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


# KHÓA LUẬN TỐT NGHIỆP

*Undergraduate Thesis*

***CHUYÊN ĐỀ:***

# HỆ THỐNG PHÂN TÍCH VÀ DỰ ĐOÁN MÔ HÌNH PHÊ DUYỆT THẺ TÍN DỤNG

**Giáo viên hướng dẫn:** Hoàng Thị Phương Thảo

**Mã lớp học phần:** MAT4082

**Sinh viên:** Tạ Quang Tùng

**Lớp:** K66A2 Toán Tin

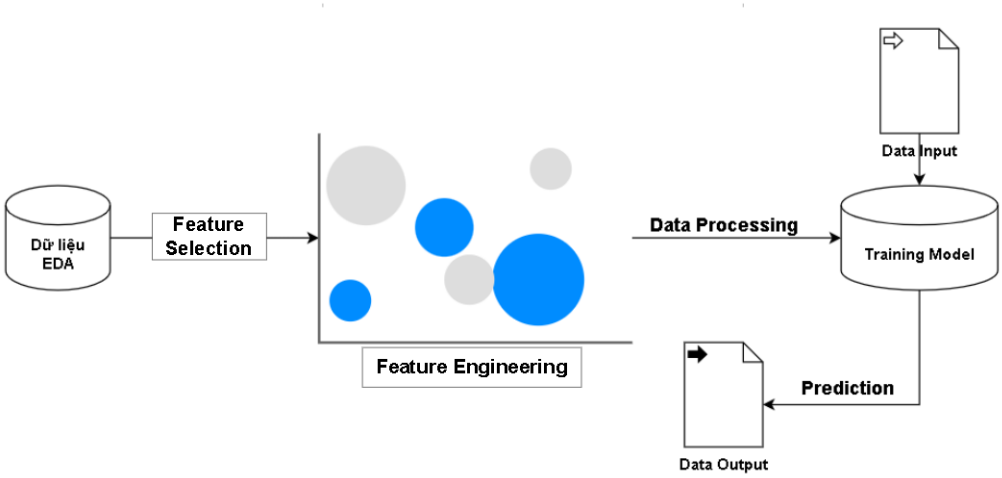
**Hà Nội, 2025**

# Mục lục

# 1 Lộ trình thực hiện dự đoán sự phê duyệt của thẻ tín dụng

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

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 1: Quy trình Dự đoán phê duyệt Thẻ Tín dụng

=> Mục đích: dự đoán xác suất thẻ ngân hàng được phê duyệt mà không ảnh hưởng đến điểm tín dụng.

## 1.2 Hồ sơ phù hợp của ứng viên:

Sau khi phân tích các đặc trưng theo các phương pháp phân tích dữ liệu bên dưới như là phân tích đơn biến, phân tích đa biến, phân tích phương sai như bên dưới thì ta có thể tạo ra hồ sơ đăng ký thẻ tín dụng điển hình của ứng viên để phục vụ cho quá trình huấn luyện và dự đoán

của mô hình.

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).

## 1.3 Tiêu chí phê duyệt trong thẻ tín dụng:

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 hữa 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 Tổng quan về bộ dữ liệu

## 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.

Link datasets: kaggle.com/datasets/credit-card-approval-prediction

## 2.2 Trường thông tin của bộ dữ liệu:

### 2.2.1 Application Record Dataset:

| 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 | |

Hồ sơ thông tin cơ bản của ứng viên

### 2.2.2 Credit Record Dataset

| 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 |

Hồ sơ tín dụng và trạng thái của ứng viên đó

# 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 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 dự án 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.*

## 3.2 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.

## 3.3 Phân tích dữ liệu:

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 bổ 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, phương sai

### 3.3.1 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.

- 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

- 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 dung cao hay không (IsHighRisk).

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

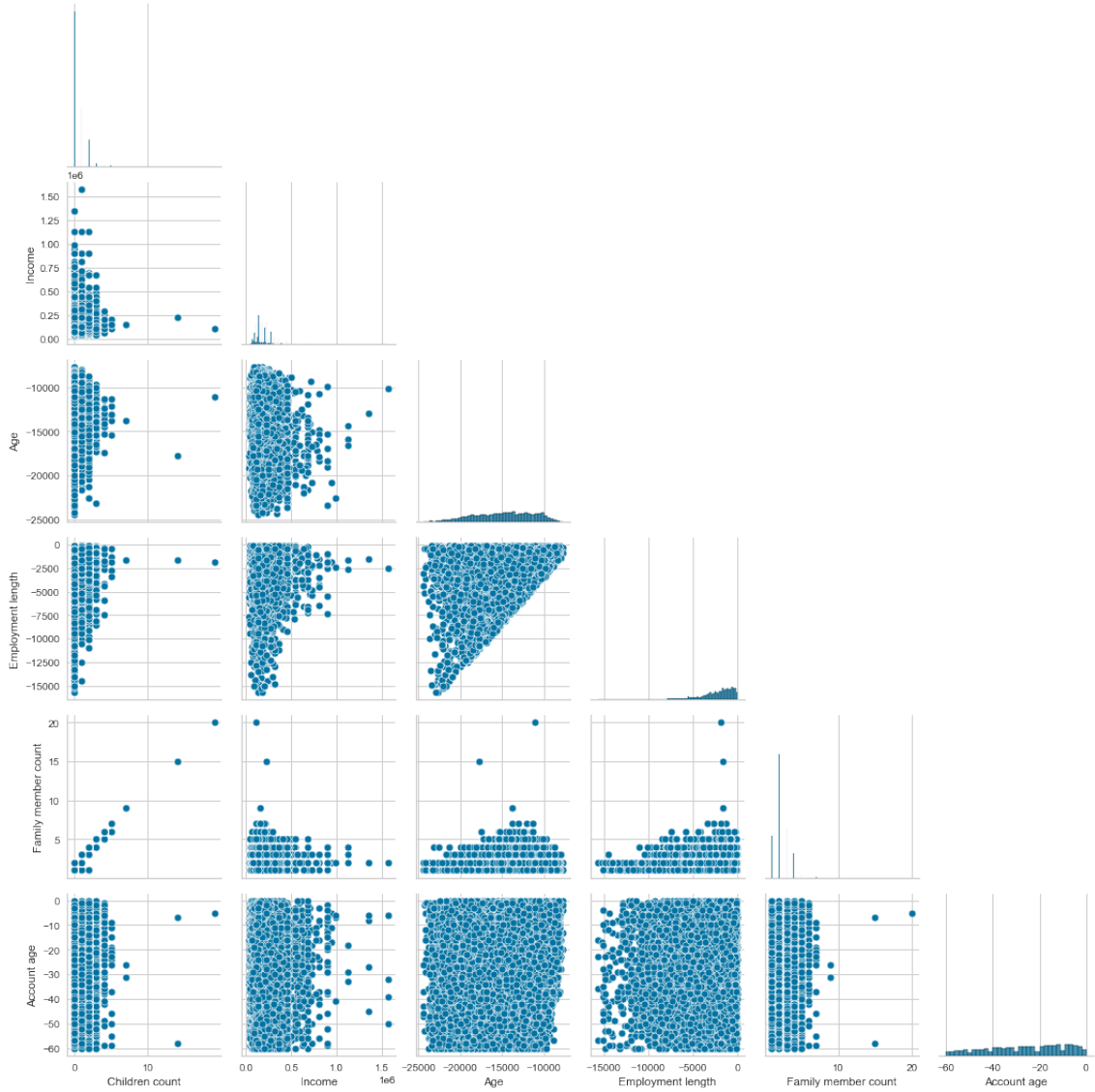
| **Đặc trưng** | **Ý nghĩa** |
|---|---|
| "Gender" | Giới tính khác hàng |
| "Age" | Độ tuổi khách hàng |
| "Marital status" | Tình trạng hôn nhân |
| "Family member count" | Số lượng thành viên trong gia đình |
| "Children count" | Số lượng con cái |
| "Dwelling" | Loại hình nhà ở |
| "Income" | Thu 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? |

### 3.3.2 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.

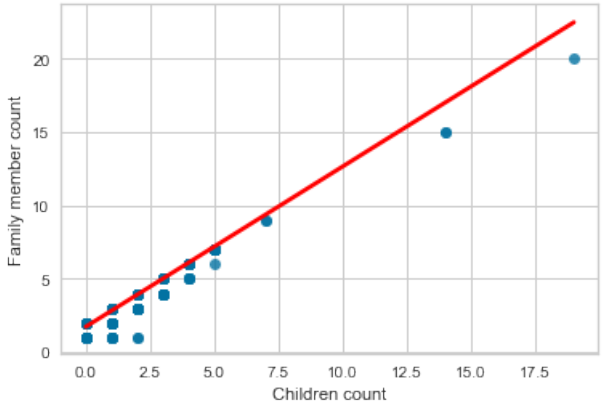
Trong bài toán này, ta cần phân tích đa biến giữa các đặc trưng khác như "Children count", "Income", "Age", "Employment length", "Family member count", "Account Age" để thấy rõ được mối quan hệ và tính tương quan giữa từng cặp biến với nhau.

Trong dự án này, ta có biểu đồ mối quan hệ giữa hai biến như sau:

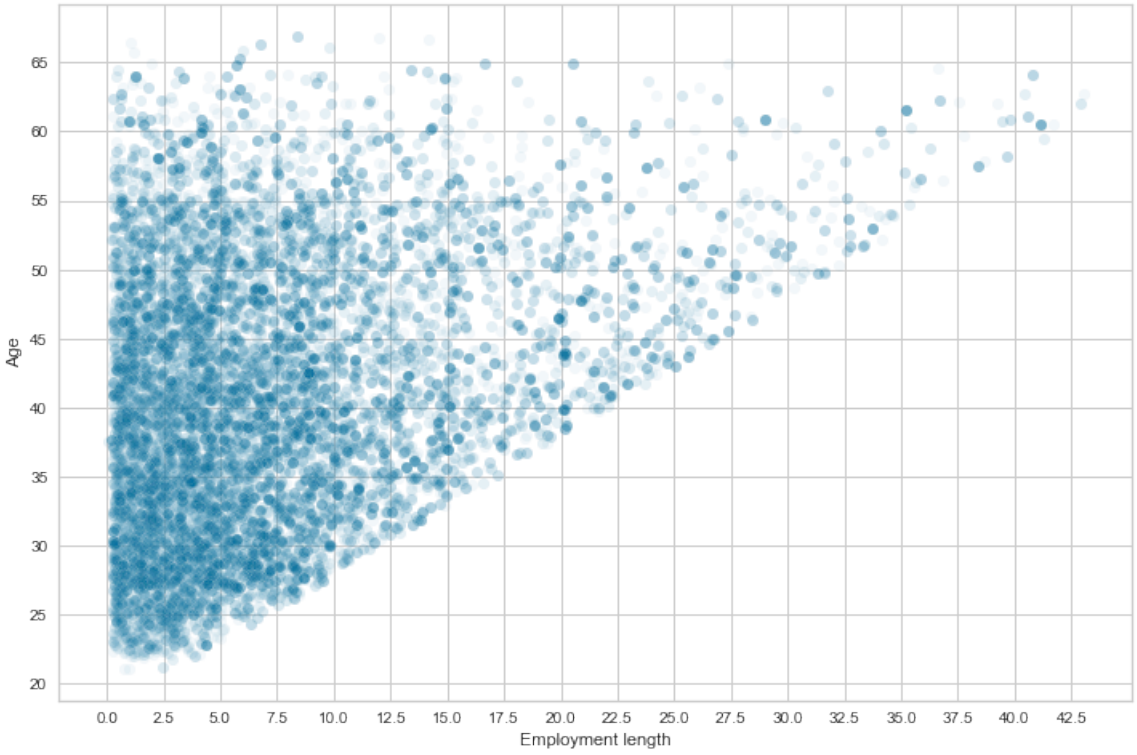



Biểu đồ thể hiện mối tương quan giữa hai biến với nhau

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.



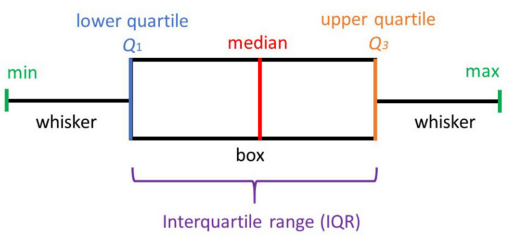
Tương tự đối với mối quan hệ giữa hai đặc trưng "Age" và "Employment Length" cũng là có mối quan hệ tuyến tính qua biểu đồ Scatter Plot nên cũng sẽ cần loại bỏ đi một trong hai đặc trưng đi để đảm bảo quá trình huấn luyện mô hình diễn ra tốt nhất.




### 3.3.3 Phân tích phương sai (ANOVA):

Phân tích phương sai ANOVA là phép kiểm định thống kê được sử dụng để đánh giá sự khác biệt giữa giá trị trung bình của nhiều hơn hai nhóm dữ liệu. Đây chính là công cụ giúp xác định ảnh hưởng của các biến độc lập đối với biến phụ thuộc trong các bài tóan về hồi quy.

Trong dự án này, ta sẽ sử dụng phân tích phương sai giữa hai đặc trưng "Age" và "Employment length" với các đặc trưng độc lập khác như là: "Gender", "Has a car", "Has a property", "Employment status", "Education level", "Maritual status", "Dwelling", "Job title" thông qua biểu đồ hộp Box Plot nhằm phân tích các giá trị ngoại lai trong đặc trưng.



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

- 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
- Chuẩn hóa 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
- Chuyển đổi dữ liệu dạng số: 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.

## 4.1 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[feature] = $\{x_1, x_2, ..., x_n\}$ 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:

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

Ta có:

- Q1 là giá trị phân vị thứ 25% (r = 1)
- Q3 là giá trị phân vị thứ 75% (r = 3)
- IQR = Q3 - Q1 là khoảng tứ phân vị

*Phương án: Loại bỏ các mẫu dữ liệu ra khỏi tập dữ liệu nếu chúng nằm ngoài khoảng [Q1-3×IQR, Q3+3×IQR]*

## 4.2 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 overfiting

*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.*

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

### 4.3.1 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.

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 sẽ 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 |
| ... | ... | ... | ... | ... | ... |

### 4.3.2 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.

Cho bảng dữ liệu sau biết rằng A > B > C > D > ..., 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 |
| ... | ... |

### 4.3.3 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:

1. 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

2. 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
   - 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:

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

3. Kết hợp dữ liệu gốc và dữ liệu mới đã tổng hợp trên để tạo thành tập dữ liệu cân bằng

## 4.4 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.

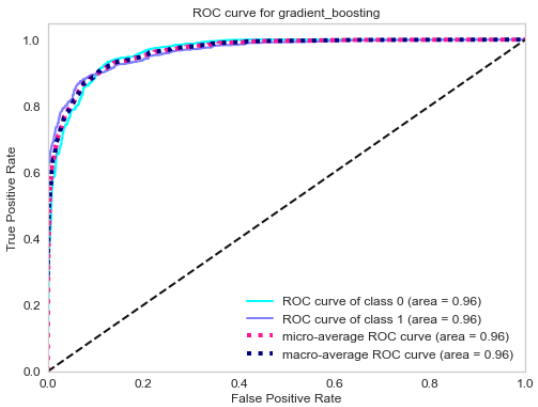
## 4.5 Lựa chọn và huấn luyện mô hình:

Ngoài việc sử dụng mô hình Gradient Boosting Classifier ngay sau đây, trong dự án này còn sử dụng một số mô hình học máy khác nhằm để so sánh độ chính xác giữa các mô hình với nhau như là: Support Vector Machine (SVM), Adaboost Classifier, ...

Sau khi thử nghiệm từng mô hình, ta có thể thấy được mô hình phù hợp nhất cho bài toán là Gradient Boosting Classifier khi dựa vào đường cong ROC và chỉ số Recall của mô hình đó.

---------- gradient_boosting ----------

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.90 | 0.91 | 0.90 | 23272 |
| 1 | 0.91 | 0.90 | 0.90 | 23272 |
| accuracy | | | 0.90 | 46544 |
| macro avg | 0.90 | 0.90 | 0.90 | 46544 |
| weighted avg | 0.90 | 0.90 | 0.90 | 46544 |



# 5 Mô hình Gradient Boosting Classifier:

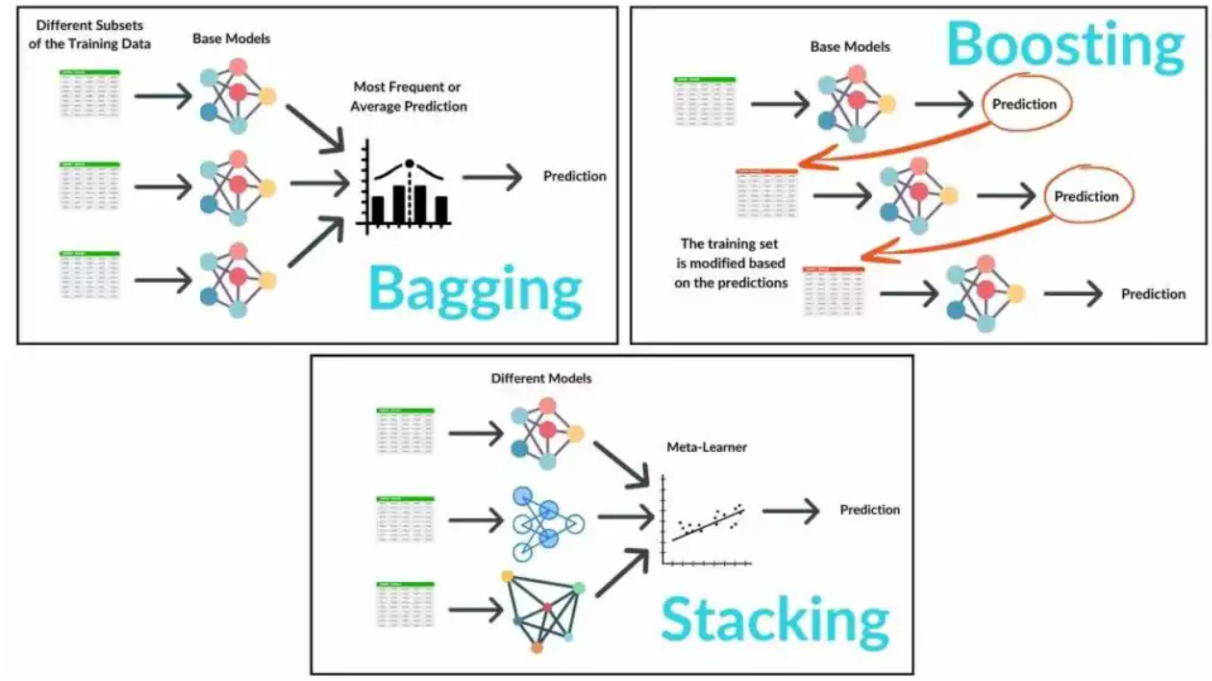
## 5.1 Phương pháp Ensemble Learning:

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:

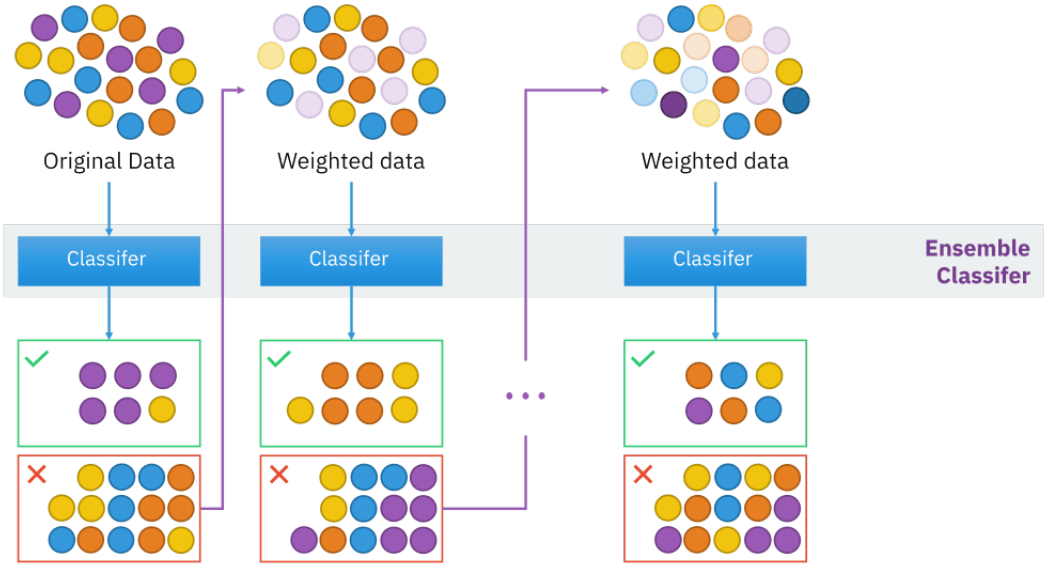
- **Bagging (đóng bao)**: xây dựng số lượng lớn các models cùng loại trên những subsamples khác nhau từ tập Training Dataset một cách song song độc lập nhằm đưa ra dự đoán tốt hơn.
- **Boosting (tăng cường)**: xây dựng số lượng lớn các models cùng loại. Tuy nhiên quá trình huấn luyện trong phương pháp 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.



## 5.2 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 (Desicion Tree).

Gradient Boosting hoạt động dựa trên ý tưởng tối ưu hóa hàm mất mát (Loss Function) bằng cách sử dụng phương pháp Gradient Descent. Phương pháp này xây dựng mô hình theo từng bước, trong đó mỗi mô hình mới được thêm vào để giảm lỗi của các dự đoán trước đó (Residual Error).




## 5.3 Quy trình hoạt động:

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

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

Biết rằng bộ dữ liệu trên thỏa mãn ánh xạ:

$$\begin{array}{ccc} \text{f:} & R^m & \longrightarrow & \text{R} \\ & (X_1, X_2, ..., X_n) & & F(X_1, X_2, ..., X_n) \end{array}$$

*Chú ý: Mỗi vector $X_i = (x_{i1}, x_{i2}, ..., x_{im})$ 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 (Inital Model): mô hình dự đoán giá trị ban đầu $F_t(X_i)$ cho tất cả các điểm dữ liệu:

$$F_0(X_i) = log(\frac{\hat{p_i}^{(0)}}{1 - \hat{p_i}^{(0)}}) \quad \text{với} \quad \hat{p_i}^{(0)} = \frac{1}{n}\sum_{j=1}^{n} y_j$$

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 mới dạng cây quyết định (Decision Tree) là $h_t(X_i)$ để dự đoán Gradient $g_i^{(t)}$ có công thức như sau:

$$h_t(X_i) = \begin{cases} \dfrac{1}{|J_t|}\displaystyle\sum_{j=1}^{n} g_j^{(t)} & \text{nếu } \hat{p_i}^{(t-1)} \geq 0.5 \\ \dfrac{1}{n - |J_t|}\displaystyle\sum_{j=1}^{n} g_j^{(t)} & \text{nếu } \hat{p_i}^{(t-1)} < 0.5 \end{cases}$$

     trong đó $|J_t|$ là số lượng lá của cây quyết định thỏa mãn điều kiện $\hat{p_i}^{(t-1)} \geq 0.5$ tại bước thứ 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:

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

     – Cập nhật lại mô hình:

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

$\Rightarrow$ 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 $|L(F_t(X_i)) - L(F_{t-1}(X_i))| < \epsilon$ 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:

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

## 5.4 Mô hình thuật toán GBC:

Trong quá trình huấn luyện mô hình, thuật toán Gradient Boosting Classifier 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 $F_0(X_i) = \log(\frac{\overline{p}}{1-\overline{p}})$ và $\overline{p} = \frac{1}{n}\sum_{j=1}^{n} y_j$
- T là số vòng lặp tối đa của thuật toán
- Dự đoán giá trị bằng hàm Sigmoid là $\hat{y_i} = \frac{1}{1+e^{-F(X_i)}} = \begin{cases} 1 \text{ nếu } \hat{y_i} \geq 0.5 \\ 0 \text{ nếu ngược lại} \end{cases}$

## 5.5 Kết quả từ phân tích và dự đoán:

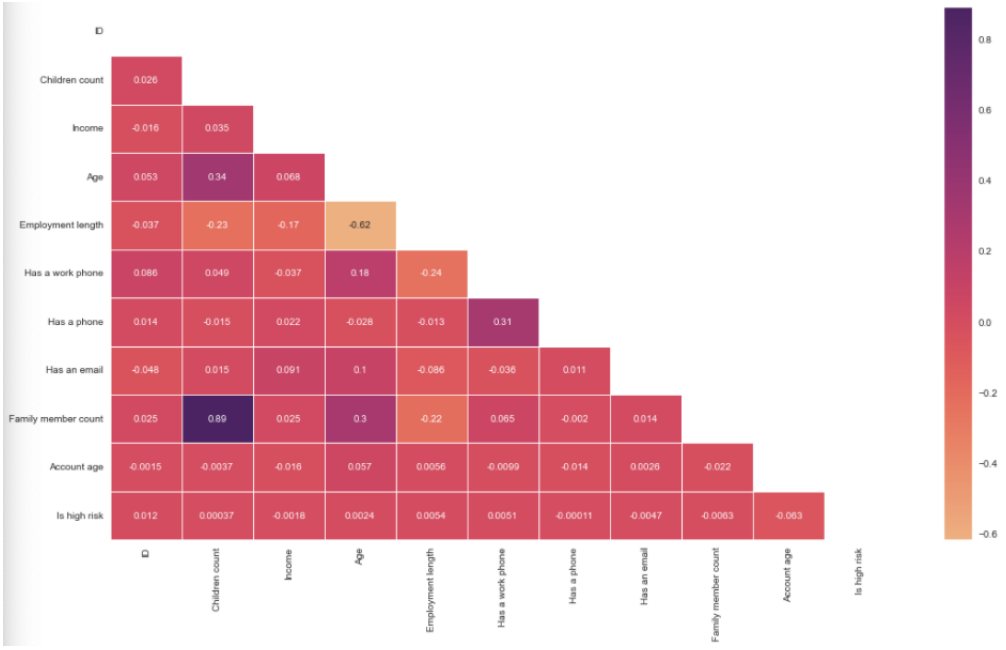
### 5.5.1 Mối tương quan giữa các đặc trưng:

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 $< 0$ => tương quan nghịch biển (A tăng B giảm, A giảm B tăng)
  - Nếu giá trị $= 0$ => không có mối tương quan giữa hai đặc trưng

- Màu sắc càng đậm thì mối tương quan càng mạnh và ngược lại

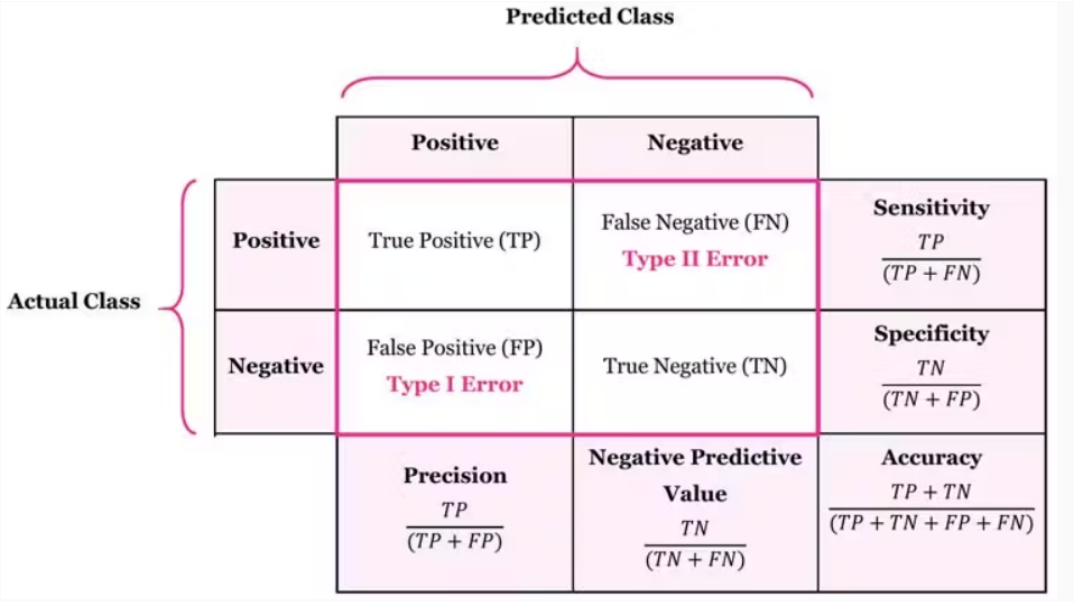



Hình 2: Ma trận tương quan các đặc trưng bằng Heatmap

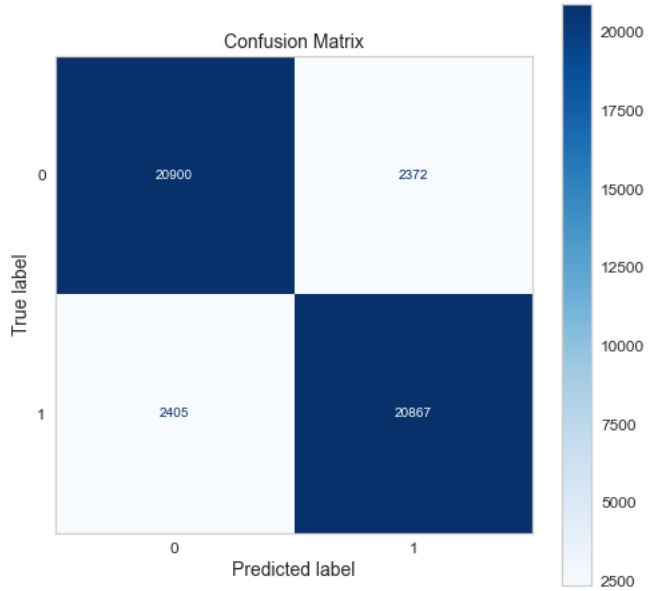
### 5.5.2 Ma trận nhầm lẫn:

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




Trong dự án này đã đưa ra ma trận nhầm lẫn (Confusion Matrix) thông qua thuật toán phân loại Gradient Boosting Classifier như sau:

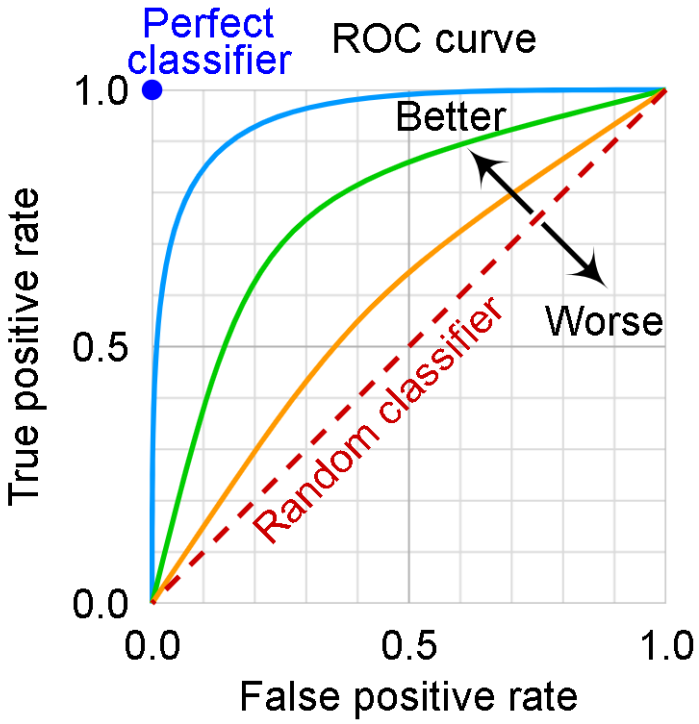



### 5.5.3 Đường cong ROC:

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 overfiting (quá khớp)




- 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

### 5.5.4 So sánh giữa các mô hình:

| Model | Recall Score |
|---|---|
| Gradient Boosting | 90% |
| Support Vector Machine | 88% |
| Adaboost | 71% |
| Logistic Regression | 61% |
| Stochastic Gradient Descent | 52% |

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:

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

- Recall đo lường khả năng của mô hình trong việc phát hiện đúng các đối tượng thuộc lớp dương tính, 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)

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

### 5.5.5 Lý do lựa chọn Gradient Boosting:

- 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 và SGD 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 mà không phải chọn Kernel phù hợp như SVM
- Ít nhạy cảm với nhiễu hơn so với AdaBoost vì chúng tối ưu hóa dựa trên gradient, đặc biệt đối với các điểm ngoại lệ Outliers

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

Sau khi đã chọn lọc và training được mô hình Gradient Boosting Classifier 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.

## 6.1 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.

## 6.2 Triển khai mô hình với Streamlit:

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.*

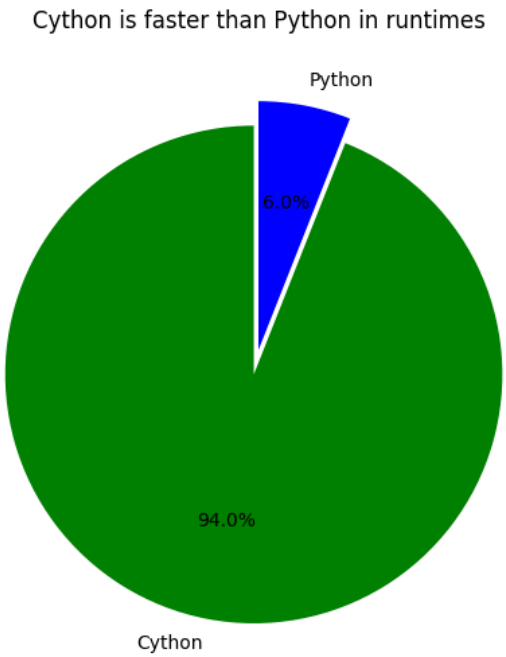
## 6.3 Ngôn ngữ lập trình mở rộng Cython:

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.




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.

# Kết luận

Demo Web: creditcardapprovalpredictiondeployment.streamlit.app
Source Code: Credit Card Project

# Tài liệu

[1] Giulio Carlone (2021), *Introduction to Credit Risk*, CRC Press, Boca Raton United States of America.

[2] Khan Academy (2025), Multivariable Calculus Tutorial

[3] Stern Semasuka (2022), Blog for Credit Card Approval Prediction

[4] Phạm Đình Khánh (2021), Credit Score Card with Deep AI Book

[5] Kaggle (2024), Free Datasets for Data Science Community