Báo cáo đồ án môn CS116

Đặng Chí Thành[1, 2], Trần Huỳnh Quốc An[1, 2], Đặng Thị Thúy Hồng[1, 2], Đinh Văn Nguyên[1, 2]
[1]Khoa Khoa học và Kỹ thuật thông tin, Trường Đại học Công nghệ Thông tin
[2]Đại học Quốc gia Thành phố Hồ Chí Minh
{20520761, 20520955, 20520523, 20520657}@gm.uit.edu.vn

Abstract

Trong bài báo cáo này, nhóm đã áp dụng các phương pháp phân tích dữ liệu (EDA), các kỹ thuật lựa chọn đặc trưng (Feature Selection), và các mô hình máy học khác nhau trên bộ dữ liệu COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) và DIABETES với mục tiêu là tìm ra phương pháp tốt nhất áp dụng cho bài toán dự đoán trên 2 bộ dữ liệu. (Video báo cáo đồ án của nhóm: link)

1 Exploratory Data Analysis

1.1 Compas

1.1.1 Giới thiệu bộ dữ liệu

Bộ dữ liệu COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) là một bộ dữ liệu lớn được sử dụng trong lĩnh vực hệ thống tư pháp để nghiên cứu và đánh giá các hệ thống dự đoán nguy cơ phạm tội của đối tượng. Bộ dữ liệu này chủ yếu tập trung vào việc dự đoán khả năng tái phạm trong vòng hai năm sau khi đối tượng ra tù.

Bộ dữ liệu COMPAS bao gồm nhiều thuộc tính, nhưng trong đề tài này, nhóm tập trung vào một số thuộc tính quan trọng, các thuộc tính được mô tả trong Bảng 1.

Trong đó, two_year_recid là biến mục tiêu với giá trị 0 là không có khả năng phạm tội và 1 là có khả năng phạm tội trong 2 năm tiếp theo.

1.1.2 Phân phối dữ liệu

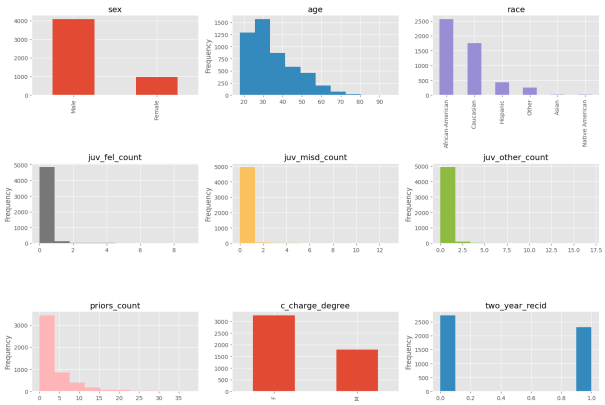
Về đặc trưng 'Sex' (Giới tính), chúng ta thấy rằng bộ dữ liệu có sự chênh lệch khá lớn về nhãn này.

Về đặc trưng 'Age' (Độ tuổi), mặc dù phạm vi rất rộng nhưng chỉ tập trung chủ yếu (Q1-Q3) trong khoảng 25 - 42 tuổi và có giá trị trung bình và trung vị lần lượt là 34.8 và 31.

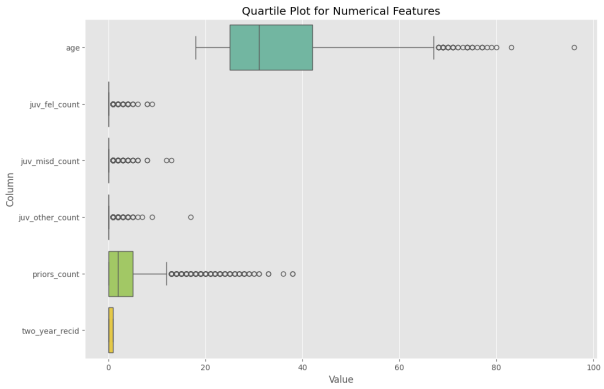
Về dữ liệu phạm tội, các đặc trưng 'juv_fel_count', 'juv_misd_count', 'juv_other_count', hầu hết giá trị đều là 0, tuy nhiên, với đặc trưng 'priors_count' (Số lần bị buộc tội) lại có xu hướng cao hơn, cụ thể Q1-Q3 là từ 0 đến 5 lần. Với đặc trưng về 'race' (Chủng tộc), dữ liệu khá đa dạng với 5 chủng tộc cụ thể và 1 biến là 'Others', tuy nhiên dữ liệu có sự chênh lệch đáng kể về số lượng giữa các nhóm chủng tộc, trong đó African-American và Caucasian chiếm phần lớn.

Về dữ liệu mức độ phạm tội 'c_charge_degree', số lượng người phạm tội nặng (F) vượt trội so với số người phạm tội nhẹ (M) với số lượng lần lượt là 3267 và 1782.

Về biến mục tiêu trong bộ dữ liệu là 'two_year_recid', có sự chênh lệch nhưng không quá lớn giữa tỉ lệ phạm tội và không phạm tội sau 2 năm (lần lượt là 2312 và 2737).



Ảnh 1: Phân bố giá trị của các thuộc tính



Ảnh 2: Boxplot của các thuộc tính có giá trị là số

1.1.3 Tương quan dữ liệu

Trong Hình 3, có sự tương quan dương (0,14) giữa 'Age' và 'priors_count'. Điều này có nghĩa là khi tuổi tăng, tổng số tội ác trước đó cũng có xu hướng tăng.

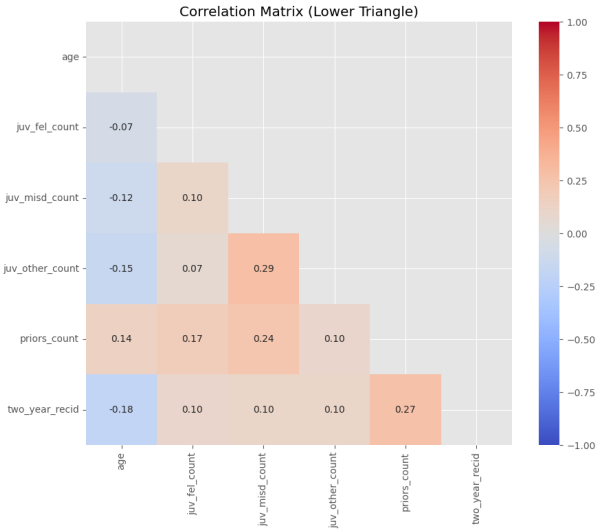
Sự tương quan dương (0,27) vừa phải giữa 'priors_count' và 'two_year_recid'. Do đó, chúng ta có thể suy luận những người có tổng số tội ác trước đó cao

Thuộc Tính	Ý Nghĩa	Phạm Vi	Ghi chú
sex	Giới Tính	Male, Female	Male: Nam Female: Nữ
age	Tuổi	> 0	Độ Tuổi
race	Sắc Tộc	Categorical	-
juv_fel_count	Tội nghiêm trọng khi còn trẻ	> 0	Lần
juv_misd_count	Tội nhẹ khi còn trẻ	> 0	Lần
juv_other_count	Tội khác khi còn trẻ	> 0	Lần
priors_count	Số lần bị buộc tội	> 0	Lần
c_charge_degree	Mức độ nghiêm trọng của tội	F, M	F: Felony - Nghiêm trọng M: Misdemeanor - Ít nghiêm trọng
two_year_recid	Khả năng tái phạm trong 2 năm	0, 1	0: Không phạm tội 1: Phạm tội

Bảng 1: Phân Tích Thuộc Tính

có khả năng cao hơn về tình trạng tái phạm trong hai năm.

Sự tương quan âm (-0.18) giữa 'Age' và biến mục tiêu 'two_year_recid'. Do đó, chúng ta có thể suy luận những người lớn tuổi hơn có khả năng ít hơn về tình trạng tái phạm trong hai năm.



Ảnh 3: Tương quan giữa các thuộc tính

1.2 Diabetes

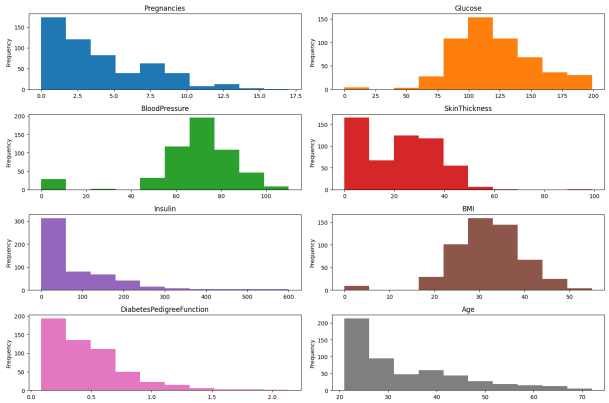
1.2.1 Giới thiệu bộ dữ liệu

Diabetes là một căn bệnh gây ra bởi lượng đường có trong máu (hay còn được gọi là đường huyết) quá cao. Lượng đường huyết này được hooc môn do tuyến tụy của con người giúp glucose từ thức ăn qua máu để đến được tế bào, qua đó tạo ra năng lượng để con người hoạt động. Tuy nhiên đôi khi cơ thể không thể tạo ra đủ lượng insulin hoặc sử dụng insulin không tốt, qua đó khiến cho lượng glucose không thể từ máu đến chỗ tế bào được. Từ đó khiến cho lượng đường huyết tăng lên. Nhằm dễ dàng hơn trong việc dự đoán bệnh tiểu đường, Kaggle đã cho ra bộ Diabetes, một bộ dữ liệu này xoay quanh thông tin sức khỏe của những người bị hoặc không bị bệnh tiểu đường. Bộ dữ liệu diabetes gồm 8 đặc trưng đầu vào và 1 cột đầu ra. Các thông tin về các đặc trưng của bộ dữ liệu được trực quan ở Bảng 2.

Output của bài toán được sử dụng để dự đoán thông tin xem người đó có bị mắc bệnh tiểu đường hay không dựa vào lượng thông tin có được từ các đặc trưng đầu vào. Output bao gồm số 0 và số 1 với: Số 0 đại diện cho việc một người không bị mắc bệnh tiểu đường Số 1 thể hiện người bị mắc bệnh tiểu đường

1.2.2 Phân phối dữ liệu

Trong quá trình phân tích bộ dữ liệu Diabetes, nhóm phát hiện ra mỗi đặc trưng đều có điểm dữ liệu có giá trị là 0. Tuy nhiên, ngoại trừ đặc trưng "Prenancies" có thể chứa điểm dữ liệu 0, việc các đặc trưng còn lại như "Glucose", "BloodPressure" hay "Insulin" có chứa dữ liệu 0 là bất hợp lý, vì lượng Glucose, Insulin hoặc huyết áp của một người không thể bằng 0. Vì thế, nhóm coi các điểm này là điểm dữ liệu bị khuyết.

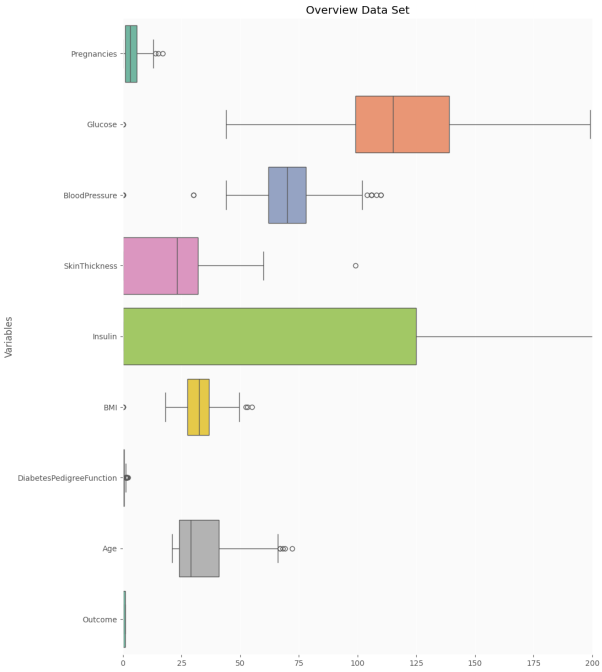


Ảnh 4: Phân bố giá trị của các đặc trưng

Nhóm xem xét các giá trị bất hợp lý và trực quan sự phân bổ các đặc trưng như ở Hình 5. Có thể thấy được ngoại trừ đặc trưng "Glucose" và "Insulin", các đặc trưng còn lại đều tồn tại một vài giá trị outlier. Sau khi thảo luận, nhóm quyết định xử lý các outlier đó như

Thuộc Tính	Ý Nghĩa	Phạm Vi	Ghi chú
Pregnancies	Số lần mang thai	Male, Female	lần
Glucose	Lượng đường glucose có trong máu	0 đến 199	mg/dL
BloodPressure	Huyết áp	0 đến 110	mm Hg
SkinThickness	Độ dày của da	0 đến 99	mm
Insulin	Lượng insulin có trong máu	0 đến 600	muU/ml
BMI	Chỉ số khối cơ thể	0 đến 55	kg/m²
DiabetesPedigreeFunction	Hệ số di truyền tiểu đường	0.084 đến 2.137	Không có đơn vị
Age	Tuổi	21 đến 72	tuổi

Bảng 2: Thông tin về các đặc trưng của bộ dữ liệu diabetes



Ảnh 5: Boxplot của các đặc trưng

Ảnh 6: Tương quan giữa các đặc trưng

cách xử lý các giá trị không hợp lệ.

1.2.3 Tương quan dữ liệu

Sau khi tiến hành trực quan sự tương quan giữa các đặc trưng với nhau và với biến mục tiêu và thể hiện qua 6, nhóm nhận thấy tất cả đặc trưng đều có tỉ lệ thuận về sự tương quan với biến mục tiêu nhưng đều không cao, chỉ trải dài từ 0.059 (BloodPressure) đến 0.47 (Glucose). Qua đó thấy được sự tác động không đáng kể của từng đặc trưng lên giá trị đầu ra. Điều này làm giảm khả năng dự đoán chính xác của mô hình.

2 Methodology

2.1 Data Preprocessing

2.1.1 Handle Inappropriate Values

Xử lý outliers: Đây là một bước quan trọng nhằm làm sạch dữ liệu và đảm bảo tính chính xác của mô hình. Outliers thường là những giá trị nằm ngoài ranh giới bình thường và có thể ảnh hưởng đến kết quả của các phương pháp thống kê và machine learning. Các kỹ thuật như sử dụng biểu đồ boxplot và xác định ngưỡng quyết định có thể giúp xác định và loại bỏ outliers.

Xử lý giá trị không hợp lý: Kiểm tra và loại bỏ các giá trị không hợp lý là bước quan trọng để đảm bảo tính nhất quán và chính xác của dữ liệu. Các giá trị không hợp lý có thể phản ánh sự sai lệch hoặc sự nhập nhằng trong quá trình thu thập dữ liệu. Bằng cách kiểm tra và phân tích, ta có thể loại bỏ những giá trị này để tăng độ tin cậy của dữ liệu.

Xử lý giá trị thiếu:

- Trong trường hợp có giá trị thiếu, kỹ thuật imputation được sử dụng để điền giá trị còn thiếu dựa trên thông tin từ các quan sát khác trong tập dữ liệu. Các phương pháp như điền bằng giá trị trung bình, trung vị hoặc sử dụng giá trị được dự đoán từ mô hình có thể được áp dụng để giữ lại sự đồng đều và tính chính xác của dữ liệu.

- Một phương pháp cao cấp hơn là sử dụng mô hình để điền giá trị khuyết. Thay vì dựa vào giá trị trung bình hoặc trung vị, ta có thể sử dụng mô hình dự đoán để ước lượng giá trị còn thiếu dựa trên mối

quan hệ và tương tác phức tạp trong dữ liệu. Điều này có thể cung cấp dự đoán chính xác hơn và giữ lại tính động của dữ liệu.

2.1.2 SMOTE

Synthetic Minority Over-sampling Technique (SMOTE) là một phương pháp trong lĩnh vực xử lý mất cân bằng dữ liệu trong học máy, đặc biệt là khi làm việc với các bài toán phân loại, nơi có sự mất cân bằng giữa các lớp. Khi dữ liệu mất cân bằng, nghĩa là một lớp có số lượng mẫu nhiều hơn rất nhiều so với lớp khác, mô hình có thể bị huấn luyện chệch và không chính xác khi dự đoán lớp thiểu số. SMOTE giúp giải quyết vấn đề này bằng cách tạo ra các mẫu dữ liệu tổng hợp (*synthetic samples*), nhằm gia tăng kích thước mẫu của lớp thiểu số.

Cụ thể, đối với mỗi mẫu thuộc lớp thiểu số, SMOTE chọn một số láng giềng gần nhất và kết hợp mẫu ban đầu với một hoặc nhiều láng giềng, tạo ra các mẫu tổng hợp. Qua đó giúp cân bằng dữ liệu và cải thiện khả năng dự đoán của mô hình.

Công thức của SMOTE có thể được mô tả một cách tổng quát như sau:

$$\text{Sample}_{\text{new}} = \text{Sample}_{\text{original}} + \lambda \times (\text{Neighbor}_{\text{random}} - \text{Sample}_{\text{original}}) \quad (1)$$

Trong đó:

- λ là tham số điều khiển mức độ biến đổi giữa mẫu tổng hợp mới và mẫu ban đầu.
- $\text{Neighbor}_{\text{random}}$ là một láng giềng ngẫu nhiên của mẫu ban đầu.
- $\text{Sample}_{\text{original}}$ là mẫu ban đầu thuộc lớp thiểu số.
- $\text{Sample}_{\text{new}}$ là mẫu tổng hợp mới được tạo ra.

Ở thời điểm hiện tại, đây là một kỹ thuật phổ biến để xử lý dữ liệu mất cân bằng. Tuy nhiên, mặc dù nó mang lại hiệu quả trong việc cải thiện hiệu suất của các mô hình học máy, SMOTE vẫn có những điểm hạn chế như tạo ra các mẫu tổng hợp mới không chính xác hoặc làm tăng kích thước của tập dữ liệu.

2.1.3 StandardScaler

Standard Scaler là một phương pháp chuẩn hóa dữ liệu thống kê, nó biến đổi các biến số của tập dữ liệu sao cho chúng có giá trị trung bình (mean) bằng 0 và độ lệch chuẩn (standard deviation) bằng 1. Điều này giúp làm giảm ảnh hưởng của các giá trị ngoại lệ và tạo ra một phân phối chuẩn (normal distribution).

Giả sử chúng ta có một tập dữ liệu X với n quan sát và p biến số. Công thức của Standard Scaler được tính như sau:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma} \quad (2)$$

Trong đó:

- X_{scaled} là ma trận mới sau khi chuẩn hóa.
- X là ma trận ban đầu của dữ liệu.
- μ là vector trung bình của từng biến số trong X.
- σ là vector độ lệch chuẩn của từng biến số trong X.

Vector trung bình μ được tính bằng:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} X_i \quad (3)$$

Vector độ lệch chuẩn σ được tính bằng:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \mu)^2} \quad (4)$$

Trong đó, X_i là giá trị của biến số i trong tập dữ liệu X.

2.2 Feature Selection

2.2.1 Compas

- **Mutual Inforamtion** là một kỹ thuật được sử dụng để đo lường độ tương tác giữa các đặc trưng trong một tập dữ liệu và giúp lựa chọn những đặc trưng quan trọng nhất đối với mục tiêu cụ thể. Đây là một cách để giảm chiều của dữ liệu và tăng cường khả năng dự đoán của mô hình thông qua việc giữ lại những đặc trưng quan trọng và loại bỏ những đặc trưng ít quan trọng. Mutual Information đo lường mức độ mà thông tin của một biến ngẫu nhiên có thể giúp dự đoán giá trị của biến ngẫu nhiên khác.

  Cụ thể, Mutual Information giữa hai biến ngẫu nhiên X và Y được định nghĩa như sau:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\left(\frac{p(x,y)}{p(x)\cdot p(y)}\right) \quad (5)$$

  Trong đó:

  - $I(X;Y)$ là Mutual Information giữa hai biến X và Y.
  - $p(x,y)$ là xác suất cộng đồng của X và Y.
  - $p(x)$ và $p(y)$ là xác suất của X và Y tương ứng.

- **Independent Component Analysis** (ICA) là một phương pháp nhằm mục đích phân tách tín hiệu (đầu vào) thành các thành phần độc lập.

  Giả sử X là ma trận dữ liệu với n quan sát và p biến số (tín hiệu hỗn hợp), S là ma trận chứa các thành phần tín hiệu độc lập, và A là ma trận trộn.

$$S = AX \quad (6)$$

  Trong đó:

  - X là ma trận ban đầu của dữ liệu.
  - A là ma trận trộn cần được tìm ra.

– S là ma trận chứa các thành phần tín hiệu độc lập.

Mục tiêu của ICA là tối ưu hóa ma trận A sao cho các thành phần trong S là độc lập tuyến tính, ta thu được S. Quá trình tối ưu hóa thường sử dụng các phương pháp như gradient descent.

2.2.2 Diabetes

- **PCA**: Thông thường để giảm chiều dữ liệu, cần phải bỏ bớt vài đặc trưng và chỉ giữ lại những đặc trưng quan trọng nhất. Nhưng do ta chưa biết đặc trưng nào quan trọng hơn đặc trưng. Hoặc trong trường hợp xấu hơn, tất cả đặc trưng đều có tầm quan trọng như nhau,khi đó việc bỏ đi bất cứ đặc trưng nào cũng sẽ dẫn tới việc bị mất lượng lớn thông tin. Phương pháp PCA sẽ sử dụng một hệ trực chuẩn nhằm tạo ra một hệ cơ sở mới sao cho thông tin dữ liệu chỉ tập trung ở một vài tọa độ, phần còn lại chỉ mang một vài thông tin. Phương pháp PCA được thực hiện theo các bước như sau:
  – Tính vector kỳ vọng của toàn bộ dữ liệu.X

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n \tag{7}$$

  – Trừ mỗi điểm dữ liệu đi vector kỳ vọng của toàn bộ dữ liệu.

$$\hat{x}_n = x_n - \bar{x} \tag{8}$$

  – Tính ma trận hiệp phương sai.

$$S = \frac{1}{N} X X^T \tag{9}$$

  – Tính các trị riêng và vector riêng có norm bằng 1 của ma trận này và sắp xếp chúng theo thứ tự giảm dần của trị riêng.
  – Lấy K trị riêng có giá trị lớn nhất, tạo ma trận U với các hàng là các vector riêng ứng với K trị riêng đã chọn.
  – Ánh xạ dữ liệu ban đầu đã chuẩn hóa xuống không gian con tìm được.
  – Thu được dữ liệu mới với toạ độ của các điểm dữ liệu trên không gian mới.

  Sau khi áp dụng PCA thì nhóm thu được bộ dữ liệu mới với 5 đặc trưng được tổng hợp từ 8 đặc trưng của bộ dữ liệu Diabetes ban đầu.

- **Variance Threshold**, còn gọi là ngưỡng sai, là một kỹ thuật được sử dụng trong quá trình tiền xử lý dữ liệu nhằm loại bỏ các biến có độ biến động (phương sai) thấp hoặc nhỏ hơn một ngưỡng cụ thể được người dùng xác định. Các biến được chọn có sự khác biệt không quá lớn giữa các mẫu dữ liệu, và do đó, chúng chỉ có ảnh hưởng nhỏ đến quá trình học máy và có thể được xem xét là không có ý nghĩa hoặc ít ý nghĩa.

  Công thức tính phương sai (*variance*) cho một biến được xác định bởi:

$$\text{Variance} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{10}$$

  Trong đó:

  – Variance là giá trị phương sai. - n là số lượng mẫu dữ liệu.
  – x_i là giá trị mẫu thứ i của biến.
  – $\bar{x}$ là giá trị trung bình của biến tính trên tất cả các mẫu.

  Khi được áp dụng lên bộ dữ liệu Diabetes, "DiabetesPedigreeFunction", với ít sự biến động hơn các thuộc tính còn lại đã được loại bỏ.

2.3 Model Selection

2.3.1 GridSearchCV

GridSearchCV là một phương pháp tìm kiếm siêu tham số (hyperparameter) phổ biến trong máy học để tối ưu hóa hiệu suất của mô hình. Một vấn đề cơ bản của các bài toán học máy rằng hiệu suất của mô hình luôn có sự phụ thuộc đáng kể vào giá trị của các siêu tham số, tuy nhiên với mỗi cơ sở dữ liệu, các siêu tham số này là khác nhau và không có cách nào để biết trước giá trị tốt nhất. Do đó, chúng ta cần thử nghiệm tất cả các giá trị có thể để tìm kiếm mô hình với những siêu tham số tốt nhất.

Đây cũng chính là ý tưởng hoạt động của GridSearchCV. Sử dụng phương pháp này có thể giúp người dùng tiết kiệm thời gian và tài nguyên hơn so với việc thực hiện tìm kiếm thủ công. Theo đó, người dùng ban đầu cần xác định một tập hợp các giá trị thử nghiệm cho các siêu tham số muốn tối ưu. Tất cả các kết hợp có thể có giữa các giá trị thử nghiệm này được tạo thành một lưới thử nghiệm và được GridSearchCV lần lượt sử dụng để huấn luyện mô hình. Sau khi hoàn tất quá trình, mô hình với hiệu suất tốt nhất sẽ được chọn dựa trên một tiêu chí đánh giá cụ thể, chẳng hạn như độ chính xác cao nhất.

Song, mặc dù GridSearchCV mang lại sự tự động hóa và tiết kiệm thời gian hơn so với việc thử nghiệm siêu tham số thủ công, nhưng không phải lúc nào đây cũng là lựa chọn tối ưu, bởi quá trình này vẫn đòi hỏi một lượng thời gian và tài nguyên đáng kể. Mặt khác, GridSearchCV sử dụng lưới siêu tham số được xác định sẵn, điều này có thể dẫn đến việc bỏ qua các giá trị tối ưu thực tế không nằm trong lưới đã xác định. Do đó, việc sử dụng GridSearchCV nên được xem xét dựa trên thời gian và nguồn tài nguyên có sẵn, cũng như cần ý thức về khả năng giới hạn của việc chọn lựa từ một lưới giá trị siêu tham số cố định.

2.3.2 Compas

- **Logistic Regression** thường được sử dụng để phân loại và phân tích dự đoán. Trong bài toán phân loại, hồi quy Logistic ước tính xác suất của một sự kiện xảy ra dựa trên các biến độc lập, kết quả thu được là xác suất nằm trong khoảng từ 0 đến 1. Để phân loại nhị phân, xác suất nhỏ hơn 0,5 sẽ dự đoán 0 trong khi xác suất lớn hơn 0,5 sẽ dự đoán 1. Tương tự thế, trong trường hợp dự đoán nhiều lớp khác nhau mỗi lớp sẽ được dự đoán một xác suất khác nhau, dự đoán cuối cùng là class có xác suất lớn nhất.

  Cơ chế hoạt động của Logistic Regression liên quan chặt chẽ đến hàm sigmoid và quá trình tối ưu hóa tham số để phân loại dữ liệu. Trong đó, hàm sigmoid (hoặc hàm logistic) là một hàm số chuyển đổi đầu vào thành một giá trị xác suất nằm trong khoảng từ 0 đến 1. Quá trình học tham số (Parameter Learning) bao gồm hàm Chi Phí (Cost Function) dùng để đo lường độ chệch giữa giá trị dự đoán với giá trị thực tế và Gradient Descent để tối ưu hóa tham số. Quá trình này được lặp lại cho đến khi hội tụ, tức là khi đạo hàm của hàm chi phí gần như bằng 0 hoặc khi đạt được một số lượng lặp tối đa được đặt trước.

  Về ưu điểm, hồi quy logistic dễ thực hiện, dễ giải thích và thường đạt kết quả tốt. Phân loại các điểm dữ liệu không xác định rất nhanh. Hồi quy logistic ít over-fitting. Về nhược điểm, các vấn đề phi tuyến tính không thể được giải quyết bằng hồi quy logistic. Hồi quy logistic có thể bị over-fitting nếu bộ dữ liệu quá nhiều chiều (L1 và L2 có thể sử dụng để tránh các trường hợp này).

  Trong đồ án này, nhóm áp dụng Logistic Regression kết hợp GridSearchCV và các phương pháp lựa chọn thuộc tính để tìm kiếm giá trị tối ưu của các siêu tham số quan trọng, lưới giá trị thử nghiệm của các siêu tham số bao gồm:

  - 'C': [0.001, 0.01, 0.1, 1, 10],
  - 'penalty': ['l1', 'l2'],
  - 'solver': ['liblinear'],
  - 'max_iter': [100, 500, 1000],

  Kết quả cho thấy, mô hình Logistic mang lại hiệu suất tốt nhất với bộ siêu tham số sau 'C': 0.01, 'max_iter': 100, 'penalty': 'l1', 'solver': 'liblinear'.

- **HistGradientBoostingClassifier** dựa trên thuật toán Gradient Boosting, một kỹ thuật học máy ensemble, nơi nhiều cây quyết định yếu (weak learners) được xây dựng và kết hợp để tạo ra một mô hình dự đoán mạnh mẽ.

  Khác với các triển khai truyền thống của Gradient Boosting, HistGradientBoostingClassifier sử dụng histogram-based learning để tăng tốc quá trình học trên dữ liệu lớn. Thay vì sử dụng toàn bộ dữ liệu để xây dựng cây quyết định, thuật toán sử dụng histogram để biểu diễn và tính toán trên các bin dữ liệu, giảm độ phức tạp tính toán và tăng tốc quá trình học. Mô hình này có khả năng xử lý dữ liệu thiếu một cách linh hoạt mà không cần phải điền giá trị thiếu trước. Mô hình có các siêu tham số (hyperparameters) được tích hợp để kiểm soát quá trình xây dựng cây và ngăn chặn việc overfitting.

  Trong đồ án này, nhóm áp dụng HistGradientBoostingClassifier kết hợp GridSearchCV và các phương pháp lựa chọn thuộc tính để tìm kiếm giá trị tối ưu của các siêu tham số quan trọng, lưới giá trị thử nghiệm của các siêu tham số bao gồm:

  - 'learning_rate': [0.01, 0.1, 0.2],
  - 'max_depth': [3, 5, 7],
  - 'max_iter': [100, 200, 300],

  Kết quả cho thấy, mô hình HistGradientBoostingClassifier mang lại hiệu suất tốt nhất với bộ siêu tham số sau 'learning_rate': 0.01, 'max_depth': 3, 'max_iter': 300.

2.3.3 Diabetes

- **K-Nearest Neighbors (KNN)** là một thuật toán học có giám sát tuy rất đơn giản nhưng mang lại hiệu quả trong một số tình huống. Ý tưởng cơ bản của KNN là giả định rằng các dữ liệu tương tự sẽ tập trung gần nhau trong một không gian. Từ đó, để dự đoán kết quả cho một điểm dữ liệu mới, ta cần tìm k điểm gần nhất và xem xét giá trị của chúng.

  Về cơ chế hoạt động, KNN không học từ dữ liệu huấn luyện. Tất cả tính toán chỉ được thực hiện khi cần dự đoán kết quả cho dữ liệu mới. Cụ thể, trong bài toán phân loại, quá trình này bao gồm việc xác định K điểm gần nhất từ tập dữ liệu huấn luyện dựa trên một độ đo khoảng cách như Euclidean Distance. Sau đó, KNN quyết định nhãn của điểm mới thông qua bầu chọn nhãn theo đa số của K điểm trên.

  Tuy nhiên, KNN khá nhạy cảm và có khả năng nhiễu cao khi số lượng K nhỏ. Mặt khác, thuật toán này đòi hỏi lưu toàn bộ dữ liệu training trong bộ nhớ, ảnh hưởng đến hiệu suất trong việc xử lý những dữ liệu lớn với số chiều cao.

  Trong đồ án này, nhóm áp dụng $\mathrm{KNeighborsClassifier}$, một biến thể của K-Nearest Neighbors được thiết kế cho tác vụ phân loại, kết hợp $\mathrm{GridSearchCV}$ và các phương pháp lựa chọn thuộc tính để tìm kiếm giá trị tối ưu của 2 siêu tham số quan trọng, bao gồm $\mathrm{n\_neighbors}$ và $\mathrm{algorithm}$. Lưới giá trị thử nghiệm của nhóm bao gồm:

  - 'n_neighbors': [5, 10, 20, 50, 100],

– 'algorithm': ["auto", "ball_tree", "kd_tree", "brute"]

Kết quả cho thấy, mô hình KNeighborsClassifier mang lại hiệu suất tốt nhất khi kết hợp SMOTE, với n_neighbors là 10 và algorithm là "brute".

- **Random Forest Classifier** (RFC) là thuật toán học có giám sát được xây dựng dựa trên phương pháp ensemble learning (học kết hợp), bằng cách kết hợp các mô hình con, mô hình được sử dụng ở đây là Decision tree. Mỗi cây quyết định sẽ được xây dựng trên một tập dữ liệu khác nhau, và chỉ sử dụng một phần các đặc trưng. Sau đó, Random Forest sẽ đưa ra kết quả dự đoán thông qua kết quả trong quá trình bỏ phiếu của các cây quyết định.

  Mỗi cây quyết định trong Random Forest được xây dựng trên một tập dữ liệu con khác nhau được lấy mẫu ngẫu nhiên từ tập dữ liệu huấn luyện. Điều này giúp giảm nguy cơ overfitting, nơi mô hình quá mức khớp với dữ liệu huấn luyện và không thể tổng quát hóa tốt cho dữ liệu mới. Hơn nữa, mỗi cây chỉ sử dụng một phần ngẫu nhiên của các đặc trưng để xây dựng, làm cho chúng đa dạng và độc lập. Việc kết hợp các cây lại giúp mô hình tăng tính tổng quát hóa tốt hơn so với một cây con riêng lẻ.

  Tuy nhiên, Random forest lại có nhược điểm về mặt giải thích cho người dùng, đặc biệt là khi số lượng cây lớn. Bên cạnh đó, việc xây dựng và dự đoán trên một Random Forest có thể tốn nhiều thời gian và tài nguyên. Đối với thuật toán Random forest, nhóm sử dụng thuật toán này với các tham số sau:

  – 'n_estimator': [50, 100, 150, 200],
  – 'max_depth': [5, 10, 15],
  – 'min_samples_split': [2, 3, 4, 5],
  – 'min_samples_leaf': [1, 2, 3, 4, 5],

  Sau khi huấn luyện trên tập train và kiểm thử trên tập dev, mô hình cho kết quả tốt nhất khi kết hợp với SMOTE và PCA với tham số của Random forest là max_depth= 10, min_samples_leaf= 1, min_samples_split= 2 và n_estimators= 100

2.4 Evaluation

2.4.1 Evaluation Metrics

F1-score là một phương pháp đánh giá hiệu suất của mô hình học máy, là trung bình điều hòa giữa độ chính xác (precision) và độ phủ (recall) của mô hình. Trong đó, **Precison** đo lường tỷ lệ các trường hợp dự đoán đúng trong số các trường hợp được dự đoán là đúng và **Recall** đo lường tỷ lệ các trường hợp dự đoán đúng trong số các trường hợp thực sự đúng.

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

		Actual	
		Positive	**Negative**
Prediction	**Positive**	True Positive	False Positive
	Negative	False Negative	True Negative

Bảng 3: Ma trận nhầm lẫn của F1

Trong đó, Precision và Recall được tính như sau:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{12}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{13}$$

2.4.2 K-Fold Cross-Validation

Nhằm đánh giá mô hình một cách toàn diện và chính xác nhất, nhóm sử dụng K-Fold Cross-Validation. Đây là một kỹ thuật quan trọng trong đánh giá hiệu suất của mô hình máy học, giúp đánh giá khả năng tổng quát hóa của mô hình trên dữ liệu mới.

Cụ thể, trong K-Fold Cross-Validation, thay vì chỉ dùng một phần dữ liệu làm tập huấn luyện thì phương pháp này sẽ dùng toàn bộ dữ liệu để dạy cho máy. Khi tham số K được lựa chọn, dữ liệu ban đầu sẽ được xáo trộn một cách ngẫu nhiên, sau đó chia thành K nhóm (K-Fold). Trong đó, một nhóm sẽ đóng vai trò là tập kiểm thử và các nhóm còn lại được sử dụng để đào tạo mô hình. Quá trình này được lặp lại K lần với mỗi lần là một tập kiểm thử khác nhau. Kết quả cuối cùng là trung bình của các kết quả từ các lượt kiểm thử. Mặc dù sẽ tốn khá nhiều thời gian để thực hiện song quy trình này giúp đảm bảo mô hình được việc đánh giá trên toàn bộ tập dữ liệu và tránh được tình trạng quá khớp hoặc chưa khớp do tập kiểm thử khác nhau.

Trong đồ án này, nhóm áp dụng phương pháp đánh giá K-Fold Cross-Validation cho các tổ hợp mô hình, với tiêu chí là F1-score, nhóm nhận thấy 02 mô hình thể hiện tốt nhất trên 02 bộ dữ liệu là RandomForestClassifier và HistGradientBoostingClassifier, thông tin cụ thể sẽ được trình bày trong Phần 3.

3 Result and Discussion

3.1 Compas

Nhận xét: Nhóm tiến hành thực nghiệm với 2 mô hình thể hiện tốt nhất trên bộ dữ liệu Compas là LogisticRegression và HistGradientBoostingClassifier và thu được kết quả như Bảng 4. Kết quả tốt nhất đạt được trên bộ dữ liệu Compas khi kết hợp HistGradientBoostingClassifier và StandardScaler với phương pháp trích xuất đặc trưng ICA đạt được khi đánh giá trên tập Dev là 69.73% và 63.34% khi đánh trên tập Test.

Tiền xử lý	Trích xuất đặc trưng	Mô hình	K-Fold Cross-Validation	Đánh giá trên X_dev	Đánh giá trên X_test
SMOTE	Mutual Information	LogisticRegression	64.94%	66.65%	59.77%
StandardScaler	Mutual Information	LogisticRegression	63.48%	64.57%	57.63%
SMOTE	ICA	HistGradientBoostingClassifier	63.72%	66.87%	60.09%
StandardScaler	ICA	HistGradientBoostingClassifier	67.03%	69.73%	63.34%

Bảng 4: Kết quả thực nghiệm trên bộ dữ liệu Compas

Tiền xử lý	Trích xuất đặc trưng	Mô hình	K-Fold Cross-Validation	Đánh giá trên X_dev	Đánh giá trên X_test
SMOTE	Variance Threshold	RandomForestClassifier	81.98%	86.9%	71.48%
StandardScaler	Variance Threshold	RandomForestClassifier	78.03%	79.23%	75.25%
SMOTE	PCA	K-Nearest Neighbors	81.14%	75.32%	73.77%
StandardScaler	PCA	K-Nearest Neighbors	72.25%	79.22%	76.65%

Bảng 5: Kết quả thực nghiệm trên bộ dữ liệu Diabetes

3.2 Diabetes

Nhận xét: Nhóm đã tiến hành thực nghiệm trên bộ dữ liệu Diabetes bằng cách kết hợp các phương pháp tiền xử lý khác nhau với 2 mô hình RandomForestClassifier và K-Nearest Neighbors và thu được kết quả như Bảng 5. Kết quả cho thấy bộ dữ liệu Diabetes khi áp dụng phương pháp tiền xử lý SMOTE luôn đạt kết quả cao hơn so với khi áp dụng phương pháp tiền xử lý Standard Scaler. Kết quả tốt nhất thu được trên tập Dev là 86.9% và trên tập Test là 71.48% với mô hình RandomForestClassifier khi áp dụng phương pháp tiền xử lý SMOTE và trích xuất đặc trưng Variance Threshold.

4 Conclusion

4.1 Compas

Tóm lại, trong đồ án lần này, nhóm đã tìm hiểu, phân tích bộ dữ liệu Compas và sử dụng một số phương pháp để tiền xử lý dữ liệu. Qua quá trình thực nghiệm trên nhiều mô hình với nhiều tham số khác nhau, nhóm nhận thấy mô hình LogisticRegression và HistGradientBoostingClassifier là hai mô hình thể hiện tốt nhất trên bộ dữ liệu Compas. Bên cạnh đó, nhóm đã kết hợp mô hình với các phương pháp tiền xử lý dữ liệu khác nhau để thực hiện bài toán phân loại và đạt được kết quả tốt nhất khi kết hợp mô hình HistGradientBoostingClassifier với phương pháp StandardScaler.

4.2 Diabetes

Đối với bộ dữ liệu Diabetes, sau khi tìm hiểu, phân tích trực quan bộ dữ liệu, nhóm đã áp dụng một số phương pháp tiền xử lý và trích xuất đặc trưng khác nhau. Sau khi huấn luyện và thực nghiệm bằng nhiều mô hình với nhiều tham số khác nhau, nhóm nhận thấy việc sử dụng SMOTE mang lại hiệu suất cao hơn so với StandardScaler trong quá trình tiền xử lý dữ liệu. Qua đó thu được kết quả tốt nhất trên tập kiểm thử là 86.9%, chứng minh hiệu quả của mô hình RandomForestClassifier khi kết hợp với kỹ thuật tiền xử lý SMOTE và phương pháp trích xuất đặc trưng Variance Threshold.