TRƯỜNG ĐẠI HỌC MỎ - ĐỊA CHẤT

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NCKH SINH VIÊN

ĐỀ TÀI: “XÂY DỰNG MÔ HÌNH HỌC MÁY DỰ ĐOÁN BỆNH TIM DỰA TRÊN TẬP DỮ LIỆU HEARTDISEASE”

Hà Nội, 5/2024

TRƯỜNG ĐẠI HỌC MỎ - ĐỊA CHẤT

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NCKH SINH VIÊN

ĐỀ TÀI: “XÂY DỰNG MÔ HÌNH HỌC MÁY DỰ ĐOÁN BỆNH TIM DỰA TRÊN TẬP DỮ LIỆU HEARTDISEASE”

* Trưởng nhóm nghiên cứu: Nguyễn Văn Huy, Lớp DCCTCLC66A1
* Thành viên tham gia thực hiện:

1. Mai Thị Diễm, DCCTCLC66A1
2. Nguyễn Hoàng Tùng Phong, DCCTCLC66A1

Giáo viên hướng dẫn: Thạc Sĩ Đặng Văn Nam

Hà Nội, 5/2024

# MỤC LỤC

[MỤC LỤC 3](#_Toc166431735)

[DANH MỤC HÌNH ẢNH 5](#_Toc166431736)

[CHƯƠNG 1: GIỚI THIỆU CHUNG VỀ ĐỀ TÀI 6](#_Toc166431737)

[1.1. Lý do chọn đề tài 6](#_Toc166431738)

[1.2. Mục tiêu của đề tài 7](#_Toc166431739)

[1.3. Nội dung nghiên cứu 7](#_Toc166431740)

[1.4. Phạm vi của đề tài 8](#_Toc166431741)

[CHƯƠNG 2: TỔNG QUAN VỀ HỌC MÁY 10](#_Toc166431742)

[2.1. Khái niệm 10](#_Toc166431743)

[2.2. Lịch sử hình thành và phát triển của học máy 10](#_Toc166431744)

[2.3. Phân loại các thuật toán học máy 12](#_Toc166431745)

[2.3.1. Học có giám sát (Supervised Learning) 12](#_Toc166431746)

[2.3.2. Học không giám sát (Unsupervised Learning) 12](#_Toc166431747)

[2.3.3. Học bán giám sát (Semi-supervised Learning) 13](#_Toc166431748)

[2.3.4. Học tăng cường (Reinforcement Learning) 13](#_Toc166431749)

[2.4. Tầm quan trọng của học máy 13](#_Toc166431750)

[2.5. Quy trình thực hiện một dự án học máy 14](#_Toc166431751)

[2.6. Các ứng dụng của học máy dữ liệu trong y tế 16](#_Toc166431752)

[CHƯƠNG 3: CÔNG CỤ VÀ MÔI TRƯỜNG THỰC HIỆN 19](#_Toc166431753)

[3.1. Giới thiệu về ngôn ngữ lập trình Python 19](#_Toc166431754)

[3.2. Một số thư viện quan trong cho phân tích dữ liệu và học máy 27](#_Toc166431755)

[3.2.1. Thư viện Pandas 28](#_Toc166431756)

[3.2.2. Thư viện Numpy 30](#_Toc166431757)

[3.2.3. Thư viện Matplotlib 32](#_Toc166431758)

[3.2.4. Thư viện Scikit-learn 33](#_Toc166431759)

[3.2.5. Thư viện Seaborn 35](#_Toc166431760)

[3.3. Môi trường lập trình Jupyter Notebook 36](#_Toc166431761)

[3.4. Một số thuật toán dùng để xây dụng mô hình học máy 40](#_Toc166431762)

[3.4.1. Thuật toán K láng giềng gần nhất (K-Nearest Neighbor) 40](#_Toc166431763)

[3.4.2. Thuật toán Cây quyết định (Decision Tree) 41](#_Toc166431764)

[3.4.3. Thuật toán Rừng ngẫu nhiên (Random Forest) 42](#_Toc166431765)

[3.5. Môi trường và ngôn ngữ dùng để lập trình web 43](#_Toc166431766)

[3.5.1. Thư viện Flask 44](#_Toc166431767)

[3.5.2. HTML 45](#_Toc166431768)

[3.5.3. CSS 46](#_Toc166431769)

[CHƯƠNG 4: XÂY DỰNG MÔ HÌNH DỰ ĐOÁN BỆNH TIM 47](#_Toc166431770)

[4.1. Giới thiệu bài toán và mô tả tập dữ liệu 47](#_Toc166431771)

[4.1.1. Giới thiệu bài toán 47](#_Toc166431772)

[4.1.2. Mô tả tập dữ liệu 48](#_Toc166431773)

[4.2. Chuẩn bị dữ liệu 49](#_Toc166431774)

[4.2.1. Lọc lấy dữ liệu những thuộc tính cần dùng cho bài toán 50](#_Toc166431775)

[4.2.2. Thông kê dữ liệu bị Missing 52](#_Toc166431776)

[4.2.3. Khám phá dữ liệu có ích từ tập dữ liệu 53](#_Toc166431777)

[4.2.4. Chuyển đổi dữ liệu chuỗi về dữ liệu số 55](#_Toc166431778)

[4.2.5. Thống kê những người mắc bệnh tim 57](#_Toc166431779)

[4.2.6. Sự ảnh hưởng của các yếu tố tới việc mắc bệnh tim 58](#_Toc166431780)

[4.2.7. Xử lý dữ liệu mất cân bằng 59](#_Toc166431781)

[4.3. Xây dựng mô hình 59](#_Toc166431782)

[4.3.1. Thuật toán K láng giềng gần nhất (K-Nearest Neighbor) 60](#_Toc166431783)

[4.3.2. Thuật toán Cây quyết định (Decision Tree) 62](#_Toc166431784)

[4.3.3. Thuật toán Rừng ngẫu nhiên (Random Forest) 64](#_Toc166431785)

[4.4. Đưa ra dự đoán 67](#_Toc166431786)

[ĐÁNH GIÁ VÀ KẾT LUẬN 69](#_Toc166431787)

[TÀI LIỆU THAM KHẢO 72](#_Toc166431788)

# DANH MỤC HÌNH ẢNH

[Hình 1: Dòng thời gian phát triển của các phiên bản Python 22](#_Toc166431838)

[Hình 2: Thư viện Pandas 28](#_Toc166431839)

[Hình 3: Thư viện NumPy 30](#_Toc166431840)

[Hình 4: Thư viên Matplotlib 32](#_Toc166431841)

[Hình 5: Thư viện Scikit-learn 34](#_Toc166431842)

[Hình 6: Thư viện Seaborn 35](#_Toc166431843)

[Hình 7: Môi trường Jupyter Notebook 36](#_Toc166431844)

[Hình 8: Mô tả bài toán phân lớp 41](#_Toc166431845)

[Hình 9: Mô hình cây quyết định 42](#_Toc166431846)

[Hình 10: Thư viện Flask 44](#_Toc166431847)

[Hình 11: HTML 45](#_Toc166431848)

[Hình 12: CSS 46](#_Toc166431849)

[Hình 13: Dữ liệu file heartdisease.csv 48](#_Toc166431850)

[Hình 14: Dữ liệu đọc từ file.csv 49](#_Toc166431851)

[Hình 15: Lọc lấy thuộc tính phục vụ bài toán 50](#_Toc166431852)

[Hình 16: Thống kê dữ liệu bị missing 52](#_Toc166431853)

[Hình 17: Loại bỏ missing 53](#_Toc166431854)

[Hình 18: Tập dữ liệu sau khi hết missing 53](#_Toc166431855)

[Hình 19: Khám phá dữ liệu số 53](#_Toc166431856)

[Hình 20: Nhóm các giá trị BMI 54](#_Toc166431857)

[Hình 21: Khám phá dữ liệu chuỗi 54](#_Toc166431858)

[Hình 22: Code chuyển đổi 56](#_Toc166431859)

[Hình 23: Code vẽ biểu đồ thống kê người mắc bệnh tim 57](#_Toc166431860)

[Hình 24: Biểu đồ phân bố người mắc bệnh tim 57](#_Toc166431861)

[Hình 25: Biểu đồ tròn phần trăm người bệnh tịm theo từng yếu tố ảnh hướng 58](#_Toc166431862)

[Hình 26: Biểu đồ cột phần trăm người bệnh tịm theo từng yếu tố ảnh hướng 58](#_Toc166431863)

[Hình 27: Xử lý dữ liệu mất cân bằng 59](#_Toc166431864)

[Hình 28: Biểu đồ thống kê tỷ lệ phân bố các lớp sau khi xử lý dữ liệu mất cân bằng 59](#_Toc166431865)

[Hình 29: Huấn luyện mô hình bằng thuật toán KNN 60](#_Toc166431866)

[Hình 30: Độ chính xác trên tập huấn luyện (KNN) 60](#_Toc166431867)

[Hình 31: Độ chính xác trên tập test (KNN) 61](#_Toc166431868)

[Hình 32: Kết quả sau khi điều chỉnh tham số (KNN) 61](#_Toc166431869)

[Hình 33: Huấn luyện mô hình bằng thuật toán cây quyết định 62](#_Toc166431870)

[Hình 34: Độ chính xác trên tập huấn luyện (Cây quyết định) 63](#_Toc166431871)

[Hình 35: Độ chính xác trên tập test (Cây quyết định) 63](#_Toc166431872)

[Hình 36: Huấn luyện mô hình bằng thuật toán rừng ngẫu nhiên 64](#_Toc166431873)

[Hình 37: Độ chính xác (Rừng ngẫu nhiên) 65](#_Toc166431874)

[Hình 38: Điều chỉnh tham số thuật toán rừng ngẫu nhiên 65](#_Toc166431875)

[Hình 39: Độ chính xác sau khi điều chỉnh tham số (Rừng ngẫu nhiên) 66](#_Toc166431876)

[Hình 40: Lưu mô hình 66](#_Toc166431877)

[Hình 41: Giao diện trang web ứng dụng mô hình học máy đã xây dựng được 67](#_Toc166431878)

[Hình 42: Kết quả AI dự đoán: Bạn không mắc bệnh tim! 67](#_Toc166431879)

[Hình 43: Kết quả AI dự đoán: Bạn có khả năng mắc bệnh tim! 68](#_Toc166431880)

# GIỚI THIỆU CHUNG VỀ ĐỀ TÀI

## Lý do chọn đề tài

Bệnh tim là một trong những nguyên nhân hàng đầu gây tử vong trên toàn cầu, và việc phát hiện và dự đoán sớm bệnh tim có thể cải thiện dự báo và điều trị. Trong bối cảnh này, việc áp dụng các kỹ thuật học máy và phân tích dữ liệu là một phương tiện hiệu quả để xây dựng các mô hình dự đoán bệnh tim. Dưới đây là những lý do về tầm quan trọng và tính hợp lý của đề tài này:

* **Sức Khỏe Công Cộng:** Bệnh tim mạch đang trở thành một vấn đề sức khỏe công cộng ngày càng nghiêm trọng trên toàn thế giới, đặc biệt là trong các nền kinh tế phát triển. Việc nắm bắt được các yếu tố nguy cơ và dự đoán bệnh tim một cách chính xác có thể giúp giảm tỷ lệ mắc bệnh và tử vong do bệnh tim mạch.
* **Ý Nghĩa Lâm Sàng:** Một hệ thống dự đoán chính xác bệnh tim có thể cung cấp thông tin quan trọng cho các bác sĩ và nhà nghiên cứu y tế. Việc phát triển mô hình học máy không chỉ giúp xác định nguy cơ bệnh tim một cách nhanh chóng mà còn có thể giúp tăng cường quyết định lâm sàng và dự báo kết quả của bệnh nhân.
* **Tiềm Năng Ứng Dụng Rộng Rãi:** Mô hình dự đoán bệnh tim có thể được tích hợp vào các hệ thống y tế điện tử, giúp tự động hóa quá trình đánh giá nguy cơ và cung cấp gợi ý điều trị cho người dùng. Điều này mang lại lợi ích to lớn trong việc cải thiện chăm sóc sức khỏe và giảm bớt gánh nặng cho các nhà cung cấp dịch vụ y tế.
* **Tiềm Năng Nghiên Cứu Thêm:** Việc phát triển một mô hình học máy dự đoán bệnh tim từ tập dữ liệu HeartDisease không chỉ là một nghiên cứu cụ thể mà còn mở ra cánh cửa cho các nghiên cứu sâu hơn về ứng dụng của học máy trong lĩnh vực y học. Các phương pháp và kỹ thuật được phát triển trong quá trình này có thể được áp dụng rộng rãi vào việc dự đoán và đối phó với các bệnh lý khác.
* **Hỗ Trợ Quyết Định Y Tế:** Các mô hình dự đoán bệnh tim có thể cung cấp cho bác sĩ và chuyên gia y tế thông tin quan trọng để hỗ trợ quyết định lâm sàng, từ việc đánh giá nguy cơ đến đề xuất điều trị phù hợp. Điều này có thể giúp tối ưu hóa quy trình chăm sóc bệnh nhân và cải thiện kết quả điều trị.

Từ tầm quan trọng trên, nhóm chúng em đã quyết định chọn đề tài “**xây dựng mô hình học máy dự đoán bệnh tim dựa trên tập dữ liệu heartdisease**” để làm Nghiên cứu Khoa học.

## Mục tiêu của đề tài

Mục tiêu của đề tài là xây dựng và đánh giá hiệu suất của một mô hình học máy có khả năng dự đoán bệnh tim dựa trên tập dữ liệu HeartDisease. Cụ thể, các mục tiêu chính bao gồm:

**Thu thập và Tiền Xử Lý Dữ Liệu:** Tập trung vào việc thu thập dữ liệu từ tập dữ liệu HeartDisease và tiền xử lý các dữ liệu này để chuẩn bị cho quá trình huấn luyện mô hình. Bước này bao gồm xử lý dữ liệu thiếu, mã hóa các biến phân loại và chuẩn hóa các biến số.

**Xây Dựng Mô Hình Học Máy:** Phát triển và tinh chỉnh các mô hình học máy để dự đoán nguy cơ mắc bệnh tim dựa trên các đặc điểm lâm sàng của bệnh nhân. Một loạt các thuật toán học máy sẽ được thử nghiệm, bao gồm các mô hình cổ điển như Random Forest, K-Nearest Neighbors (KNN), Decision Tree.

**Đánh Giá Hiệu Suất:** Đánh giá hiệu suất của các mô hình được xây dựng bằng cách sử dụng các phương pháp đánh giá chuẩn, bao gồm ma trận nhầm lẫn,

**So Sánh và Lựa Chọn Mô Hình Tốt Nhất:** So sánh hiệu suất của các mô hình khác nhau và lựa chọn mô hình tốt nhất dựa trên các tiêu chí như độ chính xác, độ nhạy và độ đặc hiệu. Mô hình được chọn sẽ được xem xét là phù hợp nhất để dự đoán bệnh tim từ các biến số lâm sàng.

Bằng cách hoàn thành các mục tiêu này, đề tài nghiên cứu này nhằm cung cấp một cách tiếp cận hiệu quả và đáng tin cậy để dự đoán bệnh tim dựa trên dữ liệu lâm sàng, đóng góp vào nỗ lực chung trong việc cải thiện sức khỏe cộng đồng và chăm sóc cá nhân.

## Nội dung nghiên cứu

Để hoàn thành mục tiêu đã đặt ra, chúng em tập trung vào các nội dung nghiên cứu sau:

* Nghiên cứu tầm quan trọng của cơ sở dữ liệu và quy trình thực hiện một dự án về phân tích dữ liệu;
* Nghiên cứu ngôn ngữ lập trình python và các công cụ cần thiết phục vụ cho việc phân tích dữ liệu;
* Nghiên cứu sử dụng một số thư viện phục vụ cho phân tích dữ liệu như Pandas, Matplotlib, Numpy, …
* Nghiên cứu các phương pháp tiền xử lý dữ liệu, trực quan hóa dữ liệu;
* Nghiên cứu tập dữ liệu HeartDisease, các thuộc tính, kiểu dữ liệu, các đặc trưng thông kê;
* Nghiên cứu các phương pháp trích rút thông tin có ích từ tập dữ liệu, cách trình bày và thể hiện kết quả;
* Nghiên cứu sử dụng các thuật toán để xây dựng mô hình học máy có độ chính xác tốt nhất.

## Phạm vi của đề tài

Phạm vi của đề tài nghiên cứu này được giới hạn vào việc xây dựng và đánh giá hiệu suất của một mô hình học máy dự đoán bệnh tim dựa trên tập dữ liệu HeartDisease. Nghiên cứu sẽ tập trung vào việc sử dụng tập dữ liệu HeartDisease, bao gồm thông tin về các biến số lâm sàng và kết quả xác định có bệnh tim của các bệnh nhân. Các biến số lâm sàng như tuổi, giới tính và các biến số khác có thể có liên quan đến bệnh tim sẽ được xem xét trong quá trình xây dựng mô hình. Phương pháp học máy như Random Forest, K-Nearest Neighbors (KNN), Decision Tree sẽ được triển khai và đánh giá hiệu suất. Đồng thời, tiền xử lý dữ liệu sẽ được thực hiện trước khi đưa vào huấn luyện mô hình, bao gồm xử lý dữ liệu thiếu, mã hóa biến phân loại, và chuẩn hóa các biến số. Hiệu suất của mô hình sẽ được đánh giá thông qua các phương pháp đánh giá chuẩn như ma trận nhầm lẫn, đường cong ROC, và các chỉ số như độ chính xác, độ nhạy và độ đặc hiệu. Tuy nhiên, nghiên cứu sẽ không bao gồm việc thu thập thêm dữ liệu hoặc thực hiện các thử nghiệm lâm sàng để xác nhận độ chính xác của mô hình trong điều kiện thực tế.

Tập dữ liệu HeartDisease – do CDC Mỹ thu thập thông qua khảo sát năm 2022 với hơn 400 nghìn người trưởng thành liên quan đến tình trạng sức khỏe của họ. Tập dữ liệu này gồm 445132 bản ghi và 40 trường thuộc tính.

# TỔNG QUAN VỀ HỌC MÁY

## Khái niệm

**Học máy hay máy học** (tiếng Anh: **machine learning**) là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống “học” tự động từ dữ liệu để giải quyết những vấn đề cụ thể. Các thuật toán học máy xây dựng một mô hình dựa trên dữ liệu mẫu, được gọi là dữ liệu huấn luyện, để đưa ra dự đoán hoặc quyết định mà không cần được lập trình chi tiết về việc đưa ra dự đoán hoặc quyết định này. Ví dụ như các máy có thể “học” cách phân loại thư điện tử xem có phải thư rác (spam) hay không và tự động xếp thư vào thư mục tương ứng. Học máy rất gần với suy diễn thống kê (statistical inference) tuy có khác nhau về thuật ngữ. Một nhánh của học máy là học sâu phát triển rất mạnh mẽ gần đây và có những kết quả vượt trội so với các phương pháp học máy khác Học máy có liên quan lớn đến thống kê, vì cả hai lĩnh vực đều nghiên cứu việc phân tích dữ liệu, nhưng khác với thống kê, học máy tập trung vào sự phức tạp của các giải thuật trong việc thực thi tính toán. Nhiều bài toán suy luận được xếp vào loại bài toán NP-khó, vì thế một phần của học máy là nghiên cứu sự phát triển các giải thuật suy luận xấp xỉ mà có thể xử lý được. [1]

## Lịch sử hình thành và phát triển của học máy

* 1950 - Nhà bác học Alan Turing đã tạo ra “Turing Test (phép thử Turing)” để xác định xem liệu một máy tính có trí thông minh thực sự hay không. Để vượt qua bài kiểm tra đó, một máy tính phải có khả năng đánh lừa một con người tin rằng nó cũng là con người.
* 1952 - Arthur Samuel đã viết ra chương trình học máy (computer learning) đầu tiên. Chương trình này là trò chơi cờ đam, và hãng máy tính IBM đã cải tiến trò chơi này để nó có thể tự học và tổ chức những nước đi trong chiến lược để giành chiến thắng.
* 1957 - Frank Rosenblatt đã thiết kế mạng nơron (neural network) đầu tiên cho máy tính, trong đó mô phỏng quá trình suy nghĩ của bộ não con người.
* 1967 - Thuật toán “nearest neighbor” đã được viết, cho phép các máy tính bắt đầu sử dụng những mẫu nhận dạng (pattern recognition) rất cơ bản. Nó được sử dụng để vẽ ra lộ trình cho một người bán hàng có thể bắt đầu đi từ một thành phố ngẫu nhiên nhưng đảm bảo anh ta sẽ đi qua tất cả các thành phố khác theo một quãng đường ngắn nhất.
* 1979 - Sinh viên tại trường đại học Stanford đã phát minh ra giỏ hàng “Stanford Cart” có thể điều hướng để tránh các chướng ngại vật trong một căn phòng.
* 1981 - Gerald Dejong giới thiệu về khái niệm Explanation Based Learning (EBL), trong đó một máy tính phân tích dữ liệu huấn luyện và tạo ra một quy tắc chung để nó có thể làm theo bằng cách loại bỏ đi những dữ liệu không quan trọng.
* 1985 - Terry Sejnowski đã phát minh ra NetTalk, nó có thể học cách phát âm các từ giống như cách một đứa trẻ tập nói.
* 1990s - Machine Learning đã dịch chuyển từ cách tiếp cận hướng kiến thức (knowledge-driven) sang cách tiếp cận hướng dữ liệu (data-driven). Các nhà khoa học bắt đầu tạo ra các chương trình cho máy tính để phân tích một lượng lớn dữ liệu và rút ra các kết luận - hay là “học” từ các kết quả đó.
* 1997 - Deep Blue của hãng IBM đã đánh bại nhà vô địch cờ vua thế giới.
* 2006 - Geoffrey Hinton đã đưa ra một thuật ngữ “deep learning” để giải thích các thuật toán mới cho phép máy tính "nhìn thấy" và phân biệt các đối tượng và văn bản trong các hình ảnh và video.
* 2010 - Microsoft Kinect có thể theo dõi 20 hành vi của con người ở một tốc độ 30 lần mỗi giây, cho phép con người tương tác với máy tính thông qua các hành động và cử chỉ.
* 2011 - Máy tính Watson của hãng IBM đã đánh bại các đối thủ là con người tại Jeopardy.
* 2011 - Google Brain đã được phát triển, và mạng deep nơron (deep neural network) của nó có thể học để phát hiện và phân loại nhiều đối tượng theo cách mà một con mèo thực hiện.
* 2012 - X Lab của Google phát triển một thuật toán machine learning có khả năng tự động duyệt qua các video trên YouTube để xác định xem video nào có chứa những con mèo.
* 2014 - Facebook phát triển DeepFace, một phần mềm thuật toán có thể nhận dạng hoặc xác minh các cá nhân dựa vào hình ảnh ở mức độ giống như con người có thể.
* 2015 - Amazon ra mắt nền tảng machine learning riêng của mình.
* 2015 - Microsoft tạo ra Distributed Machine Learning Toolkit, trong đó cho phép phân phối hiệu quả các vấn đề machine learning trên nhiều máy tính.
* 2015 - Hơn 3.000 nhà nghiên cứu AI và Robotics, được sự ủng hộ bởi những nhà khoa học nổi tiếng như Stephen Hawking, Elon Musk và Steve Wozniak (và nhiều người khác), đã ký vào một bức thư ngỏ để cảnh báo về sự nguy hiểm của vũ khí tự động trong việc lựa chọn và tham gia vào các mục tiêu mà không có sự can thiệp của con người.
* 2016 - Thuật toán trí tuệ nhân tạo của Google đã đánh bại nhà vô địch trò chơi Cờ Vây, được cho là trò chơi phức tạp nhất thế giới (khó hơn trò chơi cờ vua rất nhiều). Thuật toán AlphaGo được phát triển bởi Google DeepMind đã giành chiến thắng 4/5 trước nhà vô địch Cờ Vây. [2]

## Phân loại các thuật toán học máy

### Học có giám sát (Supervised Learning)

Học có giám sát là loại học mà máy tính được đào tạo trên một tập dữ liệu đã được gán nhãn. Nói cách khác, trong bài toán này, chúng ta cung cấp cho máy tính cả dữ liệu đầu vào và đầu ra mong muốn (nhãn). Mục đích của việc học có giám sát là dự đoán kết quả đầu ra cho dữ liệu mới chưa được gán nhãn.

Học có giám sát lại được chia thành hai nhóm chính:

* Hồi quy (Regression): Khi kết quả đầu ra là một giá trị liên tục, chúng ta sử dụng thuật toán hồi quy. Ví dụ: dự đoán giá nhà, dự đoán doanh số.
* Phân lớp (Classification): Khi kết quả đầu ra là một giá trị rời rạc, chúng ta sử dụng thuật toán phân loại. Ví dụ: phân loại văn bản, nhận dạng khuôn mặt.

### Học không giám sát (Unsupervised Learning)

Học không giám sát xử lý vấn đề khi chúng ta không biết kết quả đầu ra mong muốn và không có nhãn cho dữ liệu. Mục tiêu của học không giám sát là khám phá cấu trúc ẩn trong dữ liệu và tìm ra một cách biểu diễn dữ liệu phù hợp.

Học không giám sát chủ yếu được chia thành hai nhóm:

* Phân cụm (Clustering): Nhóm các đối tượng tương tự lại với nhau. Ví dụ: phân loại khách hàng, phân loại tài liệu.
* Giảm chiều dữ liệu (Dimensionality Reduction): Giảm số lượng biến trong dữ liệu, giữ lại các thông tin quan trọng nhất. Ví dụ: giảm chiều dữ liệu ảnh, tìm các yếu tố tiềm ẩn trong thông tin người dùng.

### Học bán giám sát (Semi-supervised Learning)

Học bán giám sát kết hợp cả hai phương pháp học có giám sát và học không giám sát, trong đó một phần dữ liệu được gán nhãn và phần còn lại không. Mục tiêu của học bán giám sát là tận dụng lợi thế của cả hai phương pháp để cải thiện độ chính xác của mô hình.

### Học tăng cường (Reinforcement Learning)

Học tăng cường là một nhánh khác của Machine Learning, trong đó một hệ thống (hay được gọi là “agent”) được xây dựng để tự động học cách tương tác với môi trường của nó, thông qua quá trình áp dụng các hành động và nhận lại phản hồi từ chính môi trường. Mục tiêu của Reinforcement Learning là điều khiển một agent sao cho đạt được điểm thưởng tối đa từ môi trường dựa vào các hành động mà agent thực hiện.

Học tăng cường được ứng dụng rộng rãi trong các lĩnh vực như điều khiển robot, ứng dụng tài chính, hệ thống gợi ý, điều khiển tự động, trò chơi và nhiều ứng dụng khác nữa. [3]

## Tầm quan trọng của học máy

Học máy đã trở thành một trong những lĩnh vực quan trọng và phát triển mạnh mẽ trong thế kỷ 21, với ảnh hưởng sâu rộng vào mọi khía cạnh của cuộc sống và công nghiệp. Dưới đây là những lý do cho tầm quan trọng của học máy:

**Tăng Cường Khả Năng Dự Đoán và Quyết Định:** Học máy cung cấp công cụ và kỹ thuật để dự đoán và phân loại dữ liệu một cách chính xác và nhanh chóng hơn. Từ việc dự đoán xu hướng tiêu dùng đến phát hiện gian lận tín dụng, học máy giúp các tổ chức và cá nhân đưa ra quyết định thông minh và hiệu quả.

**Tối Ưu Hóa Các Quy Trình Công Nghiệp:** Trong nền kinh doanh và sản xuất, học máy được sử dụng để tối ưu hóa quy trình và quản lý chuỗi cung ứng. Từ dự đoán nhu cầu hàng hóa đến tối ưu hóa lịch trình sản xuất, học máy giúp tăng cường hiệu suất và giảm thiểu lãng phí.

**Tạo Ra Sản Phẩm và Dịch Vụ Hiệu Quả Hơn:** Học máy đóng vai trò quan trọng trong việc phát triển các sản phẩm và dịch vụ mới. Từ ứng dụng di động đến công nghệ y tế, học máy giúp tạo ra các sản phẩm và dịch vụ có khả năng tương tác thông minh và cá nhân hóa.

**Tiềm Năng Trong Y Tế và Y Dược:** Trong lĩnh vực y tế, học máy có thể giúp dự đoán các bệnh lý và phân loại dữ liệu y tế. Từ phát hiện sớm bệnh ung thư đến dự đoán biến chứng sau phẫu thuật, học máy đang cung cấp những cơ hội mới để nâng cao chăm sóc sức khỏe và cải thiện kết quả điều trị.

**Phát Triển Công Nghệ Tự Động Hóa:** Học máy là một trong những yếu tố chính đằng sau sự phát triển của công nghệ tự động hóa. Từ xe tự lái đến robot dịch vụ, học máy đang định hình một tương lai mà các hệ thống tự động có khả năng học và tương tác với môi trường một cách thông minh.

**Cải Thiện Trí Tuệ Nhân Tạo:** Học máy đóng vai trò quan trọng trong việc phát triển trí tuệ nhân tạo. Bằng cách học từ dữ liệu và kinh nghiệm, các hệ thống trí tuệ nhân tạo ngày càng trở nên thông minh và có khả năng tương tác với con người một cách tự nhiên.

**Tạo Ra Cơ Hội Kinh Doanh Mới:** Học máy không chỉ giúp tối ưu hóa các quy trình hiện có mà còn tạo ra cơ hội kinh doanh mới. Từ phát triển các dịch vụ trí tuệ nhân tạo đến khai thác dữ liệu để phát triển sản phẩm và dịch vụ mới, học máy mở ra một thế giới mới của cơ hội kinh doanh.

Nhìn chung, học máy không chỉ là một công cụ mạnh mẽ để xử lý dữ liệu mà còn là một yếu tố quyết định trong việc định hình tương lai của công nghệ và xã hội. Tầm quan trọng của học máy đang được nhận ra ngày càng nhiều và tiếp tục mở ra những cơ hội mới và tiềm năng lớn cho con người.

## Quy trình thực hiện một dự án học máy

**Bước 1: Thu thập dữ liệu**

Dữ liệu cung cấp có chính xác hay không, sẽ ảnh hưởng đến độ chính xác của mô hình.

Các dữ liệu tốt là những dữ liệu chứa đầy đủ các thông tin cần dùng cho mô hình Machine Learning và chúng lặp lại liên tục, đồng thời chúng thể hiện rõ sự khác biệt giữa từng nhóm dữ liệu.

**Bước 2: Chuẩn bị dữ liệu**

Sau khi đã có đầy đủ dữ liệu, bạn cần chuẩn bị và sắp xếp lại chúng, đồng thời ngẫu nhiên hóa nó. Điều này giúp cho dữ liệu có thể được phân bổ một cách đồng đều, đồng thời việc sắp xếp thứ tự không ảnh hưởng đến quá trình học.

Việc chuẩn bị dữ liệu cũng giúp loại bỏ những dữ liệu không mong muốn, các giá trị trùng lặp… Thậm chí cần phải cấu trúc lại tệp dữ liệu theo hàng hoặc cột... để dữ liệu có thể được trực quan hóa hơn.

**Bước 3: Chọn mô hình**

Mô hình Machine Learning sẽ xác định kết quả bạn nhận được, sau khi khởi chạy một thuật toán học máy dựa trên dữ liệu nhận được.

Tính đến ngày nay, các nhà khoa học đã phát triển nhiều mô hình khác nhau như nhận dạng giọng nói, nhận dạng hình ảnh, dự đoán, …

**Bước 4: Huấn luyện mô hình**

Huấn luyện là một trong những bước cực kỳ quan trọng trong xây dựng mô hình Machine Learning.

Trong quá trình huấn luyện, chúng ta sẽ chuyển dữ liệu đã chuẩn bị trước đó đến mô hình học máy của mình, để hệ thống tìm các mẫu hữu ích và đưa ra dự đoán.

Qua đó, hệ thống có thể học hỏi từ dữ liệu để hoàn thành những nhiệm vụ đã được đặt ra. Theo thời gian, cùng với quá trình huấn luyện và đào tạo, mô hình Machine Learning sẽ hoạt động và dự đoán tốt hơn.

**Bước 5: Đánh giá mô hình**

Sau khi đã huấn luyện mô hình, chúng ta cần kiểm tra lại xem chúng đã hoạt động chính xác như mong muốn chưa.

Bạn có thể đánh giá bằng cách kiểm tra kết quả phân tích của mô hình trên một dữ liệu chưa từng thấy trước đó, xem thử hệ thống có nhận dạng được đúng và dự đoán đúng hay không.

Khi đánh giá như vậy, sẽ có được một thước đo đúng về cách mô hình của bạn hoạt động, cũng như tốc độ xử lý thông tin và dự đoán của nó.

**Bước 6: Điều chỉnh tham số**

Sau khi chúng ta đã huấn luyện và đánh giá được mô hình Machine Learning của mình, bạn có thể điều chỉnh lại tham số nếu cần để tăng độ chính xác cho hệ thống.

Tham số được hiểu là các biến đã được lưu trong mô hình, đây thường là yếu tố được lập trình viên quyết định. Ở một giá trị cụ thể nào đó của tham số, độ chính xác sẽ có thể là tối đa. Chúng ta có thể thử điều chỉnh và tìm ra con số này.

**Bước 7: Đưa ra dự đoán:** Cuối cùng, chúng ta sẽ sử dụng mô hình Machine Learning mà mình đã tạo để phân tích thông tin và đưa ra dự đoán một cách chính xác. [3]

## Các ứng dụng của học máy dữ liệu trong y tế

Phân tích dữ liệu y tế có khả năng giảm chi phí điều trị, dự đoán sự bùng phát của dịch bệnh, tránh các bệnh có thể phòng ngừa và cải thiện chất lượng cuộc sống. Tuổi thọ trung bình của con người ngày càng tăng trên dân số thế giới, điều này đặt ra những thách thức mới đối với các phương pháp điều trọ ngày nay. Các chuyên gia y tế, cũng giống như các doanh nhân kinh doanh, có khả năng thu thập một lượng dữ liệu lớn và tìm kiếm các chiến lược tốt nhất để sử những con số này.

Và chúng ta có thể thấy được các lợi ích dụng của phân tích dữ liệu trong y tế như sau:

* **Dự đoán bệnh nhân để cải thiện nhân sự:**

Một trong những bộ dữ liệu quan trong là hồ sơ nhập viện trí giá 10 năm được các nhà khoa học dữ liệu thu thập bằng kỹ thuật “phân tích chuỗi thời gian”. Những phân tích này cho phép các nhà nghiên cứu thấy được các mô hình liên quan trong tỷ lệ nhập viện. Sau đó, có thể sử dụng máy học để tìm ra các thuật toán chính xác nhất dự đoán chính xác nhất dự đoán xu hướng nhập viên trong tương lai.

* **Hồ sơ sức khỏe điện tử:**

Đây là ứng dụng rộng rãi nhất của trong y học. Mỗi bệnh nhân đều có hồ sơ kỹ thuật số của riêng mình, bao gồm nhân khẩu học, lịch sử bệnh, dị ứng, kết quả xét nghiệm trong phòng thí nghiệm, ... Hồ sơ được chia sẻ qua hệ thống thông tin an toàn và có sẵn cho các nhà cung cấp từ cả khu vực công và tư nhân. Mọi hồ sơ đều bao gồm một tệp có thể sửa đổi, có nghĩa là bác sĩ có thể thực hiện các thay đổi theo thời gian mà không cần thủ tục giấy tờ và không có nguy cơ sao chép dữ liệu.

Hồ sơ sức khỏe điện tử cũng có thể kích hoạt cảnh báo và nhắc nhở khi bệnh nhân nên làm xét nghiệm mới trong phòng thí nghiệm hoặc theo dõi các đơn thuốc để xem liệu bệnh nhân có tuân theo chỉ định của bác sĩ hay không.

* **Cảnh báo thời gian thực:**

Một số thiết bị đeo tay hiện nay có chức năng liên tục thu thập dữ liệu sức khỏe của bệnh nhân và gửi dữ liệu lên đám mây. Thông tin này sẽ được truy cập vào cơ sở dữ liệu về tình trạng sức khỏe của người dân, cho phép các bác sĩ so sánh dữ liệu này trong bối cảnh kinh tế xã hội và sửa đổi các chiến lược phân phối cho phù hợp. Các tổ chức và các nhà quản lý chăm sóc sẽ sử dụng các công cụ tinh vi để theo dõi dòng dữ liệu khổng lồ này và phản ứng lại mỗi khi kết quả bị xáo trộn.

Ví dụ: Nếu huyết áp của bệnh nhân tăng cao đáng báo động, hệ thống sẽ gửi cảnh báo theo thời gian thực cho bác sĩ, sau đó sẽ thực hiện hành động tiếp cận bệnh nhân và thực hiện các biện pháp để giảm huyết áp.

* **Sử dụng dữ liệu sức khỏe để lập kế hoạch chiến lược:**

Việc sử dụng dữ liệu lớn trong chăm sóc sức khỏe cho phép lập kế hoạch chiến lược nhờ hiểu rõ hơn về động cơ của con người. Người quản lý chăm sóc có thể phân tích kết quả kiểm tra giữa những người thuộc các nhóm nhân khẩu học khác nhau và xác định những yếu tố nào không khuyến khích mọi người tiếp tục điều trị.

# CÔNG CỤ VÀ MÔI TRƯỜNG THỰC HIỆN

## Giới thiệu về ngôn ngữ lập trình Python

Pythonlà ngôn ngữ lập trình bậc cao, phục vụ cho các mục đích lập trình đa năng. Ưu điểm nổi bật nhất đó chính là dễ đọc, dễ nhớ, dễ học. Python là ngôn ngữ có cấu trúc tương đối rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của python còn cho phép người dùng sử dụng để viết mã lệnh với số lần gõ phím tối thiểu.

Nói cách khác, khái niệm lập trình python là gì còn được hiểu là cửa ngõ để mọi người bước vào thế giới lập trình máy tính hay là một phương tiện để chúng ta nhận được một khoản lương “đáng mơ ước” từ một công việc sáng tạo.

Python mang bản chất là ngôn ngữ lập trình bậc cao, được tạo ra bởi Guido van Rossum. Ngôn ngữ python được thiết kế hướng tới đối tượng với cấu trúc hàng và cách xử lý dữ liệu đơn giản, dễ đọc. Nó sẽ giúp người dùng tạo ra những chương trình hay với số lượng dòng code ít nhất.

Python là ngôn ngữ lập trình với mã nguồn mở đa mục đích, hỗ trợ nhiều phong cách lập trình cốt lõi khác nhau, thiết kế của python đem tới sự thuận lợi trong việc đọc hiểu code, đơn giản và rõ ràng.

Python là một ngôn ngữ lập trình được sử dụng rộng rãi trong các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và máy học (ML). Các nhà phát triển sử dụng Python vì nó hiệu quả, dễ học và có thể chạy trên nhiều nền tảng khác nhau. Phần mềm Python được tải xuống miễn phí, tích hợp tốt với tất cả các loại hệ thống và tăng tốc độ phát triển.

**Lịch sử**

Python được hình thành vào cuối những năm 1980. Thời gian đó, Guido van Rossum làm việc trong một dự án tại CWI, có tên là Amoeba, một hệ điều hành phân tán. Trong một cuộc phỏng vấn với Bill Venners1, Guido van Rossum cho biết: “Vào đầu những năm 1980, tôi làm việc với tư cách là người triển khai một nhóm xây dựng ngôn ngữ có tên là ABC tại Centrum voor Wiskunde en Informatica (CWI). Tôi không biết mọi người hiểu như thế nào về Ảnh hưởng của ABC đối với Python. Tôi cố gắng đề cập đến ảnh hưởng của ABC vì tôi mang ơn tất cả những gì tôi đã học được trong dự án đó và những người đã làm việc với nó.”

Sau đó trong cùng một Cuộc phỏng vấn, Guido van Rossum tiếp tục: “Tôi nhớ lại tất cả kinh nghiệm và những điều không hài lòng của mình với ABC. Tôi quyết định cố gắng thiết kế một ngôn ngữ kịch bản đơn giản có một số đặc tính tốt hơn của ABC. Vì vậy, tôi bắt đầu nhập. Tôi đã tạo một máy ảo đơn giản, một trình phân tích cú pháp đơn giản và một thời gian chạy đơn giản. Tôi đã tạo phiên bản của riêng mình cho các phần ABC khác nhau mà tôi thích. Tôi đã tạo một cú pháp cơ bản, sử dụng thụt lề để nhóm câu lệnh thay vì dấu ngoặc nhọn hoặc khối begin-end và phát triển một số lượng nhỏ các kiểu dữ liệu mạnh mẽ: bảng băm (hoặc từ điển, như chúng tôi gọi), danh sách, chuỗi và số.”

Guido Van Rossum đã xuất bản phiên bản đầu tiên của mã Python (phiên bản 0.9.0) tại alt.sources vào tháng 2 năm 1991. Bản phát hành này đã bao gồm xử lý ngoại lệ, các hàm và các kiểu dữ liệu cốt lõi của list, dict, str và các loại khác. Nó cũng hướng đối tượng và có một hệ thống mô-đun. Phiên bản Python 1.0 được phát hành vào tháng 1 năm 1994. Các tính năng mới chính trong bản phát hành này là các công cụ lập trình chức năng lambda, map, filter và reduce, những thứ mà Guido Van Rossum không bao giờ thích. Sáu năm rưỡi sau vào tháng 10 năm 2000, Python 2.0 được giới thiệu. Bản phát hành này bao gồm toàn bộ danh sách, một bộ thu gom rác đầy đủ và nó hỗ trợ unicode.

Bắt đầu từ năm 2000, các nhà phát triển cốt lõi bắt đầu nghĩ về Python 3.0. Họ muốn hợp lý hóa ngôn ngữ, cắt các cấu trúc và chức năng ngôn ngữ không cần thiết mà Python đã tích lũy được trong gần 20 năm tồn tại của nó. Như Zen of Python nói: “Nên có một — và tốt nhất là chỉ một — cách rõ ràng để làm điều đó”. Những nỗ lực của họ đã dẫn đến Python 3.0, một phiên bản không tương thích ngược của ngôn ngữ Python được phát hành vào tháng 12 năm 2008. Thật không may, bản phát hành đã mang lại một số phức tạp. Các nhà phát triển đã không nhận ra bao nhiêu Python được sử dụng và bao nhiêu mã Python ngoài tự nhiên phụ thuộc vào các thư viện Python khác. Do đó, mặc dù dễ dàng chuyển các tập lệnh của một người sang Python 3, nhưng việc di chuyển các chương trình dựa vào thư viện của bên thứ ba lại khó hơn nhiều vì chúng không nâng cấp nhanh như vậy. Điều này dẫn đến việc chuyển đổi khá rắc rối và gây đau đớn cho một số người nhưng đã cải thiện đáng kể ngôn ngữ. Python 2 cuối cùng đã ngừng hoạt động vào năm 2020.

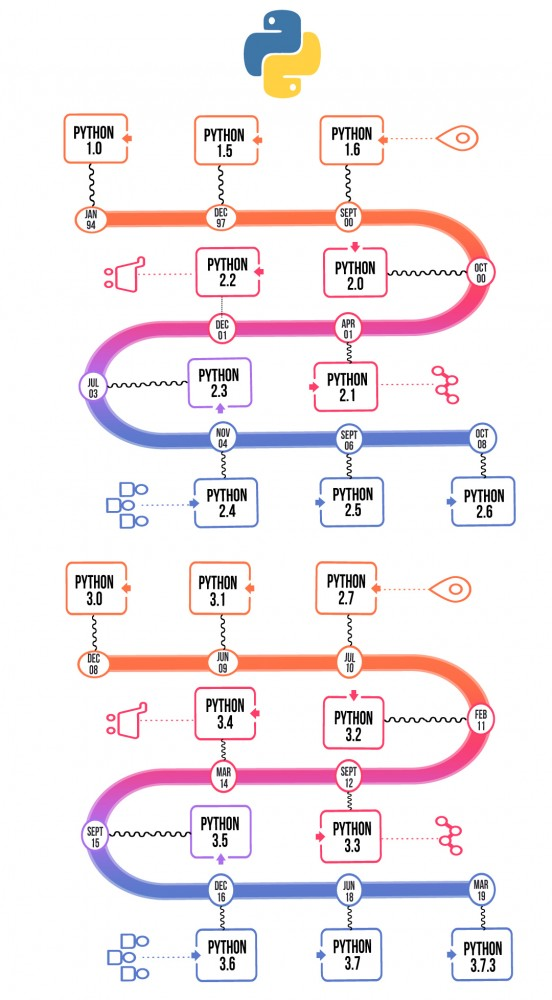
Năm 1999, Guido van Rossum xác định mục tiêu của mình cho Python:

* Một ngôn ngữ dễ dàng và trực quan cũng mạnh mẽ như ngôn ngữ của các đối thủ cạnh tranh lớn;
* Nguồn mở, vì vậy bất kỳ ai cũng có thể đóng góp vào sự phát triển của nó;
* Mã dễ hiểu như tiếng Anh đơn giản;
* Thích hợp cho các công việc hàng ngày , cho phép thời gian phát triển ngắn.

Khoảng 20 năm sau, rõ ràng là tất cả những dự định này đã được thực hiện. Mặc dù Python đang phát triển ổn định về cơ bản toàn bộ sự tồn tại của nó, từ khoảng năm 2010, nó bắt đầu trên một quỹ đạo phát triển khiến nó sớm cho phép nó sánh ngang với các ngôn ngữ lập trình hàng đầu khác, chẳng hạn như Java và JavaScript.

Đặc biệt, hành trình học máy của Python đã đạt đến điểm uốn vào khoảng tháng 9 năm 2016, theo Google Trends. Đây là một năm sau khi phát hành TensorFlow, thư viện ML của Google dựa trên học sâu và phản ánh sự gia tăng tương tự trong mối quan tâm toàn cầu đối với học máy.

Python đã trải qua một chặng đường dài để trở thành ngôn ngữ viết mã phổ biến nhất trên thế giới. Python đã là nguồn cảm hứng cho nhiều ngôn ngữ mã hóa khác như Ruby, Cobra, Boo, CoffeeScript ECMAScript, Groovy, Swift Go, OCaml, Julia, v.v.



Hình 1: Dòng thời gian phát triển của các phiên bản Python

**Các tính năng của python bao gồm:**

* Dễ học: Python có ít từ khóa, cấu trúc đơn giản và cú pháp được định nghĩa rõ ràng. Điều này cho phép người mới học tiếp cận ngôn ngữ một cách nhanh chóng.
* Dễ đọc: Mã Python được định nghĩa rõ ràng hơn và có thể nhìn thấy bằng mắt.
* Dễ bảo trì: Mã nguồn của Python khá dễ bảo trì.
* Một thư viện tiêu chuẩn rộng: Phần lớn thư viện của Python rất dễ đính kèm và đa nền tảng tương thích trên UNIX, Windows và Macintosh.
* Chế độ tương tác: Python có hỗ trợ cho chế độ tương tác cho phép kiểm tra tương tác và debug
* Portable: Python có thể chạy trên nhiều nền tảng phần cứng khác nhau và có cùng giao diện trên tất cả các nền tảng.
* Có thể mở rộng: chúng ta có thể thêm các module cấp thấp vào trình thông dịch Python. Các module này cho phép các lập trình viên thêm hoặc tùy chỉnh các công cụ của mình để hiệu quả hơn.
* Cơ sở dữ liệu: Python cung cấp phương thức giao tiếp cho tất cả các cơ sở dữ liệu.
* Lập trình GUI: Python hỗ trợ các ứng dụng GUI có thể được tạo và chuyển sang nhiều cuộc gọi hệ thống, thư viện và hệ thống cửa sổ, như Windows MFC, Macintosh và hệ thống X Window của Unix.
* Khả năng mở rộng: Python cung cấp cấu trúc và hỗ trợ tốt hơn cho các chương trình lớn hơn so với kịch bản lệnh shell.

**Những lợi ích của Python bao gồm:**

* Các nhà phát triển có thể dễ dàng đọc và hiểu một chương trình Python vì ngôn ngữ này có cú pháp cơ bản giống tiếng Anh.
* Python giúp cải thiện năng suất làm việc của các nhà phát triển vì so với những ngôn ngữ khác, họ có thể sử dụng ít dòng mã hơn để viết một chương trình Python.
* Python có một thư viện tiêu chuẩn lớn, chứa nhiều dòng mã có thể tái sử dụng cho hầu hết mọi tác vụ. Nhờ đó, các nhà phát triển sẽ không cần phải viết mã từ đầu.
* Các nhà phát triển có thể dễ dàng sử dụng Python với các ngôn ngữ lập trình phổ biến khác như Java, C và C++.
* Cộng đồng Python tích cực hoạt động bao gồm hàng triệu nhà phát triển nhiệt tình hỗ trợ trên toàn thế giới. Nếu gặp phải vấn đề, bạn sẽ có thể nhận được sự hỗ trợ nhanh chóng từ cộng đồng.
* Trên Internet có rất nhiều tài nguyên hữu ích nếu bạn muốn học Python. Ví dụ: bạn có thể dễ dàng tìm thấy video, chỉ dẫn, tài liệu và hướng dẫn dành cho nhà phát triển.
* Python có thể được sử dụng trên nhiều hệ điều hành máy tính khác nhau, chẳng hạn như Windows, macOS, Linux và Unix.

**Ngôn ngữ Python được sử dụng nhiều trong lĩnh vực phát triển ứng dụng, bao gồm những ví dụ như sau:**

* Phát triển web phía máy chủ

Phát triển web phía máy chủ bao gồm những hàm backend phức tạp mà các trang web thực hiện để hiển thị thông tin cho người dùng. Ví dụ: các trang web phải tương tác với cơ sở dữ liệu, giao tiếp với các trang web khác và bảo vệ dữ liệu khi truyền qua mạng.

Python hữu ích trong việc lập trình mã phía máy chủ bởi vì ngôn ngữ này cung cấp nhiều thư viện bao gồm mã viết sẵn cho các hàm backend phức tạp. Các nhà phát triển cũng sử dụng một loạt các khung Python cung cấp tất cả những công cụ cần thiết để xây dựng ứng dụng web một cách nhanh chóng và dễ dàng hơn.

Ví dụ: các nhà phát triển có thể tạo ứng dụng web khung trong nháy mắt bởi vì họ không cần phải lập trình nó từ đầu. Sau đó, họ có thể kiểm tra ứng dụng web này bằng cách sử dụng các công cụ kiểm thử của khung, mà không cần phụ thuộc vào những công cụ kiểm thử bên ngoài.

* Tự động hóa bằng các tập lệnh Python

Ngôn ngữ tập lệnh là một ngôn ngữ lập trình tự động hóa các tác vụ mà thường được con người thực hiện. Các lập trình viên thường xuyên sử dụng các tập lệnh Python để tự động hóa nhiều tác vụ hàng ngày như:

* Đổi tên một số lượng lớn tệp cùng lúc
* Chuyển đổi một tệp sang một loại tệp khác
* Loại bỏ các từ trùng lặp trong tệp văn bản
* Thực hiện các phép tính toán cơ bản
* Gửi email
* Tải xuống nội dung
* Thực hiện phân tích nhật ký cơ bản
* Tìm kiếm lỗi trong nhiều tệp
* Khoa học dữ liệu và máy học

Khoa học dữ liệu trích xuất thông tin quý giá từ dữ liệu và máy học (ML) dạy máy tính tự động học hỏi từ dữ liệu và đưa ra các dự đoán chính xác. Các nhà khoa học dữ liệu sử dụng Python cho các tác vụ khoa học dữ liệu sau:

* Sửa và loại bỏ dữ liệu không chính xác, hay còn được gọi là làm sạch dữ liệu
* Trích xuất và chọn lọc các đặc điểm đa dạng của dữ liệu
* Ghi nhãn dữ liệu gán tên có ý nghĩa cho dữ liệu
* Tìm các số liệu thống kê khác nhau từ dữ liệu
* Trực quan hóa dữ liệu bằng cách sử dụng các biểu đồ và đồ thị, chẳng hạn như biểu đồ đường, biểu đồ cột, biểu đồ tần suất và biểu đồ tròn

Các nhà khoa học dữ liệu sử dụng những thư viện ML của Python để đào tạo các mô hình ML và xây dựng các công cụ phân loại giúp phân loại dữ liệu một cách chính xác. Các chuyên gia từ nhiều lĩnh vực sử dụng những công cụ phân loại dựa trên Python để thực hiện các tác vụ phân loại, chẳng hạn như phân loại hình ảnh, văn bản cũng như lưu lượng truy cập mạng, nhận dạng giọng nói và nhận diện khuôn mặt. Các nhà khoa học dữ liệu cũng sử dụng Python cho deep learning, một kỹ thuật ML nâng cao.

* Phát triển phần mềm

Các nhà phát triển phần mềm thường sử dụng Python cho những tác vụ phát triển và ứng dụng phần mềm khác nhau, chẳng hạn như:

* Theo dõi lỗi trong mã của phần mềm
* Tự động xây dựng phần mềm
* Đảm nhận quản lý dự án phần mềm
* Phát triển nguyên mẫu phần mềm
* Phát triển các ứng dụng máy tính bằng cách sử dụng những thư viện Giao diện đồ họa người dùng (GUI)
* Phát triển từ các trò chơi văn bản đơn giản cho đến những trò chơi điện tử phức tạp
* Tự động hóa kiểm thử phần mềm

Kiểm thử phần mềm là quy trình kiểm tra xem kết quả thực tế từ phần mềm có khớp với kết quả mong đợi không để đảm bảo rằng phần mềm không có lỗi.

Các nhà phát triển sử dụng khung kiểm thử đơn vị Python, chẳng hạn như Unittest, Robot và PyUnit, để kiểm thử các hàm do họ viết.

Các kỹ sư kiểm thử phần mềm sử dụng Python để viết các trường hợp kiểm thử cho nhiều tình huống khác nhau. Ví dụ: họ sử dụng ngôn ngữ này để kiểm thử giao diện người dùng của một ứng dụng web, nhiều thành phần của phần mềm và những tính năng mới.

Các nhà phát triển có thể sử dụng một số công cụ để tự động chạy tập lệnh kiểm thử. Những công cụ này có tên gọi là công cụ Tích hợp liên tục/Triển khai liên tục (CI/CD). Các kỹ sư kiểm thử phần mềm cũng như những nhà phát triển sử dụng các công cụ CI/CD như Travis CI và Jenkins để tự động hóa quy trình kiểm thử. Công cụ CI/CD tự động chạy các tập lệnh kiểm thử Python và báo cáo kết quả kiểm thử bất kỳ khi nào nhà phát triển thêm vào những dòng mã mới.

**Các thư viện Python phổ biến nhất:**

* Matplotlib

Các nhà phát triển sử dụng Matplotlib để hiển thị dữ liệu dưới dạng đồ họa hai và ba chiều (2D và 3D) chất lượng cao. Thư viện này thường được sử dụng trong các ứng dụng khoa học. Với Matplotlib, bạn có thể trực quan hóa dữ liệu bằng cách hiển thị dữ liệu dưới dạng các biểu đồ khác nhau, chẳng hạn như biểu đồ cột và biểu đồ đường. Bạn cũng có thể hiển thị nhiều biểu đồ cùng lúc và các chi tiết đồ họa có thể được di chuyển qua mọi nền tảng.

* Pandas

Pandas cung cấp cấu trúc dữ liệu được tối ưu hóa và linh hoạt mà bạn có thể sử dụng để thao tác với dữ liệu chuỗi thời gian và dữ liệu có cấu trúc, chẳng hạn như bảng và nhóm. Ví dụ, bạn có thể sử dụng Pandas để đọc, ghi, hợp nhất, lọc và nhóm dữ liệu. Thư viện này được nhiều người sử dụng cho các tác vụ khoa học dữ liệu, phân tích dữ liệu và ML.

* NumPy

NumPy là một thư viện phổ biến mà các nhà phát triển sử dụng để dễ dàng tạo và quản lý nhóm, thao tác với các hình dạng logic và thực hiện các phép toán đại số tuyến tính. NumPy hỗ trợ tích hợp với nhiều ngôn ngữ như C và C ++.

* Requests

Thư viện Requests cung cấp nhiều hàm hữu ích cần thiết để phát triển web. Bạn có thể sử dụng thư viện này để gửi các yêu cầu HTTP, bổ sung tiêu đề, thêm tham số URL, bổ sung dữ liệu và thực hiện nhiều tác vụ khác khi giao tiếp với các ứng dụng web.

* OpenCV-Python

OpenCV-Python là một thư viện mà các nhà phát triển sử dụng để xử lý hình ảnh cho các ứng dụng thị giác máy tính. Thư viện này cung cấp nhiều hàm cho các tác vụ xử lý hình ảnh như đọc và ghi hình ảnh cùng lúc, xây dựng môi trường 3D từ môi trường 2D cũng như chụp và phân tích hình ảnh từ video.

* Keras

Keras là thư viện mạng nơ-ron chuyên sâu của Python với khả năng hỗ trợ tuyệt vời cho việc xử lý dữ liệu, trực quan hóa và hơn thế nữa. Keras hỗ trợ nhiều mạng nơ-ron. Thư viện này có cấu trúc mô-đun mang lại sự linh hoạt cho việc lập trình các ứng dụng sáng tạo. [4]

## Một số thư viện quan trong cho phân tích dữ liệu và học máy

Thư viện Python là một bộ tính năng trong Python giúp loại bỏ các thao tác thừa thãi khi viết lại các mã lệnh từ đầu. Các thư viện này đóng một vai trò cốt lõi trong việc phát triển Machine Learning, khoa học dữ liệu, Data Visualization, ...

### Thư viện Pandas

Nếu chúng ta đang hoạt động trong lĩnh vực khoa học dữ liệu, chắc hẳn phải biết đến thư viện lập trình Pandas (Python data analysis) và tầm quan trọng của nó trong vòng đời khoa học dữ liệu. Cùng với NymPy trong Matplotlib, Pandas là thư viện Python được sử dụng phổ biến nhất trong nghiên cứu và phát triển các ứng dụng khoa học dữ liệu. Hiện nay, số lượng bình luận trên GitHub về thư viện này đã lên đến 17.000, không thể không kể đến cộng đồng rộng rãi gồm 1.200 contributors.



Hình 2: Thư viện Pandas

Là một thư viện Python với mã nguồn mở, Pandas cung cấp cho người dùng các cấu trúc dữ liệu high-level và rất nhiều công cụ cần thiết cho thao tác phân tích dữ liệu dựa trên cấu trúc dữ liệu Dataframe. Cấu trúc dữ liệu này đem lại cho Pandas tính linh hoạt và hiệu năng đỉnh cao khi thao tác với các dữ liệu phức tạp và lập chỉ mục. Ngoài ra, thư viện lập trình này còn sở hữu rất nhiều công cụ inbuild dành cho thao tác nhóm, kết hợp và lọc dữ liệu, cũng như tính năng với dãy số thời gian.

Pandas luôn đảm bảo toàn bộ quá trình thao tác với dữ liệu được đơn giản hóa chỉ với một hoặc hai lệnh. Các tính năng nổi bật khác của thư viện Python này có thể kể đến như reindex, iteration, sorting, aggregation, concatenation và visualization.

Tổng hợp lại, dưới đây là những gì chúng ta có thể làm với thư viện Python Pandas:

* Dễ dàng xử lý dữ liệu bị thiếu với các cú pháp mạnh mẽ và tính năng mượt mà
* Tự tạo hàm và chạy hàm với các dải dữ liệu
* Tính trừu tượng (abstraction) của ứng dụng được đẩy cao
* Tạo ra ứng dụng sở hữu cấu trúc dữ liệu và công cụ hỗ trợ cao cấp
* Chọn đầu ra phù hợp nhất với phương thức áp dụng
* Chỉ số tốc độ tối ưu

Pandas được ứng dụng phổ biến trong khoa học dữ liệu, cụ thể như sau:

* Sắp xếp dữ liệu (data wrangling) và làm sạch dữ liệu (data cleaning)
* ETL sử dụng cho chuyển đổi dữ liệu (data transformation) và lưu trữ dữ liệu (data storage)
* Các thao tác phân tích dữ liệu với dãy số thời gian như tạo phạm vi dữ liệu ngày, di chuyển window, hồi quy tuyến tính,...
* Ứng dụng trong lĩnh vực giáo dục và thương mại về các mảng thống kê, tài chính và khoa học thần kinh.

**Lịch sử:**

Nhà phát triển Wes McKinney bắt đầu làm việc trên Pandas vào năm 2008 khi làm việc tại AQR Capital Management vì nhu cầu về một công cụ linh hoạt, hiệu suất cao để thực hiện phân tích định lượng trên dữ liệu tài chính. Trước khi rời AQR, anh đã thuyết phục được ban giám đốc cho phép anh mở nguồn thư viện.

Một nhân viên khác của AQR, Chang She, đã tham gia nỗ lực vào năm 2012 với tư cách là người đóng góp lớn thứ hai cho thư viện.

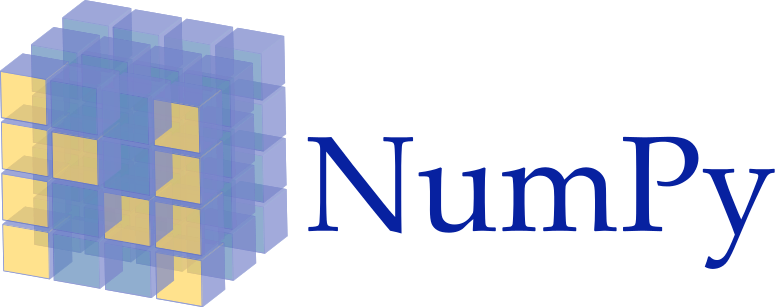
Vào năm 2015, Pandas đã ký kết với tư cách là một dự án được tài trợ về mặt tài chính của NumFOCUS, một tổ chức từ thiện phi lợi nhuận 501 ở Hoa Kỳ.

Tiến trình:

* 2008: Bắt đầu phát triển Pandas
* 2009: Pandas trở thành mã nguồn mở
* 2012: Ấn bản đầu tiên của Python để phân tích dữ liệu được xuất bản
* 2015: Pandas trở thành dự án do NumFOCUS tài trợ
* 2018: Chạy nước rút nhà phát triển cốt lõi trực tiếp đầu tiên. [5]

### Thư viện Numpy

NumPy, hay còn gọi là Numerical Python, là một trong các thư viện Python nổi tiếng nhất dành cho khoa học điện toán. Sử dụng các hàm tối ưu của mình, NumPy hỗ trợ người dùng Python với các thao tác tính toán mảng đa chiều với kích thước lớn. Thư viện này hiện có gần 18.000 bình luận trên GitHub và cộng đồng 700 contributor.



Hình 3: Thư viện NumPy

Có một sự thật thú vị là TensorFlow và các thư viện lập trình khác cũng ứng dụng NumPy vào trong một số operation trên tensor. Một trong các tính năng nổi trội. Và quan trọng hàng đầu của thư viện này chính là NumPy array, một đối tượng mảng n-chiều ở dạng hàng và cột.

Bên cạnh đó, NumPy cho phép người dùng sử dụng các tính năng mạnh mẽ khác như sau:

* Giao diện thân thiện với người dùng và đẩy mạnh tương tác
* Các công thức toán học phức tạp đã được tối giản
* Hỗ trợ tiếp cận hướng đối tượng
* Đóng góp nguồn mở

Với các tính năng mạnh mẽ của mình, NumPy có thể được ứng dụng vào các công nghệ dưới đây:

* Phân tích dữ liệu chuyên sâu
* Biểu diễn hình ảnh, sóng âm và các luồng nhị phân dưới dạng mảng số thực nhiều chiều
* Nền tảng xây dựng các thư viện Python khác, như SciPy và SciKit-Learn
* Thay thế MATLAB khi kết hợp sử dụng SciPy và matplotlib

**Lịch sử:**

Ngôn ngữ lập trình Python ban đầu không được thiết kế cho tính toán số, nhưng đã thu hút sự chú ý của cộng đồng khoa học và kỹ thuật từ rất sớm. Năm 1995, nhóm sở thích đặc biệt (SIG) ma trận-sig được thành lập với mục đích xác định một gói tính toán mảng; trong số các thành viên của nó có nhà thiết kế và bảo trì Python Guido van Rossum, người đã mở rộng cú pháp của Python (đặc biệt là cú pháp lập chỉ mục để làm cho việc tính toán mảng dễ dàng hơn.

Việc triển khai một gói ma trận đã được hoàn thành bởi Jim Fulton, sau đó được Jim Hugunin khái quát [cần giải thích thêm] và được gọi là Numeric (còn được gọi là "Phần mở rộng Python số" hoặc "NumPy"). Hugunin, một nghiên cứu sinh tại Học viện Công nghệ Massachusetts (MIT), gia nhập Công ty Sáng kiến ​​Nghiên cứu Quốc gia (CNRI) vào năm 1997 để làm việc trên JPython, rời bỏ Paul Dubois của Phòng thí nghiệm Quốc gia Lawrence Livermore (LLNL ) để đảm nhận vai trò người bảo trì. Những người đóng góp ban đầu khác bao gồm David Ascher, Konrad Hinsen và Travis Oliphant.

Một gói mới có tên Numarray được viết như một sự thay thế linh hoạt hơn cho Numeric. Giống như Numeric, nó hiện cũng không được dùng nữa. Numarray có các hoạt động nhanh hơn đối với các mảng lớn, nhưng lại chậm hơn Numeric trên các mảng nhỏ, vì vậy trong một thời gian, cả hai gói được sử dụng song song cho các trường hợp sử dụng khác nhau. Phiên bản cuối cùng của Numeric (v24.2) được phát hành vào ngày 11 tháng 11 năm 2005, trong khi phiên bản cuối cùng của numarray (v1.5.2) được phát hành vào ngày 24 tháng 8 năm 2006.

Có mong muốn đưa Numeric vào thư viện chuẩn Python, nhưng Guido van Rossum quyết định rằng mã không thể bảo trì ở trạng thái của nó sau đó.

Vào đầu năm 2005, nhà phát triển NumPy Travis Oliphant muốn thống nhất cộng đồng xung quanh một gói mảng duy nhất và chuyển các tính năng của Numarray sang Numeric, phát hành kết quả là NumPy 1.0 vào năm 2006. Dự án mới này là một phần của SciPy. Để tránh cài đặt gói SciPy lớn chỉ để lấy một đối tượng mảng, gói mới này đã được tách ra và được gọi là NumPy. Hỗ trợ cho Python 3 đã được thêm vào năm 2011 với phiên bản NumPy 1.5.0.

Năm 2011, PyPy bắt đầu phát triển việc triển khai API NumPy cho PyPy. Nó chưa hoàn toàn tương thích với NumPy. [6]

### Thư viện Matplotlib

Nói đến phân tích dữ liệu, chúng ta không thể không nhắc đến data visualization. Khi cần thực hiện các thao tác thống kê trên bảng biểu, hãy tìm kiếm Matplotlib. Đây là một thư viện Python sở hữu công cụ trực quan hóa dữ liệu vô cùng đẹp mắt mà không hề làm phai mờ sức mạnh của chúng. Chính bởi khả năng tạo ra các bảng biểu đỉnh cao, Matplotlib chính là vị cứu tinh dành cho data visualization. Hiện tại, thư viện lập trình này đã có gần 26.000 bình luận trên GitHub và cộng đồng gần 700 contributors.



Hình 4: Thư viên Matplotlib

Matplotlib là một trong những thư viện Python phổ biến nhất được sử dụng để trực quan hóa dữ liệu. Nó là một thư viện đa nền tảng để tạo các biểu đồ 2D từ dữ liệu trong các mảng. Nó cung cấp một API hướng đối tượng giúp nhúng các biểu đồ vào các ứng dụng bằng cách sử dụng bộ công cụ GUI của Python như PyQt, WxPythonotTkinter. Nó có thể được sử dụng trong trình bao Python và IPython, Jupyter notebook và máy chủ ứng dụng web.

Matplotlib là một thư viện vẽ biểu đồ cấp thấp trong python phục vụ như một tiện ích trực quan hóa.

Matplotlib được tạo bởi John D. Hunter.

Matplotlib là nguồn mở và chúng tôi có thể sử dụng nó một cách tự do.

Matplotlib chủ yếu được viết bằng python, một số phân đoạn được viết bằng C, Objective-C và Javascript để tương thích với Nền tảng.

Được tin dùng như một thư viện mã nguồn mở số 1 với các biểu đồ, Matplotlib cung cấp một số tính năng nổi bật bao gồm:

* Giao diện hướng đối tượng cho phép người dùng kiểm soát hoàn toàn các thuộc tính của biểu đồ, bao gồm trục, phông chữ, đương kẻ.
* Tạo các biểu đồ phân tán hoàn hảo
* Đơn giản hóa đồ thị qua giao diện tương tự MATLAB
* Hỗ trợ các trục thứ cấp
* Chạy mượt mà trên các hệ thống đồ họa và hệ điều hành khác nhau

Với các công cụ của thư viện lập trình này, chúng ta có thể:

* Vẽ các biểu đồ publication quality chỉ với một vài mã lệnh
* Sử dụng các biểu đồ mang tính tương tác với chức năng zoom, xoay chuyển, tự động cập nhật dữ liệu,...
* Xuất file dưới nhiều định dạng

Như vậy, chúng ta có thể dễ dàng hình dung tính năng của thư viện Python Matplotlib tương tự như tính năng vẽ biểu đồ trong Excel. Tuy nhiên, trong khi các biểu đồ Excel bị giới hạn về tính linh hoạt, chắc chắn rằng những gì chúng ta có thể làm được với Matplotlib là không giới hạn. [7]

### Thư viện Scikit-learn

Scikit-learn là một thư viện mã nguồn mở, miễn phí được viết bằng ngôn ngữ Python, chuyên dành cho các ứng dụng học máy và thống kê. Nó được xem là một trong những thư viện học máy phổ biến nhất hiện nay, được sử dụng rộng rãi bởi các nhà khoa học dữ liệu, kỹ sư phần mềm và những người đam mê học máy trên toàn thế giới.

Scikit-learn, được phát triển lần đầu vào năm 2007, là một thư viện học máy mã nguồn mở trong Python. Nó được xây dựng dựa trên thư viện NumPy, SciPy và Matplotlib, và ngày nay đã trở thành một trong những thư viện học máy phổ biến và được sử dụng rộng rãi nhất.

Thư viện được tạo ra nhằm mục đích cung cấp một công cụ dễ sử dụng, mang lại sự linh hoạt và hiệu quả trong việc xây dựng các mô hình học máy. Sự phát triển của Scikit-learn chịu ảnh hưởng từ nhu cầu ngày càng tăng về việc tự động hóa và hiểu biết sâu sắc hơn về dữ liệu.

A logo of a person

Description automatically generated with medium confidence

Hình 5: Thư viện Scikit-learn

Điểm nổi bật của Scikit-learn:

* Dễ sử dụng: Scikit-learn cung cấp API đơn giản, dễ hiểu, giúp người dùng có thể nhanh chóng bắt đầu thực hiện các dự án học máy.
* Hiệu quả: Thư viện được xây dựng dựa trên các thư viện Python tối ưu hóa như NumPy, SciPy và Matplotlib, đảm bảo hiệu suất cao trong quá trình xử lý dữ liệu và thực thi các thuật toán học máy.
* Đa dạng: Scikit-learn cung cấp một bộ sưu tập phong phú các thuật toán học máy cho nhiều nhiệm vụ khác nhau, bao gồm:
* Phân loại: Phân loại dữ liệu vào các nhóm khác nhau.
* Hồi quy: Dự đoán các giá trị liên tục dựa trên dữ liệu đầu vào.
* Phân cụm: Nhóm các dữ liệu tương tự nhau lại với nhau.
* Giảm chiều: Giảm số lượng tính năng của dữ liệu mà vẫn giữ nguyên thông tin quan trọng.
* Cộng đồng lớn: Scikit-learn sở hữu cộng đồng người dùng và nhà phát triển năng động, nhiệt tình, sẵn sàng hỗ trợ và giải đáp thắc mắc cho người mới bắt đầu.

Tóm lại, Scikit-learn là một thư viện học máy mạnh mẽ và linh hoạt, phù hợp cho nhiều đối tượng sử dụng. Với ưu điểm dễ sử dụng, hiệu quả và đa dạng, Scikit-learn là lựa chọn hàng đầu cho các ứng dụng học máy và thống kê trong Python. [8]

### Thư viện Seaborn

Seaborn là một trong những thư viện Python được đánh giá cao nhất thế giới được xây dựng nhằm mục đích tạo ra các hình ảnh trực quan đẹp mắt. Nó có thể được coi là một phần mở rộng của một thư viện khác có tên là Matplotlib vì nó được xây dựng trên đó.



Hình 6: Thư viện Seaborn

Trước khi chúng ta nói về các tính năng của Seaborn, điều quan trọng cần biết là:

Vì Seaborn là một phần mở rộng của Matplotlib, nó có nghĩa là để bổ sung cho phần sau chứ không phải thay thế hoàn toàn.

Với điều này đã nói, đây là các tính năng hàng đầu của Seaborn:

* + Rất nhiều chủ đề hoạt động với đồ họa kiểu Matplotlib
  + Khả năng trực quan hóa cả dữ liệu đơn biến và đa biến
  + Hỗ trợ trực quan hóa các loại dữ liệu mô hình hồi quy
  + Dễ dàng vẽ biểu đồ dữ liệu thống kê cho phân tích chuỗi thời gian
  + Hiệu suất liền mạch với Pandas, NumPy và các thư viện Python khác [9]

## Môi trường lập trình Jupyter Notebook



Hình 7: Môi trường Jupyter Notebook

Jupyter là một nền tảng tính toán khoa học mã nguồn mở, với khả năng nổi bật cho phép tương tác trực tiếp với từng dòng code (interactive), hỗ trợ hơn 40 ngôn ngữ lập trình, trong đó tập trung vào 3 ngôn ngữ là Julia, Python và R, cái tên Jupyter bắt nguồn từ cách chơi chữ I Python, You R, We Julia, hay một công thức ngắn gọn Jupyter = Julia + Python + R. Bên cạnh đó, Jupyter cũng là một công cụ hoàn toàn miễn phí, được tạo ra với mục đích nhắm đến khoa học dữ liệu và giáo dục, giúp mọi người cùng học lập trình dễ dàng hơn (cụ thể ở đây là Python). Jupyter có tính tương tác nên có thể sử dụng làm môi trường chạy thử và giảng dạy. Trước đây nó có tên là IPython Notebook, đến năm 2014 lại đổi tên thành Jupyter Notebook. Jupyter hỗ trợ rất nhiều các kernel cho các ngôn ngữ khác nhau, khoảng trên 40 ngôn ngữ trong đó có Python. Việc đổi tên từ IPython sang Jupyter cũng là vì mục đích hỗ trợ đa ngôn ngữ. Phần cơ bản của nó là một ứng dụng chạy trên nền web cho phép chạy Interactive Python (hay IPython), chúng ta có thể đưa cả code Python và các thành phần văn bản phức tạp như hình ảnh, công thức, video, biểu thức... vào trong cùng một file giúp cho việc trình bày trở lên dễ hiểu, giống như một file trình chiếu nhưng lại có thể thực hiện chạy code tương tác trên đó, cốt lõi của việc này chính là Markdown. Các file “notebook” này có thể được chia sẻ với mọi người và có thể thực hiện lại các công đoạn một cách nhanh chóng và chính xác như những gì chúng ta đã làm trong quá trình tạo ra file.

Jupyter Notebook được viết bằng các ngôn ngữ như Python, R và Julia, nền tảng này hiện đang được sử dụng rộng rãi. Bên cạnh đó, Jupyter còn tạo ra tài liệu, trực quan hóa dữ liệu và lưu trữ chúng một cách dễ dàng hơn rất nhiều.

Dưới đây là một số lợi ích mà Jupyter Notebook mang lại:

* Phân tích khám phá dữ liệu (Exploratory Data Analysis)

Jupyter cho phép người dùng xem kết quả của code in-line (mã inline) mà không cần phụ thuộc vào các phần khác của code.

Trong Notebook mọi ô của code có thể được kiểm tra bất cứ lúc nào, điều này đã giúp Jupyter trở nên khác biệt so với các ID như Pycharm, VSCode. Việc Jupyter có thể xuất code in-line đã giúp ích rất nhiều trong quá trình phân tích khám phá dữ liệu (EDA).

* Bộ đệm dễ dàng trong ô tích hợp

Từng ô tự duy trì trạng thái hoạt động sẽ hơi khó, nhưng với Jupyter, công việc này sẽ được thực hiện tự động. Vì Jupyter lưu trữ kết quả hoạt động của mọi ô đang chạy, cho dù là code đang đào tạo mô hình machine learning hay code đang tải xuống gigabyte dữ liệu từ một máy chủ từ xa.

* Độc lập ngôn ngữ

Jupyter Notebook ở định dạng JSON, vì thế nó được biết đến là một nền tảng độc lập cũng như độc lập về ngôn ngữ.

* Trực quan hóa dữ liệu (Data Visualisation)

Jupyter Notebook hỗ trợ trực quan hóa dữ liệu và hiển thị thêm một số đồ họa và biểu đồ. Những điều này được tạo ra từ code với sự trợ giúp của các mô -đun như Matplotlib, Plotly hoặc Bokeh. Ngoài ra, Jupyter còn cho phép người dùng cùng chia sẻ code và bộ dữ liệu hoặc thay đổi tương tác với nhau.

* Tương tác trực tiếp với code

Jupyter Notebook sử dụng “ipywidgets” packages, cung cấp cho người dùng giao diện chuẩn nhằm khám phá sự tương tác trực tiếp với code và với dữ liệu. Người dùng có thể chỉnh sửa và chạy code, làm cho code của Jupyter non-static.

Ngoài ra, nó còn cho phép người dùng kiểm soát nguồn đầu vào của code và phản hồi lại trực tiếp trên trình duyệt.

* Các mẫu code tài liệu

Jupyter giúp người dùng dễ dàng giải thích từng dòng code của họ với các phản hồi được đính kèm. Dù trong code đã có đầy đủ các chức năng nhưng người dùng vẫn có thể tăng thêm sự tương tác bằng các lời giải thích.

**Ưu điểm**

* Code ngay trong trình duyệt, hỗ trợ syntax highlight, thụt lề,... như một code editor.
* Chạy được code ngay trong trình duyệt.
* Hiển thị kết quả tính toán bằng cách sử dụng biểu diễn đa phương tiện, như HTML, LaTeX, PNG, SVG,....

Hỗ trợ markdown, khiến cho việc comment cho code trở nên trực quan hơn nhiều so với việc dùng plain text**.**

Jupyter là một dự án với mục tiêu phát triển phần mềm nguồn mở, các tiêu chuẩn mở và các dịch vụ cho máy tính tương tác trên nhiều ngôn ngữ lập trình. Nó đã được tách ra khỏi IPython vào năm 2014 bởi Fernando Pérez và Brian Granger. Tên của Project Jupyter liên quan đến ba ngôn ngữ lập trình cốt lõi được hỗ trợ bởi Jupyter, đó là Julia, Python và R. Tên và biểu tượng của nó là sự tôn kính đối với việc Galileo phát hiện ra các mặt trăng của Sao Mộc, như được ghi lại trong sổ ghi chép của Galileo. Dự án Jupyter đã phát triển và hỗ trợ các sản phẩm máy tính tương tác Jupyter Notebook, JupyterHub và JupyterLab. Jupyter được tài trợ tài chính bởi NumFOCUS.

**Lịch sử:**

Phiên bản đầu tiên của Notebooks for IPython được phát hành vào năm 2011 bởi một nhóm bao gồm Fernando Pérez, Brian Granger và Min Ragan-Kelley. Vào năm 2014, Pérez đã công bố một dự án phụ từ IPython có tên là Project Jupyter. IPython tiếp tục tồn tại dưới dạng trình bao Python và hạt nhân cho Jupyter, trong khi sổ ghi chép và các phần không thể sử dụng ngôn ngữ khác của IPython được chuyển sang tên Jupyter. Jupyter hỗ trợ các môi trường thực thi (được gọi là “hạt nhân”) bằng hàng chục ngôn ngữ, bao gồm Julia, R, Haskell, Ruby và Python (thông qua nhân IPython).

Vào năm 2015, khoảng 200.000 máy tính xách tay Jupyter đã có mặt trên GitHub. Đến năm 2018, khoảng 2,5 triệu đã có sẵn. Vào tháng 1 năm 2021, gần 10 triệu đã có sẵn, bao gồm sổ ghi chép về quan sát đầu tiên của sóng hấp dẫn và về phát hiện năm 2019 về một lỗ đen siêu lớn.

Các nhà cung cấp điện toán đám mây lớn đã sử dụng Jupyter Notebook hoặc các công cụ phái sinh làm giao diện frontend cho người dùng đám mây. Ví dụ bao gồm Amazon SageMaker Notebooks, Colaboratory của Google, và Azure Notebook của Microsoft.

Visual Studio Code hỗ trợ phát triển cục bộ sổ ghi chép Jupyter. Tính đến tháng 7 năm 2022, tiện ích mở rộng Jupyter cho VS Code đã được tải xuống hơn 40 triệu lần, khiến nó trở thành tiện ích mở rộng phổ biến thứ hai trên Thị trường mã VS.

Tờ Atlantic đã xuất bản một bài báo với tựa đề “The Scientific Paper Is Obsolete” vào năm 2018, thảo luận về vai trò của Jupyter Notebook và sổ Mathematica trong tương lai của xuất bản khoa học. Để đáp lại, nhà kinh tế học Paul Romer đã xuất bản một bài đăng trên blog, trong đó ông phản ánh kinh nghiệm của mình khi sử dụng Mathematica và Jupyter để nghiên cứu, một phần kết luận rằng Jupyter “làm tốt hơn việc truyền đạt những gì Theodore Grey đã nghĩ khi ông thiết kế sổ ghi chép Mathematica.”

Năm 2021, Nature đã đặt tên Jupyter là một trong mười dự án máy tính làm thay đổi khoa học. [10]

## Một số thuật toán dùng để xây dụng mô hình học máy

### Thuật toán K láng giềng gần nhất (K-Nearest Neighbor)

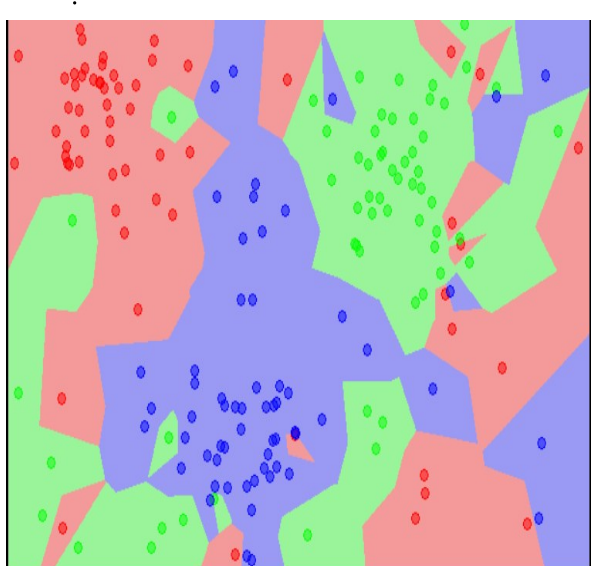
K-nearest neighbor (KNN) là một trong những thuật toán supervised learning đơn giản. Khi huấn luyện, thuật toán này không học một điều gì từ dữ liệu huấn luyện mà nhớ lại một cách máy móc toàn bộ dữ liệu đó. Đây cũng là lý do thuật toán này được xếp vào loại lazy learning, mọi tính toán được thực hiện khi nó cần dự đoán đầu ra của dữ liệu mới. KNN có thể áp dụng được vào cả classification và regression. KNN còn được gọi là một thuật toán instance-based hay memory-based learning.

Với KNN, trong bài toán classification, nhãn của một điểm dữ liệu mới được suy ra trực tiếp từ K điểm dữ liệu gần nhất trong tập huấn luyện. Nhãn đó có thể được quyết định bằng bầu chọn theo đa số (major voting) trong số K điểm gần nhất, hoặc nó có thể được suy ra bằng cách đánh trọng số khác nhau cho mỗi trong các điểm gần nhất đó rồi suy ra kết quả.

Chi tiết sẽ được nêu trong phần tiếp theo. Trong bài toán regresssion, đầu ra của một điểm dữ liệu sẽ bằng chính đầu ra của điểm dữ liệu đã biết gần nhất (trong trường hợp K = 1), hoặc là trung bình có trọng số của đầu ra của những điểm gần nhất, hoặc bằng một mối quan hệ dựa trên các điểm gần nhất đó và khoảng cách tới chúng.

Một cách ngắn gọn, KNN là thuật toán đi tìm đầu ra của một điểm dữ liệu mới bằng cách chỉ dựa trên thông tin của K điểm dữ liệu gần nhất trong tập huấn luyện.

Hình 8 mô tả một bài toán phân lớp với ba class: đỏ, lam, lục. Các hình tròn nhỏ với màu khác nhau thể hiện dữ liệu huấn luyện của các class khác nhau. Các vùng màu nền khác nhau thể hiện lãnh thổ của mỗi class. Tại một điểm bất kỳ, class của nó được xác định dựa trên class của điểm gần nó nhất trong trong tập huấn luyện. Trong hình này, có một vài vùng nhỏ xem lẫn vào các vùng lớn hơn khác màu. Ví dụ có một điểm màu lục ở gần góc 11 giờ nằm giữa hai vùng lớn với nhiều dữ liệu màu đỏ và lam. Điểm này rất có thể là nhiễu.



Hình 8: Mô tả bài toán phân lớp

Việc này nhiều khả năng sẽ dẫn đến việc phân lớp sai cho một điểm dữ liệu kiểm thử rơi vào khu vực này.

KNN là một ví dụ rõ nhất của overfitting. Với mô hình này, mọi điểm trong tập huấn luyện đều được mô hình mô tả một cách chính xác, vì vậy, nó rất nhạy cảm với nhiễu.

Mặc dù có nhiều hạn chế, KNN vẫn là một giải pháp đầu tiên nên nghĩ tới khi giải quyết một bài toán machine learning. Khi làm các bài toán machine learning nói chung, không có mô hình đúng hay sai, chỉ có mô hình cho kết quả tốt hơn. Chúng ta luôn cần một mô hình đơn giản để giải quyết bài toán, sau đó dần dần tìm cách tăng chất lượng của mô hình. [11]

### Thuật toán Cây quyết định (Decision Tree)

Cây quyết định (Decision Tree) là một cây phân cấp có cấu trúc được dùng để phân lớp các đối tượng dựa vào dãy các luật. Các thuộc tính của đối tượngncó thể thuộc các kiểu dữ liệu khác nhau như Nhị phân (Binary), Định danh (Nominal), Thứ tự (Ordinal), Số lượng (Quantitative) trong khi đó thuộc tính phân lớp phải có kiểu dữ liệu là Binary hoặc Ordinal.

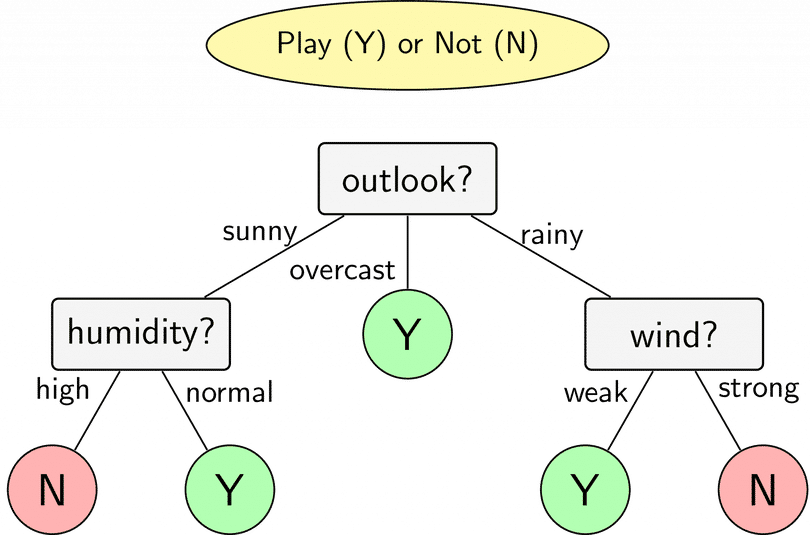
Tóm lại, cho dữ liệu về các đối tượng gồm các thuộc tính cùng với lớp (classes) của nó, cây quyết định sẽ sinh ra các luật để dự đoán lớp của các dữ liệu chưa biết.

Ta hãy xét một ví dụ 1 kinh điển khác về cây quyết định. Giả sử dựa theo thời tiết mà các bạn nam sẽ quyết định đi đá bóng hay không?

Những đặc điểm ban đầu là:

* Thời tiết
* Độ ẩm
* Gió

Dựa vào những thông tin trên, bạn có thể xây dựng được mô hình như sau:



Hình 9: Mô hình cây quyết định

Dựa theo mô hình trên, ta thấy: Nếu trời nắng, độ ẩm bình thường thì khả năng các bạn nam đi chơi bóng sẽ cao. Còn nếu trời nắng, độ ẩm cao thì khả năng các bạn nam sẽ không đi chơi bóng. [12]

### Thuật toán Rừng ngẫu nhiên (Random Forest)

Random là ngẫu nhiên, Forest là rừng, nên ở thuật toán Random Forest mình sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random). Sau đó kết quả dự đoán được tổng hợp từ các cây quyết định.

Ở bước huấn luyện thì mình sẽ xây dựng nhiều cây quyết định, các cây quyết định có thể khác nhau (phần sau mình sẽ nói mỗi cây được xây dựng như thế nào).

Sau đó ở bước dự đoán, với một dữ liệu mới, thì ở mỗi cây quyết định mình sẽ đi từ trên xuống theo các node điều kiện để được các dự đoán, sau đó kết quả cuối cùng được tổng hợp từ kết quả của các cây quyết định.

Giả sử bộ dữ liệu của mình có n dữ liệu (sample) và mỗi dữ liệu có d thuộc tính (feature).

Để xây dựng mỗi cây quyết định mình sẽ làm như sau:

1. Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping, hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.
2. Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính (k < n). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.
3. Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước

Do quá trính xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Khi dùng thuật toán Random Forest, mình hay để ý các thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách. [11]

## Môi trường và ngôn ngữ dùng để lập trình web

### Thư viện Flask

Flask là một khung web microframework mã nguồn mở được viết bằng ngôn ngữ Python. Nó được biết đến với sự đơn giản, linh hoạt và khả năng mở rộng, giúp cho việc phát triển các ứng dụng web hiện đại trở nên dễ dàng và hiệu quả.

A logo for a company

Description automatically generated

Hình 10: Thư viện Flask

Điểm nổi bật của Flask:

* Dễ sử dụng: Flask cung cấp API đơn giản, dễ hiểu, giúp người mới bắt đầu có thể nhanh chóng tạo dựng và triển khai các ứng dụng web.
* Linh hoạt: Flask không áp đặt cấu trúc hay quy ước cụ thể nào, cho phép người dùng tự do thiết kế và xây dựng ứng dụng theo ý muốn.
* Mở rộng: Flask có thể dễ dàng mở rộng bằng cách sử dụng các thư viện bên thứ ba, giúp đáp ứng nhu cầu phát triển các ứng dụng phức tạp.
* Hiệu suất cao: Flask được tối ưu hóa cho hiệu suất cao, giúp đảm bảo ứng dụng web hoạt động mượt mà và ổn định.
* Cộng đồng lớn: Flask sở hữu cộng đồng người dùng và nhà phát triển năng động, nhiệt tình, sẵn sàng hỗ trợ và giải đáp thắc mắc cho người mới bắt đầu. [13]

### HTML



Hình 11: HTML

HTML là viết tắt của cụm từ Hypertext Markup Language (Ngôn ngữ đánh dấu siêu văn bản). Đây là một ngôn ngữ đánh dấu được thiết kế để phân chia nội dung, tạo những trang web trên mạng lưới Internet: “www” (World Wide Web). Đây cũng là một ngôn ngữ được đánh giá là dễ học, dễ hiểu, thông thường là bước đệm đầu tiên dành cho người mới tìm hiểu, học lập trình website.

Trong lập trình website, HTML đóng vai trò như khung xương của một trang web, tạo nên cấu trúc cơ bản nhất của một website để các thành phần khác kết nối, lắp ráp để tạo nên một website hiệu quả và hoạt động trơn tru nhất.

Bất kể mục đích sử dụng website của bạn là gì, xây dựng trên nền tảng nào hoặc sử dụng ngôn ngữ lập trình nào thì tất cả đều cần phải có sự hỗ trợ của HTML để dữ liệu có thể được xử lý và hiển thị trên các trình duyệt web (browser). Do đó, HTML là một thành phần không thể thiếu nếu muốn xây dựng một website có cấu trúc tốt và hoạt động có hệ thống. HTML tuy không được xem là một ngôn ngữ lập trình chính thống, nhưng cũng trở thành loại ngôn ngữ cơ bản nhất mà các lập trình viên hay nhà phát triển website cần nắm rõ. [14]

### CSS



Hình 12: CSS

[CSS](https://vi.wikipedia.org/wiki/CSS) là viết tắt của cụm từ Cascading Style Sheet. Đây là một [ngôn ngữ lập trình](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh" \t "_blank) được thiết kế rất đơn giản và dễ sử dụng. Mục tiêu của CSS là giúp đơn giản hóa quá trình tạo ra các website.

CSS có nhiệm vụ thực hiện việc xử lý giao diện của một trang web. Có thể kể đến các yếu tố như màu sắc văn bản, hay khoảng cách giữa các đoạn, font chữ, hình ảnh, bố cục, màu nền,… Với sự hỗ trợ của CSS thì bạn có thể thay đổi, chỉnh sửa những yếu tố trên theo ý muốn.

Trong quá trình phát triển và thiết kế website thì CSS chính là một công cụ vô cùng cần thiết. Nó có thể giải quyết tốt những công việc mà [HTML](https://vi.wikipedia.org/wiki/HTML" \t "_blank) không xử lý được. Qua đó mang lại cho bạn khả năng kiểm soát mạnh mẽ với website của mình. [15]

# XÂY DỰNG MÔ HÌNH DỰ ĐOÁN BỆNH TIM

## Giới thiệu bài toán và mô tả tập dữ liệu

### Giới thiệu bài toán

Theo Trung tâm kiểm soát và phòng ngừa dịch bệnh CDC, bệnh tim là một trong những nguyên nhân hàng đầu gây tử vong cho người thuộc hầu hết các chủng tộc ở Mỹ (người Mỹ gốc Phi, người Mỹ da đỏ và người bản địa Alaska, và người da trắng). Khoảng một nửa số người Mỹ (47%) có ít nhất 1 trong 3 yếu tố nguy cơ chính của bệnh tim: huyết áp cao, cholesterol cao và hút thuốc. Các chỉ số quan trọng khác bao gồm tình trạng tiểu đường, béo phì (chỉ số BMI cao), không hoạt động thể chất đầy đủ hoặc uống quá nhiều rượu. Phát hiện và ngăn ngừa các yếu tố có tác động lớn nhất đến bệnh tim là rất quan trọng trong chăm sóc sức khỏe. Đổi lại, sự phát triển tính toán cho phép áp dụng các phương pháp học máy để phát hiện “mẫu” từ dữ liệu có thể dự đoán tình trạng của bệnh nhân.

Để nghiên cứu, đánh giá các yếu tố chính, quan trọng ảnh hưởng tới bệnh tim CDC tiến hành lấy khảo sát của người dân với nhiều độ tuổi, giới tính, chủng tộc… khác nhau và lưu trữ trong file: HeartDisease.csv

Tập dữ liệu này gồm 445132 mẫu bệnh nhân với 40 thuộc tính liên quan đến sức khỏe tim mạch và các bệnh lí khác, trong đó thuộc tính đầu tiên là HeartDisease cho biết bệnh nhân có mắc bệnh tim hay không (no: không, yes: có). Các thuộc tính này bao gồm giới tính, chỉ số BMI, giờ ngủ, nhóm tuổi, hút thuốc, uống rượu, đột quỵ, sức khỏe thể chất, sức khỏe tinh thần, vấn đề về đi lại, chủng tộc, bệnh tiểu đường, hoạt động thể chất, tình trạng sức khỏe, bệnh hen xuyễn, bệnh thận, ung thư da,...

Khi đã có dữ liệu khảo sát, chúng ta bắt đầu phân tích, chuẩn hóa tập dữ liệu để tìm ra các nguyên nhân dẫn đến mắc bệnh tim.

### Mô tả tập dữ liệu

A screen shot of a computer

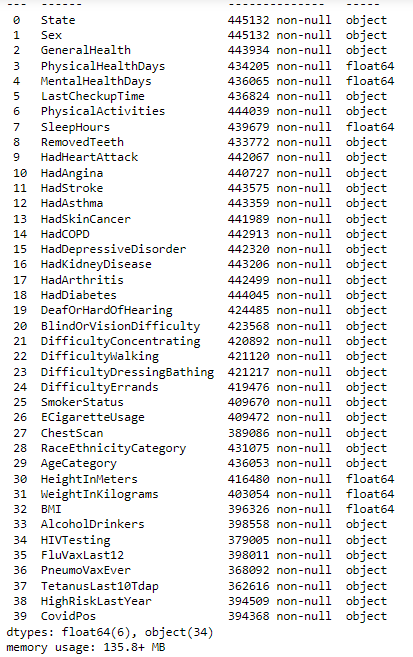
Description automatically generated

Hình 13: Dữ liệu file heartdisease.csv

Bộ dữ liệu ban đầu đến từ CDC và là một phần chính của Hệ thống giám sát yếu tố rủi ro hành vi (BRFSS), thực hiện khảo sát qua điện thoại hàng năm để thu thập dữ liệu về tình trạng sức khỏe của cư dân Hoa Kỳ. Theo mô tả của CDC: Được thành lập vào năm 1984 với 15 tiểu bang, BRFSS hiện thu thập dữ liệu ở tất cả 50 tiểu bang, Quận Columbia và ba vùng lãnh thổ của Hoa Kỳ. BRFSS hoàn thành hơn 400.000 cuộc phỏng vấn người lớn mỗi năm, trở thành cuộc phỏng vấn sức khỏe được tiến hành liên tục lớn nhất hệ thống khảo sát trên thế giới tập dữ liệu gần đây nhất bao gồm dữ liệu từ năm 2022. Nó bao gồm 445.134 hàng và 41 cột. Phần lớn các cột là những câu hỏi được đặt ra cho người trả lời về tình trạng sức khỏe của họ, chẳng hạn như “Bạn có gặp khó khăn nghiêm trọng khi đi bộ hoặc leo cầu thang không?” hoặc “Bạn có hút thuốc lá không (nếu có mức độ như nào)?”.

Trong tập dữ liệu này, chúng em nhận thấy nhiều yếu tố (câu hỏi) khác nhau ảnh hưởng trực tiếp hoặc gián tiếp đến bệnh tim, vì vậy chúng em quyết định chọn các biến có liên quan nhất từ nó và thực hiện một số làm sạch để nó có thể sử dụng được cho các dự án.

Tập dữ liệu HeartDisease bao gồm 445.133 bản ghi và 40 thuộc tính.



Hình 14: Dữ liệu đọc từ file.csv

## Chuẩn bị dữ liệu

### Lọc lấy dữ liệu những thuộc tính cần dùng cho bài toán

A screenshot of a computer

Description automatically generated

Hình 15: Lọc lấy thuộc tính phục vụ bài toán

Để phục vụ cho xây dựng mô hình học máy, chúng em đã lọc ra 21 thuộc tính phù hợp với yều cầu bài toán xây dựng mô hình dự đoán bệnh trong tim. Các thuộc tính bao gồm: 1 thuộc tính độc lập là HadHearthAttack – có 2 giá trị là Yes, No nhằm xác định 1 người có bệnh tim hay không và 20 thuộc tính độc lập, bao gồm:

* Sex: Giới tính của cá nhân (Female, Male)
* GeneralHealth: Tình trạng sức khỏe tổng thể của cá nhân (Very good, Good, Excellent, Fair, Poor)
* PhysicalHealthDays: Số ngày trong năm mà cá nhân cảm thấy sức khỏe thể chất của họ kém
* MentalHealthDays: Số ngày trong năm mà cá nhân cảm thấy sức khỏe tinh thần của họ kém
* PhysicalActivities: Mức độ hoạt động thể chất của cá nhân (Yes/No)
* SleepHours: Số giờ ngủ trung bình mỗi đêm của cá nhân (Yes/No)
* HadStroke: Cá nhân đã từng bị đột quỵ hay chưa (Yes/No)
* HadAsthma: Cá nhân đã từng bị hen suyễn hay chưa (Yes/No)
* HadSkinCancer: Cá nhân đã từng bị ung thư da hay chưa (Yes/No)
* HadDepressiveDisorder: Cá nhân đã từng bị rối loạn trầm cảm hay chưa (Yes/No)
* HadKidneyDisease: Cá nhân đã từng bị bệnh thận hay chưa (Yes/No)
* HadArthritis: Cá nhân đã từng bị viêm khớp hay chưa (Yes/No)
* HadDiabetes: Cá nhân đã từng bị tiểu đường hay chưa (Yes/No/No, pre-diabetes or borderline diabetes/Yes, but only during pregnancy (female))
* DifficultyWalking: Cá nhân có khó khăn khi đi bộ hay không (Yes/No)
* SmokerStatus: Tình trạng hút thuốc của cá nhân (Never smoked/Former smoker/Current smoker - now smokes every day/Current smoker - now smokes some days)
* RaceEthnicityCategory: Thuộc chủng tộc/dân tộc (White only, Non-Hispanic/Hispanic/Black only, Non-Hispanic/Other race only, Non-Hispanic/Multiracial, Non-Hispanic)
* AgeCategory: Nhóm tuổi của cá nhân (Age 18 to 24/Age 25 to 29/Age 30 to 34/Age 35 to 39/Age 40 to 44/Age 45 to 49/Age 50 to 54/Age 55 to 59/Age 60 to 64/Age 65 to 69/Age 70 to 74/Age 75 to 79/Age 80 or older)
* BMI: Chỉ số cơ thể của cá nhân
* AlcoholDrinkers: Cá nhân có uống rượu bia hay không (Yes/No)
* CovidPos: Cá nhân đã từng mắc COVID-19 hay chưa (Yes/No/Tested positive using home test without a health professional)

### Thông kê dữ liệu bị Missing

A screenshot of a computer

Description automatically generated

Hình 16: Thống kê dữ liệu bị missing

Qua tổng hợp dữ liệu missing, có thể thấy tập dữ liệu đang bị missing ở các thuộc tính. Để xử lý những dữ liệu missing này chúng em đã tiến hành loại bổ tất cả những dữ liệu có missing, lý do chúng em quyết định loại bỏ là vì nhận thấy tập dữ liệu có dữ liệu khá lớn hơn 400.000 dữ liệu và sau khi loại bỏ hết missing thì vẫn còn 325555 dữ liệu.

|  |  |
| --- | --- |
| Hình 17: Loại bỏ missing | Hình 18: Tập dữ liệu sau khi hết missing |

### Khám phá dữ liệu có ích từ tập dữ liệu

A screenshot of a computer

Description automatically generated

Hình 19: Khám phá dữ liệu số

Qua việc khám phá dữ liệu số, chúng em rút ra được những thông tin có ích của các thuộc tính từ giá trị tối thiểu, tối đa, giá trị trung bình của các thuộc tính. Như thuộc tính MentalHealthDays (Số ngày sức khỏe tinh thần không tốt): giá trị tối thiểu là 0 ngày và tối đa là 30 ngày, trung bình là 4 ngày.

Từ đây, chúng em còn tiến hành biến đổi các giá trị thuộc tính BMI là nhóm các khoảng giá trị với nhau chỉ còn 4 giá trị là Underweigh, Normal weigh, Overweight và Obese, theo tiêu chí như hình 20:

A screenshot of a computer code

Description automatically generated

Hình 20: Nhóm các giá trị BMI

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

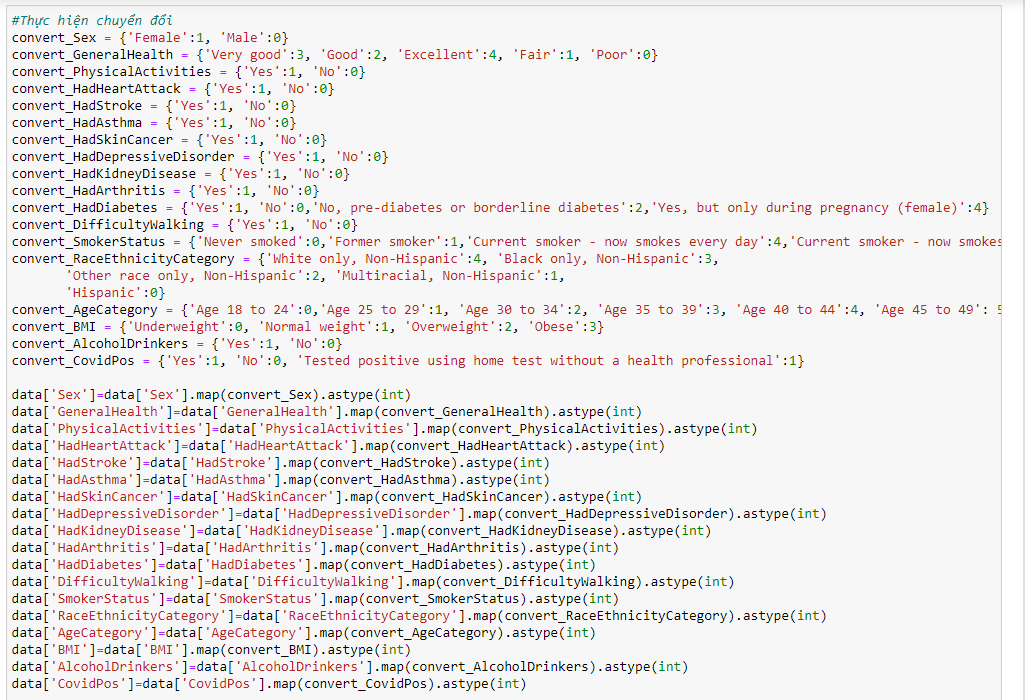
Hình 21: Khám phá dữ liệu chuỗi

Khám phá dữ liệu chuỗi, chúng em rút được số loại giá trị có trong mỗi thuộc, giá trị nào có nhiều giá trị nhất và số lượng là bao nhiêu. Ví dụ, thuộc tính Sex (giới tính) có tổng 325.555 người, có 2 giá trị, giá trị xuất hiện nhiều nhất là Female (nữ) với số lượng là 167.790 người.

### Chuyển đổi dữ liệu chuỗi về dữ liệu số

Chuyển đổi bẳng cách bằng cách gán một số nguyên duy nhất cho mỗi giá trị như sau:

* Sex: Giới tính của cá nhân (“Female”:1, “Male”:0)
* GeneralHealth: Tình trạng sức khỏe tổng thể của cá nhân (“Very good”:3, “Good”:2, “Excellent”:4, “Fair”:1, “Poor”:0)
* PhysicalHealthDays: Số ngày trong năm mà cá nhân cảm thấy sức khỏe thể chất của họ kém
* MentalHealthDays: Số ngày trong năm mà cá nhân cảm thấy sức khỏe tinh thần của họ kém
* PhysicalActivities: Mức độ hoạt động thể chất của cá nhân (“Yes”:1, “No”:0)
* SleepHours: Số giờ ngủ trung bình mỗi đêm của cá nhân
* HadHeartAttack: Cá nhân có bị đau tim hay không (“Yes”:1, “No”:0)
* HadStroke: Cá nhân đã từng bị đột quỵ hay chưa (“Yes”:1, “No”:0)
* HadAsthma: Cá nhân đã từng bị hen suyễn hay chưa (“Yes”:1, “No”:0)
* HadSkinCancer: Cá nhân đã từng bị ung thư da hay chưa (“Yes”:1, “No”:0)
* HadDepressiveDisorder: Cá nhân đã từng bị rối loạn trầm cảm hay chưa (“Yes”:1, “No”:0)
* HadKidneyDisease: Cá nhân đã từng bị bệnh thận hay chưa (“Yes”:1, “No”:0)
* HadArthritis: Cá nhân đã từng bị viêm khớp hay chưa (“Yes”:1, “No”:0)
* HadDiabetes: Cá nhân đã từng bị tiểu đường hay chưa (“Yes”:1, “No”:0, “No, pre-diabetes or borderline diabetes”:2, “Yes, but only during pregnancy (female)”:4)
* DifficultyWalking: Cá nhân có khó khăn khi đi bộ hay không (“Yes”:1, “No”:0)
* SmokerStatus: Tình trạng hút thuốc của cá nhân (“Never smoked”:0, “Former smoker”:1, “Current smoker - now smokes every day”:4, “Current smoker - now smokes some days”:3)
* RaceEthnicityCategory: Thuộc chủng tộc/dân tộc (“White only, Non-Hispanic”:4, “Black only, Non-Hispanic”:3, “Other race only, Non-Hispanic”:2, “Multiracial, Non-Hispanic”:1, “Hispanic”:0)
* AgeCategory: Nhóm tuổi của cá nhân (“Age 18 to 24”:0, “Age 25 to 29':1, “Age 30 to 34”:2, “Age 35 to 39”:3, “Age 40 to 44”:4, “Age 45 to 49”: 5, “Age 50 to 54”: 6, “Age 55 to 59”: 7, “Age 60 to 64”: 8, “Age 65 to 69”: 9, “Age 70 to 74”: 10, “Age 75 to 79”: 11, “Age 80 or older”: 12)
* BMI: Chỉ số cơ thể của cá nhân (“Underweight”:0, “Normal weight”:1, “Overweight”:2, “Obese”:3)
* AlcoholDrinkers: Cá nhân có uống rượu bia hay không (“Yes”:1, “No”:0)
* CovidPos: Cá nhân đã từng mắc COVID-19 hay chưa (“Yes”:1, “No”:0)



Hình 22: Code chuyển đổi

### **Thống kê những người** mắc bệnh tim



Hình 23: Code vẽ biểu đồ thống kê người mắc bệnh tim

A blue bar graph with white text

Description automatically generated

Hình 24: Biểu đồ phân bố người mắc bệnh tim

Ta dễ thấy, tập dữ liệu bị mất cân bằng dữ liệu, số người bị bệnh tim chiếm chỉ 5,54% trên tổng số. Yêu cầu đặt ra là đi tiến hành xử lý mất cân bằng dữ liệu.

### Sự ảnh hưởng của các yếu tố tới việc mắc bệnh tim

A colorful circle with many different colors

Description automatically generated

Hình 25: Biểu đồ tròn phần trăm người bệnh tịm theo từng yếu tố ảnh hướng

A graph with blue bars

Description automatically generated

Hình 26: Biểu đồ cột phần trăm người bệnh tịm theo từng yếu tố ảnh hướng

### Xử lý dữ liệu mất cân bằng

Sử dụng phương pháp Under-sampling: Loại bỏ bớt số mẫu của lớp đa số để cân bằng với lớp thiểu số để xử lý mất cân bằng dữ liệu.

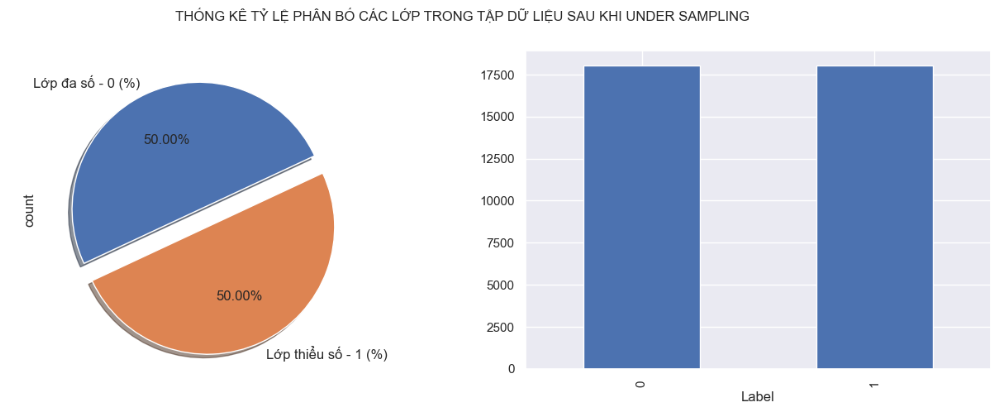
Phương pháp thực hiện sau:

* Xác định lớp thiểu số và lớp đa số: Đầu tiên, bạn cần xác định lớp thiểu số (lớp ít mẫu) và lớp đa số (lớp nhiều mẫu).
* Tính toán số lượng mẫu cần loại bỏ: Số lượng mẫu cần loại bỏ từ lớp đa số có thể được tính bằng sự chênh lệch giữa số lượng mẫu của lớp đa số và lớp thiểu số.
* Under-sampling: Loại bỏ ngẫu nhiên một số mẫu từ lớp đa số sao cho số lượng mẫu của lớp đa số giảm đi đến mức cân bằng với số lượng mẫu của lớp thiểu số.

A computer screen shot of a code

Description automatically generated

Hình 27: Xử lý dữ liệu mất cân bằng



Hình 28: Biểu đồ thống kê tỷ lệ phân bố các lớp sau khi xử lý dữ liệu mất cân bằng

## Xây dựng mô hình

### Thuật toán K láng giềng gần nhất (K-Nearest Neighbor)

A screenshot of a computer code

Description automatically generated

Hình 29: Huấn luyện mô hình bằng thuật toán KNN

Trước hết, chúng ta cần khai báo thư viện scikit-learn để sử dụng thuật toán KNN và hàm accuracy\_score để tính toán độ chính xác của mô hình.

* from sklearn.neighbors import KNeighborsClassifier: Khai báo lớp KNeighborsClassifier từ module neighbors của thư viện scikit-learn. Lớp này được sử dụng để tạo và huấn luyện mô hình KNN.
* from sklearn.metrics import accuracy\_score: Khai báo hàm accuracy\_score từ module metrics của thư viện scikit-learn. Hàm này được sử dụng để tính toán độ chính xác của mô hình trên một tập dữ liệu thử nghiệm.

Sau đó thì dùng KNN = KNeighborsClassifier() để tạo một mô hình KNN mới và lưu trữ nó trong biến KNN. Lệnh KNN.fit(X\_train,y\_train) để huấn luyện mô hình KNN trên tập dữ liệu X\_train và nhãn tương ứng y\_train.

A white rectangular box with black text

Description automatically generated

Hình 30: Độ chính xác trên tập huấn luyện (KNN)

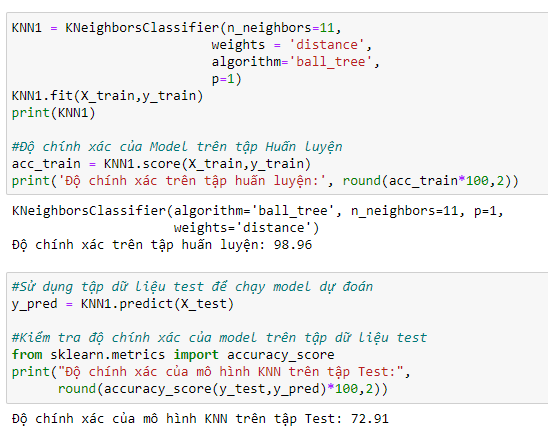
Tiếp đến, cần tính toán độ chính xác của mô hình trên tập huấn luyện X\_train và in kết quả ra màn hình.

A screenshot of a computer code

Description automatically generated

Hình 31: Độ chính xác trên tập test (KNN)

Nhận thấy độ chính xác trên tập test là 72,03% còn khá thấp so với trên tập huấn luyện là 79,02%, chúng em tiến hành điều chỉnh tham số để điều chỉnh độ chính.



Hình 32: Kết quả sau khi điều chỉnh tham số (KNN)

Sau khi điều chỉnh tham số nhận thấy mô hình bị overfitting, do độ chỉnh xác trên tập test quá thấp so với độ chính xác trên tập dữ liệu. Từ đây, chúng em quyết định sử dụng thuật toán khác để huấn luyện mô hình, đó là thuật toán cây quyết định.

### Thuật toán Cây quyết định (Decision Tree)

A screenshot of a computer program

Description automatically generated

Hình 33: Huấn luyện mô hình bằng thuật toán cây quyết định

Import thư viện DecisionTreeClassifier từ sklearn.tree:

* Dòng mã: from sklearn.tree import DecisionTreeClassifier
* Quá trình này nhúng lớp DecisionTreeClassifier từ thư viện sklearn.tree vào phạm vi làm việc hiện tại của chương trình. Lớp này là một phần của module sklearn.tree, cung cấp các công cụ cho việc xây dựng và huấn luyện các mô hình cây quyết định.

Khởi tạo một model Decision Tree gọi là Tree\_heart với các tham số mặc định:

* Dòng mã: Tree\_heart = DecisionTreeClassifier()
* Tạo ra một đối tượng của lớp DecisionTreeClassifier với các tham số mặc định. Điều này bao gồm các thiết lập chuẩn cho một mô hình cây quyết định.

In ra danh sách các tham số của model DecisionTree:

* Dòng mã: Tree\_heart.get\_params()
* Phương thức get\_params() trả về một từ điển của tất cả các tham số và giá trị của chúng cho mô hình. Thông qua in ra, ta có thể hiểu được cấu hình hiện tại của mô hình.

Huấn luyện mô hình Decision Tree trên tập dữ liệu huấn luyện (X\_train, y\_train):

* Dòng mã: Tree\_heart.fit(X\_train, y\_train)
* Sử dụng phương thức fit() của mô hình để huấn luyện nó trên dữ liệu huấn luyện X\_train và nhãn tương ứng y\_train. Điều này có nghĩa là mô hình sẽ học cách dự đoán nhãn từ các đặc trưng của dữ liệu huấn luyện.

A white box with black text

Description automatically generated

Hình 34: Độ chính xác trên tập huấn luyện (Cây quyết định)

Đánh giá độ chính xác của mô hình trên tập huấn luyện và in ra kết quả:

* Dòng mã:

acc\_train = Tree\_heart.score(X\_train, y\_train)

print("Độ chính xác của tập huấn luyện:",round(acc\_train\*100,2))

* Sử dụng phương thức score() để tính toán độ chính xác của mô hình trên dữ liệu huấn luyện X\_train và nhãn tương ứng y\_train. Kết quả được lưu vào biến acc\_train và được in ra sau đó.

A screenshot of a computer code

Description automatically generated

Hình 35: Độ chính xác trên tập test (Cây quyết định)

Đánh giá độ chính xác của mô hình trên tập dữ liệu kiểm tra (y\_test):

* Dòng mã:

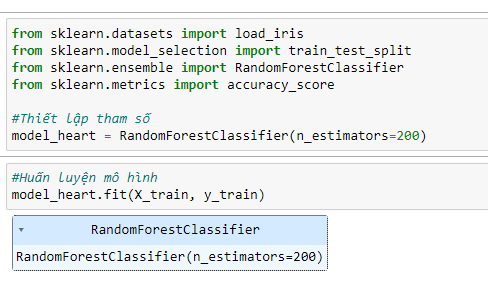
acc = accuracy\_score(y\_test, y\_pred, normalize=True)

acc\_num = accuracy\_score(y\_test, y\_pred, normalize=False)

* Sử dụng hàm accuracy\_score từ sklearn.metrics để tính toán độ chính xác của mô hình trên tập dữ liệu kiểm tra y\_test. Kết quả được lưu vào biến acc và acc\_num, trong đó acc là độ chính xác được chuẩn hóa và acc\_num là số lượng mẫu dự đoán đúng trên tổng số mẫu trong tập kiểm tra.
* Kết quả này được in ra để đánh giá hiệu suất của mô hình trên dữ liệu mới mà nó chưa từng thấy.

Sử dụng thuật toán cây quyết định, chúng em nhận thấy mô hình huấn luyện vẫn bị overfitting, nên chúng em tiến hành sử dụng thuật toán rừng ngẫu nhiên đây xây dựng mô hình.

### Thuật toán Rừng ngẫu nhiên (Random Forest)



Hình 36: Huấn luyện mô hình bằng thuật toán rừng ngẫu nhiên

Trước hết, chúng ta import RandomForestClassifier từ thư viện sklearn.ensemble và accuracy\_score từ thư viện sklearn.metrics. RandomForestClassifier là một mô hình phổ biến được sử dụng trong học máy để giải quyết bài toán phân loại, trong đó các cây quyết định được kết hợp thành một rừng cây để ra quyết định cuối cùng.

Sau đó, chúng ta thiết lập một mô hình RandomForestClassifier với tham số n\_estimators=200. Tham số này xác định số lượng cây trong rừng, và việc chọn giá trị phù hợp cho n\_estimators có thể ảnh hưởng đến hiệu suất của mô hình.

Tiếp theo, chúng ta huấn luyện mô hình bằng cách sử dụng dữ liệu huấn luyện (X\_train) và nhãn tương ứng (y\_train). Mô hình sẽ học từ dữ liệu này để có thể dự đoán nhãn cho dữ liệu mới.

A screenshot of a computer code

Description automatically generated

Hình 37: Độ chính xác (Rừng ngẫu nhiên)

Sau khi mô hình được huấn luyện, chúng ta sử dụng nó để dự đoán nhãn cho tập dữ liệu kiểm tra (X\_test).

Tiếp theo, chúng ta đánh giá hiệu suất của mô hình trên tập huấn luyện và tập kiểm tra. Điều này giúp chúng ta đánh giá xem mô hình có hiệu suất tốt và có dấu hiệu của overfitting không.

Mô hình bị overfitting, cần tiến hành điều chỉnh tham số:

A screenshot of a computer program

Description automatically generated

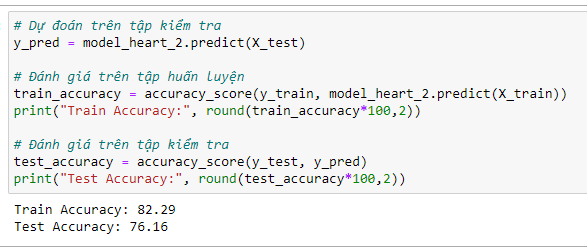
Hình 38: Điều chỉnh tham số thuật toán rừng ngẫu nhiên

Tiếp theo tùy chỉnh một số tham số của mô hình RandomForestClassifier để cải thiện hiệu suất và kiểm soát quá trình huấn luyện. Các tham số này bao gồm:

* n\_estimators: Số lượng cây trong rừng.
* max\_depth: Độ sâu tối đa của mỗi cây.
* min\_samples\_split: Số mẫu tối thiểu cần để chia một nút.
* min\_samples\_leaf: Số mẫu tối thiểu trong mỗi lá.
* random\_state: Giá trị seed cho tái lập.

Sau đó, mô hình được huấn luyện lại với các tham số mới.

Dự đoán được thực hiện trên tập kiểm tra và độ chính xác được đánh giá trên cả tập huấn luyện và tập kiểm tra.



Hình 39: Độ chính xác sau khi điều chỉnh tham số (Rừng ngẫu nhiên)

Nhận thấy, độ chính xác giữa tập huận luyện và tập test khá gần nhau và qua nhiều lần điều chỉnh tham số để có độ chính xác cao hơn chúng em thu được độ chính xác trên tập test là 76,16% là cao nhất so với các trường hợp còn lại. Qua đây, chúng em quyết định đưa mô hình vào dự đoán với mô hình được huấn luyện bằng thuật toán rừng ngẫu nhiên và có điều chỉnh tham số của thuật toán (như hình 38).

Sau cùng sử dụng thư viện pickle để lưu mô hình đã được huấn luyện vào một tệp. Điều này cho phép chúng ta lưu trữ mô hình để sau này có thể sử dụng để thực hiện dự đoán trên dữ liệu mới mà không cần phải huấn luyện lại từ đầu.

A white screen with black text

Description automatically generated with medium confidence

Hình 40: Lưu mô hình

## Đưa ra dự đoán

**A screenshot of a computer

Description automatically generated**

Hình 41: Giao diện trang web ứng dụng mô hình học máy đã xây dựng được

A screenshot of a computer

Description automatically generated

Hình 42: Kết quả AI dự đoán: Bạn không mắc bệnh tim!

A screenshot of a computer

Description automatically generated

Hình 43: Kết quả AI dự đoán: Bạn có khả năng mắc bệnh tim!

# ĐÁNH GIÁ VÀ KẾT LUẬN

**Đánh giá chung**

Nghiên cứu này tập trung vào trong lĩnh vực phòng chống và dự đoán bệnh tim dựa trên dữ liệu khảo sát từ Trung tâm Kiểm soát và Phòng ngừa Dịch bệnh CDC. Dữ liệu này bao gồm 445,132 mẫu bệnh nhân và 40 thuộc tính liên quan đến sức khỏe tim mạch và các bệnh lý khác, trong đó chúng em đã sử dụng 21 thuộc tính trên 40 thuộc tính phụ vụ cho việc xây dựng mô hình học máy. Mục tiêu của NCKH là tìm ra các nguyên nhân dẫn đến mắc bệnh tim và xây dựng một mô hình có khả năng dự đoán tình trạng bệnh tim của một bệnh nhân dựa trên các yếu tố khảo sát.

Trong quá trình thực hiện NCKH, đã có những bước cụ thể:

Chuẩn bị Dữ liệu: Dữ liệu từ CDC đã được thu thập và lưu trữ trong tập tin HeartDisease.csv. Các thuộc tính chính đã được xác định, bao gồm giới tính, chỉ số BMI, giờ ngủ, nhóm tuổi, hút thuốc, uống rượu, đột quỵ, sức khỏe thể chất, sức khỏe tinh thần, vấn đề về đi lại, chủng tộc, bệnh tiểu đường, hoạt động thể chất, tình trạng sức khỏe, bệnh hen xuyễn, bệnh thận, ung thư da, ...

Phân tích Dữ liệu: Dữ liệu đã được phân tích để hiểu rõ hơn về mối quan hệ giữa các yếu tố khảo sát và mắc bệnh tim. Các phương pháp thống kê và phân tích dữ liệu đã được áp dụng để phát hiện các mẫu và xu hướng quan trọng.

Chuẩn hóa Dữ liệu: Trước khi xây dựng mô hình, dữ liệu đã được chuẩn hóa để đảm bảo rằng các thuộc tính có phân phối đồng đều và không có giá trị ngoại lai ảnh hưởng đến quá trình huấn luyện.

Xây dựng Mô hình: Sử dụng các phương pháp học máy, một mô hình đã được xây dựng dựa trên dữ liệu huấn luyện. Mô hình này có khả năng dự đoán tình trạng bệnh tim của một bệnh nhân dựa trên các yếu tố khảo sát.

Đánh giá Mô hình: Mô hình được đánh giá để đảm bảo tính chính xác và hiệu quả. Các phương pháp đánh giá như cross-validation và confusion matrix đã được sử dụng để đo lường hiệu suất của mô hình.

Kết luận và Đề Xuất: Dựa trên kết quả của mô hình và đánh giá, kết luận được đưa ra về các yếu tố quan trọng ảnh hưởng đến bệnh tim và hiệu suất của mô hình. Đề xuất cũng có thể được đưa ra để cải thiện hiệu suất dự đoán và tối ưu hóa quá trình phòng ngừa và chăm sóc sức khỏe tim mạch.

Tóm lại, NCKH này đã mang lại cái nhìn tổng quan về mối quan hệ giữa các yếu tố khảo sát và bệnh tim, cũng như xây dựng một mô hình dự đoán hữu ích để hỗ trợ trong việc chẩn đoán và phòng ngừa bệnh tim trong cộng đồng.

**Kết luận**

Nghiên cứu này đã đạt được một số kết quả quan trọng trong việc hiểu và dự đoán bệnh tim dựa trên dữ liệu khảo sát từ CDC. Việc thực hiện các bước cụ thể như chuẩn bị dữ liệu, phân tích, chuẩn hóa, xây dựng mô hình và đánh giá đã giúp nghiên cứu này đạt được mục tiêu đề ra.

Chuẩn bị dữ liệu là bước quan trọng để đảm bảo tính chính xác của kết quả. Việc xác định các thuộc tính chính và thu thập dữ liệu lớn từ CDC đã cung cấp một cơ sở dữ liệu đáng tin cậy để phân tích. Phân tích dữ liệu chi tiết và sâu sắc đã giúp hiểu rõ hơn về mối quan hệ giữa các yếu tố khảo sát và bệnh tim.

Chuẩn hóa dữ liệu trước khi xây dựng mô hình là bước quan trọng để đảm bảo tính nhất quán và hiệu quả của mô hình. Quá trình này đã giúp loại bỏ các biến động không mong muốn và đảm bảo rằng mô hình được huấn luyện trên dữ liệu đồng nhất.

Việc xây dựng mô hình thông qua các phương pháp học máy đã tạo ra một công cụ mạnh mẽ để dự đoán tình trạng bệnh tim của một bệnh nhân dựa trên các yếu tố khảo sát. Mô hình đã được đánh giá một cách toàn diện và đạt được hiệu suất tốt, đảm bảo tính chính xác và hiệu quả trong ứng dụng thực tế.

Tuy nhiên, như mọi nghiên cứu, có những hạn chế cần được nhấn mạnh. Do thời gian và trình độ có hạn, nghiên cứu này có thể chưa thể bao phủ hết mọi khía cạnh của vấn đề. Đề xuất nên tiếp tục nghiên cứu để mở rộng phạm vi và cải thiện hiệu suất dự đoán. Đồng thời, việc kết hợp với các nguồn dữ liệu khác như môi trường sống và di truyền cũng có thể cung cấp cái nhìn toàn diện hơn về bệnh tim.

Tóm lại, nghiên cứu này đã làm rõ mối quan hệ giữa các yếu tố khảo sát và bệnh tim, đồng thời cung cấp một công cụ dự đoán hữu ích để hỗ trợ trong việc chẩn đoán và phòng ngừa bệnh tim trong cộng đồng. Đề xuất nên tiếp tục nghiên cứu để nâng cao hiệu suất và hiểu biết sâu hơn về vấn đề này.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | "Họcmáy,"[Online].Available: https://vi.wikipedia.org/wiki/H%E1%BB%8Dc\_m%C3%A1y. |
| [2] | "Tóm lược lịch sử phát triển của ngành Machine Learning," [Online]. Available: https://techmaster.vn/posts/33923/lich-su-phat-trien-machine-learning. |
| [3] | ThS. Đặng Văn Nam, Khoa Công nghê thông tin, Đại học Mỏ - Địa chất, *Bài giảng môn Kĩ nghệ tri thức và học máy.* |
| [4] | "Python là gì?," [Online]. Available: https://aws.amazon.com/vi/what-is/python/. |
| [5] | "pandas," [Online]. Available: https://pandas.pydata.org/. |
| [6] | "numpy," [Online]. Available: https://numpy.org/. |
| [7] | "matplotlib," [Online]. Available: https://matplotlib.org/. |
| [8] | "scikit-learn," [Online]. Available: https://scikit-learn.org/stable/. |
| [9] | "seaborn," [Online]. Available: https://seaborn.pydata.org/. |
| [10] | "jupyter notebook," [Online]. Available: https://jupyter.org/. |
| [11] | Vũ Hữu Tiệp, "Machine Learning cơ bản," [Online]. Available: https://machinelearningcoban.com/. |
| [12] | "Cây Quyết Định (Decision Tree)," [Online]. Available: https://trituenhantao.io/kien-thuc/decision-tree/. |
| [13] | "flask," [Online]. Available: https://flask.palletsprojects.com/en/3.0.x/. |
| [14] | "HTML LÀ GÌ? NỀN TẢNG CHO NGƯỜI MỚI BẮT ĐẦU HỌC WEB," [Online]. Available: https://www.greenacademy.edu.vn/kien-thuc-lap-trinh/html-la-gi-nen-tang-cho-nguoi-moi-bat-dau-hoc-web. |
| [15] | "CSS là gì? Kiến thức tổng quan về CSS," [Online]. Available: https://longvan.net/css-la-gi-kien-thuc-tong-quan-ve-css.html. |