**MINISTRY OF EDUCATION AND TRAINING**

**HUNGYEN UNIVERSITY OF TECHNOLOGY AND EDUCATION**

****

**FINAL PROJECT**

**DATA MINING**

**APPLYING MACHINE LEARNING ON DIAGNOSING AND ANALYZING ALZHEIMER’S DISEASE DATA**

MAJOR: DATA SCIENCE

STUDENT: **VU QUANG PHUC**

CLASS: **124221**

MENTOR: **Ph.D NGUYEN VAN QUYET**

**HUNG YEN – 2025**

**COMMENT**

**Comments from mentor:**

**MENTOR**

**Nguyen Van Quyet**

**COMMITMENT**

Em xin cam đoan bài tập lớn môn lập trình Python nâng cao có tên là “Phân tích dữ liệu về bộ dữ liệu thời trang Anh-Mỹ” là sản phẩm của bản thân em.

Những phần sử dụng tài liệu tham khảo trong bài tập lớn đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong bài tập lớn hoàn toàn là kết quả do bản thân em thực hiện.

Nếu vi phạm lời cam đoan này, em xin chịu hoàn toàn trách nhiệm trước khoa và nhà trường.

*Hưng Yên, ngày … tháng 06 năm 2024*

Sinh viên

Nguyễn Văn X

**ACKNOWLEDGEMENT**

Để có thể hoàn thành bài tập lớn này, lời đầu tiên em xin phép gửi lời cảm ơn tới bộ môn Khoa học máy tính, Khoa Công nghệ thông tin – Trường Đại học Sư phạm Kỹ thuật Hưng Yên đã tạo điều kiện thuận lợi cho em thực hiện bài tập lớn môn học này.

Đặc biệt em xin chân thành cảm ơn thầy Hoàng Quốc Việt đã rất tận tình hướng dẫn, chỉ bảo em trong suốt thời gian thực hiện bài tập lớn vừa qua.

Em cũng xin chân thành cảm ơn tất cả các Thầy, các Cô trong Trường đã tận tình giảng dạy, trang bị cho em những kiến thức cần thiết, quý báu để giúp em thực hiện được bài tập lớn này.

Mặc dù em đã có cố gắng, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hy vọng sẽ nhận được những ý kiến nhận xét, góp ý của các Thầy cô về những kết quả triển khai trong bài tập lớn.

Em xin trân trọng cảm ơn!

**TABLE OF CONTENTS**

[CHƯƠNG 1: INTRODUCTION TO THE PROBLEM 4](#_Toc187153609)

[1.1 Bài toán 4](#_Toc187153610)

[1.2 Trình bày dữ liệu bài toán 4](#_Toc187153611)

[1.3 Tiền xử lý dữ liệu 9](#_Toc187153612)

[1.4 Thống kê dữ liệu 9](#_Toc187153613)

[1.5 Trực quan hoá dữ liệu 9](#_Toc187153614)

[CHƯƠNG 2: ESSENTIAL KNOWLEDGE 10](#_Toc187153615)

[2.1 Pandas 10](#_Toc187153616)

[2.2 Matplotlib 12](#_Toc187153617)

[2.3 Sklearn 14](#_Toc187153618)

[2.4 Feature Engineering 16](#_Toc187153619)

[2.5 MaxAbsScalar 16](#_Toc187153620)

[2.6 Principal Component Analysis 16](#_Toc187153621)

[2.7 Synthetic Minority Over-sampling Technique (SMOTE) 17](#_Toc187153622)

[2.8 HyperParameter Tuning 18](#_Toc187153623)

[2.9 Cross Validation 18](#_Toc187153624)

[2.10 Machine Learning 19](#_Toc187153625)

[2.10.1 Support Vector Machine (SVM) 19](#_Toc187153626)

[2.10.2 Naive Bayes 21](#_Toc187153627)

[2.10.3 Logistic Regression 22](#_Toc187153628)

[2.10.4 XGBoost 23](#_Toc187153629)

[2.11 Confusion Matrix 24](#_Toc187153630)

[2.12 Evaluated Metrics 25](#_Toc187153631)

[2.13 ROC-AUC 27](#_Toc187153632)

[CHƯƠNG 3: SOLUTIONS 28](#_Toc187153633)

[3.1. Mã nguồn tiền xử lý dữ liệu 28](#_Toc187153634)

[3.2. Mã nguồn thống kê dữ liệu 34](#_Toc187153635)

[3.3. Mã nguồn trực quan hóa dữ liệu 40](#_Toc187153636)

[CONCLUSION 44](#_Toc187153637)

[REFERENCES 45](#_Toc187153638)

**DANH MỤC CÁC HÌNH VẼ**

[Hình 1.1 Tổng quan về bộ dữ liệu Alzheimer’s Desease 4](#_Toc186841191)

[Hình 1.2 Thông tin cơ bản về kiểu dữ liệu của từng đặc trưng 4](#_Toc186841192)

[Hình 3.1 Biểu đồ trực quan phân phối dữ liệu tuổi của bệnh nhân 23](#_Toc186841193)

[Hình 3.2 Biểu đồ trực quan tỉ lệ dữ liệu giới tính của bệnh nhân 24](#_Toc186841194)

[Hình 3.3 Biểu đồ trực quan phân phối dữ liệu MMSE của bệnh nhân 25](#_Toc186841195)

[Hình 3.4 Biểu đồ trực quan phân phối dữ liệu BMI của bệnh nhân 26](#_Toc186841196)

# INTRODUCTION TO THE PROBLEM

## Bài toán

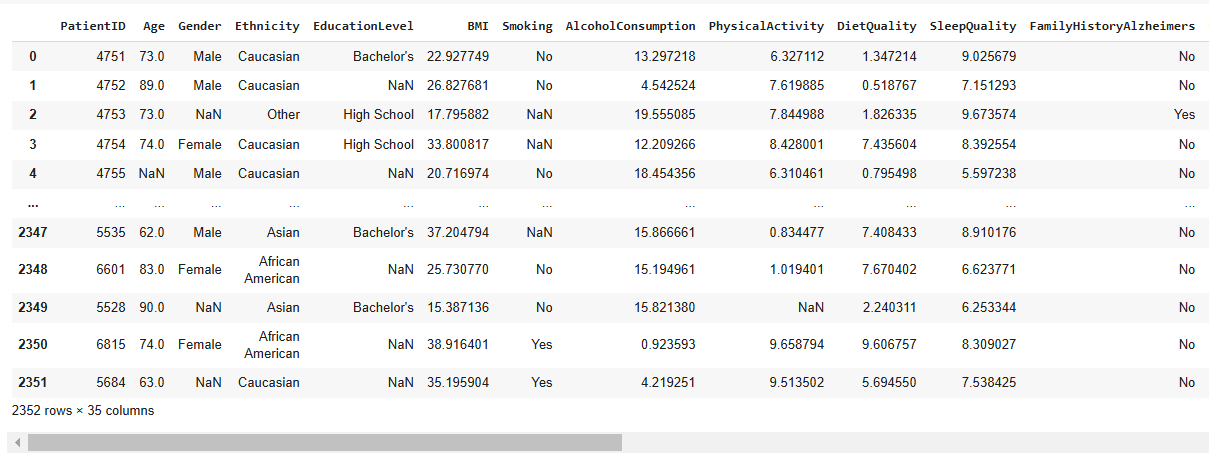
Bệnh Alzheimer là một bệnh thoái hóa não tiến triển, gây ảnh hưởng nghiêm trọng đến trí nhớ, tư duy và hành vi, là nguyên nhân phổ biến nhất của sa sút trí tuệ. Hiện nay, trên thế giới có khoảng 55 triệu người mắc các dạng sa sút trí tuệ, trong đó Alzheimer chiếm 60-70%, với dự báo số ca bệnh sẽ tăng gấp ba lần vào năm 2050. Tại Việt Nam, khoảng 500.000 người đang mắc bệnh này, và con số này sẽ còn tăng nhanh do tốc độ già hóa dân số. Ứng dụng học máy (machine learning) đang mở ra nhiều cơ hội trong việc chẩn đoán Alzheimer thông qua phân tích ảnh y tế (MRI, PET), dự đoán sớm từ dữ liệu sinh học, phân tích hành vi và ngôn ngữ, cũng như hỗ trợ điều trị và quản lý bệnh nhân. Tuy nhiên, việc triển khai tại Việt Nam đối mặt với các thách thức như thiếu dữ liệu chất lượng cao và hạ tầng y tế hạn chế. Để khai thác tiềm năng này, cần sự hợp tác chặt chẽ giữa các bên liên quan nhằm nâng cao hiệu quả chẩn đoán, điều trị, đồng thời cải thiện chất lượng cuộc sống cho bệnh nhân và gia đình.

Học máy mang lại nhiều cải tiến vượt trội so với phương pháp chẩn đoán Alzheimer truyền thống, như khả năng phát hiện sớm các dấu hiệu từ tổn thương nhỏ, xử lý lượng lớn dữ liệu phức tạp (hình ảnh, sinh học, di truyền), và tự động hóa quy trình chẩn đoán, giúp tiết kiệm thời gian và giảm sai sót. Bên cạnh đó, học máy hỗ trợ cá nhân hóa điều trị dựa trên đặc điểm riêng biệt của từng bệnh nhân và khai thác dữ liệu phi truyền thống như giọng nói hoặc hành vi. Các điều kiện thuận lợi như dữ liệu y tế phong phú, công nghệ tính toán mạnh mẽ, và sự quan tâm từ cộng đồng khoa học đang tạo nền tảng vững chắc để ứng dụng học máy vào thực tiễn, nâng cao hiệu quả chẩn đoán và điều trị bệnh Alzheimer.

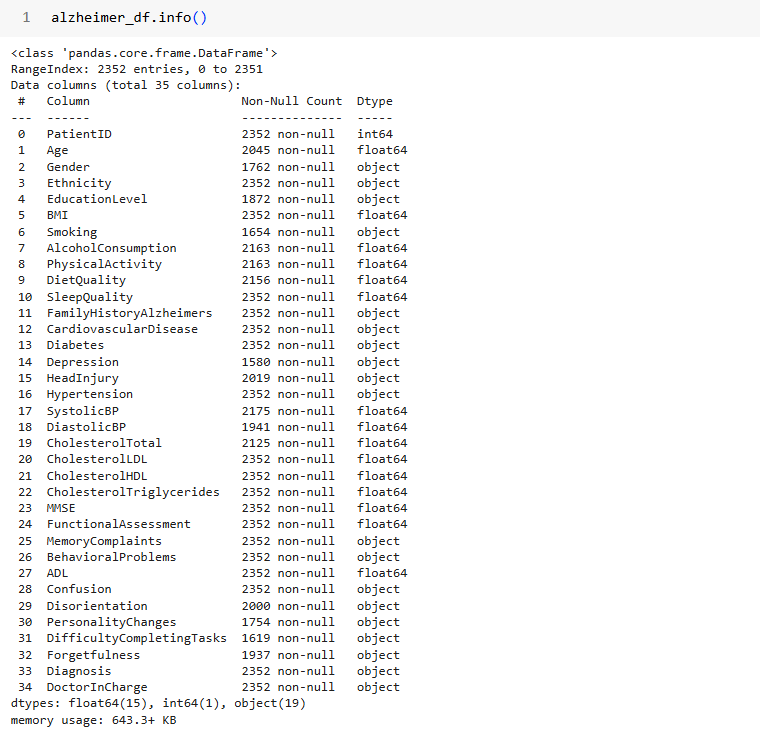
## Trình bày dữ liệu bài toán

Link dữ liệu trên Kaggle:

[Alzheimer's Desease Dataset | Kaggle](https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset/data)



Hình 1.1 Tổng quan về bộ dữ liệu Alzheimer’s Desease



Hình 1.2 Thông tin cơ bản về kiểu dữ liệu của từng đặc trưng

* Dữ liệu bài toán gồm các feature sau:

0. **PatientID**: Mã định danh duy nhất của mỗi bệnh nhân, không ảnh hưởng trực tiếp đến bệnh Alzheimer, chỉ dùng để quản lý và phân biệt bệnh nhân.

1. **Age**: Tuổi của bệnh nhân; tuổi cao là yếu tố nguy cơ hàng đầu, đặc biệt ở người trên 65 tuổi, do tế bào não thoái hóa tự nhiên.

2. **Gender**: Giới tính của bệnh nhân (Nam hoặc Nữ); nữ giới có nguy cơ mắc Alzheimer cao hơn nam giới, liên quan đến tuổi thọ cao hơn và thay đổi hormone sau mãn kinh.

3. **Ethnicity**: Dân tộc của bệnh nhân; có thể ảnh hưởng đến nguy cơ bệnh do sự khác biệt về di truyền và lối sống giữa các nhóm dân tộc.

4. **EducationLevel**: Trình độ học vấn của bệnh nhân; trình độ học vấn cao hơn thường liên quan đến nguy cơ Alzheimer thấp hơn, do dự trữ nhận thức (cognitive reserve) lớn hơn.

5. **BMI**: Chỉ số khối cơ thể; BMI cao hoặc thấp quá mức đều có thể liên quan đến nguy cơ Alzheimer thông qua các bệnh lý như tiểu đường, bệnh tim mạch, hoặc suy dinh dưỡng.

6. **Smoking**: Thói quen hút thuốc; hút thuốc làm tăng nguy cơ Alzheimer do gây tổn thương mạch máu não và tăng stress oxy hóa.

7. **AlcoholConsumption**: Mức độ tiêu thụ rượu; tiêu thụ rượu quá mức có thể làm tổn thương não, trong khi tiêu thụ vừa phải có thể có lợi ở một số trường hợp.

8. **PhysicalActivity**: Mức độ hoạt động thể chất; tập thể dục đều đặn giúp giảm nguy cơ Alzheimer do tăng cường tuần hoàn máu và bảo vệ tế bào não.

9. **DietQuality**: Chất lượng chế độ ăn; chế độ ăn giàu dinh dưỡng (như Địa Trung Hải) giúp giảm nguy cơ Alzheimer, trong khi chế độ ăn nghèo nàn tăng nguy cơ bệnh.

10. **SleepQuality**: Chất lượng giấc ngủ; giấc ngủ kém làm tăng nguy cơ Alzheimer do tích tụ beta-amyloid trong não.

11. **FamilyHistoryAlzheimers**: Tiền sử gia đình mắc Alzheimer; yếu tố di truyền có vai trò lớn, đặc biệt với các đột biến gen như APOE-ε4.

12. **CardiovascularDisease**: Bệnh tim mạch; bệnh tim mạch làm tăng nguy cơ Alzheimer thông qua tổn thương mạch máu não.

13. **Diabetes**: Tiểu đường; bệnh này liên quan đến tăng nguy cơ Alzheimer do ảnh hưởng đến tuần hoàn máu và viêm mãn tính.

14. **Depression**: Trầm cảm; trầm cảm mãn tính là yếu tố nguy cơ của Alzheimer, có thể liên quan đến thay đổi sinh lý và cấu trúc não.

15. **HeadInjury**: Tiền sử chấn thương đầu; chấn thương đầu nghiêm trọng hoặc lặp lại làm tăng nguy cơ Alzheimer, có thể do tổn thương tế bào thần kinh.

16. **Hypertension**: Tăng huyết áp; huyết áp cao kéo dài làm tăng nguy cơ tổn thương mạch máu não và dẫn đến Alzheimer.

17. **SystolicBP**: Huyết áp tâm thu; huyết áp tâm thu cao bất thường có liên quan đến nguy cơ tổn thương não.

18. **DiastolicBP**: Huyết áp tâm trương; huyết áp tâm trương không ổn định có thể gây tổn thương mạch máu nhỏ trong não.

19. **CholesterolTotal**: Tổng mức cholesterol; cholesterol cao tăng nguy cơ tổn thương mạch máu và Alzheimer.

20. **CholesterolLDL**: Cholesterol "xấu"; LDL cao làm tăng nguy cơ bệnh thông qua xơ vữa động mạch và tổn thương mạch máu não.

21. **CholesterolHDL**: Cholesterol "tốt"; mức HDL thấp có thể tăng nguy cơ Alzheimer, do giảm bảo vệ mạch máu.

22. **CholesterolTriglycerides**: Triglyceride; mức triglyceride cao liên quan đến tăng nguy cơ Alzheimer thông qua tổn thương mạch máu và viêm mãn tính.

23. **MMSE**: Điểm kiểm tra trạng thái tâm thần; điểm thấp chỉ ra suy giảm nhận thức, một dấu hiệu chính của Alzheimer.

24. **FunctionalAssessment**: Đánh giá chức năng hàng ngày; suy giảm chức năng là một dấu hiệu tiến triển của bệnh Alzheimer.

25. **MemoryComplaints**: Phàn nàn về trí nhớ; đây là triệu chứng sớm và phổ biến của Alzheimer.

26. **BehavioralProblems**: Các vấn đề hành vi; thay đổi hành vi thường gặp ở giai đoạn giữa và cuối của bệnh Alzheimer.

27. **ADL**: Khả năng thực hiện hoạt động hàng ngày; suy giảm khả năng ADL cho thấy mức độ nghiêm trọng của bệnh.

28. **Confusion**: Nhầm lẫn; nhầm lẫn ngày càng nặng là triệu chứng điển hình của Alzheimer.

29. **Disorientation**: Mất định hướng; bệnh nhân thường quên nơi chốn, thời gian hoặc người thân ở các giai đoạn tiến triển.

30. **PersonalityChanges**: Thay đổi tính cách; thay đổi cảm xúc và hành vi bất thường là đặc trưng của Alzheimer giai đoạn sau.

31. **DifficultyCompletingTasks**: Khó khăn trong hoàn thành công việc; đây là một dấu hiệu sớm của suy giảm nhận thức.

32. **Forgetfulness**: Hay quên; quên thông tin gần đây là triệu chứng khởi đầu phổ biến của Alzheimer.

33. **Diagnosis**: Chẩn đoán; kết quả chẩn đoán bệnh dựa trên các triệu chứng, kiểm tra và dữ liệu lâm sàng.

34. **DoctorInCharge**: Bác sĩ điều trị; không ảnh hưởng trực tiếp đến bệnh, nhưng phản ánh người chịu trách nhiệm quản lý bệnh nhân.

* Dữ liệu bài toán là 1 file csv gồm 2352 rows × 35 columns

Tương ứng với có 35 features và mỗi feature có 2352 dữ liệu đầu vào.

* Sau khi mô tả dữ liệu ta có:

## Tiền xử lý dữ liệu

1. Xóa các bản ghi trùng lặp
2. Xóa các cột không quan trọng trong bộ dữ liệu
3. Mã hóa các cột chứa dữ liệu văn bản
4. Phân chia dữ liệu
5. Xử lý dữ liệu mất mát
6. Xử lý dữ liệu ngoại lai
7. Xử lý mất cân bằng nhãn

## Thống kê dữ liệu

1. Đếm tổng số bệnh nhân
2. Phân phối tuổi bệnh nhân
3. Phân phối giới tính
4. Phân tích trình độ học vấn
5. BMI trung bình theo nhóm tuổi
6. Tỉ lệ bệnh nhân hút thuốc
7. Tương quan giữa giấc ngủ và MMSE
8. Phân tích tiền sử gia đình
9. Phân tích bệnh đồng mắc
10. Phân tích điểm MMSE theo giới tính
11. Phân tích ảnh hưởng của chất lượng ăn uống
12. Tỷ lệ rối loạn hành vi
13. Mối quan hệ giữa huyết áp và trí nhớ
14. Tỷ lệ mất phương hướng
15. Điểm MMSE theo nhóm có và không chấn thương đầu

## Trực quan hoá dữ liệu

1. Biểu đồ phân phối tuổi bệnh nhân
2. Biểu đồ cột tỷ lệ giới tính
3. Biểu đồ phân phối điểm MMSE
4. Biểu đồ phân bố BMI

# ESSENTIAL KNOWLEDGE

## Pandas

Pandas là một thư viện mã nguồn mở mạnh mẽ được sử dụng rộng rãi trong Python để thao tác và phân tích dữ liệu, đặc biệt trong lĩnh vực khoa học dữ liệu và học máy. Tên "Pandas" xuất phát từ cụm từ "Panel Data," một thuật ngữ trong kinh tế lượng, nhấn mạnh khả năng xử lý dữ liệu đa chiều. Pandas cung cấp hai cấu trúc dữ liệu chính: \*\*DataFrame\*\* (một bảng dữ liệu hai chiều với nhãn dòng và cột, tương tự như bảng trong Excel hoặc cơ sở dữ liệu) và \*\*Series\*\* (một mảng một chiều, giống như một cột trong DataFrame). Các tính năng chính của Pandas bao gồm: xử lý dữ liệu từ nhiều nguồn khác nhau như CSV, Excel, SQL, và JSON; cung cấp các phương thức mạnh mẽ để sắp xếp, lọc, nhóm, nối và tóm tắt dữ liệu; hỗ trợ thao tác dữ liệu chuỗi thời gian và xử lý dữ liệu thiếu (missing data). Ngoài ra, Pandas tích hợp tốt với các thư viện Python khác như NumPy, Matplotlib, và Scikit-learn, làm cho nó trở thành công cụ linh hoạt trong việc tiền xử lý dữ liệu và phân tích.

Việc sử dụng Pandas mang lại nhiều lợi ích rõ rệt. Trước hết, Pandas có một cú pháp đơn giản, dễ học, và tài liệu phong phú, giúp ngay cả những người mới bắt đầu cũng có thể nhanh chóng làm quen. Thứ hai, các thao tác của Pandas được tối ưu hóa để xử lý tập dữ liệu lớn, nhờ được xây dựng dựa trên NumPy, một thư viện xử lý mảng hiệu suất cao. Thứ ba, Pandas cung cấp các công cụ trực quan hóa dữ liệu cơ bản và tích hợp chặt chẽ với các thư viện vẽ biểu đồ khác, giúp người dùng dễ dàng khám phá và trình bày dữ liệu. Cuối cùng, khả năng quản lý và thao tác dữ liệu dễ dàng, từ việc chuẩn hóa dữ liệu đầu vào đến xử lý dữ liệu phức tạp, khiến Pandas trở thành lựa chọn lý tưởng cho các nhà khoa học dữ liệu, nhà phân tích, và kỹ sư học máy. Trong thực tế, Pandas không chỉ giúp tăng năng suất làm việc mà còn giảm nguy cơ sai sót trong xử lý dữ liệu, điều này rất quan trọng trong các dự án yêu cầu độ chính xác cao.

Việc sử dụng Pandas để xử lý bộ dữ liệu về bệnh Alzheimer có nhiều lợi ích rõ rệt nhờ các tính năng mạnh mẽ và linh hoạt của thư viện này, đặc biệt khi làm việc với dữ liệu y tế phức tạp. Dưới đây là các lý do chính:

1. Quản lý dữ liệu dễ dàng:

Bộ dữ liệu chứa nhiều đặc trưng (features) và giá trị đa dạng, từ các thông số số liệu (tuổi, huyết áp, cholesterol) đến thông tin danh mục (giới tính, dân tộc) và các giá trị nhị phân (hút thuốc, tiền sử gia đình). Pandas cung cấp cấu trúc DataFrame, cho phép tổ chức dữ liệu một cách rõ ràng, dễ dàng truy xuất, và thao tác với các cột cụ thể.

1. Xử lý dữ liệu thiếu (Missing Data):

Trong dữ liệu y tế, các giá trị thiếu (missing values) rất phổ biến. Pandas cung cấp các công cụ để phát hiện, thay thế hoặc loại bỏ dữ liệu thiếu thông qua các hàm như isnull() hoặc fillna(), giúp đảm bảo dữ liệu sạch và nhất quán.

1. Chuyển đổi dữ liệu:

Pandas hỗ trợ chuyển đổi dữ liệu hiệu quả, như chuẩn hóa các cột số liệu (tuổi, BMI) hoặc mã hóa dữ liệu danh mục (giới tính, dân tộc) thành dạng mà các thuật toán học máy có thể sử dụng, nhờ các phương thức như apply(), map(), và get\_dummies().

1. Khả năng phân tích dữ liệu ban đầu( EDA):

Với Pandas, bạn có thể dễ dàng tính toán các thống kê cơ bản (trung bình, độ lệch chuẩn, tần suất xuất hiện) để hiểu sâu hơn về từng đặc trưng. Ví dụ, describe() cung cấp tổng quan về dữ liệu số, giúp nhanh chóng xác định các giá trị ngoại lệ (outliers) hoặc xu hướng bất thường.

1. Lọc và nhóm dữ liệu (Filtering and Grouping):

Pandas hỗ trợ thao tác với các nhóm dữ liệu dựa trên điều kiện cụ thể. Ví dụ, bạn có thể dễ dàng nhóm bệnh nhân theo giới tính, độ tuổi, hoặc chẩn đoán để so sánh các thông số khác nhau bằng cách sử dụng groupby() và filter().

1. Tích hợp tốt với học máy:

Bộ dữ liệu này là tiền đề cho các mô hình học máy. Pandas hỗ trợ chuẩn bị dữ liệu đầu vào cho các thư viện học máy như Scikit-learn hoặc TensorFlow, từ việc chia nhỏ dữ liệu (train-test split) đến xuất dữ liệu dưới dạng NumPy array.

1. Hiệu quả trong thao tác dữ liệu lớn:

Pandas được tối ưu hóa để xử lý lượng lớn dữ liệu một cách nhanh chóng và hiệu quả. Bộ dữ liệu về Alzheimer có thể có nhiều dòng (bệnh nhân) và cột (đặc trưng), nhưng Pandas giúp quản lý và thao tác dễ dàng nhờ hiệu suất cao.

1. Tích hợp với trực quan hóa:

Với Pandas, bạn có thể dễ dàng tạo các biểu đồ như histogram, boxplot, hoặc scatter plot để phân tích mối quan hệ giữa các đặc trưng (ví dụ: mối quan hệ giữa tuổi và nguy cơ Alzheimer).

## Matplotlib

1. Trực quan hóa dữ liệu dễ dàng:

Matplotlib cung cấp các công cụ mạnh mẽ để tạo ra nhiều loại biểu đồ như histogram, scatter plot, line plot, bar chart, box plot, phù hợp với dữ liệu đa dạng trong bộ dữ liệu Alzheimer. Ví dụ: biểu đồ scatter có thể giúp minh họa mối quan hệ giữa tuổi và điểm MMSE, trong khi biểu đồ histogram giúp phân phối các đặc trưng như chỉ số BMI hoặc cholesterol.

1. Tùy chỉnh mạnh mẽ:

Matplotlib cho phép tùy chỉnh toàn diện các yếu tố của biểu đồ, bao gồm màu sắc, kiểu đường, nhãn, tiêu đề, kích thước, và chú giải. Điều này rất hữu ích để tạo ra các biểu đồ chuyên nghiệp hoặc nhấn mạnh vào các yếu tố quan trọng, chẳng hạn như hiển thị tỷ lệ bệnh nhân với huyết áp cao theo từng nhóm tuổi.

1. Khám phá mối quan hệ giữa các đặc trưng:

Matplotlib hỗ trợ minh họa mối quan hệ giữa các đặc trưng qua biểu đồ hai chiều hoặc ba chiều. Chẳng hạn, bạn có thể sử dụng scatter plot để kiểm tra sự tương quan giữa cholesterol LDL và huyết áp tâm thu trong việc dự đoán nguy cơ Alzheimer, hoặc biểu đồ heatmap để hiển thị mức độ liên quan giữa nhiều đặc trưng.

1. Hỗ trợ phân tích so sánh:

Với Matplotlib, bạn có thể dễ dàng so sánh các nhóm dữ liệu khác nhau. Ví dụ: sử dụng bar chart để so sánh tỷ lệ mắc bệnh Alzheimer giữa nam và nữ, hoặc box plot để so sánh giá trị cholesterol giữa các nhóm bệnh nhân có hoặc không có tiền sử gia đình.

1. Xử lý dữ liệu thời gian:

Bộ dữ liệu Alzheimer thường có các đặc trưng liên quan đến thời gian, chẳng hạn như diễn biến triệu chứng hoặc kết quả đánh giá theo thời gian. Matplotlib hỗ trợ vẽ biểu đồ line để minh họa xu hướng thay đổi của điểm MMSE hoặc chức năng ADL qua các năm.

1. Tích hợp tốt với Pandas:

Matplotlib hoạt động mượt mà với các DataFrame của Pandas. Bạn có thể dễ dàng truyền dữ liệu trực tiếp từ Pandas vào Matplotlib để tạo biểu đồ, ví dụ như biểu đồ phân phối tuổi hoặc biểu đồ scatter giữa huyết áp và chỉ số BMI.

1. Phân phối dữ liệu và giá trị ngoại lệ:

Matplotlib hỗ trợ trực quan hóa phân phối dữ liệu và phát hiện các giá trị ngoại lệ thông qua các biểu đồ như box plot hoặc violin plot, giúp bạn nhanh chóng nhận ra bất kỳ sự bất thường nào trong dữ liệu, chẳng hạn như huyết áp cao bất thường hoặc BMI rất thấp.

1. Tạo báo cáo và trình bày dữ liệu:

Với Matplotlib, bạn có thể tạo các biểu đồ có chất lượng cao phù hợp cho việc trình bày báo cáo, nghiên cứu khoa học, hoặc chia sẻ trong nhóm. Ví dụ, biểu đồ cột có thể được sử dụng để minh họa tỷ lệ phần trăm bệnh nhân có các triệu chứng như mất định hướng hoặc thay đổi tính cách.

1. Hiệu quả và linh hoạt:

Matplotlib rất linh hoạt khi làm việc với các tập dữ liệu lớn, cho phép bạn chia nhỏ các tập con hoặc tổng hợp dữ liệu thành các biểu đồ trực quan. Điều này rất quan trọng khi cần so sánh nhiều nhóm bệnh nhân, chẳng hạn giữa các nhóm tuổi hoặc các mức độ nghiêm trọng của bệnh.

1. Hỗ trợ biểu đồ nâng cao với Seaborn:

Mặc dù Matplotlib là công cụ cơ bản, nó tích hợp rất tốt với Seaborn (một thư viện cao cấp hơn) để tạo ra các biểu đồ thống kê đẹp mắt và dễ dàng minh họa các mối quan hệ phức tạp, như biểu đồ heatmap giữa các đặc trưng hoặc pair plot để kiểm tra tương quan giữa các cột số liệu.

## Sklearn

Scikit-learn (sklearn) cung cấp một loạt các công cụ mạnh mẽ giúp xử lý và tiền xử lý dữ liệu một cách hiệu quả. Trong bộ dữ liệu về bệnh Alzheimer, bạn có thể sử dụng các công cụ như `StandardScaler` và `MinMaxScaler` để chuẩn hóa và điều chỉnh các đặc trưng đầu vào như huyết áp, cholesterol, BMI. Việc chuẩn hóa này giúp mô hình học máy có thể xử lý dữ liệu hiệu quả hơn, đặc biệt là khi các đặc trưng có phạm vi hoặc đơn vị khác nhau.

Scikit-learn cũng cung cấp một hàm hữu ích gọi là `train\_test\_split`, cho phép bạn chia bộ dữ liệu thành các tập huấn luyện và kiểm tra một cách dễ dàng. Điều này rất quan trọng trong việc kiểm tra độ chính xác của mô hình sau khi huấn luyện, giúp đảm bảo rằng mô hình có thể tổng quát hóa tốt trên dữ liệu chưa thấy và tránh tình trạng overfitting.

Với Scikit-learn, bạn có thể dễ dàng chọn các mô hình học máy phù hợp. Thư viện này hỗ trợ nhiều thuật toán học giám sát như hồi quy tuyến tính, cây quyết định, SVM (Support Vector Machine), hay các mô hình học không giám sát như k-means. Ví dụ, bạn có thể sử dụng hồi quy logistic để dự đoán nguy cơ mắc bệnh Alzheimer dựa trên các đặc trưng như tuổi tác, huyết áp, cholesterol, hoặc SVM để phân loại bệnh nhân thành các nhóm nguy cơ cao và thấp.

Khi đã lựa chọn được mô hình, Scikit-learn cung cấp nhiều công cụ để đánh giá hiệu suất của mô hình như kỹ thuật cross-validation và grid search. Những công cụ này giúp bạn tối ưu hóa các siêu tham số của mô hình, từ đó nâng cao hiệu quả dự đoán. Bạn có thể sử dụng `cross\_val\_score` để đánh giá mô hình qua nhiều lần phân chia dữ liệu khác nhau, hoặc dùng `GridSearchCV` để tìm bộ tham số tối ưu cho mô hình của mình.

Một vấn đề thường gặp trong các bộ dữ liệu y tế là giá trị thiếu. Scikit-learn cung cấp các công cụ như SimpleImputer để thay thế các giá trị thiếu bằng các giá trị trung bình, trung vị hoặc mode của cột, giúp duy trì tính toàn vẹn của bộ dữ liệu và không làm gián đoạn quá trình huấn luyện mô hình.

Về mặt kỹ thuật biến đổi đặc trưng, Scikit-learn hỗ trợ các công cụ như OneHotEncoding và PolynomialFeatures, giúp cải thiện khả năng học của mô hình. Ví dụ, các đặc trưng phân loại như giới tính hay dân tộc có thể được mã hóa dưới dạng các cột nhị phân bằng `OneHotEncoder`, giúp mô hình học máy có thể xử lý chúng một cách hiệu quả.

Một điểm mạnh của Scikit-learn là khả năng cung cấp các mô hình học máy có thể giải thích được. Các mô hình như cây quyết định hay hồi quy tuyến tính có thể đưa ra các quyết định rõ ràng về cách các đặc trưng ảnh hưởng đến kết quả, điều này rất quan trọng trong các bài toán y tế như Alzheimer, nơi việc giải thích các quyết định của mô hình có thể giúp bác sĩ và chuyên gia y tế hiểu được các yếu tố ảnh hưởng đến nguy cơ bệnh.

Ngoài ra, Scikit-learn có thể dễ dàng tích hợp với các thư viện khác như Pandas và Matplotlib, giúp người dùng dễ dàng thao tác với dữ liệu và trực quan hóa kết quả. Sau khi huấn luyện mô hình, bạn có thể sử dụng Pandas để phân tích kết quả, sau đó sử dụng Matplotlib để trực quan hóa các chỉ số quan trọng như độ chính xác, độ nhạy và độ đặc hiệu.

Scikit-learn cũng rất hiệu quả trong việc xử lý các bộ dữ liệu lớn, giúp bạn huấn luyện các mô hình học máy phức tạp mà không gặp phải vấn đề về hiệu suất. Điều này đặc biệt quan trọng khi làm việc với các bộ dữ liệu có quy mô lớn như bộ dữ liệu bệnh Alzheimer.

Mặc dù Scikit-learn chủ yếu tập trung vào các thuật toán học máy truyền thống, thư viện này cũng hỗ trợ các mô hình học sâu đơn giản thông qua các công cụ như MLPClassifier và MLPRegressor, cho phép giải quyết các bài toán phức tạp hơn trong việc dự đoán các bệnh lý như Alzheimer, đặc biệt khi dữ liệu có sự tương tác phức tạp giữa các đặc trưng.

## Feature Engineering

Feature Engineering là một bước quan trọng trong quy trình xử lý dữ liệu, giúp cải thiện hiệu suất của mô hình học máy bằng cách chuyển đổi hoặc tạo ra các đặc trưng quan trọng từ dữ liệu thô. Dưới đây là các kỹ thuật thường được sử dụng

## MaxAbsScalar

MaxAbsScaler là một kỹ thuật chuẩn hóa đặc trưng, biến đổi các giá trị trong tập dữ liệu về khoảng [-1, 1] bằng cách chia mỗi giá trị cho giá trị tuyệt đối lớn nhất trong cột tương ứng.

Ưu điểm:

* Bảo toàn dạng thưa (sparse) của dữ liệu, phù hợp với dữ liệu có nhiều số 0.
* Đơn giản, hiệu quả với dữ liệu có giá trị lớn nhưng không lệch quá nhiều.

Nhược điểm:

* Không xử lý tốt khi dữ liệu chứa nhiễu hoặc các ngoại lệ.

## Principal Component Analysis

PCA là một kỹ thuật giảm chiều dữ liệu, chuyển đổi các đặc trưng ban đầu thành các thành phần chính (principal components), đảm bảo giữ lại phần lớn thông tin của dữ liệu gốc.

Ưu điểm:

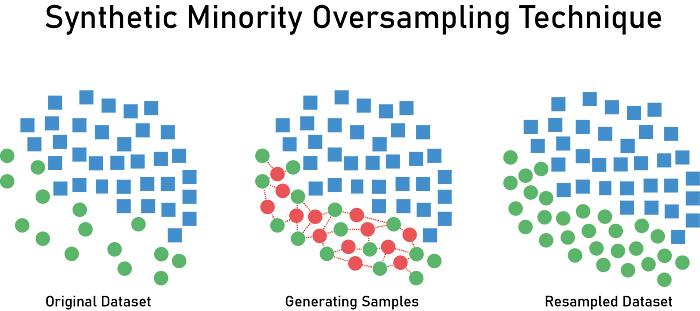
* Giảm chiều dữ liệu, tăng tốc độ huấn luyện mô hình.
* Loại bỏ mối quan hệ tương quan giữa các đặc trưng.

Nhược điểm:

* Khó diễn giải ý nghĩa của các thành phần chính.
* Mất thông tin khi giảm chiều dữ liệu quá mức.

## Synthetic Minority Over-sampling Technique (SMOTE)

SMOTE là một kỹ thuật xử lý dữ liệu mất cân bằng bằng cách tạo thêm các mẫu giả lập cho lớp thiểu số. Phương pháp này sử dụng nội suy giữa các điểm dữ liệu hiện có để sinh ra các mẫu mới.



Hình 2.1: Synthetic Minority Oversampling Techique

Ưu điểm:

* Cân bằng dữ liệu hiệu quả mà không làm mất dữ liệu ban đầu.
* Tăng cường khả năng học của mô hình trên lớp thiểu số.

Nhược điểm:

* Có thể sinh ra các mẫu không tự nhiên, dẫn đến giảm tính chính xác.
* Không xử lý được sự chồng lấn giữa các lớp dữ liệu.

## HyperParameter Tuning

Tinh chỉnh siêu tham số là quá trình điều chỉnh các tham số của mô hình học máy (siêu tham số) để tối ưu hóa hiệu suất của mô hình. Siêu tham số không được học trực tiếp từ dữ liệu mà được thiết lập trước khi huấn luyện mô hình, như số lượng cây trong rừng ngẫu nhiên, tỷ lệ học trong mô hình mạng nơ-ron, hoặc độ sâu của cây quyết định.

Phương pháp:

* **Grid Search**: Kiểm tra tất cả các kết hợp có thể của siêu tham số trong một phạm vi đã định.
* **Random Search**: Chọn ngẫu nhiên các giá trị từ một phạm vi và kiểm tra chúng, giúp giảm thời gian tính toán.
* **Bayesian Optimization**: Sử dụng các mô hình thống kê để tối ưu hóa siêu tham số, thay vì thử tất cả các kết hợp có thể.

Ưu điểm:

* Giúp tối ưu hóa hiệu suất mô hình.
* Cải thiện khả năng dự đoán của mô hình.

Nhược điểm:

* Tốn thời gian và tài nguyên tính toán, đặc biệt đối với mô hình phức tạp.
* Không phải lúc nào cũng dễ dàng xác định phạm vi siêu tham số hợp lý.

## Cross Validation

Kiểm định chéo là một kỹ thuật đánh giá mô hình học máy nhằm giảm thiểu vấn đề overfitting và tăng tính tổng quát của mô hình. Quá trình kiểm định chéo chia dữ liệu thành nhiều phần (folds), mỗi phần sẽ lần lượt được sử dụng làm dữ liệu kiểm tra trong khi các phần còn lại được sử dụng làm dữ liệu huấn luyện.

Phương pháp:

* **K-fold Cross Validation:** Dữ liệu được chia thành K phần, và mô hình được huấn luyện và đánh giá K lần, mỗi lần sử dụng một phần khác nhau làm dữ liệu kiểm tra.
* **Leave-One-Out Cross Validation (LOOCV):** Mỗi quan sát trong dữ liệu sẽ được sử dụng làm dữ liệu kiểm tra một lần duy nhất, và phần còn lại làm dữ liệu huấn luyện.
* **Stratified K-fold Cross Validation:** Phân chia dữ liệu sao cho tỷ lệ các lớp trong mỗi fold giống với tỷ lệ của toàn bộ tập dữ liệu.

Ưu điểm:

* Cung cấp một ước tính chính xác hơn về hiệu suất mô hình.
* Giảm thiểu bias và overfitting bằng cách sử dụng tất cả dữ liệu để huấn luyện và kiểm tra.

Nhược điểm:

* Tốn thời gian tính toán, đặc biệt với các mô hình phức tạp hoặc tập dữ liệu lớn.
* Các phương pháp kiểm định chéo có thể gây ra sự chồng chéo nếu dữ liệu không được chia đúng cách.

## Machine Learning

### Support Vector Machine (SVM)

1. Nền tảng lý thuyết

Support Vector Machine (SVM) là một thuật toán học máy có giám sát, được sử dụng phổ biến trong các bài toán phân loại và hồi quy. SVM được áp dụng để tìm ra siêu phẳng (hyperplane) tối ưu phân tách các lớp khác nhau trong không gian đặc trưng.

SVM hoạt động dựa trên nguyên tắc:

* Tìm kiếm siêu phẳng tối ưu có khoảng cách lớn nhất (maximum margin) đến các điểm dữ liệu gần nhất của mỗi lớp, gọi là các vector hỗ trợ (support vectors).
* Với các bài toán không thể phân tách tuyến tính, SVM sử dụng hàm kernel để ánh xạ dữ liệu từ không gian ban đầu sang không gian có chiều cao hơn, nơi dữ liệu có thể được phân tách tuyến tính.

1. Công thức tính toán

- Siêu phẳng phân tách tuyến tính:

Một siêu phẳng trong không gian d-chiều có thể được biểu diễn bởi phương trình:

w⋅x+b=0

* w: vector trọng số (weight vector).
* x: vector đặc trưng của dữ liệu.
* b: hằng số (bias).

- Hàm mục tiêu:

Khoảng cách giữa các siêu phẳng và các vector hỗ trợ là:

SVM tối ưu hóa bài toán:

Với điều kiện ràng buộc:

* : nhãn (label) của dữ liệu (+1 hoặc -1).
* : vector đặc trưng của dữ liệu.

c, Hàm Kernal

Khi dữ liệu không thể phân tách tuyến tính, SVM sử dụng hàm kernel để ánh xạ dữ liệu sang chiều không gian cao hơn. Một số hàm kernel phổ biến:

* Linear kernel
* Polynomial kernel
* RBF kernel (Gaussian)

1. Ưu điểm

* Tính tổng quát cao: Nhờ tối ưu hóa khoảng cách giữa các lớp, SVM thường đạt hiệu suất tốt trên tập dữ liệu chưa thấy.
* Hiệu quả trên dữ liệu phi tuyến: Hàm kernel cho phép xử lý các bài toán phức tạp, phi tuyến.
* Khả năng xử lý tốt với dữ liệu chiều cao: Thích hợp cho dữ liệu văn bản, vốn thường có số chiều rất lớn.

1. Nhược điểm

* Chi phí tính toán cao: SVM không hiệu quả trên các tập dữ liệu lớn do thời gian tính toán tăng nhanh với kích thước dữ liệu.
* Khó chọn tham số và kernel: Cần thử nghiệm nhiều để chọn được kernel và các siêu tham số (C, γ) phù hợp.
* Không hỗ trợ phân loại xác suất trực tiếp: SVM chỉ cho ra kết quả phân lớp, không đưa ra xác suất dự đoán.

### Naive Bayes

1. Nền tảng lý thuyết

Naive Bayes (NB) là một thuật toán học máy có giám sát, dựa trên Định lý Bayes và giả định rằng các đặc trưng đầu vào là độc lập với nhau (giả định "naive" - đơn giản). Dù giả định này không phải lúc nào cũng chính xác, nhưng NB vẫn thường đạt hiệu quả tốt, đặc biệt trong các bài toán phân loại.

1. Công thức tính toán

- Giả định độc lập có điều kiện:

NB giả định ràng tất cả các đặc trưng , , …, trong tập đặc trưng X là độc lập:

P(X|C) = P(|C) + P(|C) + … + P(|C)

Phương trình phân loại trở thành:

C = arg max(C)

- Tính xác suất:

* Xác suất tiên nghiệm P(C): Tỷ lệ xuất hiện của mỗi lớp trong tập huấn luyện.
* Xác suất có điều kiện : Tỷ lệ xuất hiện của đặc trưng trong các văn bản thuộc lớp C.

- NB trong phân loại:

* Multinomial Naive Bayes: Dùng cho dữ liệu rời rạc.
* Bernoulli Naive Bayes: Dùng cho dữ liệu nhị phân.
* Gaussian Naive Bayes: Dùng cho dữ liệu liên tục.

1. Ưu điểm

* Tốc độ nhanh: Naive Bayes rất nhẹ, phù hợp với tập dữ liệu lớn.
* Hiệu quả tốt trên dữ liệu nhiều chiều: Thích hợp cho dữ liệu văn bản, nơi mỗi từ là một đặc trưng.
* Không đòi hỏi nhiều siêu tham số: NB không cần tinh chỉnh tham số phức tạp.
* Khả năng mở rộng: NB có thể được mở rộng với các dạng phân phối khác nhau (Multinomial, Gaussian).

1. Nhược điểm

Giả định độc lập không thực tế: Các từ trong văn bản thường có liên quan với nhau, điều này có thể làm giảm độ chính xác.

Không linh hoạt với dữ liệu phức tạp: NB không thể nắm bắt tốt các mối quan hệ phức tạp giữa các đặc trưng.

Không xử lý tốt dữ liệu mất cân bằng: Nếu một lớp xuất hiện quá ít, NB có thể không dự đoán tốt lớp đó.

### Logistic Regression

1. Nền tảng lý thuyết

Logistic Regression là một thuật toán học máy có giám sát thường được sử dụng cho bài toán phân loại nhị phân. Tuy nhiên, nó có thể được mở rộng để xử lý các bài toán phân loại đa nhãn (k>2) thông qua hai phương pháp chính:

* One-vs-Rest (OvR): Huấn luyện k mô hình logistic regression, mỗi mô hình phân biệt một lớp với các lớp còn lại.
* Softmax Regression (Multinomial Logistic Regression): Mở rộng logistic regression để dự đoán trực tiếp xác suất của từng lớp bằng hàm softmax.

1. Công thức tính toán

- Softmax Regression:

P(y = i|x) = ∀i∈{1,2,...,k}

Hàm softmax đảm bảo rằng:

* P(y = i|x) ∈ [0, 1].
* Tổng tất cả xác suất các nhãn bằng 1.

1. Ưu điểm

* Đơn giản và dễ hiểu: Logistic Regression dễ cài đặt và giải thích kết quả.
* Tính toán nhanh: Phù hợp với tập dữ liệu nhỏ đến trung bình.
* Khả năng mở rộng: Dễ dàng áp dụng cho bài toán đa nhãn thông qua OvR hoặc softmax.

d, Nhược điểm

* Hạn chế với dữ liệu phi tuyến: Logistic Regression chỉ hoạt động tốt khi dữ liệu có thể phân tách tuyến tính trong không gian đặc trưng.
* Không phù hợp với số lượng lớn đặc trưng: Khi dữ liệu có nhiều chiều, Logistic Regression có thể bị overfitting nếu không có kỹ thuật điều chuẩn.
* Phụ thuộc vào tiền xử lý: Hiệu quả của Logistic Regression phụ thuộc lớn vào cách trích xuất và biểu diễn đặc trưng.

### XGBoost

1. Nền tảng lý thuyết

XGBoost (Extreme Gradient Boosting), được thiết kế cho bài toán phân loại. Đây là một thuật toán học máy mạnh mẽ dựa trên cây quyết định, áp dụng kỹ thuật boosting để cải thiện hiệu suất của mô hình.

**1. Boosting**

Boosting là một kỹ thuật ensemble học máy, nơi nhiều mô hình yếu (weak learners) được huấn luyện tuần tự, mỗi mô hình cố gắng sửa lỗi của mô hình trước đó. Mục tiêu là kết hợp các mô hình yếu để tạo ra một mô hình mạnh với hiệu suất cao hơn.

**2. Gradient Boosting**

Gradient Boosting tối ưu hóa mô hình thông qua việc giảm thiểu hàm mất mát bằng cách xây dựng tuần tự các cây quyết định, sử dụng đạo hàm gradient để hướng dẫn.

**3. XGBoost**

XGBoost là một cải tiến của Gradient Boosting, được tối ưu hóa về tốc độ và hiệu suất. Một số đặc điểm nổi bật:

* Tối ưu hóa hiệu suất: Tích hợp xử lý song song và quản lý bộ nhớ hiệu quả.
* Hàm mất mát tùy chỉnh: Hỗ trợ hàm mất mát log-loss cho bài toán phân loại.
* Regularization (L1 và L2): Giảm overfitting thông qua điều chuẩn trọng số.
* Xử lý dữ liệu thiếu: Tự động xử lý các giá trị bị thiếu trong dữ liệu.

1. Công thức tính toán

Các tham số quan trọng:

* max\_depth: Độ sâu tối đa của mỗi cây quyết định (quyết định khả năng phân chia dữ liệu).
* learning\_rate: Tốc độ học, xác định mức độ điều chỉnh mô hình qua mỗi cây.
* n\_estimators: Số lượng cây quyết định trong rừng.
* subsample: Tỷ lệ mẫu dữ liệu sử dụng cho mỗi cây (để giảm overfitting).
* colsample\_bytree: Tỷ lệ cột được chọn cho mỗi cây.
* objective: Hàm mục tiêu, thường là "binary: logistic" cho phân loại nhị phân hoặc "multi: softprob" cho đa nhãn.
* reg\_lambda và reg\_alpha: Điều chuẩn L2 và L1.

1. Ưu điểm

* Hiệu suất cao: XGBoost thường đạt kết quả tốt trên nhiều bài toán thực tế.
* Khả năng tổng quát hóa tốt: Điều chỉnh regularization giúp giảm overfitting.
* Tùy biến cao: Hỗ trợ nhiều tham số và hàm mất mát tùy chỉnh.
* Xử lý dữ liệu thiếu: Không yêu cầu tiền xử lý dữ liệu thiếu.

d, Nhược điểm

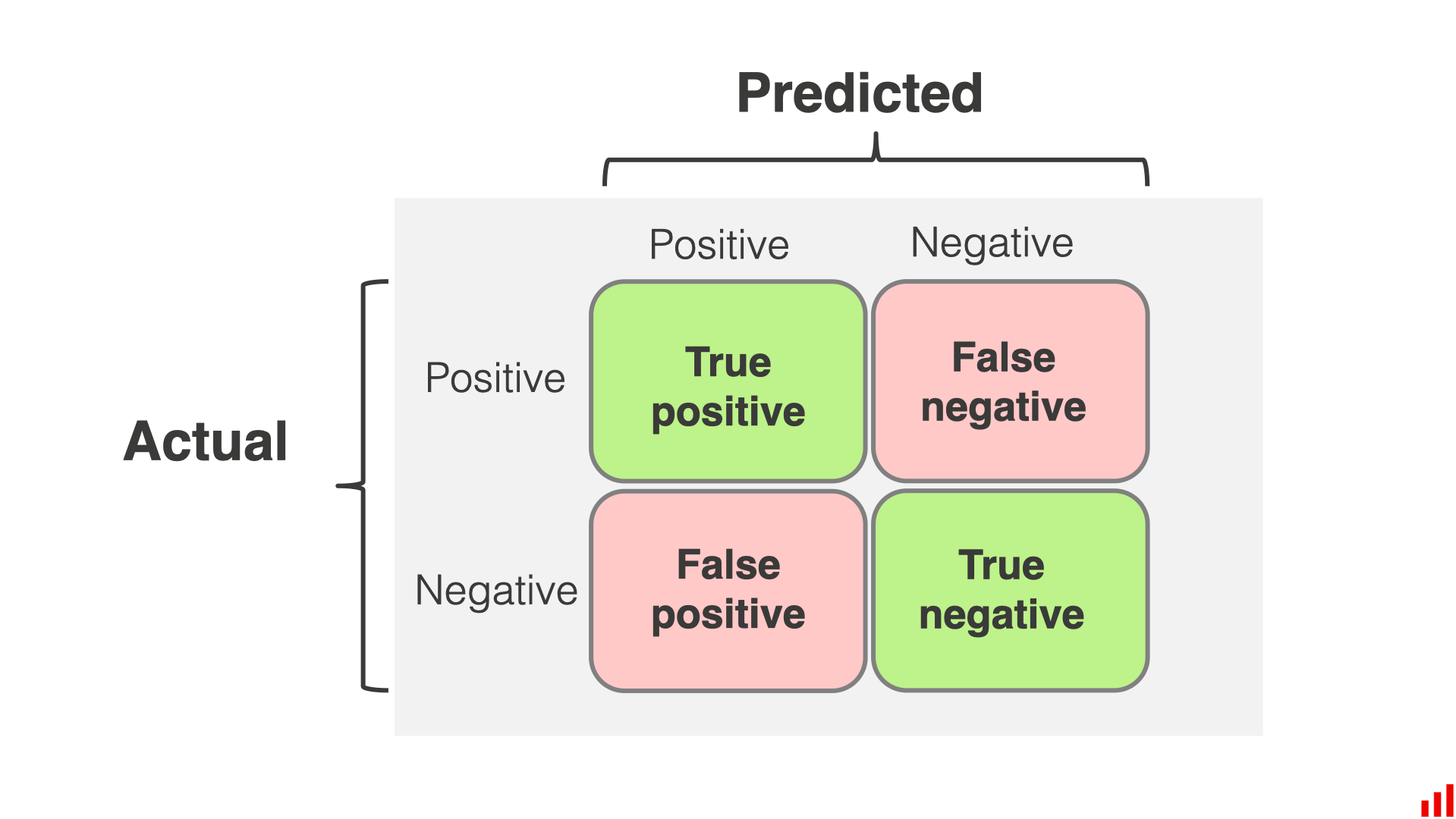
* Chi phí tính toán cao: Việc huấn luyện có thể mất nhiều thời gian, đặc biệt với dữ liệu lớn.
* Nhạy cảm với siêu tham số: Cần tinh chỉnh tham số (tuning) để đạt hiệu suất tối ưu.
* Không phù hợp cho dữ liệu ít: XGBoost cần một lượng dữ liệu đáng kể để phát huy hiệu quả.

## Confusion Matrix

Confusion Matrix (Ma trận nhầm lẫn) là một công cụ đánh giá mô hình phân loại, giúp so sánh giữa các dự đoán của mô hình và các giá trị thực tế trong tập kiểm tra. Ma trận này cung cấp cái nhìn tổng quan về hiệu suất của mô hình, từ đó giúp chúng ta nhận diện được các lỗi mà mô hình mắc phải.

Ma trận nhầm lẫn cho một bài toán phân loại nhị phân sẽ có 4 ô:

* True Positive (TP): Số lượng mẫu mà mô hình dự đoán đúng là lớp dương (positive).
* False Positive (FP): Số lượng mẫu mà mô hình dự đoán sai là lớp dương, trong khi thực tế là lớp âm (negative).
* True Negative (TN): Số lượng mẫu mà mô hình dự đoán đúng là lớp âm.
* False Negative (FN): Số lượng mẫu mà mô hình dự đoán sai là lớp âm, trong khi thực tế là lớp dương.



Hình 2.9: Confusion matrix

**Ưu điểm**:

* Giúp phân tích chi tiết các loại lỗi mà mô hình mắc phải (ví dụ: sai thành lớp dương hay lớp âm).
* Cung cấp cái nhìn rõ ràng về hiệu suất của mô hình trong từng lớp.

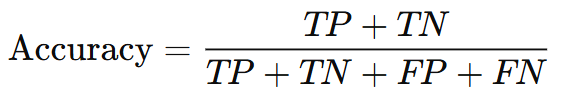
**Nhược điểm**:

* Không cung cấp thông tin về các lỗi liên quan đến độ chính xác tổng thể của mô hình, mà chỉ tập trung vào các phân loại chính xác và sai lệch giữa các lớp.

## Evaluated Metrics

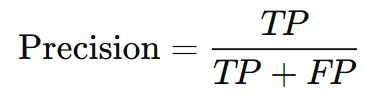
Các **metrics đánh giá** giúp chúng ta đo lường hiệu suất của mô hình từ nhiều góc độ khác nhau. Dưới đây là một số metric quan trọng trong việc đánh giá mô hình phân loại:

1. **Accuracy (Độ chính xác)**:
   * **Định nghĩa**: Là tỷ lệ giữa số mẫu được phân loại đúng trên tổng số mẫu.
   * **Công thức**:



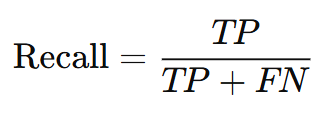
* + **Ưu điểm**: Dễ hiểu và tính toán nhanh.
  + **Nhược điểm**: Không hiệu quả khi dữ liệu mất cân bằng (class imbalance), vì mô hình có thể đạt độ chính xác cao bằng cách dự đoán đúng lớp chiếm ưu thế.

1. **Precision (Độ chính xác)**
   * **Định nghĩa**: Là tỷ lệ giữa số mẫu dự đoán đúng là lớp dương so với tổng số mẫu được mô hình dự đoán là lớp dương.
   * **Công thức**:



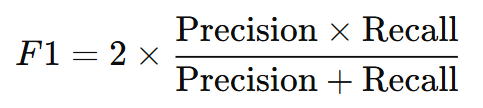
* + **Ưu điểm**: Chỉ ra được độ chính xác của mô hình khi dự đoán lớp dương.
  + **Nhược điểm**: Nếu lớp dương hiếm, precision có thể không phản ánh được khả năng mô hình phân biệt tốt giữa các lớp.

1. **Recall (Độ nhạy)**
   * **Định nghĩa**: Là tỷ lệ giữa số mẫu dự đoán đúng là lớp dương so với tổng số mẫu thực sự thuộc lớp dương.
   * **Công thức**:



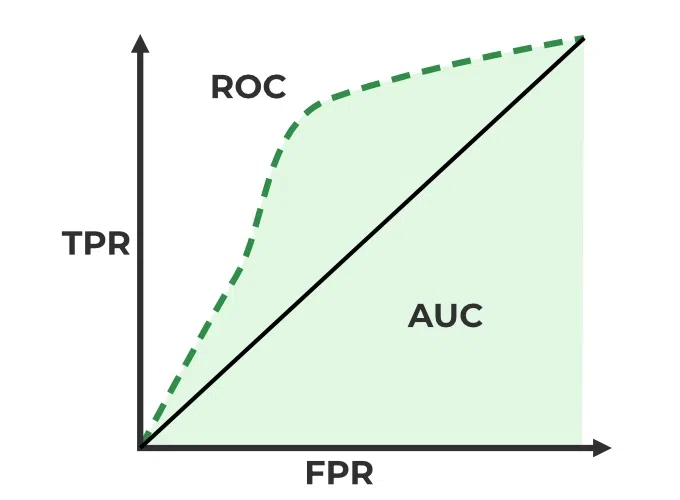
* + **Ưu điểm**: Giúp đánh giá khả năng của mô hình trong việc nhận diện tất cả các mẫu dương.
  + **Nhược điểm**: Một mô hình có recall cao có thể dẫn đến nhiều dự đoán sai (false positives).

1. **F1-Score**
   * **Định nghĩa**: F1-Score là trung bình hài hòa của Precision và Recall. Nó giúp cân bằng giữa độ chính xác và độ nhạy, đặc biệt hữu ích khi dữ liệu bị mất cân bằng.
   * **Công thức**:



* + **Ưu điểm**: Cung cấp cái nhìn cân bằng về độ chính xác và độ nhạy, đặc biệt trong các bài toán phân loại mất cân bằng.
  + **Nhược điểm**: Không phải lúc nào cũng cung cấp thông tin đầy đủ nếu không có sự cân nhắc về các yếu tố khác như Accuracy.

## ROC-AUC

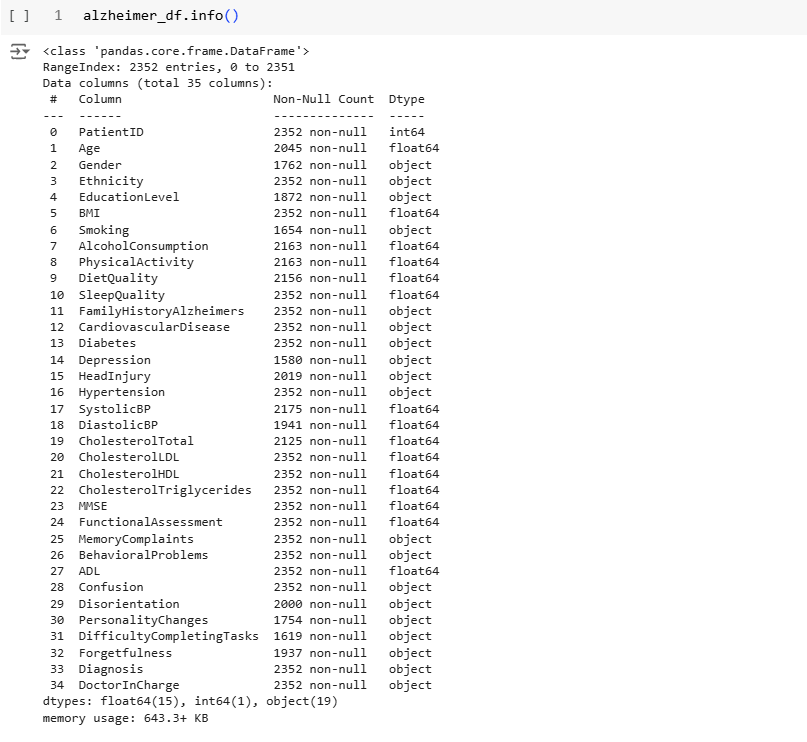


* + **Định nghĩa**: **Receiver Operating Characteristic (ROC)** là một biểu đồ thể hiện mối quan hệ giữa Tỷ lệ dương giả (False Positive Rate - FPR) và Tỷ lệ dương thật (True Positive Rate - TPR) ở các ngưỡng khác nhau. **AUC (Area Under the Curve)** là diện tích dưới đường cong ROC, đại diện cho khả năng phân biệt của mô hình.
  + **Ưu điểm**: Phản ánh được khả năng phân loại tổng thể của mô hình trên toàn bộ các ngưỡng phân loại.
  + **Nhược điểm**: Cần mô hình có độ phân giải tốt và phụ thuộc vào việc điều chỉnh các ngưỡng.

# SOLUTIONS

## Mã nguồn tiền xử lý dữ liệu

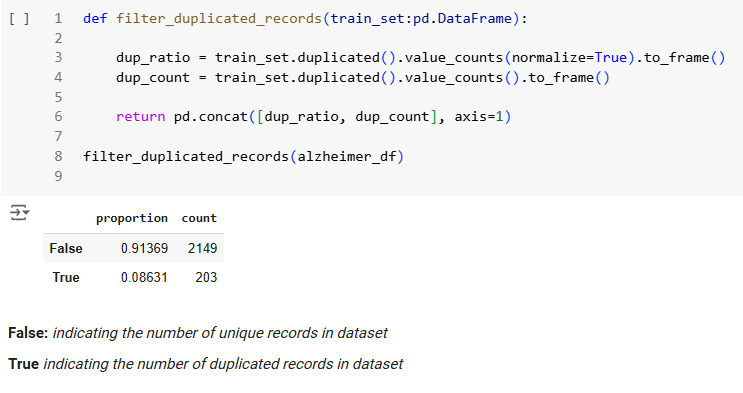
Đầu tiên em sử dụng hàm Alzheimer\_df.info() để kiểm tra các đặc trưng nào bị thiếu và có kiểu dữ liệu thế nào. Qua đó có thể biết thêm 1 số thông tin về kích thước của bộ dữ liệu, tên của các đặc trưng đó sẽ như nào, kiểu dữ liệu đã chuẩn hóa đúng chưa.



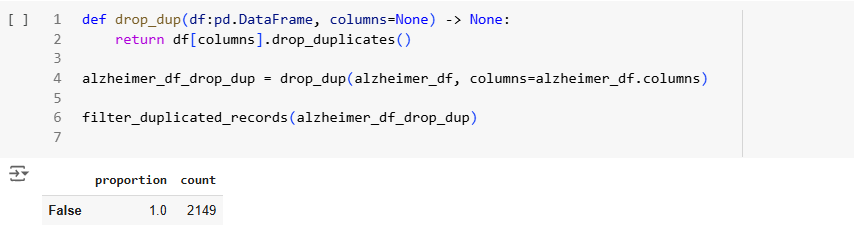
1. Xóa các bản ghi trùng lặp

Em sử dụng hàm filter\_duplicated\_records() để xuất ra tỉ lệ lặp lại của các bản ghi trong bộ dữ liệu này, như dưới hình có thể thấy, tỉ lệ các bản ghi trùng lặp hay giống y đúc nhau vào khoảng 0.08631 phần trăm, số lượng cụ thể là 203 bản ghi.

Các bản ghi trùng lặp sẽ bị xóa bỏ để tránh dư thừa dữ liệu cho quá trình huấn luyện mô hình và tiền xử lý dữ liệu.

Hàm bên dưới đây sẽ xóa tất cả các bản ghi trùng lặp trong DataFrame thông qua phương thức drop\_duplicated() được hỗ trợ trong thư viện pandas.

Sau khi gọi hàm và kiểm tra lại có thể thấy dữ liệu trùng lặp đã được loại bỏ.

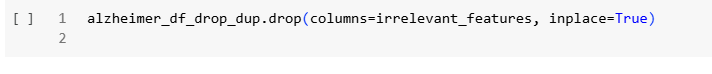


1. Xóa các cột không quan trọng trong bộ dữ liệu

Các cột bên dưới đây bao gồm “DoctorInCharge” and “PatientID” đều không quan trọng trong quá trình phân tích và huấn luyện mô hình học máy cho dữ liệu nên sẽ bị loại bỏ.

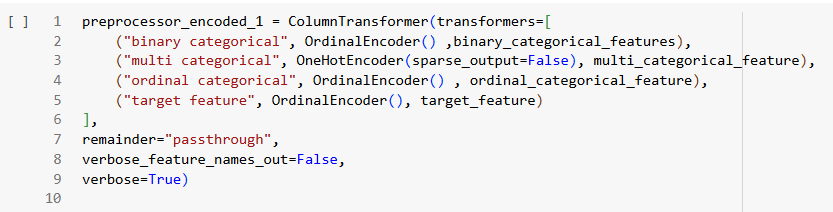


Ta sẽ loại bỏ các cột hay các đặc trưng này bằng cách sử dụng phương thức drop() được hỗ trợ trong thư viện pandas.



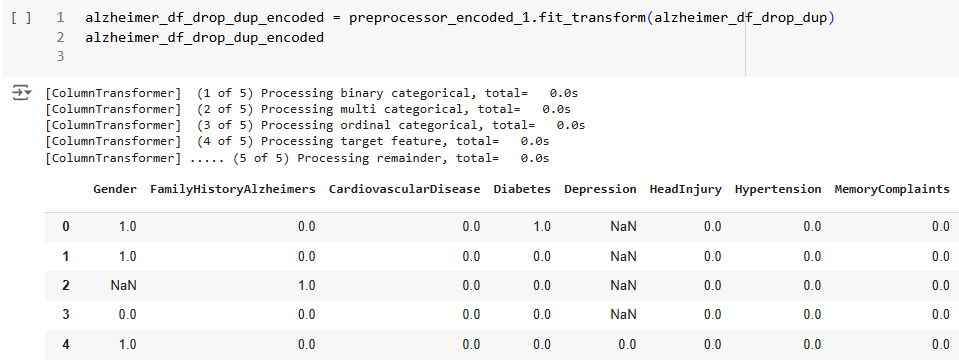
1. Mã hóa các cột chứa dữ liệu văn bản

Ta sẽ mã hóa các giá trị văn bản bằng cách sử dụng ColumnTransformer(), 1 công cụ cho phép thay đổi, mã hóa dữ liệu dạng văn bản xang dạng số 1 cách linh hoạt và tối ưu nhất. Như hình bên dưới em có định nghĩa ColumnsTransformer() cộng thêm danh sách các cột các kiểu mã hóa khác nhau cho từng loại đặc trưng, với kiểu dữ liệu và tính chất khác nhau.



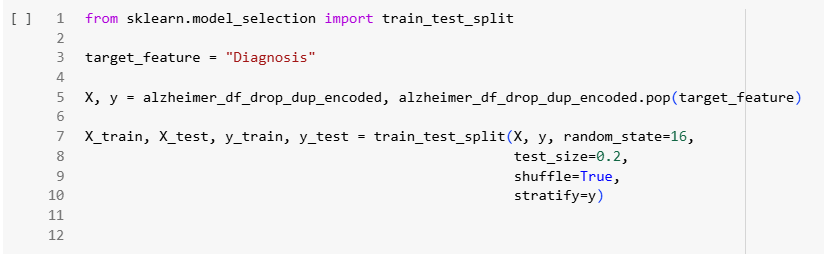
Quá trình bên dưới mô tả cách áp dụng hàm và kết quả đầu ra của dữ liệu ban đầu

Đã được mã hóa, không còn các dữ liệu văn bản nữa.



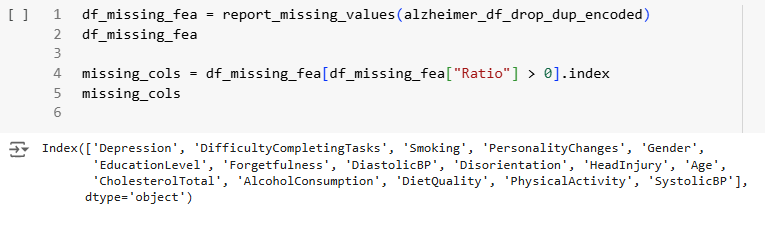
1. Phân chia dữ liệu

Em sẽ sử dụng hàm train\_test\_split được hỗ trợ trong sklearn để có thể chưa dữ liệu ra làm 2 bộ, bộ train và bộ test với tỉ lệ train:test là 80:20. Ta sẽ sử dụng Stratified sampling, hay chọn mẫu phân tổ để có thể chia đều label cho từng tập



1. Xử lý dữ liệu mất mát

Các cột bên dưới được in ra dưới màn hình là những cột đều mang các giá trị thiếu Các cột này sẽ cần phải được thêm dữ liệu vào những chỗ trống để có thể phù hợp cho mô hình huấn luyện và phân tích.

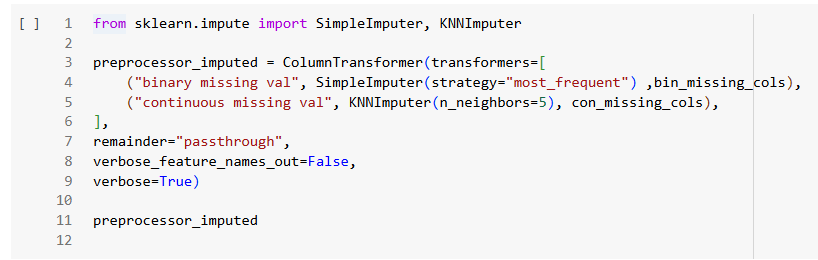


Để có thể thêm những dữ liệu mới vào chỗ trống ta sẽ sử dụng phương pháp

Thế trong thư viện sklearn.impute. Hình bên dưới sử dụng SimpleImputer() and

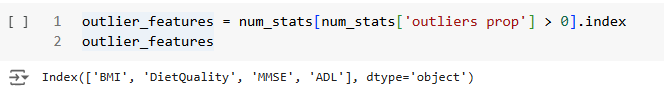
KNNImputer(), cả 2 phương pháp đều có những cách triển khai khác nhau.

Chúng ta sẽ sử dụng 2 phương pháp này kết hợp với những kiểu dữ liệu khác nhau.



1. Xử lý dữ liệu ngoại lai

Bên dưới là danh sách các cột có khả năng cao sẽ chứa những dữ liệu ngoại lai hay dữ liệu nhiễu, ta cần phải tìm cách loại bỏ chúng để tối thiểu sai số cho mô hình và quá trình phân tích.

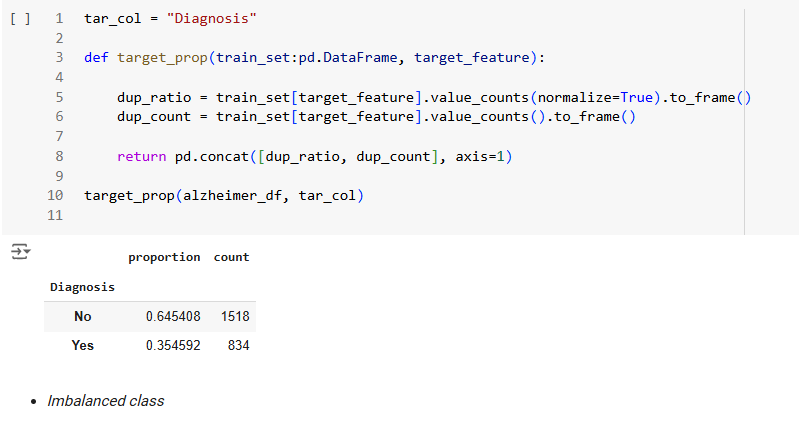


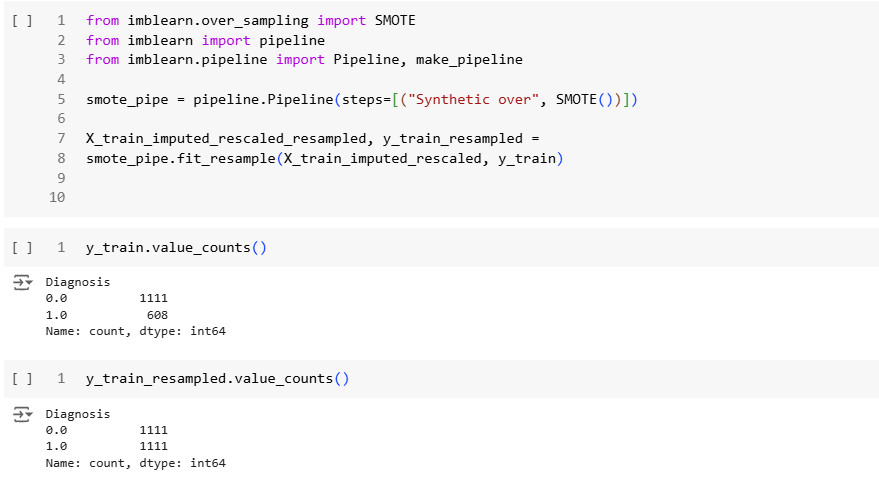


Ta sẽ loại bỏ dữ liệu ngoại lai, hay nhiễu bằng cách sử dụng RobustScalar() cho những cột chứa dữ liệu ngoại lai này.

1. Xử lý mất cân bằng nhãn

Đầu tiên em sử dụng hàm target\_prop() để hiển thị tỉ lệ mất cân bằng của nhãn trong dữ liệu. Cột nhãn mà ta đang xét là “Diagnosis” và tỉ lệ mất cân bằng của nhãn rơi vào khoảng 64:35, tỉ lệ khá lệch về nhãn No. Nhưng mà đây là vấn đề thường thấy ở những bộ dữ liệu liên quan đến y tế, là điều không thể tránh khỏi.

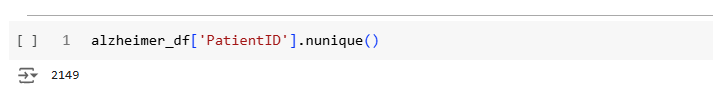




Ta sử dụng SMOTE, một kĩ thuật sử dụng để lấy mẫu lại cho class có số lượng các bản ghi thấp hơn. Ta sử dụng SMOTE và kết quả cho thấy các nhãn đã được cân bằng về mức 50:50.

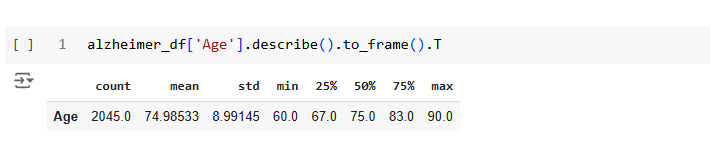
## Mã nguồn thống kê dữ liệu

1. Đếm tổng số bệnh nhân



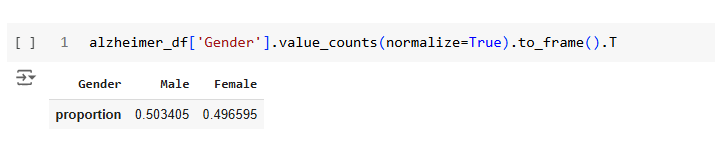
* Mục đích: Xác định tổng số bệnh nhân trong bộ dữ liệu.
* Insight: Xác định quy mô mẫu để đánh giá độ tin cậy của kết quả. Số lượng mẫu lớn giúp tăng độ chính xác khi xây dựng mô hình chẩn đoán.

1. Phân phối tuổi bệnh nhân



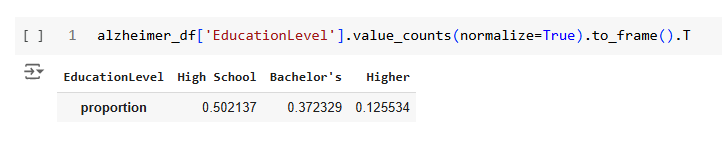
* Mục đích: Tóm tắt thống kê (giá trị trung bình, min, max) về tuổi.
* Insight: Phát hiện nhóm tuổi có nguy cơ cao. Nếu phần lớn bệnh nhân thuộc nhóm trên 65 tuổi, việc tập trung chẩn đoán ở nhóm này sẽ hiệu quả hơn.

1. Phân phối giới tính



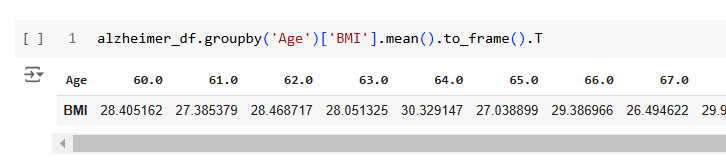
* Mục đích: Tính tỉ lệ nam và nữ trong dữ liệu.
* Insight: Nếu nữ giới chiếm tỷ lệ cao, các biện pháp dự phòng và chẩn đoán sớm cần ưu tiên nhóm này.

1. Phân tích trình độ học vấn



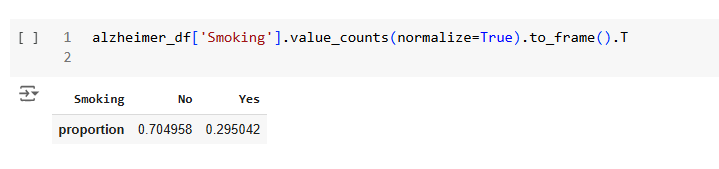
* Mục đích: Xem phân phối trình độ học vấn.
* Insight: Phát hiện mối liên hệ giữa học vấn và nguy cơ mắc bệnh. Trình độ thấp có thể liên quan đến tỷ lệ mắc Alzheimer cao hơn.

1. BMI trung bình theo nhóm tuổi



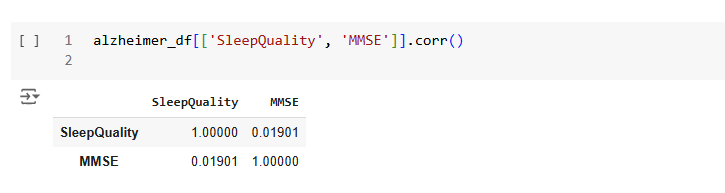
* Mục đích: Tìm hiểu xu hướng BMI theo độ tuổi.
* Insight: BMI bất thường ở nhóm tuổi cao có thể là yếu tố dự đoán Alzheimer, cần xem xét thêm.

1. Tỉ lệ bệnh nhân hút thuốc



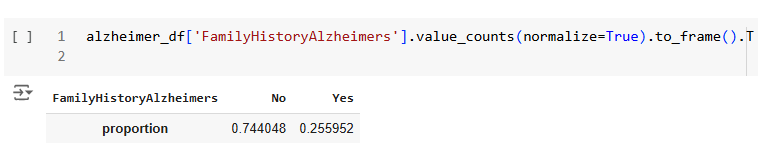
* Mục đích: Tính phần trăm bệnh nhân có thói quen hút thuốc.
* Insight: Hút thuốc làm tăng nguy cơ tổn thương mạch máu não, giúp phát triển mô hình cảnh báo cho nhóm này.

1. Tương quan giữa giấc ngủ và MMSE



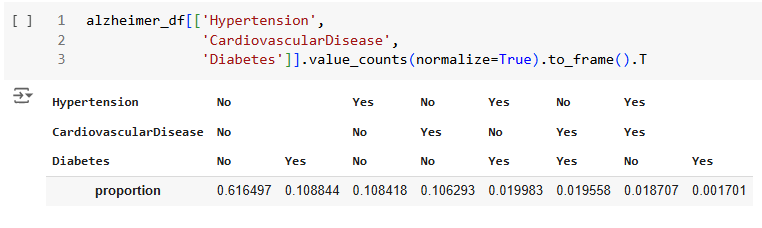
* Mục đích: Đánh giá mối quan hệ giữa chất lượng giấc ngủ và điểm MMSE (Mini-Mental State Examination).
* Insight: Giấc ngủ kém có thể liên quan đến suy giảm nhận thức, là dấu hiệu sớm của Alzheimer.

1. Phân tích tiền sử gia đình



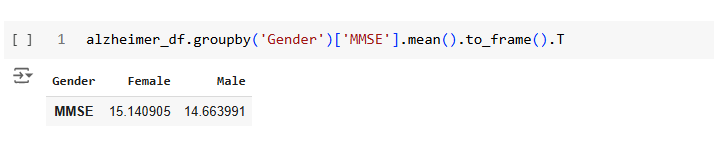
* Mục đích: Xác định tỷ lệ bệnh nhân có tiền sử gia đình mắc Alzheimer.
* Insight: Nhóm này có nguy cơ cao, cần được ưu tiên trong chương trình dự phòng và theo dõi.

1. Phân tích bệnh đồng mắc



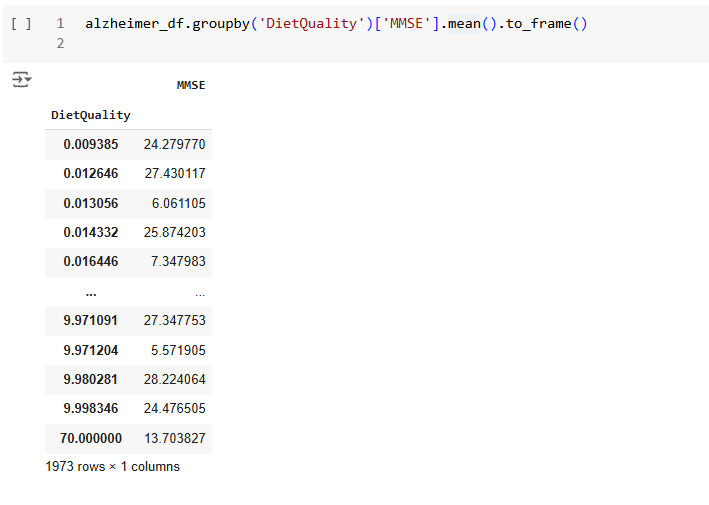
* Mục đích: Tính số bệnh nhân mắc các bệnh đồng mắc phổ biến.
* Insight: Các bệnh này làm tăng nguy cơ Alzheimer, cần kết hợp điều trị đa bệnh lý để giảm nguy cơ.

1. Phân tích điểm MMSE theo giới tính



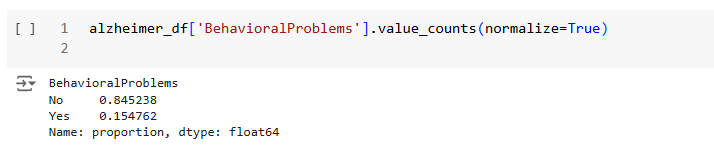
* Mục đích: Tìm hiểu sự khác biệt trong điểm nhận thức giữa nam và nữ.
* Insight: Giúp thiết kế các chương trình chẩn đoán và can thiệp phù hợp với giới tính.

1. Phân tích ảnh hưởng của chất lượng ăn uống



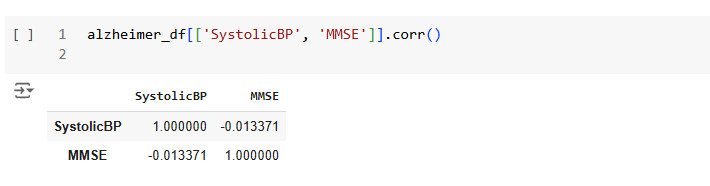
* Mục đích: Đánh giá tác động của chế độ ăn uống đến nhận thức.
* Insight: Chế độ ăn kém có thể liên quan đến suy giảm nhận thức, cần nhấn mạnh vai trò của dinh dưỡng trong dự phòng Alzheimer.

1. Tỷ lệ rối loạn hành vi



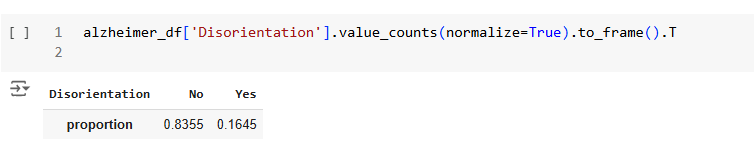
* Mục đích: Xem tỷ lệ bệnh nhân gặp rối loạn hành vi.
* Insight: Những triệu chứng này thường xuất hiện ở giai đoạn giữa và cuối của Alzheimer, hỗ trợ theo dõi mức độ tiến triển.

1. Mối quan hệ giữa huyết áp và trí nhớ



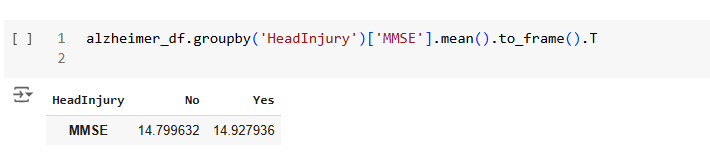
* Mục đích: Đánh giá mối liên hệ giữa huyết áp tâm thu và điểm nhận thức.
* Insight: Huyết áp cao có thể ảnh hưởng tiêu cực đến nhận thức, là yếu tố dự đoán sớm Alzheimer.

1. Tỷ lệ mất phương hướng (Disorientation)



* Mục đích: Phân tích tỷ lệ bệnh nhân gặp tình trạng mất phương hướng.
* Insight: Đây là dấu hiệu lâm sàng quan trọng để đánh giá mức độ nghiêm trọng của Alzheimer.

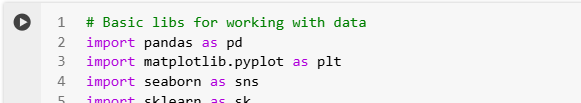
1. Điểm MMSE theo nhóm có và không chấn thương đầu



* Mục đích: So sánh mức độ nhận thức giữa bệnh nhân có và không có tiền sử chấn thương đầu.
* Insight: Chấn thương đầu làm tăng nguy cơ suy giảm nhận thức, cần được xem xét trong các chương trình chẩn đoán.

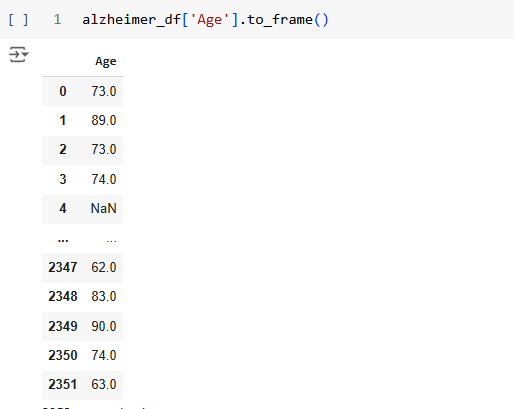
Các thống kê trên sẽ cung cấp góc nhìn đa chiều về dữ liệu và hỗ trợ xác định yếu tố nguy cơ, dấu hiệu sớm và mức độ nghiêm trọng của Alzheimer, từ đó tối ưu hóa quy trình chẩn đoán và can thiệp.

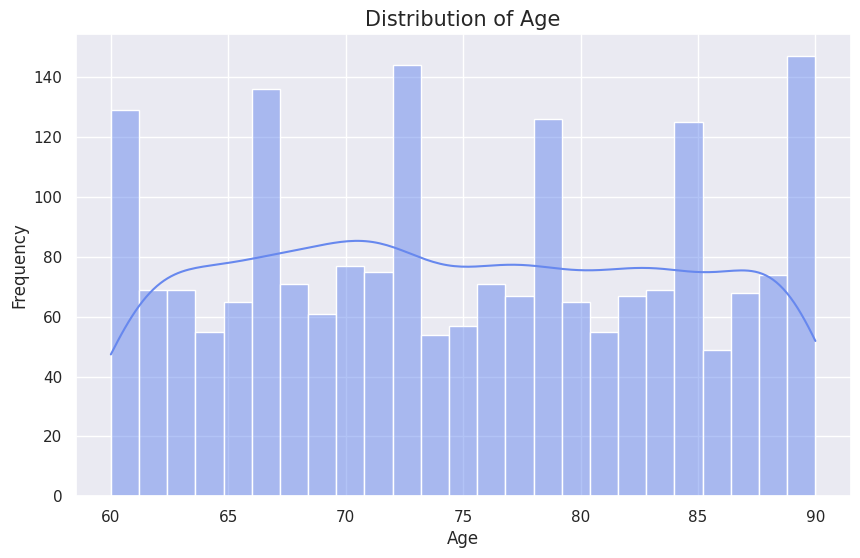
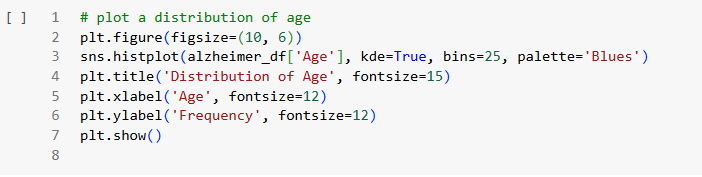
## Mã nguồn trực quan hóa dữ liệu



Khai báo các thư viện cần thiết để trực quan hóa dữ liệu

1. Biểu đồ phân phối tuổi bệnh nhân

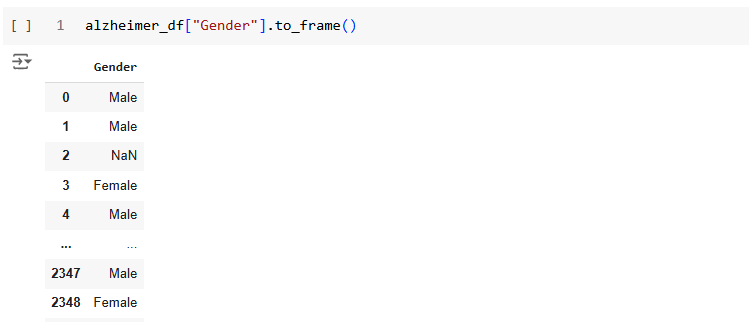


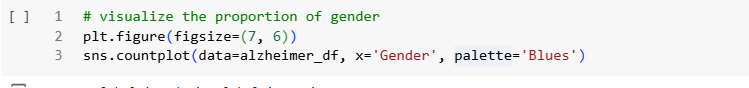


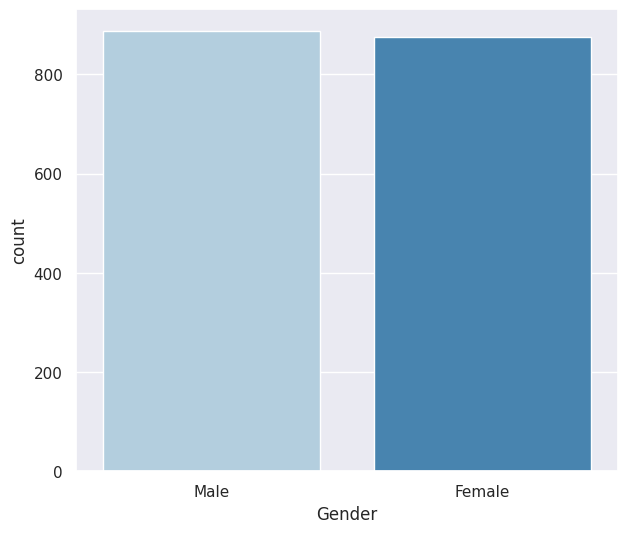
Hình 3.1 Biểu đồ trực quan phân phối dữ liệu tuổi của bệnh nhân

Em sử dụng biểu đồ histogram để có thể trực quan hóa được phân phối của dữ liệu tuổi. Thư viện seaborn hỗ trợ 1 hàm histplot() để có thể làm việc này 1 cách đơn giản hơn. Hàm histplot nhận đầu vào là 1 cột dữ liệu số liên tục từ DataFrame. Sau đó vẽ lên biểu đồ phân phối của dữ liệu đó.

1. Biểu đồ cột tỷ lệ giới tính



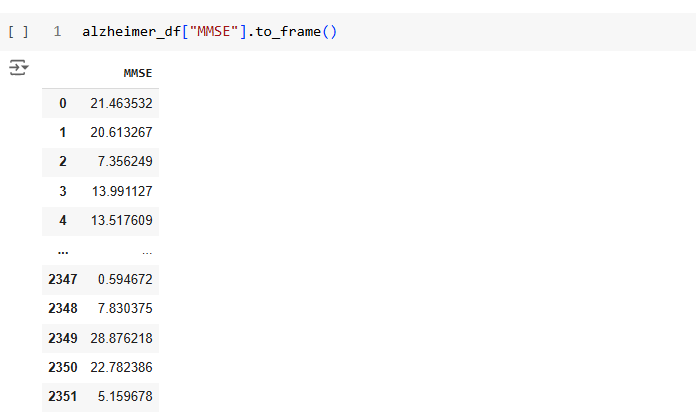


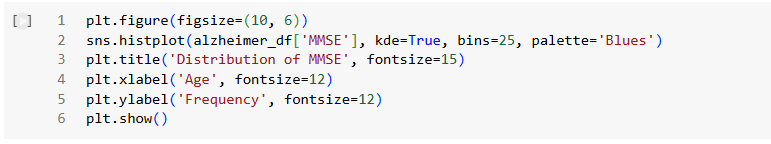


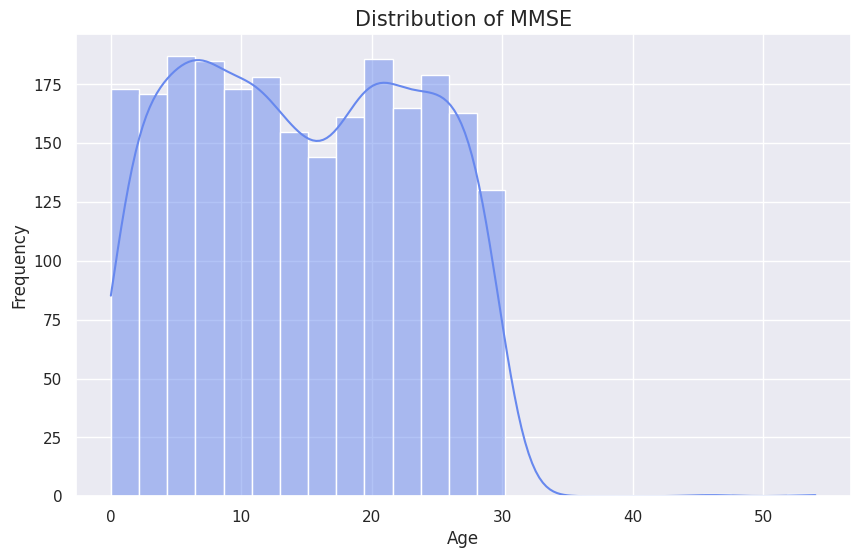
Hình 3.2 Biểu đồ trực quan tỉ lệ dữ liệu giới tính của bệnh nhân

Em sử dụng hàm countplot() được hỗ trợ trong thư viện seaborn để trực quan được dữ liệu tỉ lệ giới tính trong bộ dữ liệu này, Tỉ lệ nam và nữ được thể hiện tương đối cân bằng thông qua 2 cột trên. Điều đó cho thấy việc mắc các căn bệnh về trí nhớ không liên quan nhiều đến việc giới tính của bệnh nhân là nam hay nữ.

1. Biểu đồ phân phối điểm MMSE



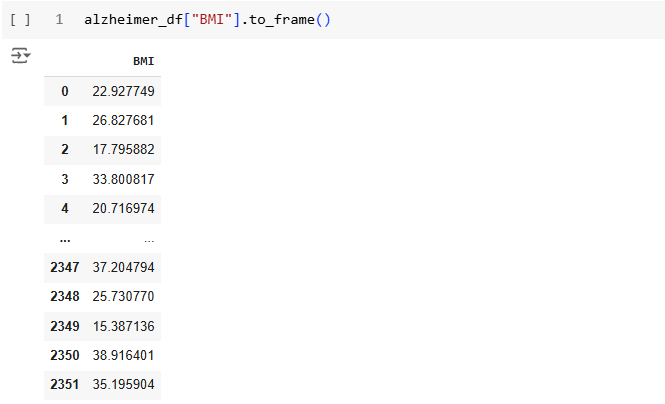


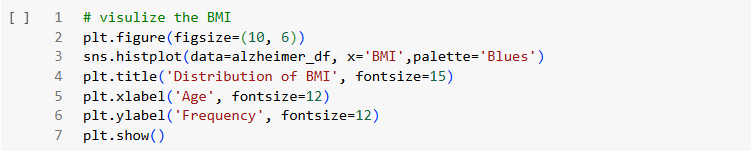


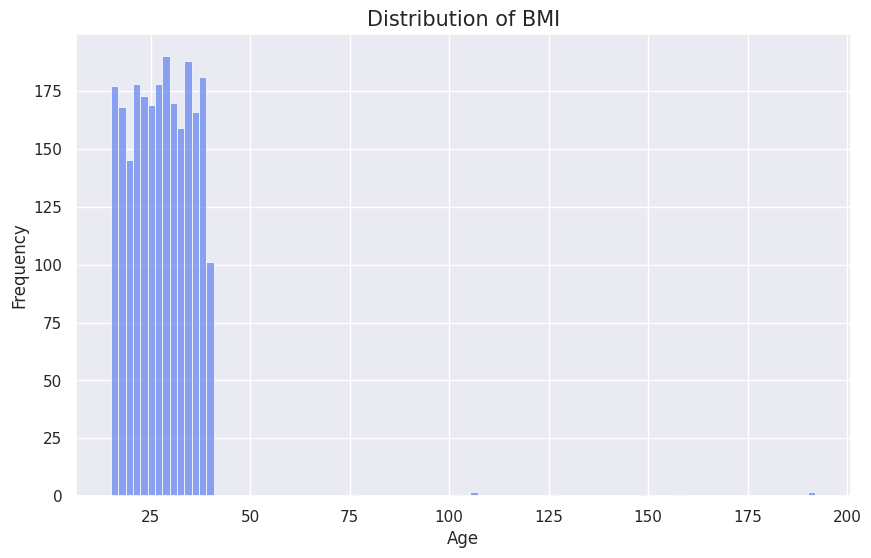
Hình 3.3 Biểu đồ trực quan phân phối dữ liệu MMSE của bệnh nhân

Biểu đồ trên cho thấy phân phối dữ liệu MMSE (Mini mental state examination) của bệnh nhân khá đồng đều ngoại trừ nửa từ 40-50 nhìn có vẻ như nằm ngoài phân phối của dữ liệu, khả năng cao sẽ là nhiễu có thể cần phải loại bỏ về sau.

1. Biểu đồ phân bố BMI







Hình 3.4 Biểu đồ trực quan phân phối dữ liệu BMI của bệnh nhân

Biểu đồ phân bố BMI (Body Mass Index) cũng cho thấy nhiều điểm dữ liệu nhiễu nằm ngoài phân phối tập trung của đa số các điểm dữ liệu còn lại.

# CONCLUSION

# REFERENCES

|  |  |
| --- | --- |
| [1] | Dataset link: *https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset/data* |
| [2] | Documents and slides of mentor Nguyen Van Quyet |