A picture containing picture frame, rectangle, screenshot, frame

Description automatically generated

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

Ảnh có chứa văn bản, mẫu họa

Mô tả được tạo tự động**KHOA CÔNG NGHỆ THÔNG TIN**

**Môn học: Khai Phá Dữ Liệu**

**Đề tài**

**ỨNG DỤNG CÁC KỸ THUẬT HỌC MÁY VÀO PHÂN TÍCH, DỰ ĐOÁN**

**NGUY CƠ BỊ ĐỘT QUỴ**

GVHD: **Th.S Trần Trọng Bình**

Nhóm thực hiện đồ án: **Nhóm 3**

TP Hồ Chí Minh, tháng 05 năm 2024

# **BẢNG PHÂN CÔNG NHIỆM VỤ NHÓM 3**

**Tên đề tài: ỨNG DỤNG CÁC KỸ THUẬT HỌC MÁY VÀO PHÂN TÍCH, DỰ ĐOÁN NGUY CƠ BỊ ĐỘT QUỴ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | HỌ TÊN | MSSV | CÔNG VIỆC | KẾT QUẢ |
| 1 | Nguyễn Tấn Sương | 21133078 | * Tìm hiểu tổng quan, giới thiệu về tập dữ liệu. * Tiền xử lý dữ liệu: làm sạch, tích hợp, thu giảm, biến đổi và rời rạc hóa, trực quan hóa dữ liệu. * Áp dụng kỹ thuật SVM vào việc dự đoán nguy cơ bị đột quỵ | 100 |
| 2 | Nguyễn Trung Đức | 21142261 | * Áp dụng Decision Tree vào việc dự đoán nguy cơ bị đột quỵ * Ứng dụng Artificxial Neural Network (ANN) vào việc dự đoán nguy cơ bị đột quỵ | 100 |
| 3 | Trần Trọng Thoại | 21133084 | * Áp dụng RanDom Forest vào việc dự đoán nguy cơ bị đột quỵ * Ứng dụng K-Nearest Neighbor(KNN) vào việc dự đoán nguy cơ bị đột quỵ | 100 |
| 4 | Nguyễn Đức Duy | 21133020 | * Áp dụng Logistic Regression vào việc dự đoán nguy cơ bị đột quỵ * Ứng dụng Gradient Boosting vào việc dự đoán nguy cơ bị đột quỵ | 100 |

**MỤC LỤC**

[**BẢNG PHÂN CÔNG NHIỆM VỤ NHÓM 3** 2](#_Toc168012455)

[**MỤC LỤC** 3](#_Toc168012456)

[**PHẦN 1: MỞ ĐẦU** 4](#_Toc168012457)

[**1.** **Lời cảm ơn:** 4](#_Toc168012458)

[**2.** **Giới thiệu đề tài:** 4](#_Toc168012459)

[**PHẦN 2: NỘI DUNG** 6](#_Toc168012460)

[**Chương 1: Tổng quan về tập dữ liệu** 6](#_Toc168012461)

[**1.1** **Giới thiệu về dataset** 6](#_Toc168012462)

[**1.2** **Các đặc điểm chính của tập dữ liệu** 6](#_Toc168012463)

[**Chương 2: Tiền xử lý, làm sạch, trực quan hóa dữ liệu:** 6](#_Toc168012464)

[**2.1.** **Các bước tiền xử lý và làm sạch dữ liệu:** 6](#_Toc168012465)

[**2.2.** **Trực quan hóa dữ liệu (EDA)** 9](#_Toc168012466)

[**Chương 3: Áp dụng các mô hình, kỹ thuật, thuật toán vào việc dự đoán nguy cơ bị đột quỵ** 14](#_Toc168012467)

[**3.1. Mô hình Support Vector Machine (SVM)** 14](#_Toc168012468)

[**3.2. Mô hình Random Forest** 17](#_Toc168012469)

[**3.3. Mô hình K-Nearest Neighbors** 20](#_Toc168012470)

[**3.4. Mô hình Decision Tree** 23](#_Toc168012471)

[**3.5. Mạng Nơ-ron Nhân Tạo (ANN) với PyTorch** 25](#_Toc168012472)

[**3.6. Mô hình hồi quy Logistic** 36](#_Toc168012473)

[**3.7 Mô hình Extreme Gradient Boosting (XGBoost)** 42](#_Toc168012474)

[**PHẦN 3: TỔNG KẾT** 48](#_Toc168012475)

[**1. Kết quả đạt được:** 48](#_Toc168012476)

[**2. Hướng phát triển:** 49](#_Toc168012477)

[**3. Quá trình làm việc của nhóm và đánh giá:** 49](#_Toc168012478)

[**TÀI LIỆU THAM KHẢO** 50](#_Toc168012479)

# **PHẦN 1: MỞ ĐẦU**

## **Lời cảm ơn:**

Lời đầu tiên, chúng em xin được bày tỏ lòng biết ơn chân thành và sâu sắc nhất đến Khoa Công Nghệ Thông Tin - Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh, vì đã tạo điều kiện cho chúng em được học tập và phát triển nền tảng kiến thức sâu sắc và thực hiện đề tài này.

Bên cạnh đó, chúng em muốn gửi lời cảm ơn sâu sắc nhất đến thầy Trần Trọng Bình. Qua quá trình học tập và thực hiện đề tài, cô đã luôn đồng hành và hỗ trợ chúng em trong đề tài này.

Với sự hướng dẫn tận tình và giảng dạy đầy đủ kiến thức của thầy Trần Trọng Bình, chúng em đã học tập và hiểu được những kiến thức cơ bản về môn Phân tích dữ liệu lớn.

Tuy nhiên, kiến thức là vô tận và do khả năng hạn hẹp của chúng em, đã có những thiếu sót không tránh khỏi. Chúng em hy vọng nhận được sự góp ý tận tình từ cô, nhằm giúp chúng em rút ra bài học kinh nghiệm và hoàn thiện, cải thiện, nâng cấp sản phẩm của chúng em một cách tốt nhất có thể.

Chúng em xin chân thành cảm ơn!

1. **Giới thiệu đề tài:**

Trong những năm gần đây, đột quỵ đã trở thành một trong những nguyên nhân hàng đầu gây tử vong và tàn tật trên toàn thế giới. Đột quỵ không chỉ gây ảnh hưởng nghiêm trọng đến sức khỏe của cá nhân mà còn gây ra gánh nặng kinh tế và xã hội lớn đối với gia đình và cộng đồng. Theo Tổ chức Y tế Thế giới (WHO), mỗi năm có hàng triệu người bị đột quỵ, trong đó một phần đáng kể là những người trẻ tuổi. Điều này đặt ra yêu cầu cấp bách về việc nghiên cứu và phát triển các phương pháp hiệu quả để dự đoán và phòng ngừa đột quỵ.

Trong bối cảnh đó, các kỹ thuật học máy (Machine Learning) đã nổi lên như một công cụ mạnh mẽ trong việc phân tích dữ liệu y tế và dự đoán nguy cơ bệnh tật. Học máy không chỉ giúp phát hiện các mô hình ẩn trong dữ liệu mà còn có khả năng cải thiện độ chính xác của các dự báo nhờ vào khả năng học hỏi và tự cải thiện theo thời gian. Các thuật toán học máy có thể xử lý một lượng lớn dữ liệu phức tạp, bao gồm cả các thông tin lâm sàng, sinh học và hình ảnh y khoa, để đưa ra các dự đoán chính xác về nguy cơ đột quỵ.

Chính vì thế nhóm đã quyết định chọn đề tài "Ứng dụng các kỹ thuật học máy vào phân tích, dự đoán nguy cơ bị đột quỵ" nhằm mục tiêu khám phá và triển khai các mô hình học máy tiên tiến để phân tích các yếu tố nguy cơ và dự đoán khả năng xảy ra đột quỵ. Đồ án này sẽ tập trung vào việc thu thập và xử lý dữ liệu, xây dựng các mô hình học máy, và đánh giá hiệu quả của chúng trong việc dự đoán đột quỵ. Qua đó góp phần dự đoán nguy cơ đột quỵ của một người để phòng ngừa và điều trị sớm nhất có thể.

# **PHẦN 2: NỘI DUNG**

# **Chương 1: Tổng quan về tập dữ liệu**

## **Giới thiệu** **về dataset**

Dữ liệu được lấy trên Kaggle có URL: <https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset>

Bộ dữ liệu này được lấy từ trang kaggle chứa dữ liệu các bệnh nhân có bị đột quỵ hoặc là không bị đột quỵ và các thông tin liên quan như giới tính, tuổi tác, các bệnh nền và tình trạng hôn nhân, có hút thuốc không, … Mỗi hàng trong dữ liệu cung cấp thông tin liên quan về bệnh nhân.

Bộ dữ liệu gồm 5110 hàng và 12 cột, trong đó gồm 11 cột chứa biến có ích cho việc phân tích và cột id là không cần thiết.

## **Các đặc điểm chính của tập dữ liệu**

Dữ liệu có 5120 dòng và 12 cột Dữ liệu được mô tả như sau:

- id: định danh duy nhất

- gender: “Nam”, “Nữ” hoặc “Khác”

- age: tuổi của bệnh nhân

- hypertension: 0 nếu bệnh nhân không bị tăng huyết áp, 1 nếu bệnh nhân bị tăng huyết áp

- heart\_disease: 0 nếu bệnh nhân không mắc bất kỳ bệnh tim nào, 1 nếu bệnh nhân có bệnh tim

- ever\_married : đã từng kết hôn “Không” hoặc “Có”

- work\_type: “children”, “Govt\_jov”, “Never\_worked”, “Tư nhân” hoặc “Tự kinh doanh”

- Residence\_type: “Nông thôn” hoặc “Thành thị”

- avg\_glucose\_level: mức đường trung bình trong máu

- BMI: chỉ số khối cơ thể

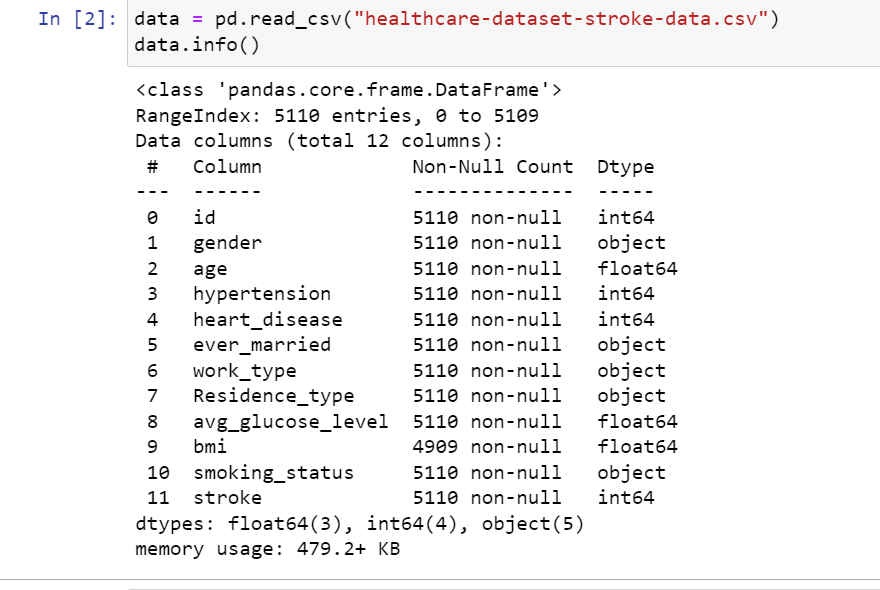
- smoking\_status: “trước đây đã hút thuốc”, “chưa bao giờ hút thuốc”, “hút thuốc”hoặc “Không xác định” (“Không xác định” trong smoking\_status có nghĩa là thông tin không có sẵn cho bệnh nhân này)

- stroke: 1 nếu bệnh nhân bị đột quỵ hoặc 0 nếu không

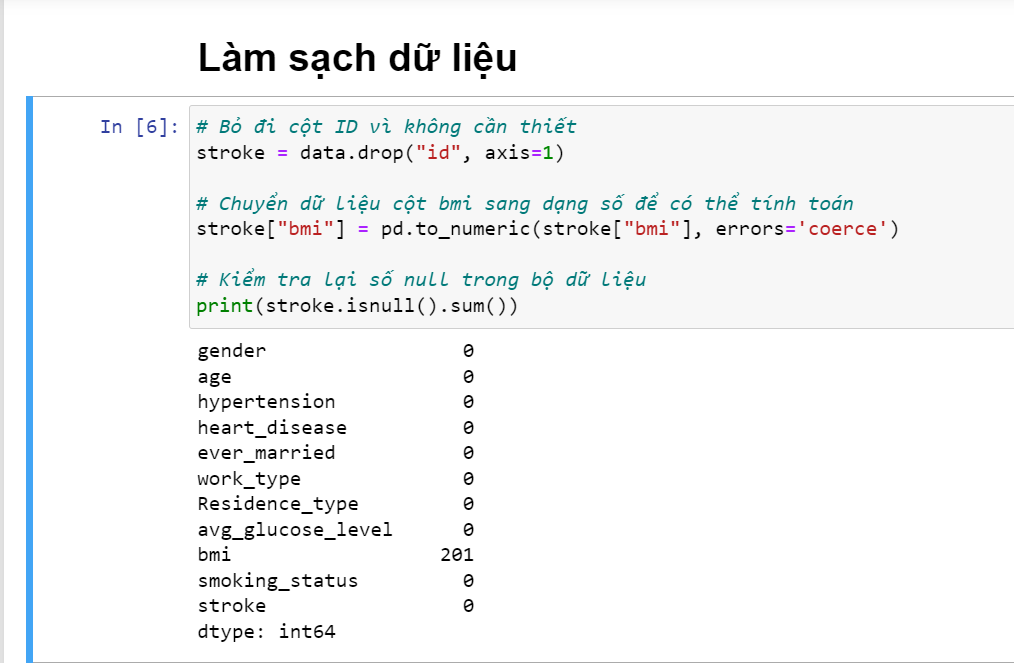
# **Chương 2: Tiền xử lý, làm sạch, trực quan hóa dữ liệu:**

* 1. **Các bước tiền xử lý và làm sạch dữ liệu:**

**Tiến hành đọc file .csv**



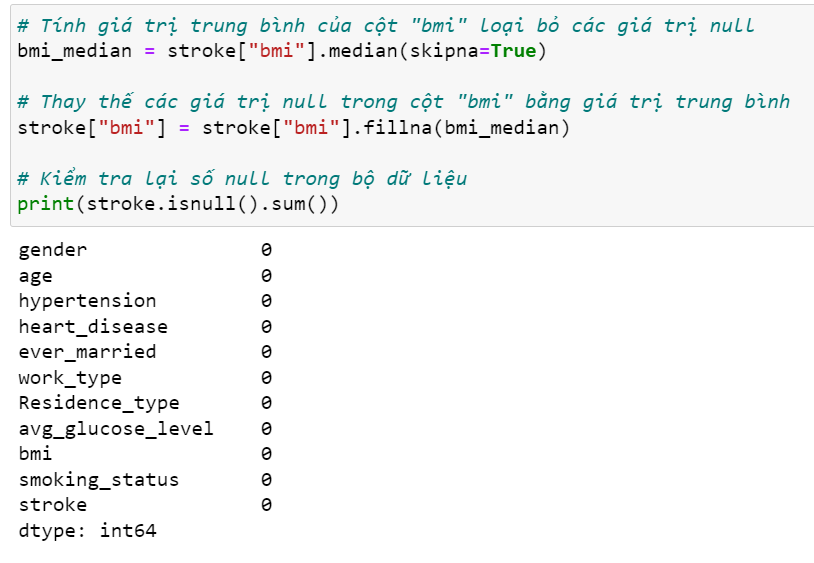
**Tiến hành làm sạch dữ liệu**



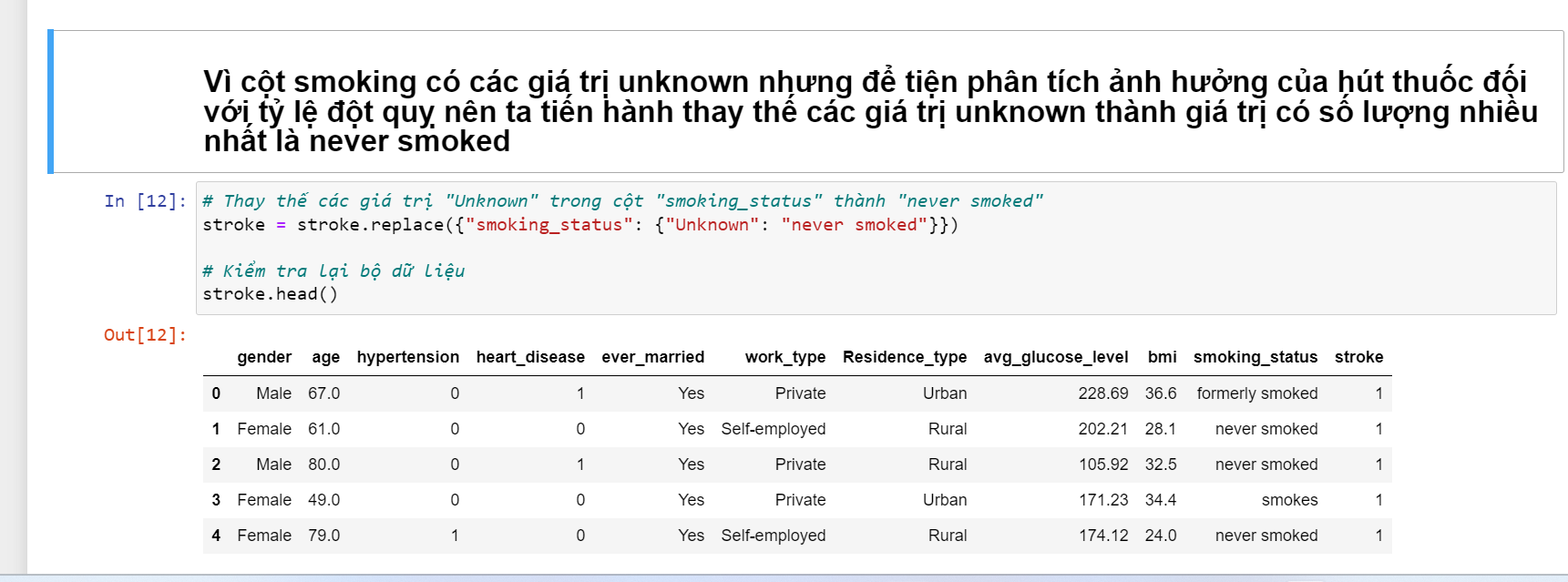
**Xử lý outlier**



**Thay thế giá trị null của cột bmi bằng giá trị trung bình của các giá trị thực của cột**



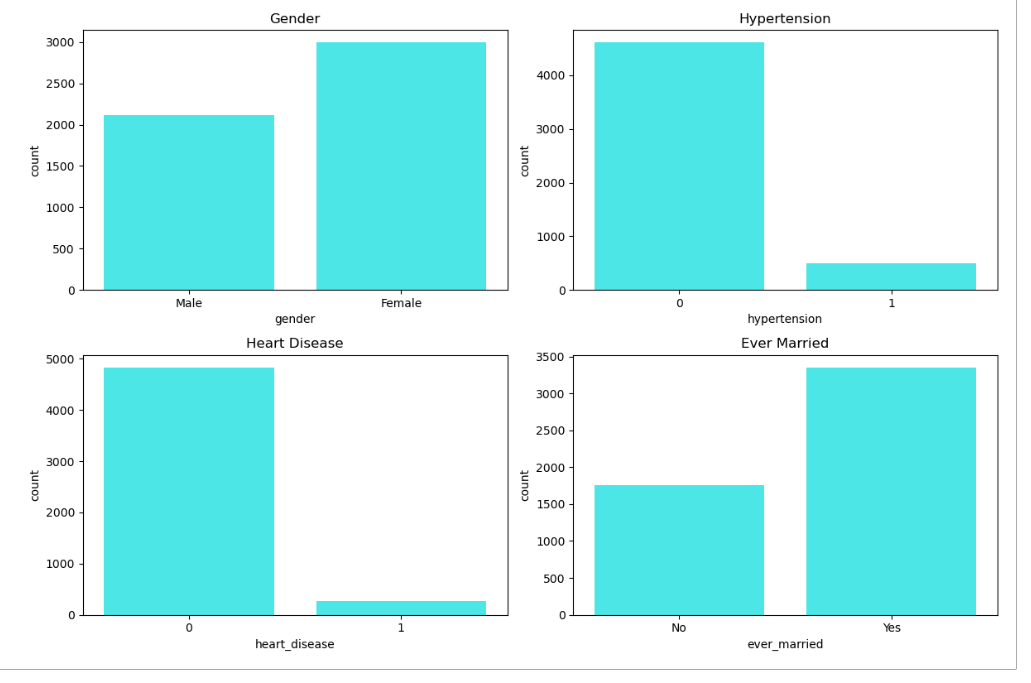
**Vì cột smoking có các giá trị unknown nhưng để tiện phân tích ảnh hưởng của hút thuốc đối với tỷ lệ đột quỵ nên ta tiến hành thay thế các giá trị unknown thành giá trị có số lượng nhiều nhất là never smoked**



* 1. **Trực quan hóa dữ liệu (EDA)**

Tạo ra 4 biểu đồ cột trong một grid 2x2 để visualize các biến phân loại trong tập dữ liệu "stroke" với một màu chung cho tất cả các biểu đồ.

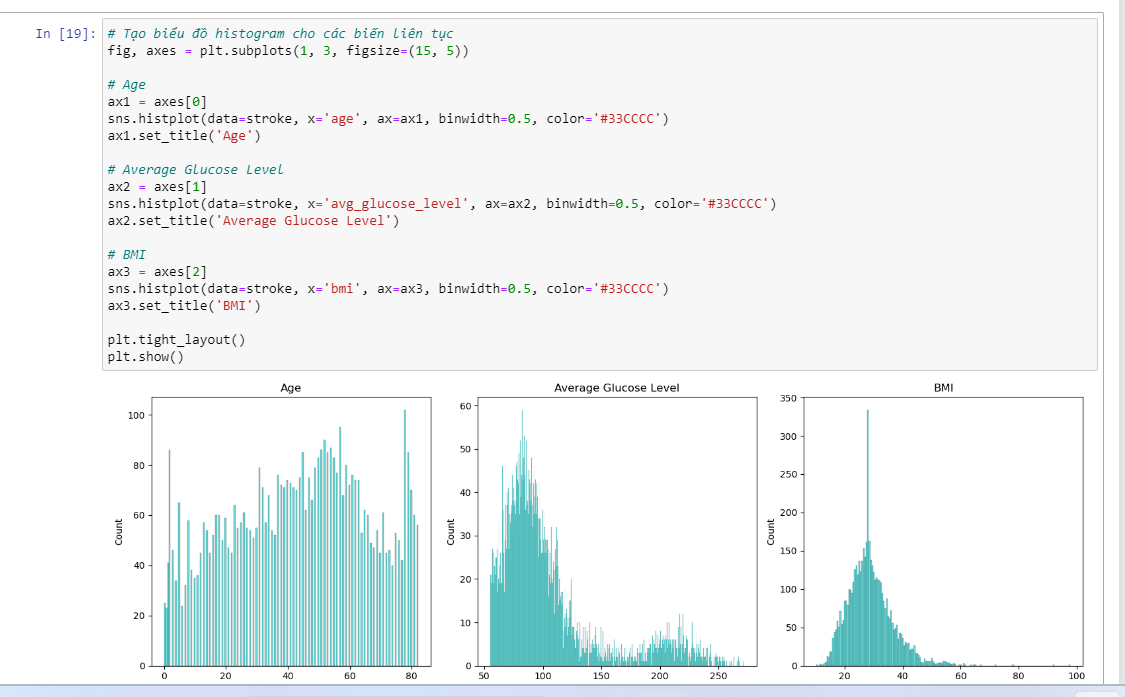




**Dựa vào biểu dồ trên, có thể thấy phần lớn trường hợp bệnh nhân không mắc bệnh tim và huyết áp, người đã kết hôn và nữ giới chiếm đa số trong tập dữ liệu.**

Tạo ra 3 biểu đồ histogram trong một grid 1x3 để visualize các biến liên tục như "age", "avg\_glucose\_level", và "bmi" trong tập dữ liệu "stroke". Tất cả các biểu đồ sử dụng cùng một màu xanh lam nhạt (#33CCCC).

Tạo biểu đồ histogram cho các biến liên tục



**Nhận xét: Tuổi tác có xu hướng lệch phải nhẹ nhưng vẫn phân bố khá đều hơn lượng đường và chỉ số bmi đang có xu hướng lệch trái rõ rệt**

**Tiếp theo ta sẽ kiểm tra mối quan hệ giữa các biến đầu vào và biến kết quả của chúng thông qua boxplot và từ đó đưa ra nhận xét “CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN TỶ LỆ ĐỘT QUỴ VÀ ẢNH HƯỞNG CỦA CHÚNG”.**

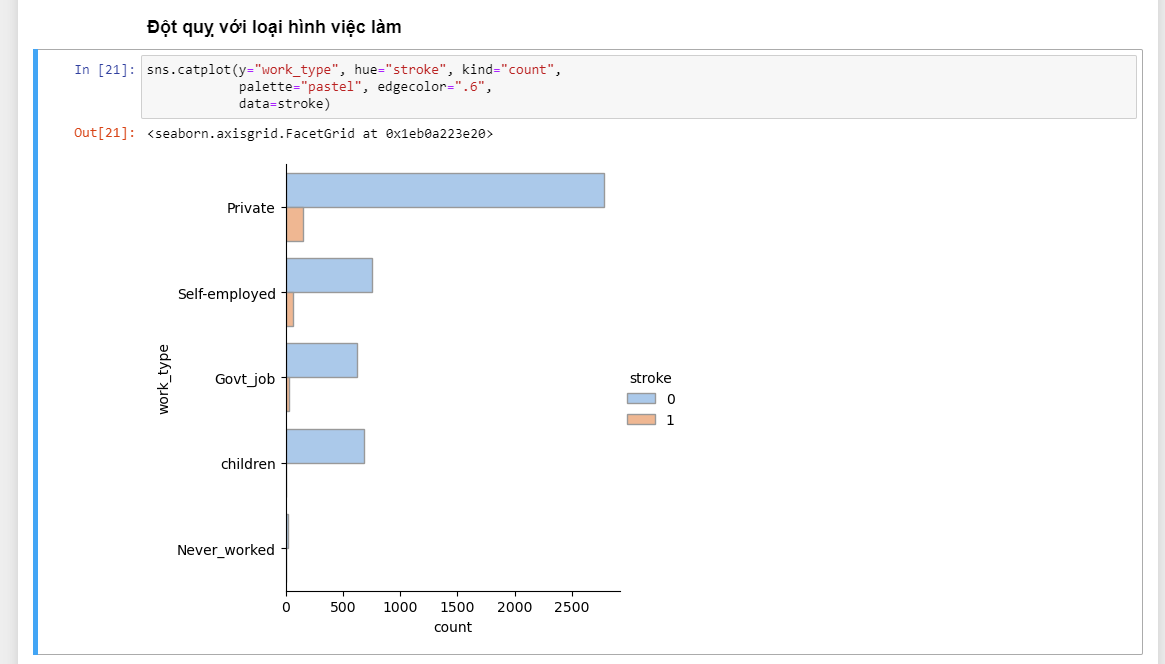
Vẽ biểu đồ hộp cho các biến liên tục và biến lưỡng giá

**Đột quỵ với tuổi tác**



* **Từ những biểu đồ trên, có thể kết luận được những người bị đột quỵ có độ tuổi, lượng đường và chỉ số cơ thể cao hơn so với người không mắc bệnh.**

Đột quỵ với loại hình việc làm



**Xét về tỷ lệ, tư nhân và tự kinh doanhn đều có số lượng người bị đột quỵ cao. Tuy nhiên, những người từ chính phủ có khả năng ít bị đột quỵ hơn 2 nhóm trên và trẻ em cũng không có nhiều khả năng bị đột quỵ. Có lẽ điều đó có thể được giải thích do mức độ áp lực của người lao động**

# **Chương 3: Áp dụng các mô hình, kỹ thuật, thuật toán vào việc dự đoán nguy cơ bị đột quỵ**

## **3.1. Mô hình Support Vector Machine (SVM)**

**Giới thiệu về mô hình được sử dụng**

SVM là một thuật toán học máy được sử dụng phổ biến cho các bài toán phân loại và hồi quy. Nó hoạt động bằng cách tìm ra siêu phẳng (hyperplane) tối ưu nhất để phân chia các lớp dữ liệu.

SVM hoạt động bằng cách tìm ra siêu phẳng tối ưu nhất để phân chia các lớp dữ liệu, sao cho độ rộng của vùng biên (margin) là lớn nhất. Các vector hỗ trợ (support vectors) chính là các điểm dữ liệu nằm trên biên này. Mô hình SVM có thể được sử dụng cho các bài toán phân loại nhị phân, đa nhãn, hồi quy và một số ứng dụng khác như phát hiện bất thường

**Phân tích ưu và nhược điểm của mô hình**

Về ưu điểm, SVM hoạt động hiệu quả trong không gian đặc trưng có nhiều chiều, cho phép xử lý dữ liệu phi tuyến tính một cách hiệu quả nhờ sử dụng các hàm kernel. Ngoài ra, SVM cũng có thể đạt độ chính xác cao ngay cả với kích thước tập dữ liệu huấn luyện nhỏ, và ít bị ảnh hưởng bởi các điểm dữ liệu ngoại lai. Những ưu điểm này khiến SVM trở thành một lựa chọn hấp dẫn trong nhiều bài toán phân loại.

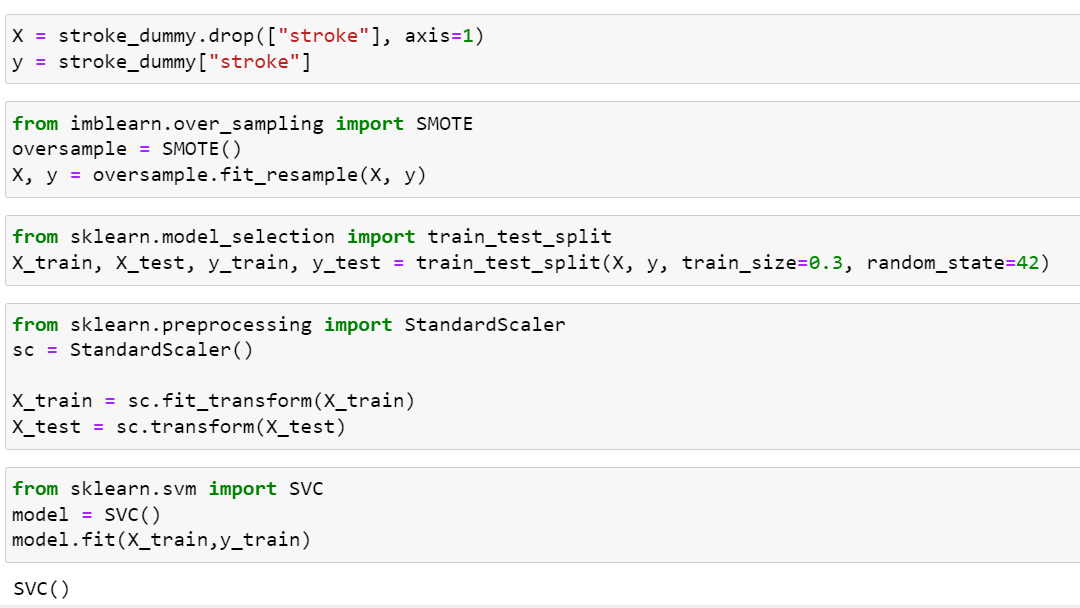
Tuy nhiên, SVM cũng có một số hạn chế đáng lưu ý. Khi kích thước tập dữ liệu trở nên rất lớn, SVM trở nên kém hiệu quả về mặt tính toán và bộ nhớ. Ngoài ra, SVM là một mô hình khá phức tạp, đòi hỏi nhiều hiểu biết về toán học và lý thuyết học máy để có thể hiểu và sử dụng hiệu quả. Hơn nữa, SVM không trực tiếp cung cấp xác suất của các lớp dự đoán như một số mô hình khác. Cuối cùng, việc chọn hàm kernel và tham số phù hợp cho SVM cũng là một thách thức, ảnh hưởng đáng kể đến hiệu suất của mô hình.

**Lý do lựa chọn mô hình**

Khi áp dụng mô hình Support Vector Machine (SVM) để dự đoán các trường hợp mắc bệnh đột quỵ, có một số lý do chính khiến SVM trở thành một lựa chọn hợp lý. Đầu tiên, SVM có khả năng xử lý dữ liệu phi tuyến tính, điều này rất phù hợp với bệnh đột quỵ, vì bệnh thường phụ thuộc vào nhiều yếu tố phức tạp và có mối quan hệ phi tuyến với các biến dự báo. Thứ hai, SVM hoạt động hiệu quả ngay cả với kích thước tập dữ liệu huấn luyện nhỏ, điều này rất phù hợp khi việc thu thập dữ liệu y tế, đặc biệt là các trường hợp mắc bệnh đột quỵ, thường gặp khó khăn. Thứ ba, SVM tương đối chịu đựng được các điểm dữ liệu ngoại lai, giúp tăng độ chính xác của mô hình, vì trong dữ liệu y tế thường có những trường hợp bất thường.

Cuối cùng, mặc dù SVM được coi là "hộp đen", nhưng các giá trị trọng số và vector hỗ trợ của mô hình có thể được sử dụng để hiểu và giải thích các yếu tố quan trọng trong dự đoán bệnh đột quỵ, đồng thời SVM cũng có thể được triển khai và tích hợp vào các hệ thống y tế một cách tương đối dễ dàng.

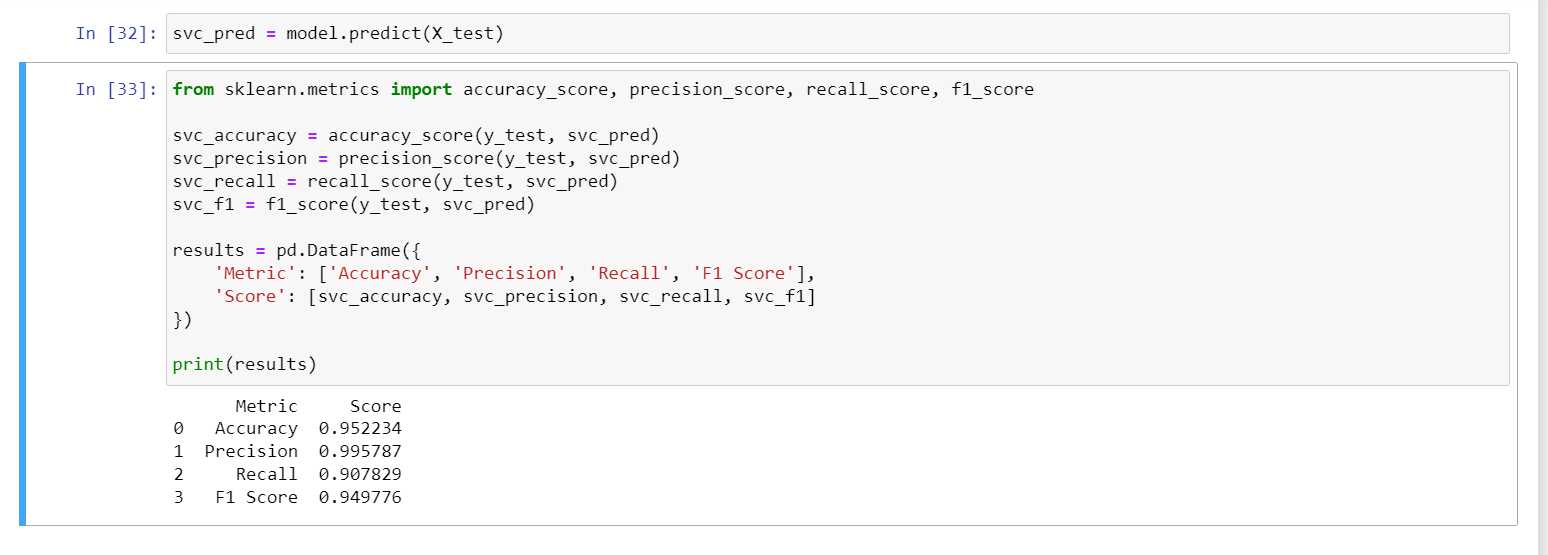
**Giải thích ý nghĩa của kết quả mô hình**



Sử dụng kỹ thuật over-sampling để cân bằng dữ liệu không cân bằng bằng cách sử dụng SMOTE (Synthetic Minority Over-sampling Technique) từ thư viện imblearn.over\_sampling.

Chuẩn hóa dữ liệu (data normalization) là một bước quan trọng trước khi huấn luyện mô hình, đặc biệt khi các feature có phạm vi giá trị khác nhau. StandardScaler sẽ chuẩn hóa các feature về cùng một phạm vi, thường là trung bình 0 và độ lệch chuẩn 1.

Sau khi chuẩn hóa dữ liệu, sử dụng SVC (Support Vector Classifier) để xây dựng mô hình phân loại. SVC là một mô hình học máy phổ biến, được sử dụng rộng rãi trong nhiều ứng dụng phân loại.



**Tạo một dataframe pandas với các chỉ số hiệu suất và in ra kết quả.**

**Đánh giá hiệu suất:**

1. Accuracy (Độ chính xác): 0.952234

- Điều này có nghĩa là mô hình SVC dự đoán chính xác 95.22% các mẫu trong tập dữ liệu kiểm tra.

- Độ chính xác cao là một dấu hiệu tích cực, cho thấy mô hình có khả năng dự đoán tốt.

2. Precision (Độ chính xác): 0.995787

- Độ chính xác này cho biết trong số các dự đoán dương tính, 99.58% là chính xác.

- Điều này chỉ ra rằng mô hình SVC có rất ít dự đoán sai lầm là dương tính.

- Độ chính xác cao là một dấu hiệu tích cực, cho thấy mô hình có khả năng phân loại chính xác các mẫu dương tính.

3. Recall (Độ nhạy): 0.907829

- Độ nhạy này cho biết mô hình SVC phát hiện chính xác 90.78% các mẫu dương tính.

- Điều này cho thấy mô hình có khả năng phát hiện hầu hết các mẫu dương tính, mặc dù không phải 100%.

- Độ nhạy ở mức 90% trở lên là một kết quả tốt.

4. F1-score (Điểm F1): 0.949776

- F1-score là trung bình điều hòa của precision và recall, được coi là một chỉ số tổng hợp hiệu suất của mô hình.

- Điểm F1 ở mức 0.949776 cho thấy mô hình SVC có hiệu suất tổng thể rất tốt, khi cân bằng giữa độ chính xác và độ nhạy.

**Nhận xét:**

Nhìn chung, kết quả đánh giá hiệu suất của mô hình SVC là rất tốt, với độ chính xác, độ chính xác, độ nhạy và điểm F1 đều ở mức cao. Điều này cho thấy mô hình SVC đã được huấn luyện tốt và có khả năng dự đoán chính xác trên tập dữ liệu kiểm tra.

## **3.2. Mô hình Random Forest**

**Giới thiệu về mô hình được sử dụng**

Random Forest là một phương pháp học máy thuộc nhóm ensemble learning, được phát triển từ cây quyết định (Decision Tree). Mô hình Random Forest tạo ra một "rừng" gồm nhiều cây quyết định và kết hợp kết quả của chúng để đưa ra dự đoán cuối cùng. Quá trình này giúp tăng cường độ chính xác và giảm thiểu khả năng quá khớp (overfitting) so với việc sử dụng một cây quyết định đơn lẻ.

**Phân tích ưu và nhược điểm của mô hình**

Ưu điểm của việc sử dụng mô hình: Random Forest thường có độ chính xác cao hơn so với các mô hình đơn giản khác, nhờ vào việc kết hợp nhiều cây quyết định. Bên cạnh đó, sự kết hợp của nhiều cây giúp giảm thiểu khả năng quá khớp, vì mỗi cây chỉ được xây dựng từ một phần của tập dữ liệu và tập hợp các đặc trưng khác nhau. Không những thế Random Forest có khả năng làm việc tốt với các tập dữ liệu có giá trị thiếu mà không cần phải loại bỏ hoặc ước lượng các giá trị này. Cuối cùng chính là tính linh hoạt, Random Forest có thể được sử dụng cho cả bài toán phân loại và hồi quy.

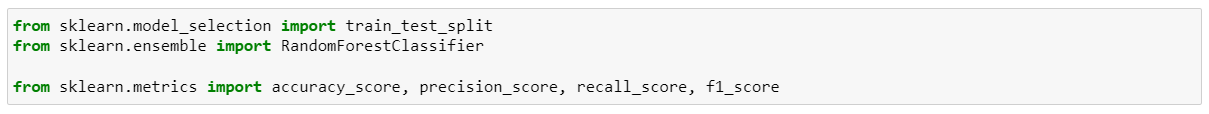
Nhược điểm của việc sử dụng mô hình: Việc xây dựng nhiều cây quyết định và tổng hợp kết quả có thể tốn thời gian và tài nguyên tính toán, đặc biệt với các tập dữ liệu lớn. Mặc dù Random Forest có thể cung cấp tầm quan trọng của các đặc trưng, nhưng việc giải thích từng cây quyết định và quá trình tổng hợp kết quả cuối cùng vẫn khá phức tạp. Khả năng bị bias với dữ liệu lệch nghĩa là nếu dữ liệu huấn luyện không cân bằng, mô hình có thể bị thiên vị về phía các lớp có số lượng mẫu lớn hơn.

**Lý do lựa chọn mô hình**

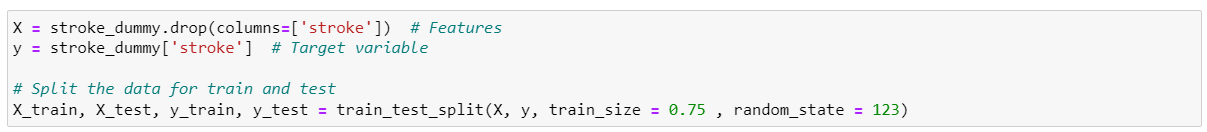
Random Forest thường cho kết quả chính xác và ổn định, phù hợp với loại bài toán dự đoán dựa trên dữ liệu phân loại khác nhau. Không những thế khả năng xử lý đa dạng dữ liệu của mô hình này có thể làm việc tốt với các dữ liệu có giá trị thiếu, không cần phải xử lý trước quá nhiều. Khả năng đánh giá tầm quan trọng của các đặc trưng giúp hiểu rõ hơn về dữ liệu và hỗ trợ trong việc tối ưu hóa mô hình. Và cuối cùng là giảm thiểu overfitting với cơ chế kết hợp nhiều cây quyết định, Random Forest giúp giảm thiểu khả năng quá khớp, điều này đặc biệt quan trọng với các tập dữ liệu phức tạp.

**Giải thích ý nghĩa của kết quả mô hình**

Import những thư viện cần thiết.



Chọn các nhãn đầu vào là X bao gồm toàn bộ cột trong dataframe loại trừ cột ‘stroke’ và biến mục tiêu y là cột ‘stroke’ để dự đoán xem người đó có bị đột quỵ không. Chia tập dữ liệu với tập train là 75% và test là 25%.



Sau đó, kiểm tra thử số lượng dữ liệu của y\_test sau khi chia thành 2 tập train, test và bên cạnh đó kiểm tra lại những cột là nhãn đầu vào của X\_train đã đủ và đúng chưa.

A screenshot of a computer

Description automatically generated

Tạo và huấn luyện một mô hình phân loại Random Forest với dữ liệu đầu vào và nhãn đã cho với bộ tham số ngẫu nhiên của mô hình là 29.

A white rectangular object with a white border

Description automatically generated

Tạo một biến mới là y\_pred để dự đoán nhãn cho tập dữ liệu kiểm tra sau khi mô hình Random Forest đã được huấn luyện và đánh giá hiệu quả mô hình thông qua Accuracy score, Precision score, Recall score và F1 score dựa vào y\_test và y\_pred.

A screenshot of a computer

Description automatically generated

Đánh giá hiệu quả của mô hình:

* + **Accuracy** (0.952269): Mô hình dự đoán đúng 95.23% các mẫu. Đây là chỉ số tốt khi dữ liệu không bị lệch.
  + **Precision** (0.285714): Trong số các mẫu được dự đoán là dương tính, chỉ có 28.57% là chính xác.
  + **Recall** (0.034483): Chỉ có 3.45% các mẫu thực sự dương tính được dự đoán đúng.
  + **F1 Score** (0.061538): Sự cân bằng giữa Precision và Recall cho thấy mô hình không hiệu quả trong việc dự đoán các mẫu dương tính.

Nhận xét:

Các chỉ số Precision, Recall và F1 Score đều khá thấp, điều này có thể chỉ ra rằng mô hình đang gặp khó khăn trong việc dự đoán chính xác lớp dương tính. Điều này có thể xảy ra trong các tình huống dữ liệu bị lệch (rất ít mẫu dương tính so với mẫu âm tính). Trong trường hợp này, có thể cần xem xét các kỹ thuật như tái cân bằng dữ liệu (resampling), điều chỉnh ngưỡng dự đoán, hoặc sử dụng các thuật toán khác phù hợp hơn với dữ liệu lệch.

## **3.3. Mô hình K-Nearest Neighbors**

**Phân tích về áp dụng mô hình**

**Giới thiệu về mô hình được sử dụng**

K-Nearest Neighbors (KNN) là một trong những thuật toán học máy đơn giản và phổ biến nhất, được sử dụng chủ yếu cho các bài toán phân loại và hồi quy. KNN là một mô hình học không tham số, nghĩa là nó không giả định bất kỳ hình thức cụ thể nào cho hàm số ánh xạ từ dữ liệu đầu vào sang đầu ra. Nguyên tắc hoạt động của KNN rất đơn giản: dựa trên khoảng cách giữa các điểm dữ liệu trong không gian đặc trưng, mô hình sẽ dự đoán nhãn của một điểm mới bằng cách xem xét "K" điểm lân cận gần nhất của nó.

**Phân tích ưu và nhược điểm của mô hình**

Ưu điểm của việc sử dụng mô hình: Mô hình KNN đơn giản và dễ hiểu, rất dễ triển khai và trực quan, không yêu cầu nhiều về kiến thức toán học phức tạp. Vì KNN không xây dựng mô hình, nó không cần giai đoạn đào tạo, do đó tiết kiệm được thời gian và tài nguyên. Hiệu quả với các bộ dữ liệu nhỏ nghĩa là KNN hoạt động tốt với các bộ dữ liệu nhỏ, nơi chi phí tính toán khoảng cách không phải là vấn đề lớn. Cuối cùng không yêu cầu giả định về dữ liệu, vì không giả định hình thức của hàm số, KNN có thể áp dụng cho nhiều loại dữ liệu khác nhau.

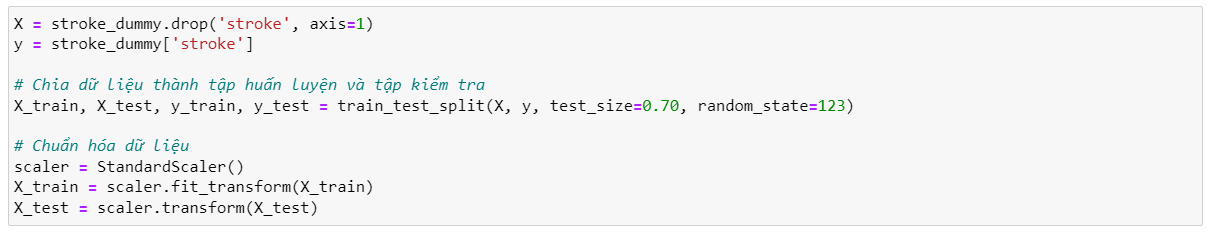
Nhược điểm của việc sử dụng mô hình: Chi phí tính toán cao nghĩa là khi kích thước dữ liệu tăng, chi phí tính toán khoảng cách giữa các điểm trở nên rất lớn, gây ra sự chậm trễ và tiêu tốn tài nguyên. Nhạy cảm với nhiễu và dữ liệu không cân bằng có thể bị ảnh hưởng lớn bởi các điểm dữ liệu nhiễu và các lớp có kích thước không cân bằng, dẫn đến dự đoán không chính xác. Khó khăn trong việc lựa chọn K. Giá trị của K cần được lựa chọn cẩn thận, nếu không kết quả sẽ không đáng tin cậy nghĩa là K quá nhỏ có thể dẫn đến nhiễu, K quá lớn có thể làm mất chi tiết. Và cuối cùng không mở rộng tốt, đối với các bài toán với dữ liệu lớn hoặc có nhiều đặc trưng, KNN không phải là lựa chọn tốt nhất vì nó đòi hỏi nhiều bộ nhớ và tài nguyên tính toán.

**Lý do lựa chọn mô hình**

KNN là một mô hình lý tưởng cho người mới bắt đầu học về học máy, cũng như cho các ứng dụng mà việc giải thích mô hình là quan trọng. Hiệu quả với dữ liệu nhỏ và không yêu cầu quá nhiều tài nguyên, khi bộ dữ liệu không quá lớn, KNN có thể được áp dụng một cách hiệu quả mà không cần nhiều tài nguyên tính toán. Linh hoạt với nhiều loại dữ liệu khác nhau, có thể được áp dụng cho cả bài toán phân loại và hồi quy, và có thể xử lý các dạng dữ liệu khác nhau mà không cần giả định trước về phân phối dữ liệu. Không yêu cầu giả định về hàm số, điều này làm cho KNN phù hợp với các bài toán mà hình dạng của hàm số ánh xạ từ đầu vào sang đầu ra không rõ ràng hoặc phức tạp.

**Giải thích ý nghĩa của kết quả mô hình**

Tương tự với các bước ở Random Forest, đầu tiên xác định nhãn đầu vào và biến mục tiêu của mô hình. Sau đó chia tập dữ liệu thành 2 tập train, test với train 70% và test 30%. Và cuối cùng là chuẩn hóa dữ liệu của nhãn đầu vào X bằng StandardScaler.



Tạo và huấn luyện một mô hình phân loại KNN với dữ liệu đầu vào và nhãn đã cho. Sau đó, tạo một biến y\_pred để dự đoán nhãn cho tập dữ liệu kiểm tra sau khi mô hình KNN đã được huấn luyện và đánh giá hiệu quả mô hình thông qua Accuracy score.

A white rectangular object with a black border

Description automatically generated

Vi đây là accuracy score của mô hình KNN chưa được xác định tham số n\_neighbors nên không thể khẳng định được là nó có thật sự là chính xác cao nhất hay chưa nên ta gán cho n\_neighbors từ 1 đến 10 vào mô hình để dự đoán và tìm ra với n là bao nhiêu thì accuracy score cao nhất.

A computer screen shot of a computer code

Description automatically generated

Với n\_neighbors = 5 thì accuracy score đạt mức cao nhất nên ta chọn rồi áp dụng vào KNN để tính các thông số đánh giá còn lại như: Precision score, Recall score và F1 score dựa vào y\_test và y\_pred.

A screenshot of a computer

Description automatically generated

Đánh giá hiệu quả mô hình:

* **Accuracy** (0.950517): Cao cho thấy mô hình hoạt động tốt đối với các lớp chính, nhưng điều này có thể bị gây hiểu lầm do sự không cân bằng trong dữ liệu.
* **Precision** (0.636364): tương đối cao hơn so với Recall, cho thấy mô hình chính xác hơn trong các dự đoán dương tính, nhưng số lượng dự đoán dương tính rất ít.
* **Recall** (0.038889): cực kỳ thấp, cho thấy mô hình không tìm ra được nhiều trường hợp dương tính, dẫn đến việc bỏ sót nhiều trường hợp quan trọng.
* **F1 Score** (0.073298): thấp phản ánh việc mô hình không cân bằng tốt giữa Precision và Recall.

Nhận xét:

Kết quả đánh giá cho thấy mô hình KNN có vẻ không phù hợp với dữ liệu này, đặc biệt là với những bài toán cần phải phát hiện các trường hợp dương tính chính xác và đầy đủ. Điều này có thể do dữ liệu không cân bằng hoặc do mô hình KNN chưa được tối ưu hóa đúng cách. Có thể cần thử nghiệm với các mô hình khác, kỹ thuật xử lý dữ liệu hoặc điều chỉnh tham số K để cải thiện hiệu suất.

## **3.4. Mô hình Decision Tree**

**Giới thiệu về kĩ thuật khai phá**

Cây quyết định, cụ thể là thuật toán CART (Cây phân loại và hồi quy), là một kỹ thuật khai thác dữ liệu phổ biến được sử dụng cho các nhiệm vụ phân loại và hồi quy. Nó liên quan đến việc tạo ra một mô hình quyết định dạng cây dựa trên giá trị của các đặc tính đầu vào, được sử dụng để dự đoán giá trị của biến mục tiêu

**Các khái niệm và thuật ngữ trong thuật toán cây quyết định Decision Tree:**

**Root Node (Nút gốc)**: Là nút bắt đầu của cây quyết định, từ nút này bắt đầu phân chia theo các đặc tính khác nhau.

**Parent Node (Nút cha)**: Là nút mà có các nút con bên dưới.

**Child Node (Nút con)**: Là nút mà có nút cha.

**Decision Nodes (Nút quyết định):**Là các nút mà chúng ta nhận được sau khi chia tách nút.

**Leaf Nodes (Nút lá):** Là các nút mà không thể chia tách thêm được, nút này còn gọi là nút cuối.

**Sub-tree (Cây con):** Là một phần nhỏ của đồ thị, nó tương tự như một phần con của cây quyết định Decision Tree.

**Pruning (Cắt tỉa)**: Không có gì thay đổi hay ảnh hưởng khi chúng ta cắt giảm một số nút để không bị overfitting.

**Gini index** :Chỉ số Gini dùng để xác định mức độ phân tách của cây quyết định.Nó là một chỉ số dùng để đo và đánh giá việc phân chia ở node điều kiện có tốt hay không.Nó dùng để tính độ lệch Gini của nốt cha với tổng các giá trị Gini có đánh trọng số của các node con.

Công thức tính chỉ số Gini :

A black and white math equation

Description automatically generated

Trong đó:

1. **𝐶**: Đây là số lượng lớp hoặc hạng mục (classes) trong tập dữ liệu. Ví dụ, trong một bài toán phân loại nhị phân, 𝐶=2, còn nếu có ba lớp thì 𝐶=3.
2. **𝑝𝑖​**: Đây là xác suất của một mẫu thuộc về lớp i. Nó được tính bằng cách lấy số lượng mẫu thuộc lớp i chia cho tổng số lượng mẫu.

**Ưu và nhược điểm**

**Ưu điểm:**

* **Dễ hiểu và diễn giải:** Cây quyết định rất đơn giản để hiểu và giải thích. Cấu trúc cây giúp dễ dàng hình dung quá trình ra quyết định.
* **Ít yêu cầu tiền xử lý dữ liệu:** Chúng yêu cầu xử lý trước dữ liệu ở mức tối thiểu, chẳng hạn như chia tỷ lệ hoặc chuẩn hóa.
* **Có khả năng xử lý cả dữ liệu số và dữ liệu phân loại:** Cây quyết định có thể xử lý cả dữ liệu số và dữ liệu phân loại..
* **Không cần giả định về mối quan hệ giữa các biến:** Chúng không giả định bất kỳ mối quan hệ nào trước đó giữa các đặc điểm và biến mục tiêu..

**Nhược điểm:**

* **Dễ bị overfitting:** Cây quyết định có thể dễ dàng bị quá khớp, đặc biệt khi chúng ở sâu và có nhiều nhánh.
* **Không ổn định:** Những thay đổi nhỏ trong dữ liệu có thể dẫn đến một cây hoàn toàn khác.
* **Không hiệu quả với dữ liệu lớn:** Chúng có thể không hoạt động tốt với các tập dữ liệu rất lớn vì chúng có thể trở nên đắt đỏ về mặt tính toán.

## **3.5. Mạng Nơ-ron Nhân Tạo (ANN) với PyTorch**

**Giới thiệu kĩ thuật khai phá**

Mạng Nơ-ron Nhân Tạo (ANN) là một mô hình học sâu (deep learning) được thiết kế để mô phỏng cách hoạt động của não người. ANN bao gồm nhiều lớp neuron liên kết với nhau, nơi mỗi lớp thực hiện các phép tính để dự đoán hoặc phân loại dữ liệu.

Pytorch chính là một framework hỗ trợDeep learningđược phát triển bởi Facebook.

Phát triển Pytorch với mã nguồn mở Facebook đã tạo được một cộng đồng chia sẻ rất lớn. Với nguồn tài nguyên rộng lớn trên cộng đồng này các vấn đề mà bạn đang mắc phải có thể đã có ai đó giải quyết và được chia sẻ lên cộng đồng, hãy dành một chút thời gian để tìm kiếm.

**Ưu điểm và nhược điểm**

**Ưu Điểm**

* Khả năng học từ các dữ liệu phức tạp và phi tuyến tính.
* Tính linh hoạt cao trong việc điều chỉnh cấu trúc mô hình (số lớp, số neuron).
* PyTorch cung cấp công cụ mạnh mẽ cho việc xây dựng và huấn luyện mô hình.

**Nhược Điểm**

* Đòi hỏi thời gian và tài nguyên tính toán lớn để huấn luyện.
* Dễ bị quá khớp (overfitting) nếu không có kỹ thuật điều chỉnh tốt.
* Khó giải thích kết quả.

**Lý do chọn kỹ thuật**

Chúng ta lựa chọn sử dụng ANN với PyTorch và CART vì mỗi kỹ thuật có những điểm mạnh riêng biệt phù hợp với yêu cầu của dự án. ANN với PyTorch được lựa chọn vì khả năng học từ dữ liệu phức tạp và khả năng mở rộng tốt, trong khi CART được lựa chọn vì tính dễ hiểu và dễ giải thích kết quả.

**Giải thích ý nghĩa của việc Khai phá**

**Kết quả từ Pytorch**

Ta sẽ đi tạo 1 một lớp mô hình học sâu (neural network) tên là **TabularModel** sử dụng PyTorch để xử lý dữ liệu bảng. Đây là một mô hình kết hợp giữa các đặc trưng phân loại (categorical features) và liên tục (continuous features).

A computer screen shot of a program

Description automatically generated

Ta tiếp tục đặt hạt ngẫu nhiên (seed) để đảm bảo tính tái lập của kết quả. Điều này có nghĩa là mỗi khi bạn chạy mã này, bạn sẽ nhận được cùng một kết quả.

Khởi tạo mô hình với tham số:

* **Tham Số:**
  + **emb\_szs**: Danh sách các kích thước embedding cho các đặc trưng phân loại.
  + **conts.shape[1]**: Số lượng các đặc trưng liên tục.
  + **2**: Kích thước đầu ra (số lớp output).
  + **[400, 200, 100]**: Danh sách các lớp ẩn với số lượng đơn vị tương ứng.
  + **p=0.2**: Tỷ lệ dropout.

A screen shot of a computer

Description automatically generated

Ta xác định hàm mất và tối ưu hóa

**Hàm mất mát CrossEntropyLoss:**

* Đây là hàm mất mát được sử dụng phổ biến cho các bài toán phân loại đa lớp (multi-class classification).
* Nó kết hợp giữa hàm softmax và hàm mất mát entropy chéo, giúp đánh giá sự khác biệt giữa dự đoán của mô hình và nhãn thực tế.
* Hàm này yêu cầu đầu vào là các logit (chưa qua hàm softmax) từ lớp cuối cùng của mô hình.

**Bộ tối ưu hóa Adam:**

* Adam (Adaptive Moment Estimation) là một trong những bộ tối ưu hóa phổ biến trong học sâu nhờ tính hiệu quả và khả năng hội tụ nhanh.
* model.parameters(): Chỉ định các tham số của mô hình cần được tối ưu hóa.
* lr=0.001: Thiết lập tốc độ học (learning rate), đây là một siêu tham số điều chỉnh bước cập nhật của các tham số mô hình trong mỗi lần lặp.

A screen shot of a computer

Description automatically generated

Tiến hành chia tập Train/Test

Tập Huấn Luyện (Training Set): Gồm các dữ liệu và nhãn từ đầu đến batch\_size **-** test\_size (tức là 9000 - 492 = 8508 mẫu). Tập này được sử dụng để huấn luyện mô hình.

Tập Kiểm Tra (Test Set): Gồm các dữ liệu và nhãn từ batch\_size - test\_size đến batch\_size (tức là từ 8508 đến 9000 mẫu, tổng cộng là 492 mẫu). Tập này được sử dụng để đánh giá hiệu suất của mô hình sau khi huấn luyện

A screen shot of a computer code

Description automatically generated

Training model

Chúng ta sẽ khởi tạo thời gian bắt đầu để đo thời gian huấn luyện. Với epochs : Số lần lặp là 320. “Losses” là Danh sách lưu trữ giá trị hàm mất mát sau mỗi epoch.

import time

start\_time = time.time()

epochs = 320

losses = []

for i in range(epochs):

    i+=1

    y\_pred = model(cat\_train, con\_train)

    loss = criterion(y\_pred, y\_train)

    losses.append(loss.detach().cpu().numpy())

    if i%25 == 1:

        print(f'epoch: {i:3}  loss: {loss.item():10.8f}')

    optimizer.zero\_grad()

    loss.backward()

    optimizer.step()

print(f'epoch: {i:3}  loss: {loss.item():10.8f}')

print(f'\nDuration: {time.time() - start\_time:.0f} seconds')

Ta vẽ đồ thị này thể hiện quá trình giảm giá trị hàm mất mát Cross Entropy trong suốt quá trình huấn luyện mô hình. Trục x là số epoch và trục y là giá trị của hàm mất mát Cross Entropy. Đây là một đồ thị điển hình được sử dụng để theo dõi sự hội tụ của mô hình học máy.

A graph with a red line

Description automatically generated

**Phân Tích Đồ Thị**

1. **Giai Đoạn Ban Đầu (0-50 Epochs):**
   * **Giảm Mạnh:** Trong giai đoạn này, giá trị hàm mất mát giảm rất nhanh. Điều này cho thấy mô hình đang học các đặc trưng quan trọng từ dữ liệu huấn luyện một cách hiệu quả.
   * **Tốc Độ Giảm Cao:** Đây là dấu hiệu tốt cho thấy mô hình có khả năng học nhanh trong giai đoạn đầu.
2. **Giai Đoạn Trung Gian (50-200 Epochs):**
   * **Giảm Đều:** Sau giai đoạn ban đầu, tốc độ giảm của hàm mất mát chậm lại và bắt đầu giảm đều. Điều này cho thấy mô hình đang tiếp tục học nhưng với tốc độ chậm hơn.
   * **Hội Tụ Dần:** Mô hình dần dần hội tụ, giá trị hàm mất mát giảm từ từ.
3. **Giai Đoạn Cuối (200-320 Epochs):**
   * **Giảm Chậm:** Trong giai đoạn này, giá trị hàm mất mát giảm rất chậm, tiệm cận giá trị tối thiểu. Điều này cho thấy mô hình đã học được hầu hết các đặc trưng quan trọng từ dữ liệu.
   * **Ổn Định:** Sự thay đổi của giá trị hàm mất mát trở nên nhỏ hơn, cho thấy mô hình đang ổn định và hội tụ tốt.

Ta đi tính toán giá trị hàm mất mát (Cross Entropy Loss) trên tập kiểm tra để đánh giá hiệu suất của mô hình trên dữ liệu. Với giá trị CE Loss 0.29529357 cho thấy mô hình hoạt động khá tốt trên tập kiểm tra. Tuy nhiên, để có đánh giá toàn diện hơn về hiệu suất của mô hình, cần kiểm tra thêm các chỉ số khác như độ chính xác và F1-score, và so sánh giá trị hàm mất mát trên tập huấn luyện và tập kiểm tra

A computer code on a black background

Description automatically generated

Ta tiến hành kiểm tra và hiển thị kết quả dự đoán của mô hình trên một số mẫu dữ liệu từ tập kiểm tra. Nó so sánh giá trị dự đoán với nhãn thực tế và tính toán tỷ lệ chính xác

A computer screen shot of a program code

Description automatically generated

A black background with white text

Description automatically generated

Kết quả cho thấy mô hình đã dự đoán đúng 180 trên tổng số 200 mẫu dữ liệu, với tỷ lệ chính xác là 90.00%.

Đánh giá mô hình qua F-1 score và độ chính xác accuracy

A screen shot of a computer code

Description automatically generated

**Kết Quả**

* **F1-score:** **0.9047619047619048**
  + F1-score là một chỉ số kết hợp giữa precision và recall, cung cấp một đánh giá tổng thể về hiệu suất của mô hình.
  + F1-score khoảng **0.90** cho thấy mô hình có hiệu suất rất tốt, đặc biệt khi độ chính xác và độ nhạy đều cao.
* **Accuracy:** **0.9**
  + Độ chính xác là tỷ lệ phần trăm của các dự đoán đúng trên tổng số mẫu.
  + Độ chính xác **0.9** (90%) cho thấy mô hình đã dự đoán đúng 90% số mẫu dữ liệu trong tập kiểm tra.

**Đánh Giá Hiệu Suất**

1. **F1-score:**
   * F1-score là một chỉ số quan trọng trong các bài toán phân loại, đặc biệt là khi có sự mất cân bằng giữa các lớp (class imbalance).
   * Giá trị F1-score 0.9047619047619048 cho thấy mô hình có khả năng dự đoán chính xác và đáng tin cậy.
2. **Accuracy:**
   * Độ chính xác là chỉ số dễ hiểu và phổ biến, cho thấy tỷ lệ phần trăm của các dự đoán đúng.
   * Độ chính xác **0.9** là một kết quả rất tốt, cho thấy mô hình hoạt động hiệu quả trên tập kiểm tra.

**2 Kết quả từ CART**

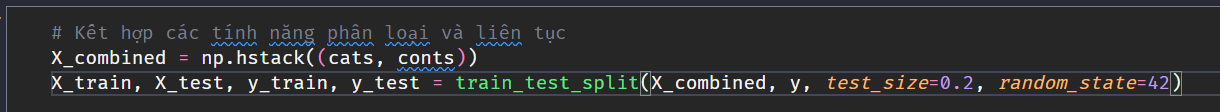
Import thư viện vào

A screen shot of a computer code

Description automatically generated

Ta kết hợp tính năng phân loại và liên tục qua Sử dụng hàm hstack của NumPy để kết hợp các đặc trưng phân loại và liên tục thành một mảng duy nhất theo chiều ngang (horizontal stack). Mỗi hàng của X\_combined sẽ chứa các giá trị của cả đặc trưng phân loại và liên tục cho một mẫu dữ liệu.

Ta chia dữ liệu 80% cho tập train và 20% còn lại cho tập test



Ta xác định

**Parameter Grid:** Việc xác định bảng tham số giúp tìm kiếm các giá trị tối ưu cho các siêu tham số của mô hình. Điều này giúp cải thiện hiệu suất và độ chính xác của mô hình.

**Decision Tree Classifier:** Tạo một mô hình cây quyết định với một hạt ngẫu nhiên cố định, sẵn sàng để tìm kiếm và huấn luyện với các tham số tối ưu.

A computer screen shot of a code

Description automatically generated

Ta tiếp tục đi tìm kiếm lưới (Grid Search) để tìm các tham số tối ưu cho mô hình cây quyết định (Decision Tree Classifier) bằng cách sử dụng GridSearchCV của scikit-learn

A screen shot of a computer

Description automatically generated

**Fitting 5 folds for each of 72 candidates, totalling 360 fits:** Thông tin chi tiết về quá trình tìm kiếm lưới. Có 72 tổ hợp tham số cần thử nghiệm và mỗi tổ hợp được đánh giá bằng cách sử dụng 5-fold cross-validation, tổng cộng có 360 lần huấn luyện mô hình.

**Best parameters found: {'criterion': 'gini', 'max\_depth': 10, 'min\_samples\_leaf': 2, 'min\_samples\_split': 2}:** Các tham số tốt nhất được tìm thấy:

* **criterion**: **'gini'**
* **max\_depth**: **10**
* **min\_samples\_leaf**: **2**
* **min\_samples\_split**: **2**

Sau khi tìm ra các tham số tốt nhất, bạn có thể sử dụng chúng để tạo và huấn luyện mô hình cuối cùng. Dưới đây là cách thực hiện:

A computer screen shot of a code

Description automatically generated

Đánh giá mô hình

A screen shot of a computer

Description automatically generated

Qua việc đánh giá, ta thấy được

* Độ chính xác của mô hình trên tập kiểm tra là **0.88** (88%). Điều này cho thấy mô hình dự đoán đúng 88% số mẫu trong tập kiểm tra.
* F1-score của mô hình là 0.89. Đây là một chỉ số quan trọng phản ánh sự cân bằng giữa precision và recall của mô hình.

**Nhận xét 2 mô hình:**

Mô hình Mạng Nơ-ron Nhân Tạo (ANN, Pytorch): Mô hình ANN đạt được kết quả ấn tượng với giá trị hàm mất mát Cross Entropy là 0.29529357. Độ chính xác và F1-score của mô hình này đều cao là 90%, cho thấy khả năng phân loại chính xác nguy cơ đột quỵ dựa trên các đặc trưng đầu vào. Điều này chứng tỏ ANN có khả năng học và tổng quát hóa tốt từ dữ liệu phức tạp.

Mô hình Cây Quyết Định (CART): Mô hình CART cũng đạt được hiệu suất cao với độ chính xác 88% và F1-score là 0.89. Báo cáo phân loại cho thấy mô hình dự đoán chính xác và hiệu quả cho cả hai lớp. Quá trình tìm kiếm tham số tối ưu thông qua GridSearchCV giúp mô hình đạt được cấu hình tốt nhất, nâng cao hiệu suất tổng thể.

Tóm lại, cả hai mô hình ANN và CART đều chứng minh được hiệu suất tốt trong việc dự đoán nguy cơ đột quỵ. ANN cho thấy khả năng mạnh mẽ trong việc xử lý dữ liệu phức tạp và phi tuyến tính, trong khi CART cung cấp các dự đoán dễ hiểu và chính xác với thời gian huấn luyện nhanh chóng. Việc lựa chọn mô hình nào phù hợp phụ thuộc vào yêu cầu cụ thể của bài toán và sự ưu tiên giữa hiệu suất dự đoán và khả năng giải thích mô hình

## **3.6. Mô hình hồi quy Logistic**

**Giới thiệu về mô hình được sử dụng**

Hồi quy Logistic là một mô hình hồi quy thuộc nhóm học có giám sát (Supervised learning), được sử dụng để dự đoán một biến phụ thuộc nhị phân (Binary) hoặc đa phân (Multinomial). Mô hình này sử dụng hàm Sigmoid để biến đổi kết quả từ một giá trị liên tục về một khoảng giá trị từ 0 đến 1, từ đó có thể sử dụng một ngưỡng để phân loại đối tượng.

Quá trình huấn luyện mô hình hồi quy Logistic bao gồm việc ước lượng các hệ số hồi quy từ tập dữ liệu huấn luyện, sử dụng phương pháp Maximum Likelihood Estimation (ước lượng khả năng tối đa). Mô hình sẽ tìm ra các hệ số tối ưu để giảm thiểu sai số giữa dự đoán và giá trị thực tế.

**Phân tích ưu và nhược điểm của mô hình**

**Ưu điểm**:

Hồi quy Logistic tạo ra một mô hình dễ hiểu, với các hệ số hồi quy giúp ta có thể hiểu rõ mối quan hệ giữa các biến độc lập và biến phụ thuộc. Mô hình thường đạt hiệu suất cao khi các biến độc lập và biến phụ thuộc có mối quan hệ tuyến tính. Bên cạnh đó, không bị ảnh hưởng nhiều bởi các giá trị ngoại lai trong tập dữ liệu huấn luyện. Cuối cùng, hồi quy Logistic tương đối đơn giản và có thể được huấn luyện nhanh chóng, đặc biệt trên các tập dữ liệu nhỏ đến trung bình.

**Nhược điểm:**

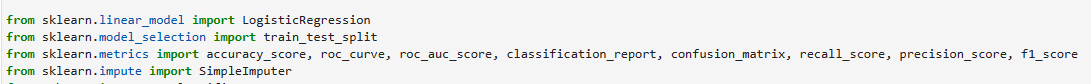
Hồi quy Logistic chỉ có thể mô hình hóa các mối quan hệ tuyến tính giữa các biến độc lập và biến phụ thuộc, không thể xử lý các mối quan hệ phi tuyến. Nếu tập dữ liệu huấn luyện bị lệch, mô hình Hồi quy Logistic có thể bị thiên vị về phía lớp có số lượng mẫu lớn hơn. Kế đó, Hồi quy Logistic chỉ phù hợp với các bài toán phân loại nhị phân hoặc đa phân đơn giản, không thể xử lý các bài toán phân loại phức tạp hơn. Cuối cùng, hồi quy Logistic có thể xử lý các tương tác giữa các biến độc lập, nhưng phải được định nghĩa trước trong mô hình.

**Lý do lựa chọn mô hình**

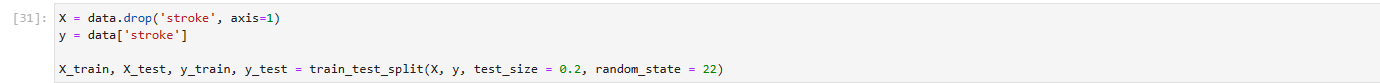
Hồi quy Logistic thường cho được sử dụng khi biến phụ thuộc là một biến phân loại nhị phân. Có thể ước tính xác suất xảy ra của biến phụ thuộc dựa trên các biến độc lập, cho phép ra quyết định dựa trên các xác suất này. Các hệ số hồi quy trong mô hình hồi quy Logistic có thể được diễn giải dễ dàng về ảnh hưởng của các biến độc lập đến xác suất của biến phụ thuộc. Cuối cùng là mô hình hồi quy Logistic không đòi hỏi các giả định về phân phối của dữ liệu như bình thường hóa, tính tuyến tính,… và có thể được áp dụng trong nhiều lĩnh vực.

**Giải thích ý nghĩa của kết quả mô hình**

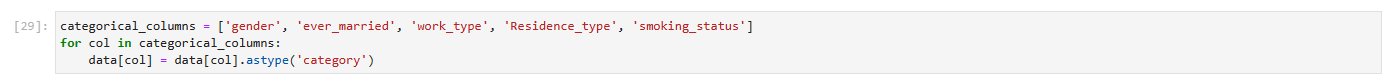
Import những thư viện cần thiết.



Chọn các nhãn đầu vào là X bao gồm toàn bộ cột trong dataframe loại trừ cột ‘stroke’ và biến mục tiêu y là cột ‘stroke’ để dự đoán xem người đó có bị đột quỵ không. Chia tập dữ liệu với tập train là 80% và test là 20%.



Sau đó, chuyển đổi các cột có dữ liệu phân loại (categorical) trong tập dữ liệu data sang định dạng category.

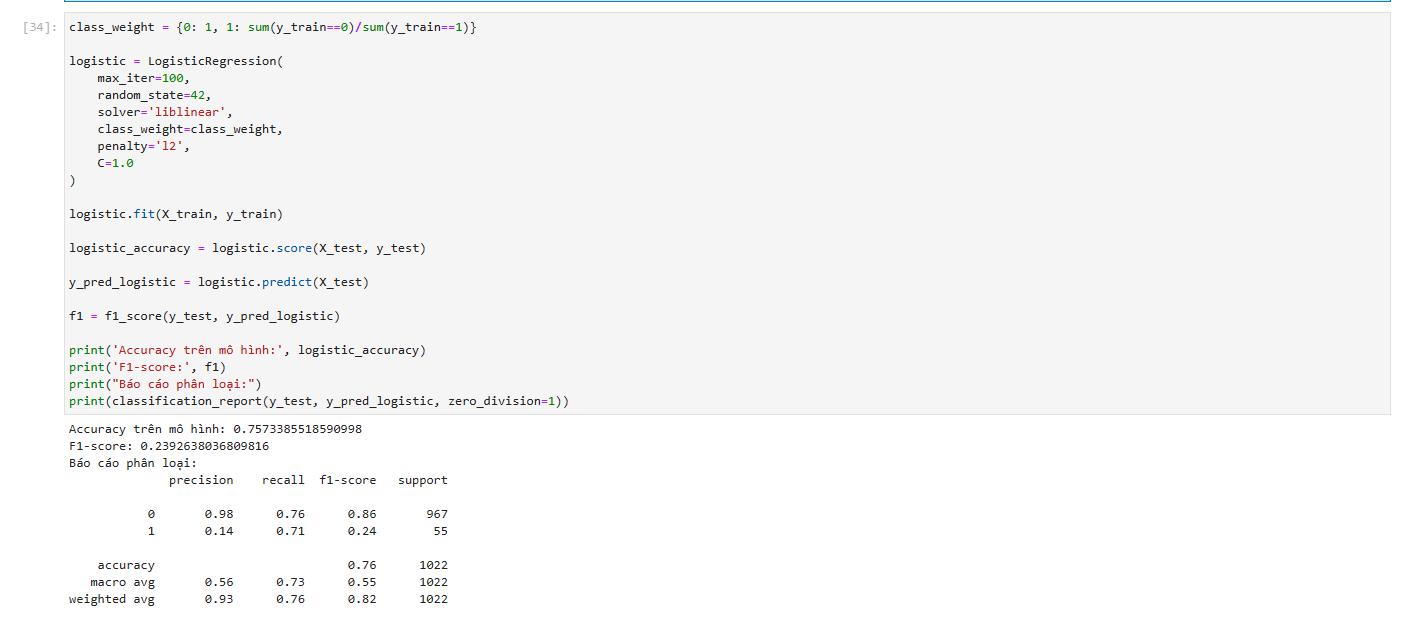


Tiếp đến, thực hiện việc mã hóa các đặc trưng phân loại (categorical features) trong tập dữ liệu data sử dụng kỹ thuật "one-hot encoding".



Giải thích: Mã hóa one-hot là một kỹ thuật phổ biến để xử lý các đặc trưng phân loại trong các bài toán học máy. Nó giúp các thuật toán học máy có thể hiểu và xử lý tốt hơn các đặc trưng này.

Cuối cùng, tạo một biến mới là y\_pred\_logistic để dự đoán nhãn cho tập dữ liệu kiểm tra sau khi mô hình Hồi quy Logistic đã được huấn luyện và đánh giá hiệu quả mô hình thông qua Accuracy score, Precision score, Recall score và F1 score dựa vào y\_test và y\_pred\_logistic.



Đánh giá hiệu quả của mô hình:

* + **Accuracy** (0.75733): Mô hình dự đoán đúng 75.73% các mẫu. Đây là chỉ số tốt khi dữ liệu hơi lệch.
  + **Precision** (0.14): Trong số các mẫu được dự đoán là dương tính, chỉ có 14% là chính xác.
  + **Recall** (0.71): Chỉ có 71% các mẫu thực sự dương tính được dự đoán đúng.
  + **F1 Score** (0.24): Sự cân bằng giữa Precision và Recall cho thấy mô hình không hiệu quả trong việc dự đoán các mẫu dương tính.

Nhận xét:

Chỉ số Precision và F1 Score của mô hình đều khá thấp, điều này chỉ ra rằng mô hình đang gặp khó khăn trong việc dự đoán chính xác lớp dương tính (positive class). Nguyên nhân của việc Precision và F1 Score thấp có thể do tập dữ liệu bị lệch, tức là số lượng mẫu dương tính (positive samples) rất ít so với số lượng mẫu âm tính (negative samples). Trong trường hợp này, có thể cần xem xét các kỹ thuật như tái cân bằng dữ liệu (resampling), điều chỉnh ngưỡng dự đoán, hoặc sử dụng các thuật toán khác phù hợp hơn với dữ liệu lệch.

Biểu đồ Ma trận nhầm lẫn:

A green and white chart

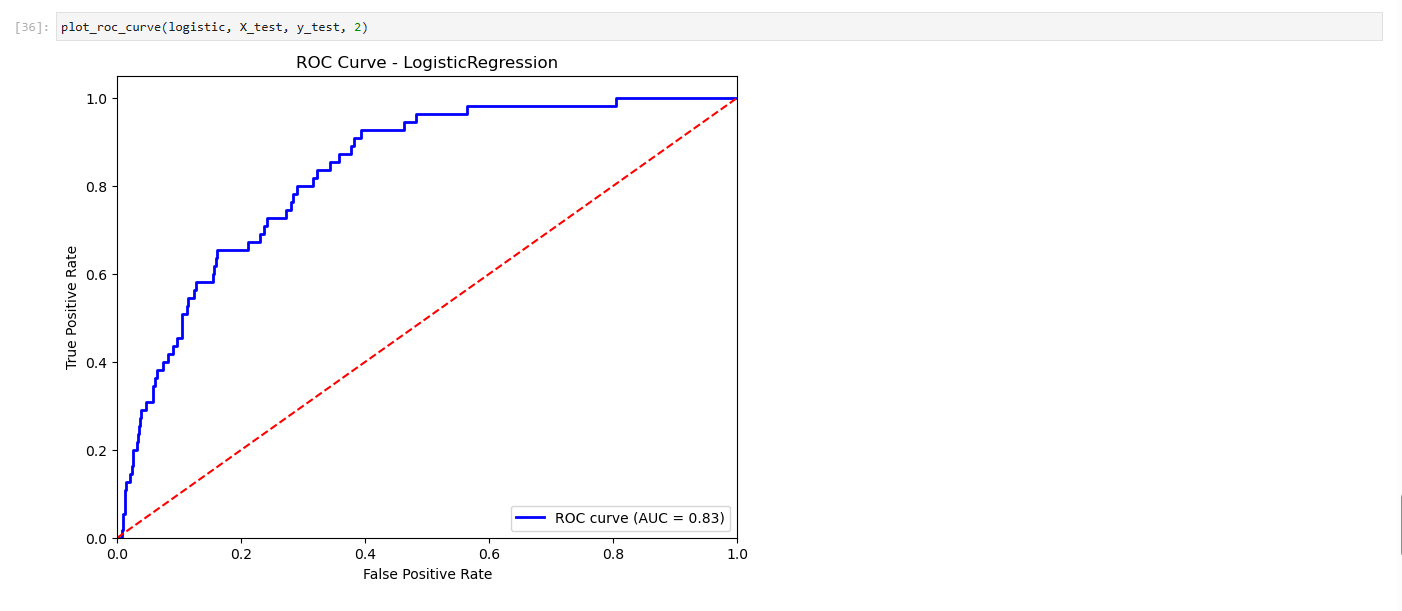
Description automatically generated

Đánh giá:

Trong biểu đồ dữ liệu âm tính giả chiếm đa số, kế tiếp là dương tính giả.

Hai dòng dương tính thật và âm tính giả chiếm không đáng kể.

Biểu đồ đường cong ROC:



Đánh giá:

Trong biểu đồ cho thấy giá trị AUC = 0,83 nằm trong khoảng từ 0,8 đến 0,9, điều này cho thấy mô hình có chất lượng phân loại tốt.

## **3.7 Mô hình Extreme Gradient Boosting (XGBoost)**

**Giới thiệu về mô hình được sử dụng**

Extreme Gradient Boosting (XGBoost) là một mô hình lý tưởng cho những người học máy đã có kiến thức cơ bản, cũng như cho các ứng dụng yêu cầu hiệu suất cao và khả năng tối ưu hóa mạnh mẽ. Hiệu quả với dữ liệu lớn và có khả năng tận dụng tối đa tài nguyên tính toán, XGBoost có thể được áp dụng một cách hiệu quả ngay cả khi bộ dữ liệu lớn và phức tạp. Linh hoạt với nhiều loại dữ liệu khác nhau, XGBoost có thể được áp dụng cho cả bài toán phân loại và hồi quy, và có thể xử lý các dạng dữ liệu khác nhau mà không cần giả định trước về phân phối dữ liệu. Khả năng xử lý tốt các biến không tuyến tính và tự động tối ưu hóa thông qua quá trình gradient boosting, điều này làm cho XGBoost phù hợp với các bài toán có mối quan hệ phức tạp giữa đầu vào và đầu ra.

**Phân tích ưu và nhược điểm của mô hình**

**Ưu điểm:**

XGBoost thường đạt được kết quả tốt hơn các phương pháp Gradient Boosting truyền thống, đặc biệt là trên các tập dữ liệu lớn và phức tạp. Nó có thể cạnh tranh với các mô hình học sâu và đạt được độ chính xác cao. Nhờ các tối ưu hóa về bộ nhớ và tính toán, mô hình có tốc độ xử lý nhanh hơn nhiều so với các thuật toán tăng cường truyền thống. Điều này làm cho nó rất hữu ích trong các ứng dụng thời gian thực. Bên cạnh đó, có thể làm việc với các kiểu dữ liệu khác nhau như dữ liệu số, dữ liệu phân loại, dữ liệu thiếu,… Điều này làm cho nó trở thành một lựa chọn linh hoạt và phổ biến. XGBoost còn cung cấp nhiều tham số để người dùng tùy chỉnh và tối ưu hóa mô hình. Điều này cho phép người dùng điều chỉnh mô hình để phù hợp với các yêu cầu cụ thể.

**Nhược điểm:**

Mặc dù XGBoost nhanh hơn nhiều so với các thuật toán tăng cường truyền thống, nó vẫn có độ phức tạp tính toán cao hơn các mô hình như Random Forest. Giống như các mô hình cây quyết định khác, XGBoost có thể khó hiểu và khó giải thích so với các mô hình tuyến tính đơn giản. Cũng như các mô hình cây quyết định khác, có thể bị ảnh hưởng bởi nhiễu trong dữ liệu và cần phải được chuẩn bị dữ liệu kỹ lưỡng. Cuối cùng, Hiệu suất của XGBoost phụ thuộc rất nhiều vào chất lượng của đặc trưng được sử dụng. Nếu không chọn được đặc trưng tốt, hiệu suất của mô hình có thể sẽ thấp.

**2.1.3. Lý do lựa chọn mô hình**

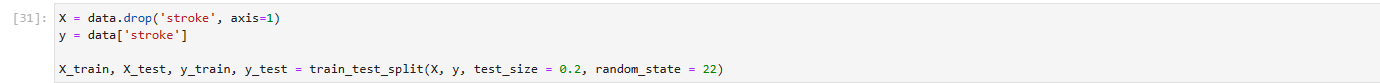
XGBoost được thiết kế để đạt hiệu suất dự đoán cao, giúp tối ưu hóa mô hình một cách hiệu quả thông qua quá trình gradient boosting, có khả năng tận dụng tối đa tài nguyên tính toán và xử lý hiệu quả các bộ dữ liệu lớn và phức tạp. Mô hình có thể áp dụng cho cả bài toán phân loại và hồi quy, và không yêu cầu giả định trước về phân phối dữ liệu, giúp xử lý tốt nhiều dạng dữ liệu khác nhau, XGBoost còn hỗ trợ tính toán song song và phân tán, giúp giảm thời gian huấn luyện mô hình đáng kể khi làm việc với các tập dữ liệu lớn. Các kỹ thuật như regularization và pruning được tích hợp sẵn trong XGBoost giúp giảm thiểu hiện tượng quá khớp (overfitting), đặc biệt hữu ích khi làm việc với các tập dữ liệu phức tạp. Nhìn chung, XGBoost cung cấp các công cụ để đánh giá tầm quan trọng của các đặc trưng, giúp hiểu rõ hơn về dữ liệu và cải thiện quá trình chọn lựa đặc trưng.

**2.2. Giải thích ý nghĩa của kết quả mô hình**

Import những thư viện cần thiết.



Đầu tiên xác định nhãn đầu vào và biến mục tiêu của mô hình. Sau đó chia tập dữ liệu thành 2 tập train, test với train 80% và test 20%.



Tạo và huấn luyện một mô hình phân loại XGB với dữ liệu đầu vào và nhãn đã cho. Sau đó, tạo một biến y\_pred\_rgb để dự đoán nhãn cho tập dữ liệu kiểm tra sau khi mô hình XGBoost đã được huấn luyện và đánh giá hiệu quả mô hình thông qua Accuracy trên mô hình.

A screenshot of a computer

Description automatically generated

Đánh giá hiệu quả mô hình:

* **Accuracy** (0.87084): Cao cho thấy mô hình hoạt động tốt đối với các lớp chính, nhưng điều này có thể bị gây hiểu lầm do sự không cân bằng trong dữ liệu.
* **Precision** (0.20): tương đối thấp hơn so với Recall.
* **Recall** (0.45): ở mức trung bình, cho thấy mô hình tìm ra được nhiều trường hợp dương tính.
* **F1 Score** (0.27): thấp phản ánh việc mô hình không cân bằng tốt giữa Precision và Recall.

Nhận xét:

Kết quả đánh giá cho thấy mô hình XGBoost không phù hợp với dữ liệu này, đặc biệt là với những bài toán cần phải phát hiện các trường hợp dương tính chính xác và đầy đủ. Điều này có thể do dữ liệu không cân bằng hoặc do mô hình XGBoost chưa được tối ưu hóa đúng cách. Có thể cần thử nghiệm với các mô hình khác, kỹ thuật xử lý dữ liệu hoặc điều chỉnh tham số K để cải thiện hiệu suất.

Biểu đồ Ma trận nhầm lẫn:

A screenshot of a computer

Description automatically generated

Đánh giá:

Trong biểu đồ dữ liệu âm tính giả chiếm đa số, kế tiếp là dương tính giả.

Hai dòng dương tính thật và âm tính giả chiếm không đáng kể.

Biểu đồ đường cong ROC:

A screen shot of a graph

Description automatically generated

Đánh giá:

Trong biểu đồ cho thấy giá trị AUC = 0,82 nằm trong khoảng từ 0,8 đến 0,9, điều này cho thấy mô hình có chất lượng phân loại tốt.

# **PHẦN 3: TỔNG KẾT**

## **1. Kết quả đạt được:**

* SVM (Support Vector Machine): Mô hình SVM đạt được kết quả rất tốt trên các chỉ số đánh giá với độ chính xác 95,22%, độ chính xác dương 99,58%, độ nhạy 90,78% và điểm F1 94,98%. Đây là mô hình có hiệu suất cao nhất trong số các mô hình đã thử nghiệm.
* Random Forest: Mặc dù độ chính xác của Random Forest cũng khá cao 95,23%, nhưng các chỉ số khác như độ chính xác dương 28,57%, độ nhạy 3,45% và điểm F1 6,15% lại tương đối thấp. Điều này gợi ý rằng mô hình có khả năng phát hiện các trường hợp dương còn hạn chế.
* KNN (K-Nearest Neighbors): Tương tự như Random Forest, KNN có độ chính xác tổng thể 95,05% nhưng các chỉ số khác như độ chính xác dương 63,64%, độ nhạy 3,89% và điểm F1 7,33% cũng thấp. Điều này cho thấy mô hình KNN cũng chưa thể phát hiện đầy đủ các trường hợp dương.
* ANN (Artificial Neural Network): ANN đạt được kết quả tương đối tốt với độ chính xác 88,68%, độ chính xác dương 85,49%, độ nhạy 92,86% và điểm F1 89,03%. Mô hình này có thể cân bằng tốt giữa độ chính xác và độ nhạy.
* Decision Tree: Decision Tree có độ chính xác tổng thể 95,01% nhưng các chỉ số khác như độ chính xác dương 27,27%, độ nhạy 6,52% và điểm F1 10,53% lại khá thấp. Điều này cho thấy mô hình còn hạn chế trong việc phát hiện các trường hợp dương.
* Hồi Quy Logistic: Mô hình Hồi Quy Logistic có kết quả tương đối kém với độ chính xác 74,27%, độ chính xác dương 11,39%, độ nhạy 69,57% và điểm F1 19,57%. Đây là mô hình có hiệu suất thấp nhất trong số các mô hình đã thử.
* XGBoost: XGBoost đạt được độ chính xác 85,32%, độ chính xác dương 15,79%, độ nhạy 52,17% và điểm F1 24,24%. Kết quả của mô hình này cũng chưa thực sự ấn tượng.
* Nhìn chung, mô hình SVM đã cho kết quả rất tốt và có thể là lựa chọn tối ưu cho bài toán dự đoán đột quỵ. Các mô hình như ANN, Random Forest và KNN cũng có tiềm năng và có thể được cải thiện thêm.

**Kết luận:**

Dựa trên tổng thể các chỉ số, mô hình SVM có vẻ là mô hình có hiệu suất dự đoán tốt nhất, với các chỉ số cao nhất về độ chính xác (Accuracy), độ chính xác (Precision), và điểm F1 (F1 Score). Mặc dù ANN có recall cao nhất, nhưng các chỉ số khác của ANN thấp hơn SVM.

Vì vậy, mô hình SVM được đánh giá là mô hình có dự đoán chính xác nhất trong trường hợp này.

## **2. Hướng phát triển:**

Tìm hiểu kỹ hơn về các đặc trưng dữ liệu và cân nhắc việc lựa chọn/loại bỏ các đặc trưng để cải thiện hiệu suất của các mô hình.

Thử nghiệm các kỹ thuật xử lý dữ liệu không cân bằng như oversampling, undersampling hoặc SMOTE để cải thiện độ nhạy của các mô hình.

Tối ưu hóa các siêu tham số của các mô hình (ví dụ như số cây trong Random Forest, số láng giềng trong KNN, cấu trúc mạng thần kinh trong ANN, v.v.) để nâng cao hiệu suất.

Thử nghiệm các kỹ thuật ensemble (ví dụ như Voting Classifier, Stacking) để tận dụng ưu điểm của các mô hình riêng lẻ.

Tìm hiểu và áp dụng các mô hình học sâu (Deep Learning) có thể mang lại hiệu suất tốt hơn so với các mô hình máy học truyền thống.

## **3. Quá trình làm việc của nhóm và đánh giá:**

**3.1. Quá trình làm việc của nhóm:**

* Nhóm đã phân chia công việc cụ thể, với mỗi thành viên chịu trách nhiệm về một phần việc liên quan đến phân tích dữ liệu bệnh nhân đột quỵ.
* Nhóm đã nghiên cứu và vận dụng tổng cộng 7 mô hình phân tích khác nhau để tìm ra mô hình phù hợp và hiệu quả nhất.
* Các thành viên thường xuyên họp, thảo luận về tiến độ, chia sẻ kết quả phân tích, đưa ra nhận xét và gợi ý cải thiện.
* Nhóm đã làm việc chặt chẽ, phối hợp tốt để hoàn thành nhiệm vụ.

**3.2. Đánh giá của nhóm trưởng:**

* Nhóm trưởng đánh giá cao sự phân công công việc rõ ràng và hiệu quả của nhóm.
* Việc vận dụng 7 mô hình khác nhau đã giúp nhóm có cái nhìn toàn diện và so sánh được ưu nhược điểm của từng mô hình.
* Dựa trên kết quả phân tích, nhóm trưởng cho rằng mô hình SVM là hiệu quả nhất cho việc phân tích dữ liệu bệnh nhân đột quỵ. Mô hình này đạt được độ chính xác (Accuracy) 95,22%, Precision 99,58% và F1-Score 94,98%, vượt trội so với các mô hình khác.
* Nhóm trưởng đánh giá cao sự phối hợp và tinh thần làm việc nhóm của các thành viên, điều này đã góp phần quan trọng vào thành công của nhóm.

# **TÀI LIỆU THAM KHẢO**

[1] Logistic Regression - Bài toán cơ bản trong Machine Learning.

<https://viblo.asia/p/logistic-regression-bai-toan-co-ban-trong-machine-learning-924lJ4rzKPM>

[2] Decision Tree in Python Programming.

[https://www.w3schools.com/python/python\_ml\_decision\_tree.asp](https://www.w3schools.com/python/python_ml_decision_tree.asp%20)

[3] Random Forest algorithm.

[https://machinelearningcoban.com/tabml\_book/ch\_model/random\_forest.html](https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html%20)

[4] R Squared in R - How to Calculate R2 in R?.

<https://www.digitalocean.com/community/tutorials/r-squared-in-r-programming>

[5] Trí tuệ nhân tạo: Các phương pháp đánh giá một mô hình phân loại.

<https://tapit.vn/cac-phuong-phap-danh-gia-mot-mo-hinh-phan-loai/?fbclid=IwAR1zQMzhFr3-DC3Q6DHOGHl0ancYKxC8YT5_IPeW8KT2PYMghCrfKZxYO78>

[6] Machine Learning Library (MLlib) Guide (2024)

<https://spark.apache.org/docs/latest/ml-guide.html>

[7] Support Vector Machines (2024)

[https://scikit-learn.org/stable/modules/svm.html](https://scikit-learn.org/stable/modules/svm.html%20)

[8] Thuật Toán K-Nearest Neighbors (KNN) (2024)

<https://tek4.vn/khoa-hoc/machine-learning-co-ban/thuat-toan-k-nearest-neighbors-knn>

[9] Thuật Toán Gradient Boosting (2024)

<https://viblo.asia/p/gradient-boosting-tat-tan-tat-ve-thuat-toan-manh-me-nhat-trong-machine-learning-YWOZrN7vZQ0>