BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

BÀI TẬP LỚN

NHẬP MÔN KHOA HỌC DỮ LIỆU

ỨNG DỤNG HỌC MÁY TRONG DỰ ĐOÁN BỆNH UNG THƯ PHỔI

NGÀNH: KHOA HỌC MÁY TÍNH

SINH VIÊN: **DƯƠNG VIỆT HÙNG**

LỚP: **124221**

NGƯỜI HƯỚNG DẪN: **PGS. TS. NGUYỄN MINH TIẾN**

HƯNG YÊN – 2025

NHẬN XÉT

Nhận xét của giáo viên hướng dẫn

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

GIÁO VIÊN HƯỚNG DẪN

Nguyễn Minh Tiến

LỜI CAM ĐOAN

Em xin cam đoan đồ án "Ứng dụng học máy trong dự đoán bệnh ung thư phổi" là kết quả thực hiện của bản thân em dưới sự hướng dẫn của thầy PSG. TS. Nguyễn Minh Tiến.

Những phần sử dụng tài liệu tham khảo trong đồ án đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong đồ án và chương trình xây dựng được 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 30 tháng 05 năm 2025

Sinh viên

Dương Việt Hùng

LỜI CẢM ƠN

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 Nguyễn Minh Tiến 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 hết sức, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hi vọng sẽ nhận được những ý kiến nhận xét, góp ý của các Thầy giáo, cô giáo về những kết quả triển khai trong bài tập lớn.

Em xin trân trọng cảm ơn!

MỤC LỤC

DANH MỤC CÁC HÌNH VẼ

CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN

1.1 Bài toán

Chẩn đoán và dự đoán bệnh ung thư phổi là một trong những ứng dụng nổi bật và đầy tiềm năng của học máy (machine learning) trong lĩnh vực y học hiện đại. Ung thư phổi hiện là một trong những nguyên nhân gây tử vong hàng đầu trên toàn cầu, đặc biệt tại các quốc gia đang phát triển, nơi mà điều kiện tầm soát và chẩn đoán còn hạn chế. Tại Việt Nam, số ca mắc mới và tử vong do ung thư phổi vẫn ở mức cao, phần lớn do bệnh được phát hiện ở giai đoạn muộn. Việc ứng dụng học máy mang lại khả năng phân tích và xử lý dữ liệu để hỗ trợ phát hiện sớm và phân loại nguy cơ mắc bệnh với độ chính xác cao. Các thuật toán tiên tiến như rừng ngẫu nhiên (Random Forest), XGBoost đã chứng minh khả năng vượt trội trong việc nhận diện tổn thương phổi, phân biệt khối u lành tính và ác tính, cũng như dự đoán tiến triển của bệnh theo thời gian.

Ngoài ra, học máy còn hỗ trợ cá nhân hóa phương pháp điều trị thông qua phân tích đặc điểm di truyền của từng bệnh nhân (genomics), từ đó tối ưu hóa phác đồ hóa trị, xạ trị hoặc liệu pháp miễn dịch. Tuy nhiên, để các mô hình này phát huy hiệu quả trong thực tế lâm sàng, cần đảm bảo nguồn dữ liệu chất lượng cao, đa dạng và được chú thích chính xác, đồng thời đáp ứng các yêu cầu đạo đức và bảo mật thông tin y tế. Tại Việt Nam, những thách thức về hạ tầng công nghệ, nguồn nhân lực chuyên môn và hành lang pháp lý vẫn đang là rào cản lớn. Do đó, việc phát triển và triển khai hệ thống dự đoán ung thư phổi thông minh cần có sự phối hợp liên ngành giữa các cơ sở y tế, viện nghiên cứu, trường đại học và doanh nghiệp công nghệ nhằm xây dựng nền tảng dữ liệu y tế số hóa, thúc đẩy nghiên cứu ứng dụng AI trong y học, góp phần nâng cao chất lượng khám chữa bệnh và giảm tỷ lệ tử vong do ung thư phổi trong tương lai.

1.2 Trình bày dữ liệu bài toán

Link dữ liệu trên Kaggle:

Lung Cancer Dataset | Kaggle

	AGE	GENDER	SMOKING	FINGER_DISCOLORATION	MENTAL_STRESS	EXPOSURE_TO_POLLUTION	LONG_TERM_ILLNESS
0	68	1	1	1	1	1	0
1	81	1	1	0	0	1	1
2	58	1	1	0	0	0	0
3	44	0	1	0	1	1	0
4	72	0	1	1	1	1	1
...	...	...	...	...	...	...	...
4995	32	0	1	1	0	0	1
4996	80	0	1	1	1	1	1
4997	51	1	0	0	1	0	0
4998	76	1	0	1	0	0	0
4999	33	0	1	0	0	1	1

5000 rows × 18 columns

Hình 1.1 Tổng quan về bộ dữ liệu Lung Cancer

```
[56]  df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   AGE                     5000 non-null   int64
 1   GENDER                  5000 non-null   int64
 2   SMOKING                 5000 non-null   int64
 3   FINGER_DISCOLORATION    5000 non-null   int64
 4   MENTAL_STRESS           5000 non-null   int64
 5   EXPOSURE_TO_POLLUTION   5000 non-null   int64
 6   LONG_TERM_ILLNESS       5000 non-null   int64
 7   ENERGY_LEVEL            5000 non-null   float64
 8   IMMUNE_WEAKNESS         5000 non-null   int64
 9   BREATHING_ISSUE         5000 non-null   int64
 10  ALCOHOL_CONSUMPTION     5000 non-null   int64
 11  THROAT_DISCOMFORT       5000 non-null   int64
 12  OXYGEN_SATURATION       5000 non-null   float64
 13  CHEST_TIGHTNESS         5000 non-null   int64
 14  FAMILY_HISTORY          5000 non-null   int64
 15  SMOKING_FAMILY_HISTORY  5000 non-null   int64
 16  STRESS_IMMUNE           5000 non-null   int64
 17  PULMONARY_DISEASE       5000 non-null   object
dtypes: float64(2), int64(15), object(1)
memory usage: 703.3+ KB
```

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

Dữ liệu bài toán gồm các feature sau:

1. **AGE**: Tuổi của đối tượng (số nguyên). Tuổi tác có thể ảnh hưởng đến nguy cơ mắc ung thư phổi, đặc biệt ở những người lớn tuổi.
2. **GENDER**: Giới tính (0 hoặc 1, có thể đại diện cho nữ hoặc nam). Giới tính có thể liên quan đến tỷ lệ mắc bệnh do khác biệt về lối sống hoặc sinh học.
3. **SMOKING**: Thói quen hút thuốc (0: không hút, 1: hút thuốc). Hút thuốc là yếu tố nguy cơ chính gây ung thư phổi.
4. **FINGER_DISCOLORATION**: Tình trạng đổi màu ngón tay (0: không, 1: có). Đây có thể là dấu hiệu của các vấn đề sức khỏe liên quan đến phổi hoặc tuần hoàn.
5. **MENTAL_STRESS**: Mức độ căng thẳng tinh thần (0: không, 1: có). Căng thẳng có thể ảnh hưởng gián tiếp đến sức khỏe tổng thể và hệ miễn dịch.
6. **EXPOSURE_TO_POLLUTION**: Tiếp xúc với ô nhiễm (0: không, 1: có). Ô nhiễm không khí (như PM2.5, PM10, NO2) là yếu tố nguy cơ môi trường.
7. **LONG_TERM_ILLNESS**: Bệnh mãn tính lâu dài (0: không, 1: có). Các bệnh mãn tính có thể làm tăng nguy cơ ung thư phổi.
8. **ENERGY_LEVEL**: Mức năng lượng (giá trị liên tục, có thể là phần trăm). Mức năng lượng thấp có thể liên quan đến các triệu chứng của bệnh phổi.
9. **IMMUNE_WEAKNESS**: Suy yếu hệ miễn dịch (0: không, 1: có). Hệ miễn dịch yếu có thể làm tăng nguy cơ mắc bệnh nghiêm trọng.
10. **BREATHING_ISSUE**: Vấn đề về hô hấp (0: không, 1: có). Khó thở là triệu chứng phổ biến liên quan đến ung thư phổi.
11. **ALCOHOL_CONSUMPTION**: Tiêu thụ rượu bia (0: không, 1: có). Uống rượu có thể ảnh hưởng gián tiếp đến sức khỏe phổi.
12. **THROAT_DISCOMFORT**: Khó chịu ở cổ họng (0: không, 1: có). Đây có thể là triệu chứng sớm của các vấn đề về đường hô hấp.
13. **OXYGEN_SATURATION**: Độ bão hòa oxy trong máu (giá trị liên tục, thường là phần trăm). Giá trị thấp có thể chỉ ra vấn đề về phổi.
14. **CHEST_TIGHTNESS**: Cảm giác tức ngực (0: không, 1: có). Tức ngực là triệu chứng tiềm năng của ung thư phổi hoặc bệnh phổi khác.
15. **FAMILY_HISTORY**: Tiền sử gia đình mắc ung thư phổi (0: không, 1: có). Yếu tố di truyền có thể làm tăng nguy cơ.

16. **SMOKING_FAMILY_HISTORY**: Tiền sử gia đình có người hút thuốc (0: không, 1: có). Tiếp xúc thụ động với khói thuốc là yếu tố nguy cơ.
17. **STRESS_IMMUNE**: Căng thẳng ảnh hưởng đến hệ miễn dịch (0: không, 1: có). Căng thẳng kéo dài có thể làm suy yếu khả năng miễn dịch.
18. **PULMONARY_DISEASE**: Bệnh phổi hiện có (NO: không, YES: có). Đây là biến mục tiêu (target variable) trong tập dữ liệu, chỉ ra liệu đối tượng có mắc bệnh phổi (ung thư phổi) hay không.

Dữ liệu bài toán là 1 file csv gồm 5000 rows × 18 columns

Tương ứng với có 18 features và mỗi feature có 5000 dữ liệu đầu vào.

1.3 Tiền xử lý dữ liệu

a) Xử lý giá trị khuyết thiếu, trùng lặp
b) Xử lý giá trị ngoại lai
c) Feature selection
d) Phân tách dữ liệu
e) Xử lý mất cân bằng nhãn
f) Xử lý chuẩn hóa dữ liệu

1.4 Trực quan hoá dữ liệu

a) Phân bố nhãn
b) Phát hiện ngoại lai
c) Phân bố tuổi
d) Phân bố giới tính
e) Quan hệ giữa hút thuốc và ung thư phổi
f) Quan hệ giữa gia đình và ung thư phổi
g) Ma trận tương quan

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 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:

a) 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.

b) 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.

c) 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().

d) 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.

e) 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().

f) 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.

g) 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.

h) 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).

2.2 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.

2) 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.

3) 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.

4) 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.

5) 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.

6) 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.

7) 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.

8) 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.

9) 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.

10) 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.

2.3 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.

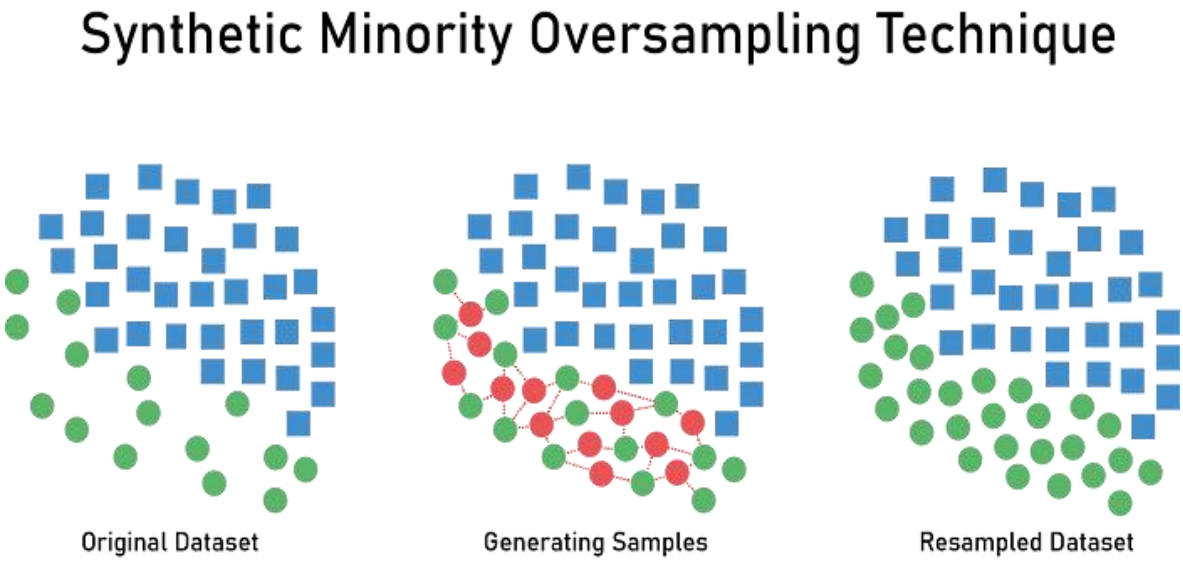
Ngoài ra, Scikit-learn có thể dễ dàng tích hợp với các thư viện khác như Pandas và Matplotlib, giúp người dùng dễ dàng thao tác với dữ liệu và trực quan hóa kết quả. Sau khi huấn luyện mô hình, bạn có thể sử dụng Pandas để phân tích kết quả, sau đó sử dụng Matplotlib để trực quan hóa các chỉ số quan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

2.4 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.

2.5 Random Oversampling Technique

Random Oversampling là một kỹ thuật xử lý dữ liệu mất cân bằng bằng cách nhân bản ngẫu nhiên các mẫu hiện có của lớp thiểu số để cân bằng dữ liệu. Phương pháp này đơn giản và không yêu cầu nội suy giữa các điểm dữ liệu.

Ưu điểm:

- Cân bằng dữ liệu hiệu quả mà không làm mất dữ liệu ban đầu.
- Giữ nguyên cấu trúc, giúp mô hình dễ dàng học tập hơn trên lớp thiểu số.

Nhược điểm:

- Có thể dẫn đến hiện tượng overfitting do việc nhân bản dữ liệu.
- Không tạo ra thêm thông tin mới, chỉ lặp lại dữ liệu hiện có.

2.6 Data Scaling

Scaling là một kỹ thuật trong Data Transformation nhằm thay đổi phạm vi giá trị của các đặc trưng (features) trong dữ liệu để đảm bảo sự đồng nhất về độ lớn giữa các đặc trưng và giúp các thuật toán hoạt động hiệu quả hơn.

Phương pháp:

- **Min - Max Scaler**: Biến đổi dữ liệu về một phạm vi giá trị cụ thể (thường là [0, 1]).

Ưu điểm:

- Scaling giúp tăng độ chính xác cho các thuật toán phụ thuộc vào khoảng cách, như K-Means hoặc KNN.
- Giảm sự chênh lệch về độ lớn giữa các đặc trưng, giúp các thuật toán như SVM, KNN, hoặc Gradient Descent hoạt động hiệu quả hơn.

Nhược điểm:

- Sau khi scale, giá trị dữ liệu không còn phản ánh ý nghĩa ban đầu (ví dụ: chiều cao từ cm chuyển thành số nhỏ như 0.6).
- Phương pháp như Min-Max Scaling rất nhạy cảm với giá trị ngoại lệ, có thể làm méo mó phạm vi giá trị của dữ liệu.

2.7 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.

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.

2.8 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.

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.

2.9 Machine Learning

2.9.1 Decision Tree

a) Nền tảng lý thuyết

Decision Tree (Cây quyết định) là một thuật toán học máy có giám sát (supervised learning) được sử dụng cho cả bài toán phân loại và hồi quy. Thuật toán xây dựng một cây quyết định, trong đó mỗi nút đại diện cho một đặc trưng (feature), mỗi nhánh là một điều kiện, và mỗi lá (leaf) đại diện cho kết quả dự đoán.

b) Công thức tính toán

- Tiêu chí phân tách:

Decision Tree sử dụng các phép đo để đánh giá chất lượng phân tách tại mỗi nút. Một số tiêu chí phổ biến:

Entropy (đo lường mức độ hỗn loạn của dữ liệu):

$$\text{Entropy}(S) = -i=1\sum npilog2(pi)$$

Information Gain (đo lường hiệu quả phân tách):

$$\text{Gain}(S,A) = \text{Entropy}(S) - v \in \text{Values}(A)\sum \mid S| Sv| .\text{Entropy}(Sv)$$

Gini Impurity (đo lường xác suất chọn sai lớp nếu phân loại ngẫu nhiên):

$$\text{Gini}(S) = 1 - i=1\sum n\ pi2$$

c) Ưu điểm

Kết quả của Decision Tree có thể được trực quan hóa, dễ dàng giải thích cho người không chuyên về kỹ thuật.

Không cần chuẩn hóa hay scale dữ liệu. Decision Tree hoạt động tốt với cả dữ liệu danh mục (categorical) và số (numerical).

Cây có thể điều chỉnh để tập trung vào các lớp hiếm thông qua việc tối ưu tiêu chí phân tách.

d) Nhược điểm

Cây sâu hoặc quá chi tiết có thể dẫn đến việc học thuộc dữ liệu huấn luyện, giảm khả năng tổng quát hóa.

Các giá trị ngoại lệ hoặc nhiễu trong dữ liệu có thể làm giảm độ chính xác của cây.

Khi số lượng đặc trưng hoặc giá trị của các đặc trưng rất lớn, việc xây dựng cây có thể trở nên tốn kém về thời gian và tài nguyên.

2.9.2 Naive Bayes

a) Nền tảng lý thuyết

Naive Bayes là một thuật toán học máy có giám sát (supervised learning) được sử dụng chủ yếu trong các bài toán phân loại. Dựa trên Định lý Bayes, NB đưa ra giả định "naive" rằng các đặc trưng (features) là độc lập có điều kiện với nhau, tức là không có sự phụ thuộc giữa các đặc trưng khi đã biết lớp (class).

b) Công thức tính toán

- Giả định độc lập có điều kiện:

NB giả định ràng tất cả các đặc trưng x_1, x_2, …, x_n trong tập đặc trưng X là độc lập:

P(X|C) = P(x_1|C) + P(x_2|C) + … + P(x_n|C)

Phương trình phân loại trở thành:

C = arg max(C)$\prod_{i=1}^{n} P(x_i|C)$

- 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 $P(x_i|C)$: Tỷ lệ xuất hiện của đặc trưng x_i 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.

c) 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)

d) 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.

2.9.3 XGBoost

a) 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.

Công thức tính toán

$$\hat{y} = \sum_{t=1}^{T} f_t(x)$$

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.

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.

2.9.4 Random Forest

a) Nền tảng lý thuyết

Random Forest là thuật toán học máy có giám sát, sử dụng kỹ thuật học tổ hợp (ensemble learning) bằng cách kết hợp nhiều cây quyết định. Mỗi cây được huấn luyện trên tập dữ liệu con ngẫu nhiên (bagging) và chọn tập con đặc trưng ngẫu nhiên tại mỗi nút, tăng tính đa dạng và giảm overfitting. Kết quả cuối cùng được tính bằng đa số phiếu (phân loại) hoặc trung bình (hồi quy).

b) Công thức tính toán

- Dựa trên Decision Tree với các tiêu chí như Entropy, Information Gain, hoặc Gini Impurity.
- Kết hợp dự đoán:

$Y = \text{mode}(Y_1, Y_2, ..., Y_n)$ (phân loại) hoặc $Y = \text{mean}(Y_1, Y_2, ..., Y_n)$ (hồi quy)

c) Ưu điểm

- Giảm overfitting nhờ kết hợp nhiều cây.
- Xử lý tốt dữ liệu mất cân bằng và nhiễu.
- Không yêu cầu chuẩn hóa dữ liệu, hoạt động tốt với cả dữ liệu số và danh mục.

d) Nhược điểm

- Tốn tài nguyên tính toán khi số cây lớn.
- Khó giải thích trực quan với số lượng cây nhiều.
- Có thể overfitting nếu không điều chỉnh độ sâu cây hoặc số lượng cây.

2.9.5 KNN

a) Nền tảng lý thuyết

KNN là thuật toán học máy có giám sát, không tham số, phân loại hoặc dự đoán dựa trên K điểm láng giềng gần nhất trong không gian đặc trưng. Khoảng cách (Euclidean, Manhattan, hoặc Minkowski) được sử dụng để xác định láng giềng, và kết quả được tính bằng đa số phiếu (phân loại) hoặc trung bình (hồi quy).

b) Công thức tính toán

- Khoảng cách Euclidean:

  $d(x, y) = \sqrt{(\sum(i=1 \text{ to } n) (x\_i - y\_i)^2)}$

- Dự đoán phân loại:

  $Y = \text{mode}(Y\_1, Y\_2, ..., Y\_k)$

c) Ưu điểm

- Đơn giản, dễ triển khai, không cần huấn luyện phức tạp.
- Hiệu quả với dữ liệu phi tuyến.
- Linh hoạt cho cả bài toán phân loại và hồi quy.

d) Nhược điểm

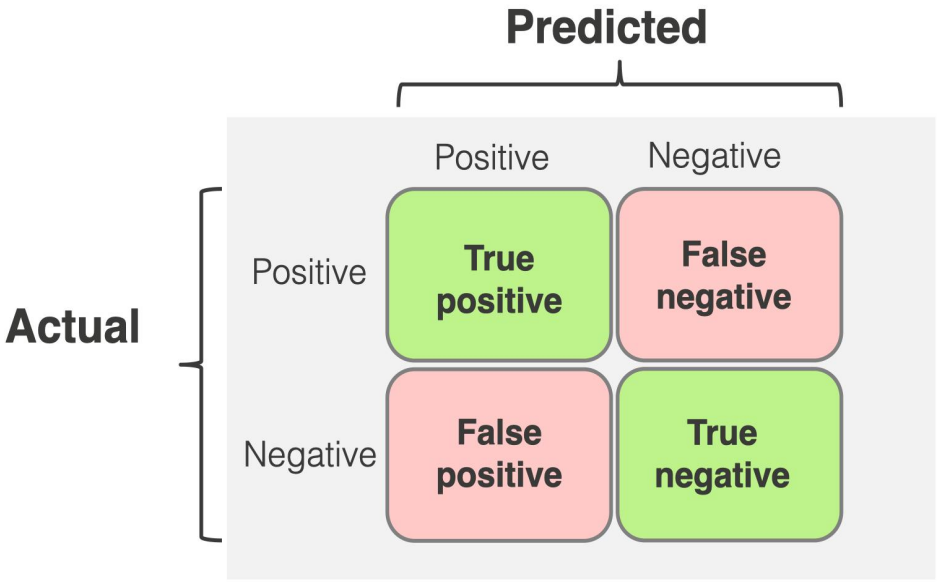
- Tốn tài nguyên tính toán với tập dữ liệu lớn.
- Nhạy cảm với giá trị k và dữ liệu mất cân bằng.
- Yêu cầu chuẩn hóa dữ liệu để đảm bảo tính đồng nhất.

2.10 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 2.2: 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.

2.11 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**:

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

    - **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.
2. **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**:

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

    - **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.

3. **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**:

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

- **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).

4. **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**:

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

- **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.

CHƯƠNG 3: GIẢI PHÁP

3.1. Tiền xử lý dữ liệu

a) Xử lý giá trị khuyết thiếu, trùng lặp

Kiểm tra các giá trị khuyết thiếu

```
[17]
# Check null data
df.isnull().sum()
```

```
AGE                        0
GENDER                     0
SMOKING                    0
FINGER_DISCOLORATION       0
MENTAL_STRESS              0
EXPOSURE_TO_POLLUTION      0
LONG_TERM_ILLNESS          0
ENERGY_LEVEL               0
IMMUNE_WEAKNESS            0
BREATHING_ISSUE            0
ALCOHOL_CONSUMPTION        0
THROAT_DISCOMFORT          0
OXYGEN_SATURATION          0
CHEST_TIGHTNESS            0
FAMILY_HISTORY             0
SMOKING_FAMILY_HISTORY     0
STRESS_IMMUNE              0
PULMONARY_DISEASE          0
dtype: int64
```

Hình 3.1: Kiểm tra giá trị khuyết thiếu

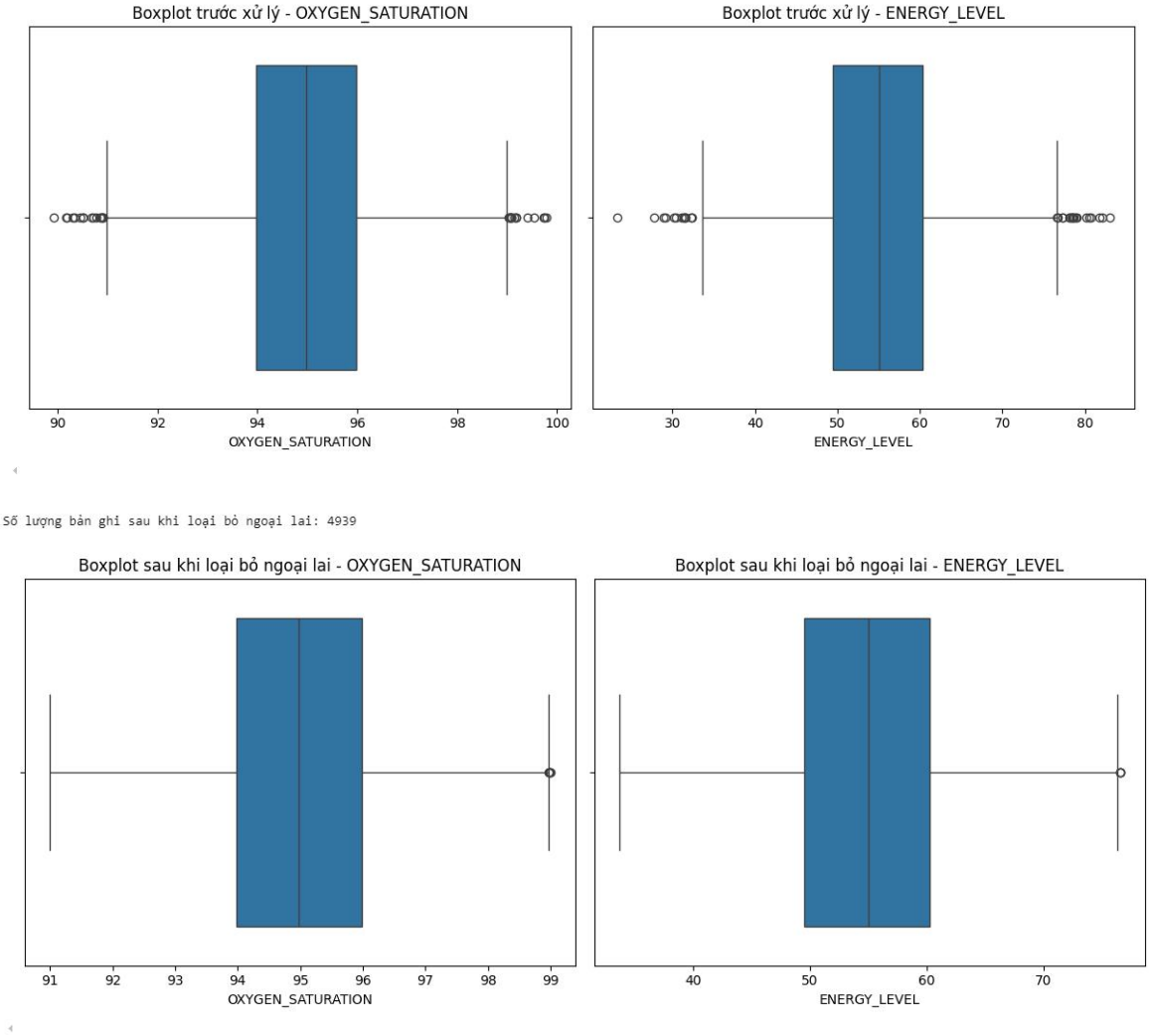
Kiểm tra các giá trị trùng lặp

```
]
# Check duplicated data
df.duplicated().sum()
```

```
np.int64(0)
```

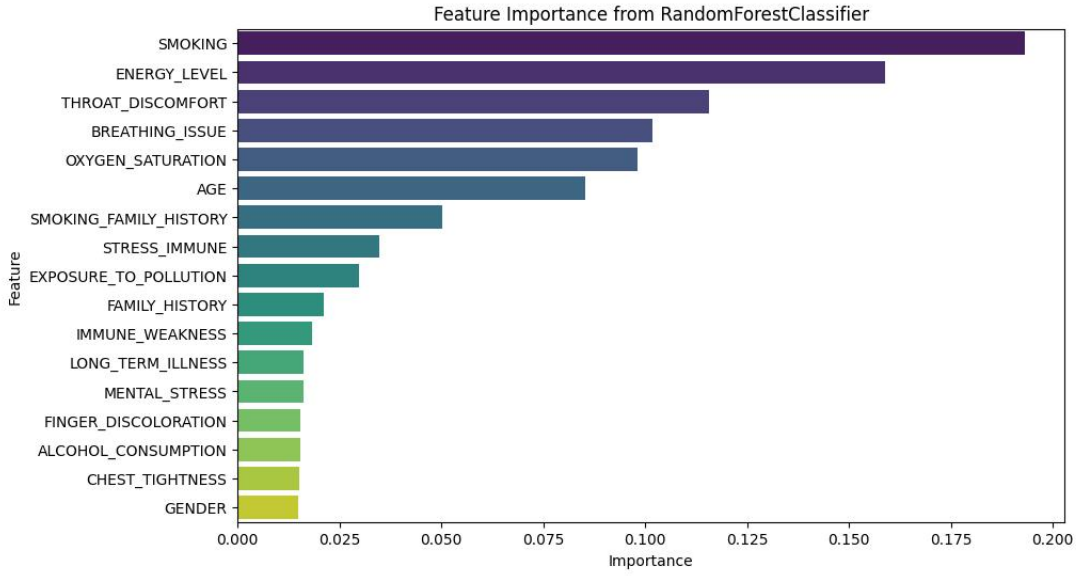
Hình 3.2: Kiểm tra giá trị trùng lặp

b) Xử lý giá trị ngoại lai



Hình 3.3: Xử lý giá trị ngoại lai

c) Feature selection



Hình 3.4: Feature selection

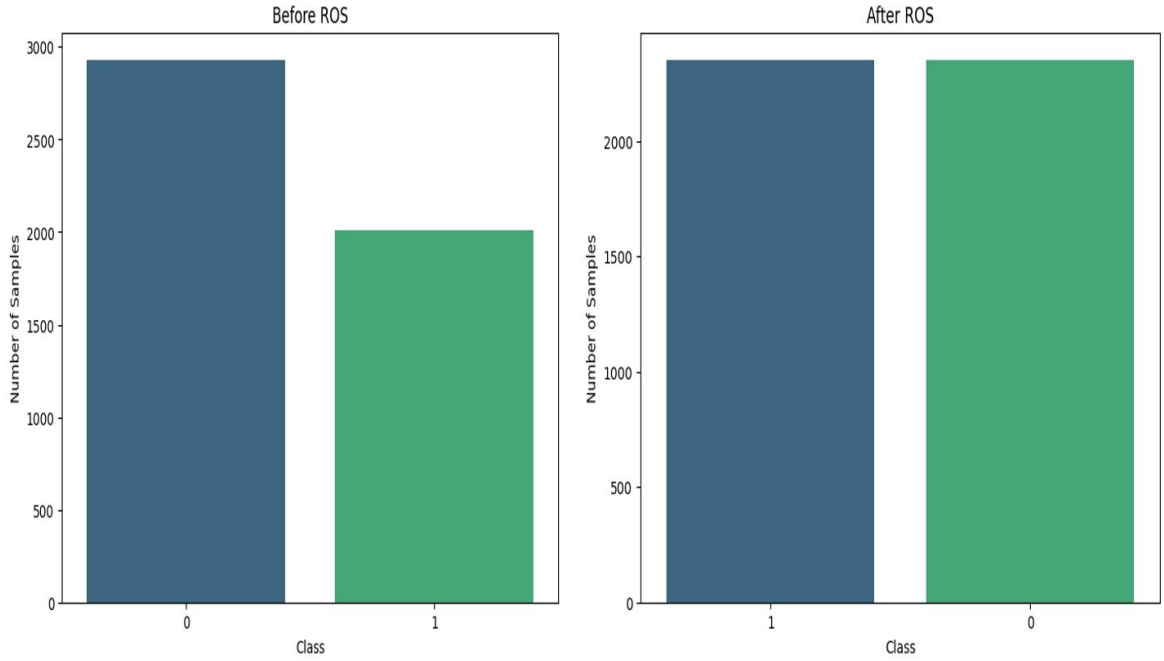
d) Data Split

```
columns_to_drop = [
    "PULMONARY_DISEASE",
    "GENDER",
    "CHEST_TIGHTNESS",
    "ALCOHOL_CONSUMPTION",
    "FINGER_DISCOLORATION",
    "MENTAL_STRESS",
    "LONG_TERM_ILLNESS",
    "IMMUNE_WEAKNESS",
    "FAMILY_HISTORY"
]
X = df.drop(columns=columns_to_drop)
y = df["PULMONARY_DISEASE"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train.head()
```

	AGE	SMOKING	EXPOSURE_TO_POLLUTION	ENERGY_LEVEL	BREATHING_ISSUE	THROAT_DISCOMFORT	OXYGEN_SATURATION	SMOKING_FAMILY_HISTORY	STRESS_IMMUNE
2265	35	1	1	68.442800	1	1	93.510972	1	0
1130	66	1	0	57.370063	1	1	94.843254	0	0
4680	57	0	0	50.147466	1	0	95.858824	0	0
1974	82	0	0	34.225550	1	1	94.188979	0	1
1567	52	1	0	55.064677	1	1	95.530578	0	0

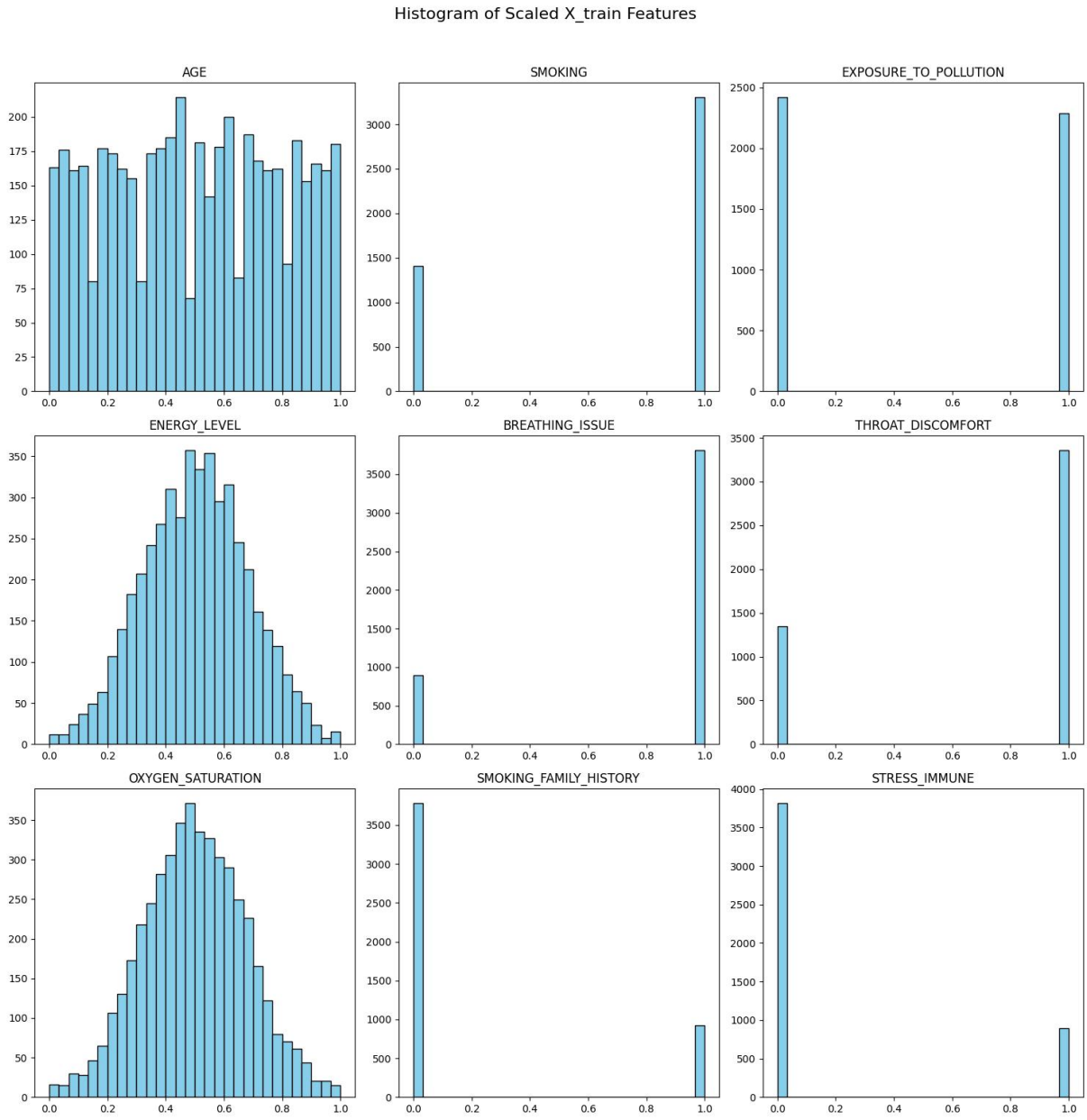
Hình 3.5: Data Split

e) Cân bằng dữ liệu



Hình 3.6: Cân bằng dữ liệu

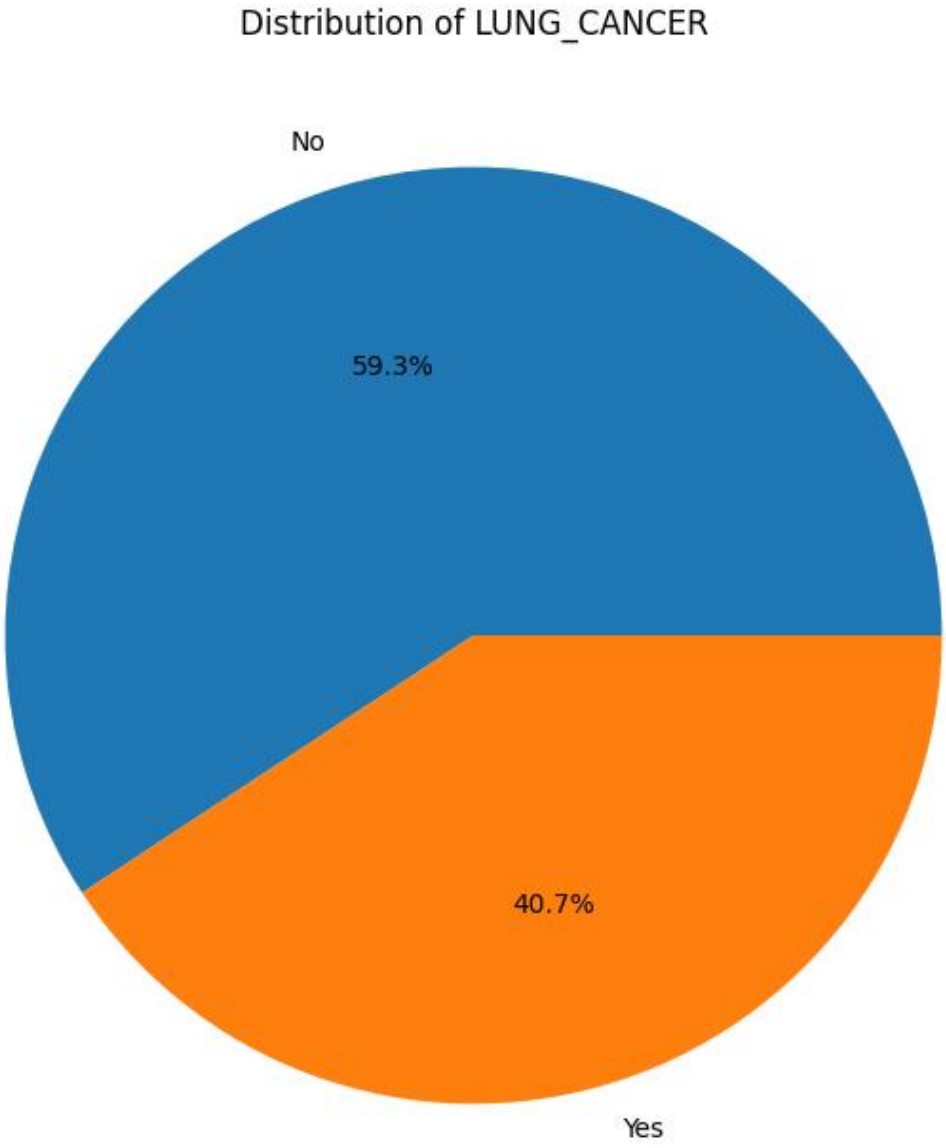
f) Scale dữ liệu



Hình 3.7: Scale dữ liệu

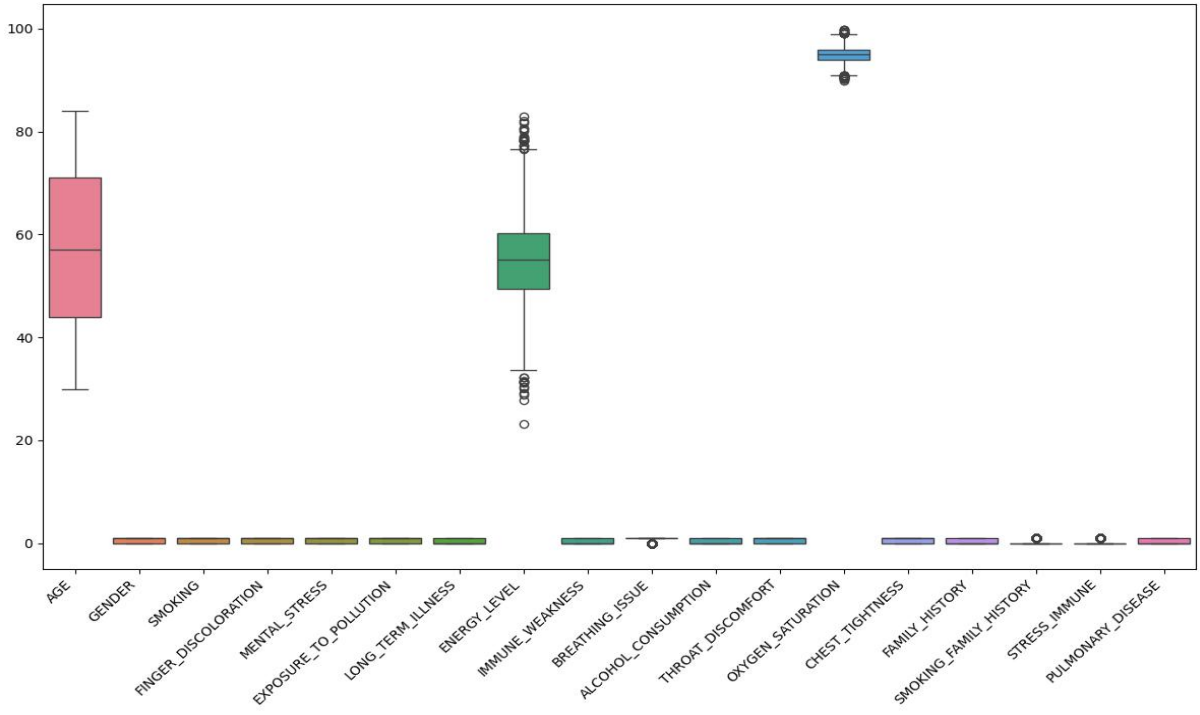
3.2. Trực quan hóa dữ liệu

1. Phân phối nhãn dữ liệu



Hình 3.8: Phân phối của nhãn dữ liệu

2. Phát hiện ngoại lai

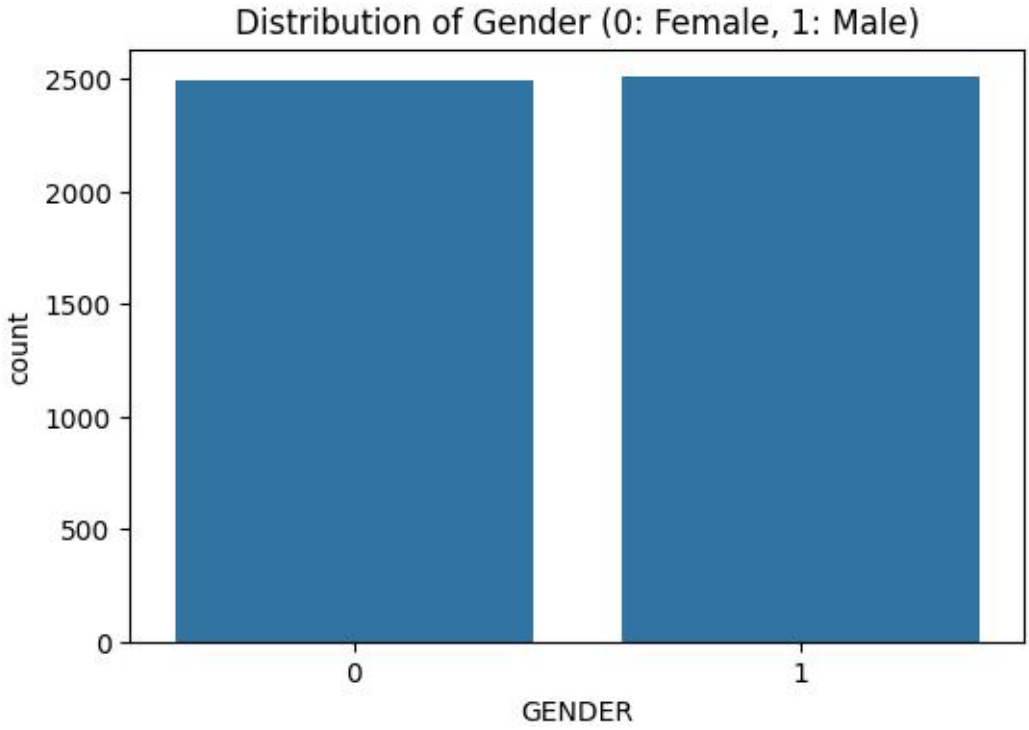


Hình 3.9: Phát hiện ngoại lai

3. Phân bố tuổi

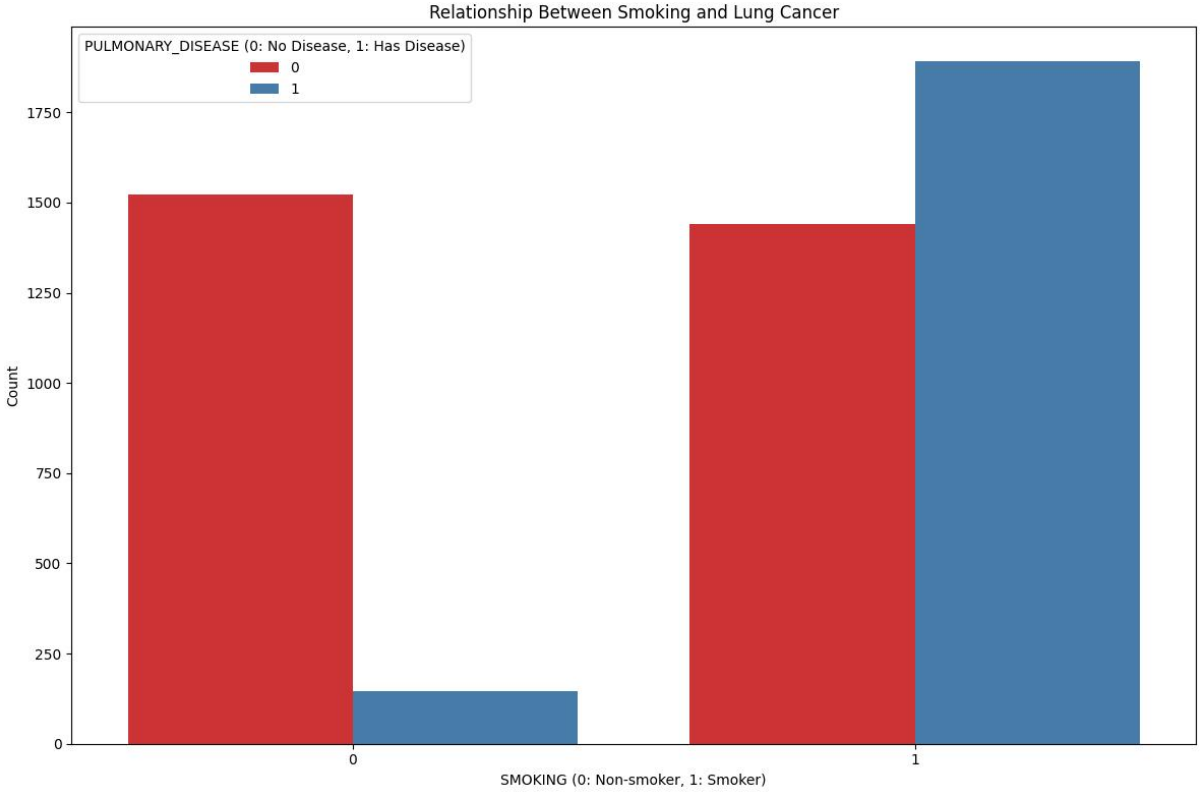
Hình 3.10: Phân bố tuổi

4. Phân bố giới tính



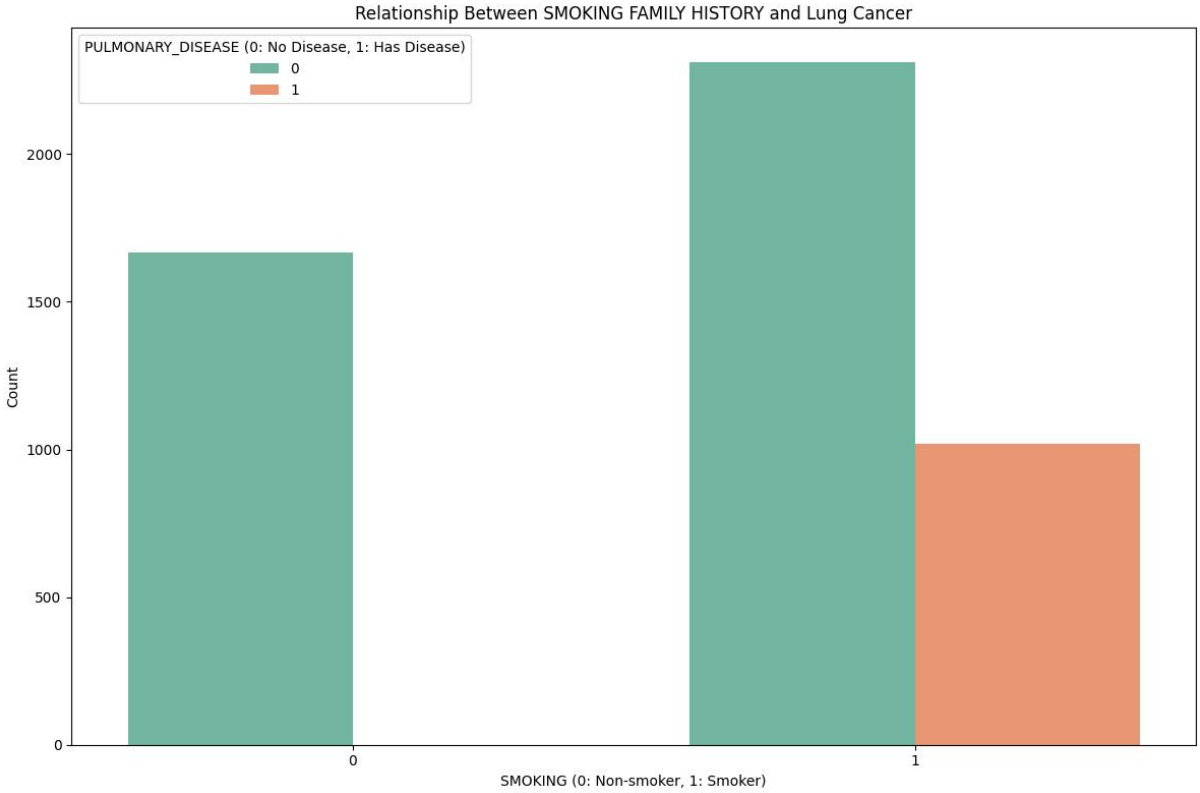
Hình 3.11: Phân bố giới tính

5. Quan hệ giữa hút thuốc và ung thư phổi



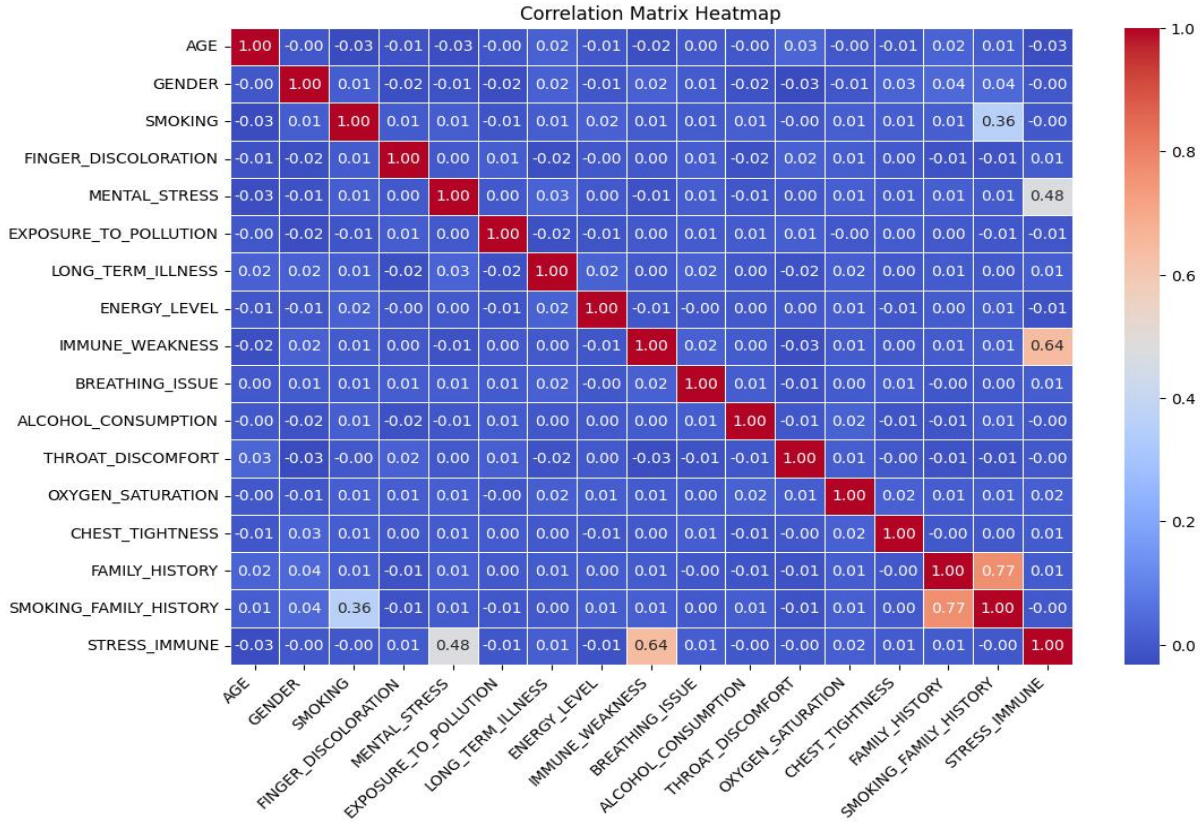
Hình 3.12: Quan hệ giữa hút thuốc và ung thư phổi

6. Quan hệ giữa gia đình và ung thư phổi



Hình 3.13: Quan hệ giữa gia đình và ung thư phổi

7. Ma trận tương quan



Hình 3.14: Ma trận tương quan

3.3. Đánh giá mô hình

Lần 1: (chưa tiền xử lý, tham số mặc định)

Model	Dataset	Accuracy	Precision	Recall	F1-Score
Decision Tree	Train	100.00%	1.0000	1.0000	1.0000
	Test	84.50%	0.8049	0.7772	0.7908
	Difference	15.50%	0.1951	0.2228	0.2092
Random Forest	Train	100.00%	1.0000	1.0000	1.0000
	Test	91.20%	0.8937	0.8700	0.8817
	Difference	8.80%	0.1063	0.1300	0.1183
K-NN	Train	78.50%	0.7454	0.7319	0.7386
	Test	64.50%	0.5281	0.5491	0.5384
	Difference	14.00%	0.2173	0.1829	0.2002
Naive Bayes	Train	86.17%	0.8118	0.8681	0.8390
	Test	86.90%	0.8030	0.8647	0.8327
	Difference	-0.73%	0.0089	0.0034	0.0063
XGBoost	Train	99.40%	0.9886	0.9970	0.9928
	Test	90.60%	0.8856	0.8621	0.8737
	Difference	8.80%	0.1031	0.1349	0.1191

Hình 3.15: Lần 1: (chưa tiền xử lý, tham số mặc định)

Lần 2: (sau tiền xử lý, SMOTE)

Model	Dataset	Accuracy	Precision	Recall	F1-Score
Decision Tree	Train	100.00%	1.0000	1.0000	1.0000
	Test	82.79%	0.7655	0.8439	0.8028
	Difference	17.21%	0.2345	0.1561	0.1972
Random Forest	Train	100.00%	1.0000	1.0000	1.0000
	Test	90.18%	0.8717	0.8951	0.8833
	Difference	9.82%	0.1283	0.1049	0.1167
K-NN	Train	89.04%	0.8917	0.8887	0.8902
	Test	88.97%	0.8444	0.9000	0.8713
	Difference	0.07%	0.0473	-0.0113	0.0189
Naive Bayes	Train	82.02%	0.8133	0.8313	0.8222
	Test	84.31%	0.7766	0.8732	0.8220
	Difference	-2.29%	0.0367	-0.0419	0.0002
XGBoost	Train	97.45%	0.9674	0.9822	0.9747
	Test	87.85%	0.8420	0.8707	0.8561
	Difference	9.60%	0.1254	0.1114	0.1186

Hình 3.16: Lần 2: (sau tiền xử lý, SMOTE)

Lần 2: (sau tiền xử lý, ROS)

Model	Dataset	Accuracy	Precision	Recall	F1-Score
Decision Tree	Train	100.00%	1.0000	1.0000	1.0000
	Test	85.53%	0.8414	0.8024	0.8215
	Difference	14.47%	0.1586	0.1976	0.1785
Random Forest	Train	100.00%	1.0000	1.0000	1.0000
	Test	91.50%	0.9055	0.8878	0.8966
	Difference	8.50%	0.0945	0.1122	0.1034
K-NN	Train	92.01%	0.9152	0.9261	0.9206
	Test	90.18%	0.8735	0.8927	0.8830
	Difference	1.83%	0.0417	0.0334	0.0376
Naive Bayes	Train	85.74%	0.8405	0.8823	0.8609
	Test	85.53%	0.8000	0.8683	0.8327
	Difference	0.22%	0.0405	0.0140	0.0281
XGBoost	Train	98.53%	0.9774	0.9936	0.9855
	Test	90.28%	0.8811	0.8854	0.8832
	Difference	8.25%	0.0964	0.1083	0.1022

Hình 3.17: Lần 2: (sau tiền xử lý, ROS)

Lần 3: (Tùy chỉnh tham số, SMOTE)

Model	Dataset	Accuracy	Precision	Recall	F1-Score
Decision Tree	Train	88.50%	0.9030	0.8627	0.8824
	Test	87.55%	0.8284	0.8829	0.8548
	Difference	0.95%	0.0746	-0.0202	0.0276
Random Forest	Train	98.90%	0.9873	0.9907	0.9890
	Test	88.97%	0.8508	0.8902	0.8701
	Difference	9.93%	0.1365	0.1004	0.1189
K-NN	Train	100.00%	1.0000	1.0000	1.0000
	Test	89.98%	0.8642	0.9000	0.8817
	Difference	10.02%	0.1358	0.1000	0.1183
Naive Bayes	Train	82.02%	0.8133	0.8313	0.8222
	Test	84.31%	0.7766	0.8732	0.8220
	Difference	-2.29%	0.0367	-0.0419	0.0002
XGBoost	Train	97.60%	0.9706	0.9817	0.9761
	Test	89.37%	0.8605	0.8878	0.8739
	Difference	8.23%	0.1101	0.0939	0.1022

Hình 3.18: Lần 3: (Tùy chỉnh tham số, SMOTE)

Lần 3: (Tùy chỉnh tham số, ROS)

Model	Dataset	Accuracy	Precision	Recall	F1-Score
Decision Tree	Train	94.65%	0.9476	0.9452	0.9464
	Test	88.87%	0.8713	0.8585	0.8649
	Difference	5.78%	0.0763	0.0866	0.0815
Random Forest	Train	100.00%	1.0000	1.0000	1.0000
	Test	90.79%	0.8958	0.8805	0.8881
	Difference	9.21%	0.1042	0.1195	0.1119
K-NN	Train	100.00%	1.0000	1.0000	1.0000
	Test	91.50%	0.8937	0.9024	0.8981
	Difference	8.50%	0.1063	0.0976	0.1019
Naive Bayes	Train	85.74%	0.8405	0.8823	0.8609
	Test	85.53%	0.8000	0.8683	0.8327
	Difference	0.22%	0.0405	0.0140	0.0281
XGBoost	Train	97.54%	0.9706	0.9805	0.9755
	Test	91.19%	0.9007	0.8854	0.8930
	Difference	6.34%	0.0698	0.0951	0.0825

Hình 3.19: Lần 3: (Tùy chỉnh tham số, ROS)

KẾT LUẬN

1. **Thành tựu:**
   - Dự án đã đạt được những kết quả đáng kể trong việc phân loại nguy cơ ung thư phổi bằng cách sử dụng các mô hình học máy như Random Forest và XGBoost. Các mô hình này thể hiện hiệu quả cao trong việc phân tích các đặc trưng như tuổi, thói quen hút thuốc, tiếp xúc với ô nhiễm, độ bão hòa oxy, và các triệu chứng như khó thở hoặc tức ngực, từ đó dự đoán chính xác khả năng mắc bệnh phổi.
   - Khả năng xử lý các bộ dữ liệu lớn với nhiều đặc trưng đa dạng (bao gồm yếu tố môi trường, lối sống, và tiền sử gia đình) đã cung cấp những hiểu biết sâu sắc, hỗ trợ xây dựng các chiến lược phòng ngừa và điều trị hiệu quả hơn.
2. **Hạn chế:**
   - **Hạn chế dữ liệu:** Tập dữ liệu có thể thiếu thông tin chi tiết về các yếu tố khác như mức độ tiếp xúc với ô nhiễm (ví dụ: nồng độ PM2.5, PM10) hoặc các yếu tố di truyền cụ thể, đặc biệt ở các khu vực có hệ thống thu thập dữ liệu y tế kém phát triển. Ngoài ra, dữ liệu có thể không đại diện cho toàn bộ dân số do thiếu sự đa dạng về địa lý hoặc nhân khẩu học.
   - **Khả năng tổng quát hóa:** Các mô hình được huấn luyện trên dữ liệu từ một nhóm đối tượng cụ thể có thể không đạt hiệu quả cao khi áp dụng cho các nhóm khác với đặc điểm lối sống, môi trường, hoặc yếu tố di truyền khác biệt.
3. **Hướng phát triển:**
   - **Cải thiện dữ liệu:** Cần thu thập và tích hợp các bộ dữ liệu đa dạng hơnbao gồm dữ liệu thời gian thực từ các thiết bị y tế (như máy đo độ bão hòa oxy), thông tin về mức độ ô nhiễm môi trường chi tiết hơn, và dữ liệu di truyền dài hạn để nâng cao tính toàn diện và khả năng áp dụng của mô hình.
   - **Tăng độ chính xác:** Áp dụng các kỹ thuật học máy tiên tiến như học sâu (deep learning) để nâng cao độ chính xác trong việc phát hiện sớm các dấu hiệu ung thư phổi, đặc biệt ở những trường hợp có triệu chứng không rõ ràng.

TÀI LIỆU THAM KHẢO

[1] Dataset link: *https://www.kaggle.com/datasets/shantanugarg274/lung-cancer-prediction-dataset*

[2] Scikit-learn Developers. *Scikit-learn: Machine Learning in Python*

[3] P. Vinothini and P. Rajalakshmi, "Predicting Lung Cancer using Machine Learning Algorithms," *International Journal of Scientific & Engineering Research*

[4] "Lung cancer detection using machine learning: A review," *Computer Methods and Programs in Biomedicine*, vol. 208, p. 106288, 2021