**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC VÀ KĨ THUẬT THÔNG TIN**



**BÁO CÁO KHAI PHÁ DỮ LIỆU**

**ỨNG DỤNG KHAI PHÁ DỮ LIỆU**

**XÂY DỰNG HỆ HỖ TRỢ CHUẨN DOÁN BỆNH TIM MẠCH**

GVHD: Ths. Huỳnh Thiện Ý

Nhóm sinh viên thực hiện:

1. Ngô Minh Nhí MSSV: 16520889

2. Trần Văn Quang MSSV: 16521005

🙡🙢Tp. Hồ Chí Minh, 12/04/2019🙠🙣

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2019*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**LỜI MỞ ĐẦU**

Sự phát triển nhanh chóng của mạng Internet và Intranet đã sinh ra một khối lượng khổng lồ các dữ liệu dạng siêu văn bản (dữ liệu Web). Cùng với sự thay đổi và phát triển hàng ngày, hàng giờ về nội dung cũng như số lượng các trang Web trên Internet thì vấn đề tìm kiếm thông tin đối với người sử dụng lại càng khó khăn. Có thể nói nhu cầu tìm kiếm thông tin trên một CSDL phi cấu trúc đã được phát triển chủ yếu cùng với sự phát triển của Internet. Tuy nhiên cũng với sự đa dạng và số lượng lớn thông tin như vậy đã nảy sinh vấn đề quá tải thông tin. Người ta không thể tự tìm kiếm địa chỉ trang Web chứa thông tin mà mình cần, do vậy đòi hỏi phải có một trình tiện ích quản lý nội dung của các trang Web và cho phép tìm thấy các địa chỉ trang Web có nội dung giống với yêu cầu của người tìm kiếm. Các tiện ích này quản lý dữ liệu như các đối tượng phi cấu trúc. Hiện nay chúng ta đã làm quen với một số các tiện ích như vậy, đó là: yahoo, google, alvista…

Mặt khác, giả sử chúng ta có các trang Web về các vấn đề Tin học, Thể thao, Kinh tế - Xã hội và xây dựng… Căn cứ vào nội dung của các tài liệu mà khách hàng xem hoặc download về, sau khi phân lớp chúng ta sẽ biết khách hàng hay tập trung vào nội dung gì trên trang Web của chúng ta, từ đó chúng ta sẽ bổ sung thêm nhiều các tài liệu về các nội dung mà khách hàng quan tâm và ngược lại. Còn về phía khách hàng sau khi phân tích chúng ta cũng biết được khách hàng hay tập trung về vấn đề gì, từ đó có thể đưa ra những hỗ trợ thêm cho khách hàng đó. Từ những nhu cầu thực tế trên , phân lớp và tìm kiếm trang Web vẫn là bài toán hay và cần phát triển nghiên cứu hiện nay.

Sau đây, chúng tôi sẽ trình bày về một số thuật toán phổ biến trong khai phá dữ liệu bằng ngôn ngữ lập trình python qua các chương sau:

* Chương 1: Thuật toán Apriori
* Chương 2: Thuật toán Generalized Sequential Patterns (GSP)
* Chương 3: Thuật toán Decision Trees
* Chương 4: Thuật toán Naive Bayes
* Chương 5: Thuât toán tìm túi từ

# **THÔNG TIN NHÓM:**

Bảng 1: Thông tin nhóm

|  |  |  |
| --- | --- | --- |
| Họ và tên | MSSV | Email |
| Ngô Minh Nhí | 16520889 | 16520889[@gm.uit.edu.vn](mailto:15520667@gm.uit.edu.vn) |
| Trần Văn Quang | 16521005 | 16521005[@gm.uit.edu.vn](http://15520555@gm.uit.edu.vn) |

**PHÂN CÔNG, ĐÁNH GIÁ THÀNH VIÊN:**

Bảng 2: Đánh giá thành viên

|  |  |  |  |
| --- | --- | --- | --- |
| Họ và tên | MSSV | Phân công | Đánh giá |
| Ngô Minh Nhí | 16520889 | * Tìm hiểu về thuật toán Decision Tree và Naïve Bayes. * Tóm tắt tư liệu tìm kiếm. | * Tỉ lệ % hoàn thành công việc: 95%. * Nhận xét:   + Làm việc nhiệt tình, năng nổ.  + Hoàn thành tốt nhiệm vụ được giao. |
| Trần Văn Quang | 16521005 | * Tìm hiểu về thuật toán Decision Tree và Naïve Bayes. * Tổng hợp thông tin, tạo bìa và viết file báo cáo. * Chỉnh sửa thông tin cho các thành viên. | * Tỉ lệ % hoàn thành công việc: 95%. * Nhận xét:   + Làm việc nhiệt tình, năng nổ.  + Hoàn thành tốt nhiệm vụ được giao. |

# **LỜI MỞ ĐẦU**

# Sự cần thiết của đề tài

Bệnh tim mạch ngày nay vẫn đang là vấn đề thời sự vì sự gia tăng nhanh chóng trong cộng đồng. Ở những nghiên cứu trên những bệnh nhân cao tuổi thì tỷ lệ mắc bệnh tim mạch cao hơn rất nhiều so với những nhóm khác. Theo hống kê của Tổ chức Y tế Thế giới (WHO) cho thấy có đến 17,5 triệu người tử vong mỗi năm do các bệnh liên qua đến tim mạch và con số bệnh nhân tích lũy ngày một nhiều. Việt Nam chưa có thống kê cụ thể về con số, tuy nhiên thống kê nguyên nhân tử vong hàng đầu tại các bệnh viện là do bệnh lý tim mạch.

Công việc chẩn đoán bệnh tim tương đối dễ dàng dựa vào trị số đo được sau khi đo các chỉ số theo đúng quy trình. Ngưỡng chẩn đoán bệnh tim mạch thay đổi tùy theo từng thời điểm, để dự đoán một bệnh nhân mắc bệnh tim mạch thì còn nhiều khó khăn trong thực tế lâm sàng. Do vậy vẫn còn bệnh nhân tử vong cao, cũng như chi phí điều trị cao cho những bệnh nhân.

# Tính thực tiễn của đề tài

Y học là một lĩnh vực khoa học ứng dụng liên quan đến nghệ thuật chữa bệnh, bao gồm nhiều phương pháp chăm sóc sức khỏe nhằm duy trì, hồi phục cơ thể từ việc phòng ngừa và chữa bệnh. Y học hiện đại ứng dụng các kiến thức khoa học sức khỏe, nghiên cứu về y sinh học và công nghệ y học để chẩn đoán và chữa trị bệnh tật thông qua thuốc men, phẫu thuật hoặc bằng nhiều phương pháp trị liệu phong phú khác. Là môn khoa học không ngừng phát triển, nhằm tiếp cận và cập nhật hóa thông tin y học chứng cớ và y học thực chứng từ những cơ sở dữ liệu, để nâng cao chất lượng chăm sóc sức khỏe cho nhân dân là điều không thể là điều không thể thiếu trong thực hành lâm sàng.

Ngày nay, với sự phát triển vượt bậc của công nghệ thông tin, các ứng dụng công nghệ thông tin đã góp phần thay đổi cuộc sống của con người, hỗ trợ chúng ta trong hầu như tất cả các lĩnh vực: kinh tế, xã hội, giáo dục, y tế, khoa học, … nó đã trở thành một phần không thể thiếu được trong cuộc sống hàng ngày của con người. Tuy nhiên, song hành cùng sự phát triển này là sự gai tăng bùng nổ của dữ liệu. Khai phá dữ liệu là một trong những lĩnh vực đang phát triển nhanh chóng và đã được ứng dụng trong nhiều lĩnh vực khác nhau. Những lĩnh vực ứng dụng tiêu biểu của khai phá dữ liệu là trong kinh doanh thương mại, khoa học kĩ thuật. Trong những lĩnh vực ứng dụng đó, y học là một trong những lĩnh vực mà khai phá dữ liệu đã có những đóng góp đáng kể và trở thành một trong những giải pháp cho nhiều vấn đề y học. Nhiều hệ thống phát hiện tri thức y học đã được phát triển và thu được rất nhiều lợi ích. Khai phá dữ liệu giúp tìm ra manh mối liên hệ giữa các triệu chứng lâm sàng, chẩn đoán bệnh. Tuy nhiên khai phá dữ liệu trong lĩnh vực y khoa ở nước ta còn rất ít, gặp nhiều khó khăn, do hiện nay nhiều bệnh viện trên cả nước chưa có bệnh án điện tử. Việc khai phá trong lĩnh vực này thực sự mang lại nhiều ý nghĩa cho y học chứng cớ và y học thực chứng để hỗ trợ cho các bác sĩ, chẩn đoán bệnh sớm và điều trị bệnh có hiệu quả, giảm bớt tử vong cũng như chi phí điều trị, đây là một nhu cầu thiết thực trong các bệnh viện.

Từ những lý do trên và xu hướng tất yếu đồ án chọn đề tài “Khai phá dữ liệu tim mạch tại kaggle ” để nghiên cứu cho đồ án thực hành của nhóm.

# Mục đích nghiên cứu

* Tìm ra các quy luật từ dữ liệu thực tế nhằm chẩn đoán một số bệnh thường gặp (Bệnh tim mạch).
* Xây dựng các thuật toán hỗ trợ chẩn đoán bệnh tim mạch.
* Ứng dụng các thuật toán vào công việc chẩn đoán bệnh.

# Đối tượng và phạm vi nghiên cứu

Chúng tôi lấy tập dữ liệu cho các nhà nghiên cứu machine learning trên trang web kaggle. ( Với 303 dòng dữ liệu và 14 thuộc tính)

Link: <https://www.kaggle.com/ronitf/heart-disease-uci>

# Phương pháp nghiên cứu

* Tìm hiểu nghiệp vụ y tế liên quan đến một số bệnh cụ thể phân chia theo độ tuổi, giới tính, …
* Thu thập và tiền xử lý dữ liệu tại kaggle.
* Tìm hiểu bài toán phân lớp trong khai phá dữ liệu, lựa chọn thuật toán phù hợp (cây quyết định và thuật toán Naïve Bayes).
* Phân tích và đánh giá.

# Kết cấu đồ án

Đồ án gồm 3 chương: ngoài phần mở đầu, tham khảo, phụ lục.

* Chương 1: Tổng quan về khai phá dữ liệu
* Chương 2: Xây dựng hệ hỗ trợ chuẩn đoán y khoa
* Chương 3: Ứng dụng luật phân lớp vào y khoa
* Chương 4: Hiện thực hóa mô hình với Chatbot
* Chương 5: Tổng kết

# **MỤC LỤC**

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc7516013)

[LỜI MỞ ĐẦU 3](#_Toc7516014)

[THÔNG TIN NHÓM: 4](#_Toc7516015)

[LỜI MỞ ĐẦU 5](#_Toc7516016)

[1. Sự cần thiết của đề tài 5](#_Toc7516017)

[2. Tính thực tiễn của đề tài 5](#_Toc7516018)

[3. Mục đích nghiên cứu 6](#_Toc7516019)

[4. Đối tượng và phạm vi nghiên cứu 6](#_Toc7516020)

[5. Phương pháp nghiên cứu 6](#_Toc7516021)

[6. Kết cấu đồ án 7](#_Toc7516022)

[MỤC LỤC 8](#_Toc7516023)

[DANH MỤC CÁC TỪ VIẾT TẮT 10](#_Toc7516024)

[CHƯƠNG 1: TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU 11](#_Toc7516025)

[1.1 Sự cần thiết của khai phá dữ liệu (datamining) 11](#_Toc7516026)

[1.2 Khai phá dữ liệu là gì 11](#_Toc7516027)

[1.3 Các phương pháp khai thác dữ liệu 11](#_Toc7516028)

[1.4 Ứng dụng của khai phá dữ liệu 12](#_Toc7516029)

[CHƯƠNG 2: XÂY DỰNG HỆ HỖ TRỢ CHUẨN ĐOÁN Y KHOA 13](#_Toc7516030)

[2.1 Tổng quan về mô hình chuẩn đoán 13](#_Toc7516031)

[2.2 Tổng quan về kho dữ liệu 13](#_Toc7516032)

[2.2.1 Danh sách thuộc tính 13](#_Toc7516033)

[2.2.2 Tiền xử lý dữ liệu 15](#_Toc7516034)

[2.2.3 Phân tích dữ liệu bệnh tim 15](#_Toc7516035)

[2.2.4 Tổng quan về thuật toán phân lớp 15](#_Toc7516036)

[2.2.5 Bệnh án mẫu 18](#_Toc7516037)

[2.2.6 Chẩn đoán 18](#_Toc7516038)

[CHƯƠNG 3: ỨNG DỤNG LUẬT PHÂN LỚP VÀO Y KHOA 19](#_Toc7516039)

[3.1 Decision Tree (Cây quyết định, ID3) 19](#_Toc7516040)

[3.1.1 Một số thư viện cần sử dụng 19](#_Toc7516041)

[3.1.2 Tạo class TreeNode 19](#_Toc7516042)

[3.1.3 Tạo class DecisionTreeID3 20](#_Toc7516043)

[3.1.4 Hàm tính entropy dựa trên tuần suất 20](#_Toc7516044)

[3.1.5 Tính entropy của nút với các vị trí ids 21](#_Toc7516045)

[3.1.6 Phân loại cho nút lá 21](#_Toc7516046)

[3.1.7 Trả về danh sách node con 21](#_Toc7516047)

[3.1.8 Dự đoán kết quả phân loại 23](#_Toc7516048)

[3.1.9 Tạo cấu trúc cây 24](#_Toc7516049)

[3.2 Naïve Bayes 25](#_Toc7516050)

[3.2.1 Một số thư viện cần sử dụng 25](#_Toc7516051)

[3.2.2 Tạo tập dữ liệu và tải dữ liệu 26](#_Toc7516052)

[3.2.3 Tính độ lệch chuẩn 26](#_Toc7516053)

[3.2.4 Tiền xử lý dữ liệu 27](#_Toc7516054)

[3.2.5 Tính xác suất của từng biến liên tục theo phân phối Gausian 28](#_Toc7516055)

[3.2.6 Dự đoán dựa vào xác suất 29](#_Toc7516056)

[3.3 Kiểm tra tính chính xác của thuật toán 30](#_Toc7516057)

[3.3.1 Thông tin tập dữ liệu 30](#_Toc7516058)

[3.3.2 Decision Tree 30](#_Toc7516059)

[3.3.3 Naïve Bayes 33](#_Toc7516060)

[3.4 Xây dựng mô hình cây quyết định bằng Sklearn 35](#_Toc7516061)

[3.5 Đánh giá 36](#_Toc7516062)

[CHƯƠNG 4: HIỆN THỰC HÓA MÔ HÌNH VỚI CHATBOT 38](#_Toc7516063)

[CHƯƠNG 5: TỔNG KẾT 39](#_Toc7516064)

[5.1 Tổng kết 39](#_Toc7516065)

[5.2 Ưu điểm 39](#_Toc7516066)

[5.3 Hạn chế của đề tài 39](#_Toc7516067)

[5.4 Hướng phát triển 40](#_Toc7516068)

[Bảng đánh giá chéo các thành viên trong nhóm 41](#_Toc7516069)

[TÀI LIỆU THAM KHẢO 43](#_Toc7516070)

# **DANH MỤC CÁC TỪ VIẾT TẮT**

# **TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU**

## Sự cần thiết của khai phá dữ liệu (datamining)

Khoảng hơn một thập kỷ trở lại đây, lượng thông tin được lưu trữ trên các thiết bị điện tử (đĩa cứng, CD-ROM, băng từ, .v.v.) không ngừng tăng lên. Sự tích lũy dữ liệu này xảy ra với một tốc độ bùng nổ. Người ta ước đoán rằng lượng thông tin trên toàn cầu tăng gấp đôi sau khoảng hai năm và theo đó số lượng cũng như kích cỡ của các cơ sở dữ liệu (CSDL) cũng tăng lên một cách nhanh chóng. Nói một cách hình ảnh là chúng ta đang “ngập” trong dữ liệu nhưng lại “đói” tri thức. Câu hỏi đặt ra là liệu chúng ta có thể khai thác được gì từ những “núi” dữ liệu tưởng chừng như “bỏ đi” ấy không ?

“Necessity is the mother of invention” - Data Mining ra đời như một hướng giải quyết hữu hiệu cho câu hỏi vừa đặt ra ở trên . Khá nhiều định nghĩa về Data Mining và sẽ được đề cập ở phần sau, tuy nhiên có thể tạm hiểu rằng Data Mining như là một công nghệ tri thức giúp khai thác những thông tin hữu ích từ những kho dữ liệu được tích trữ trong suốt quá trình hoạt động của một công ty, tổ chức nào đó.

## Khai phá dữ liệu là gì

Khai phá dữ liệu (data mining) Là quá trình tính toán để tìm ra các mẫu trong các bộ dữ liệu lớn liên quan đến các phương pháp tại giao điểm của máy học, thống kê và các hệ thống cơ sở dữ liệu. Đây là một lĩnh vực liên ngành của khoa học máy tính.. Mục tiêu tổng thể của quá trình khai thác dữ liệu là trích xuất thông tin từ một bộ dữ liệu và chuyển nó thành một cấu trúc dễ hiểu để sử dụng tiếp. Ngoài bước phân tích thô, nó còn liên quan tới cơ sở dữ liệu và các khía cạnh quản lý dữ liệu, xử lý dữ liệu trước, suy xét mô hình và suy luận thống kê, các thước đo thú vị, các cân nhắc phức tạp, xuất kết quả về các cấu trúc được phát hiện, hiện hình hóa và cập nhật trực tuyến. Khai thác dữ liệu là bước phân tích của quá trình "khám phá kiến thức trong cơ sở dữ liệu" hoặc KDD.

## Các phương pháp khai thác dữ liệu

* Phân loại (Classification): Là phương pháp dự báo, cho phép phân loại một đối tượng vào một hoặc một số lớp cho trước.
* Hồi qui (Regression): Khám phá chức năng học dự đoán, ánh xạ một mục dữ liệu thành biến dự đoán giá trị thực.
* Phân nhóm (Clustering): Một tác vụ mô tả chung trong đó người ta tìm cách xác định một tập hợp hữu hạn các danh mục hoặc cụm để mô tả dữ liệu.
* Tổng hợp (Summarization): Một tác vụ mô tả bổ sung liên quan đến các phương pháp tìm mô tả nhỏ gọn cho một tập hợp (hoặc tập hợp con) dữ liệu.
* Mô hình ràng buộc (Dependency modeling): Tìm một mô hình cục bộ mô tả các phụ thuộc đáng kể giữa các biến hoặc giữa các giá trị của một tính năng trong một tập dữ liệu hoặc trong một phần của tập dữ liệu.
* Dò tìm biến đổi và độ lệch (Change and Deviation Dectection): Khám phá những thay đổi quan trọng nhất trong bộ dữ liệu.

## Ứng dụng của khai phá dữ liệu

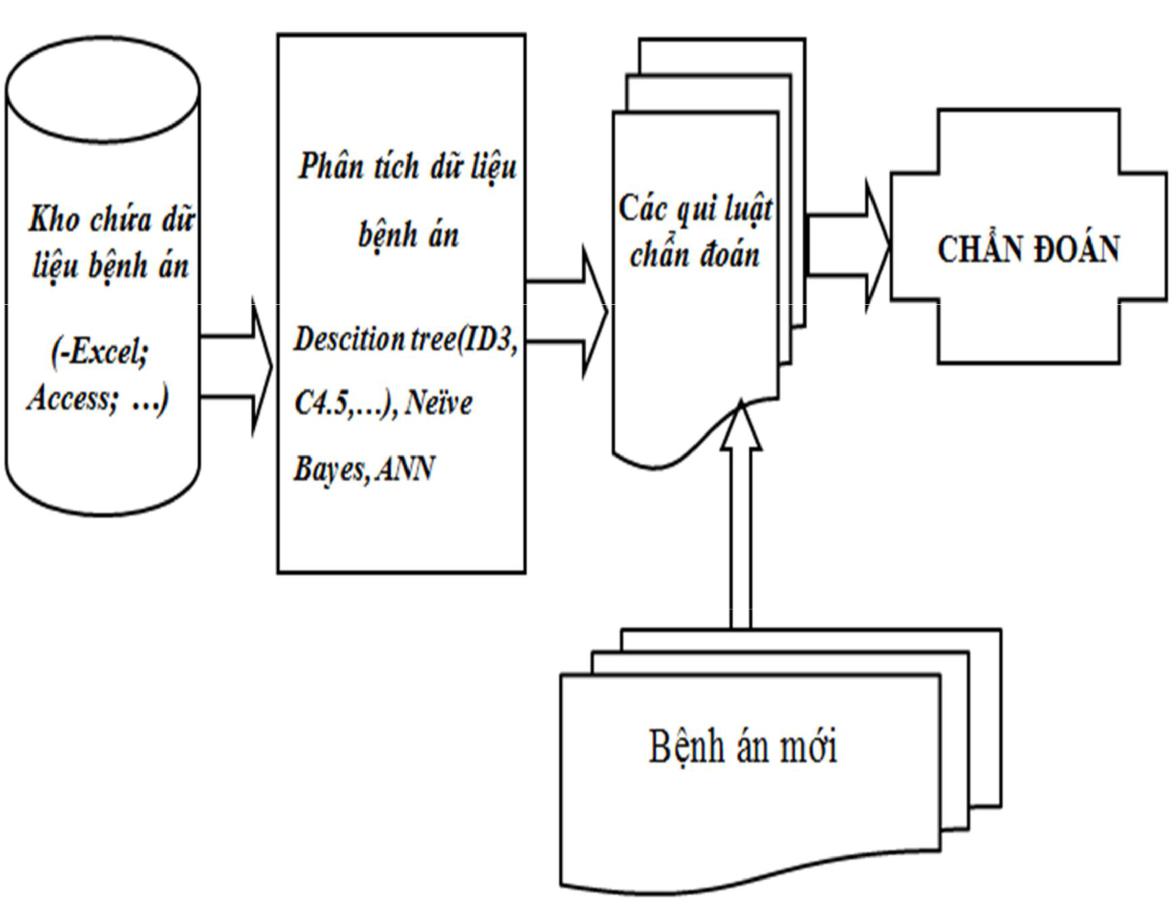
Sau đây là một số lĩnh vực mà Data mining đang được ứng dụng rộng rãi:

* Phân tích dữ liệu tài chính (Financial Data Analysis)
* Công nghiệp bán lẻ (Retail Industry)
* Công nghiệp viễn thông (Telecommunication Industry)
* Phân tích dữ liệu sinh học (Biological Data Analysis)
* Phát hiện xâm nhập (Intrusion Detection)
* Một số ứng dụng trong khoa học (Scientific Applications)

# **XÂY DỰNG HỆ HỖ TRỢ CHUẨN ĐOÁN Y KHOA**

## Tổng quan về mô hình chuẩn đoán

Sau khi thu thập dữ liệu ta cần xây dựng cơ sở dữ liệu, lưu trữ các thông tin cần thiết cho bộ điều khiển theo mô hình sau:



Hình 1: Mô hình xây dựng giải pháp hỗ trợ chẩn đoán bệnh

## Tổng quan về kho dữ liệu

### Danh sách thuộc tính

Cơ sở dữ liệu này chứa 76 thuộc tính, nhưng tất cả các thử nghiệm được xuất bản đều đề cập đến việc sử dụng tập hợp con gồm 14 thuộc tính. Đặc biệt, cơ sở dữ liệu của Cleveland là cơ sở dữ liệu duy nhất được các nhà nghiên cứu ML sử dụng cho đến ngày nay. Trường "mục tiêu" đề cập đến sự hiện diện của bệnh tim ở bệnh nhân. Nó là số nguyên có giá trị từ 0 (không có sự hiện diện) đến 4.

Link: <https://www.kaggle.com/ronitf/heart-disease-uci>

Thông tin thuộc tính có trong cơ sở dữ liệu bệnh tim:

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Thuộc tính** | **Kiểu** | **Mô tả** |
| 1 | Age | Numeric | Tuổi theo năm |
| 2 | Sex | Numeric | Giới tính (1 = nam; 0 = nữ) |
| 3 | chest pain type (cp) | Numeric | cp: Loại đau ngực   * Giá trị 1: đau thắt ngực điển hình * Giá trị 2: đau thắt ngực không điển hình * Giá trị 3: đau không đau thắt ngực * Giá trị 4: không có triệu chứng |
| 4 | resting blood pressure (trestbps) | Numeric | Huyết áp khi nghỉ ngơi (tính bằng mm Hg khi nhập viện) |
| 5 | serum cholestoral (chol) | Numeric | cholestoral huyết thanh tính theo mg / dl |
| 6 | fasting blood sugar (fbs) | Numeric | Đường huyết lúc đói> 120 mg / dl) (1 = đúng; 0 = sai) |
| 7 | resting electrocardiographic results (restecg) | Numeric | restecg: Kết quả điện tâm đồ  - Giá trị 0: bình thường  - Giá trị 1: có bất thường sóng ST-T (đảo ngược sóng T và / hoặc ST chênh lên hoặc trầm cảm> 0,05 mV)  - Giá trị 2: cho thấy chứng phì đại thất trái có thể xảy ra hoặc xác định theo tiêu chí của Estes |
| 8 | maximum heart rate achieved (thalach) | Numeric | Nhịp tim tối đạt |
| 9 | exercise induced angina (exang) | Numeric | Tập thể dục gây đau thắt ngực (1 = có; 0 = không) |
| 10 | oldpeak | Numeric | Chỉ số trầm cảm |
| 11 | the slope of the peak exercise ST segment (slope**)** | Numeric | Độ dốc của bài tập đỉnh ST đoạn : - Giá trị 1: upsloping  - Giá trị 2: flat  - Giá trị 3 : downsloping |
| 12 | number of major vessels (ca) | Numeric | Number of major vessels (0-3) colored by flourosopy |
| 13 | thal | Numeric | Giá trị 3: bình thường;  Giá trị 6: khuyết tật cố định  Giá trị 7: khuyết tật có thể đảo ngược |
| 14 | target | Numeric | Giá trị phân lớp:  Giá trị 0: Không mắc bệnh  Giá trị 1: Mắc bệnh |

### Tiền xử lý dữ liệu

Dữ liệu sau khi thu thập sẽ lưu vào file excel sau đó import vào database. Dữ liệu được nhập trên một hàng (tupe) bao gồm các thuộc tính:

*Age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, target.*

Để thực hiện mô hình khai phá luật kết hợp ta cần hiệu chỉnh lại dữ liệu và loại bỏ các thuộc tính không cần thiết:

* Loại bỏ các thuộc tính mà dữ liệu bị thiếu hoặc bị nhiễu quá nhiều.
* Rời rạc hóa dữ liệu

### Phân tích dữ liệu bệnh tim

* Phân tích dữ liệu bệnh án điện tử để tìm qui luật chẩn đoán khi được cung cấp các thông tin về triệu chứng lâm sàng và cận lâm sàng của bệnh nhân.
* Các giải thuật được sử dụng để xây dựng hệ thống là sử dụng giải thuật tiêu biểu: Cây quyết định với thuật toán ID3 dùng Gain – Entropy làm độ đo để lựa chọn thuộc tính tốt nhất và thuật toán Naïve Bayes

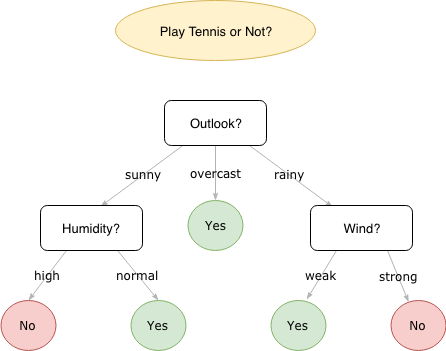
### Tổng quan về thuật toán phân lớp

1. Decision Tree (Cây quyết định, ID3)

Cây quyết định (gọi tắt là DT) là mô hình đưa ra quyết định dựa trên các câu hỏi.

* Dưới đây là mô hình DT về một ví dụ kinh điển.

Câu hỏi có chơi tennis hay không? Quyết định đưa ra dựa trên các yếu tố về thời tiết: outlook, humidity, wind.

1. 

**Iterative Dichotomiser 3** (ID3) là thuật toán nổi tiếng để xây dựng Decision Tree, áp dụng cho bài toán Phân loại (Classification) mà tất các các thuộc tính để ở dạng category.

*Hàm xây dựng cây quyết định trong thuậ t toán ID3:*

**Function** induce\_tree(tập\_ví\_dụ, tập\_thuộc\_tính)

**begin**

**if** mọi ví dụ trong tập\_ví\_dụ đều nằm trong cùng một lớp **then**

**return** một nút lá được gán nhãn bởi lớp đó

**else** **if** tập\_thuộc\_tính là rỗng **then**

**return** nút lá được gán nhãn bởi tuyển của tất cả các lớp trong tập\_ví\_dụ

**else** **begin**

chọn một thuộc tính P, lấy nó làm gốc cho cây hiện tại;

xóa P ra khỏi tập\_thuộc\_tính;

với mỗi giá trị V của P

**begin**

tạo một nhánh của cây gán nhãn V;

Đặt vào phân\_vùngV các ví dụ trong tập\_ví\_dụ có giá trị V tại

thuộc tính P;

Gọi induce\_tree(phân\_vùngV, tập\_thuộc\_tính), gắn kết quả vào nhánh V

**end**

**end**

**end**

1. Naïve Bayes

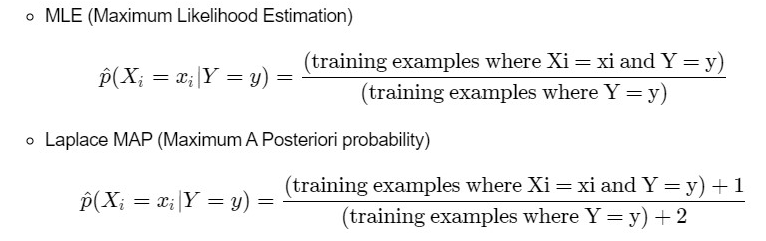
**Naive Bayes Classification** (NBC) là một thuật toán phân loại dựa trên tính toán xác suất áp dụng định lý Bayes

* Trên thực tế thì ít khi tìm được dữ liệu mà các thành phần là hoàn toàn độc lập với nhau. Tuy nhiên giả thiết này giúp cách tính toán trở nên đơn giản, training data nhanh, đem lại hiệu quả bất ngờ với các lớp bài toán nhất định.
* Cách xác định các thành phần (class) của dữ liệu dựa trên giả thiết này có tên là **Naive Bayes Classifier**.

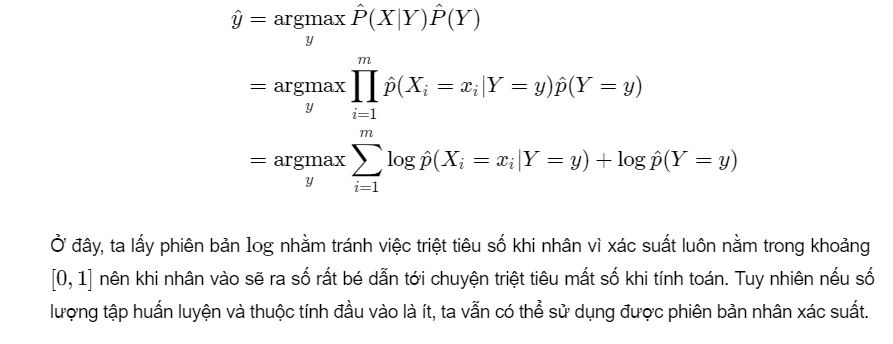
*Giải thuật Naive Bayes được trình bày như sau:*

* **Huấn luyện**

Mục tiêu của quá trình huấn luyện là đánh giá xác suất *P*(*Y*) và *P*(*Xi*​∣*Y*) cho mọi thuộc tính  . Để làm việc này, ta có thể sử dụng 1 trong 2 cách đánh giá sau:



* **Dự đoán**

****

### Bệnh án mẫu

Bệnh án mẫu là hồ sơ bệnh án của bệnh nhân bao gồm các thông tin về các triệu chứng lâm sàng, cận lâm sàng và kết luận chẩn đoán của bệnh nhân. Bệnh án mẫu được dùng để đưa các thông tin về lâm sàng và cận lâm sàng của bệnh nhân vào hệ thống, các thông tin này sẽ được phân tích dựa trên các qui luật chẩn đoán của hệ thống.

### Chẩn đoán

Sau khi đưa thông tin từ bệnh án mẫu vào hệ thống, chương trình dựa vào các luật đã được rút ra trong quá trình huấn luyện sẽ phân bệnh án mẫu vào lớp tương ứng của hệ thống để cho ra kết quả chẩn đoán. Kết quả này sẽ được so sánh với kết luận chẩn đoán trong bệnh án mẫu.

# **ỨNG DỤNG LUẬT PHÂN LỚP VÀO Y KHOA**

Tập dữ liệu: <https://www.kaggle.com/ronitf/heart-disease-uci>

## Decision Tree (Cây quyết định, ID3)

### Một số thư viện cần sử dụng

from \_\_future\_\_ import print\_function

import numpy as np

import pandas as pd

import csv

import random

import time

start = time.time()

### Tạo class TreeNode

Tạo class TreeNode(object) tạo ra các tham số trong lớp.

Hàm set\_properties() truyền vào các thuộc tính.

Hàm set\_label() truyền và label.

class TreeNode(object):

def \_\_init\_\_(self, ids=None, children=[], entropy=0, depth=0):

self.ids = ids # index of data in this node

self.entropy = entropy # entropy, will fill later

self.depth = depth # distance to root node

self.split\_attribute = None # which attribute is chosen, it non-leaf

self.children = children # list of its child nodes

self.order = None # order of values of split\_attribute in children

self.label = None # label of node if it is a leaf

def set\_properties(self, split\_attribute, order):

self.split\_attribute = split\_attribute

self.order = order

def set\_label(self, label):

self.label = label

### Tạo class DecisionTreeID3

Tương tự như class TreeNode(object), class này tạo ra các tham số trong lớp.

class DecisionTreeID3(object):

def \_\_init\_\_(self, max\_depth=13, min\_samples\_split=2, min\_gain=1e-4):

self.root = None

self.max\_depth = max\_depth

self.min\_samples\_split = min\_samples\_split

self.Ntrain = 0

self.min\_gain = min\_gain

### Hàm tính entropy dựa trên tuần suất

Trong hàm này, chúng ta phải chú ý bỏ các tần suất bằng 0 đi vì logarit tại đây không xác định.

def entropy(freq):

# Tính entropy

freq\_0 = freq[np.array(freq).nonzero()[0]]

prob\_0 = freq\_0 / float(freq\_0.sum())

return -np.sum(prob\_0 \* np.log(prob\_0))

### Tính entropy của nút với các vị trí ids

Hàm \_entropy dùng để trả về mảng số lần xuất hiện của phần tử rồi từ đó truyền tham số này vào hàm entropy để tính entropy cho từng phần tử của ids.

def \_entropy(self, ids):

# Tính entropy của nút với các vị trí ids

if len(ids) == 0: return 0

ids = [i + 1 for i in ids] # Chỉ số bắt đầu từ 1

freq = np.array(self.target[ids].value\_counts()) # Trả về 1 mảng số lần suất hiện của các giá trị khác nhau trong thuộc tính kết quả

return entropy(freq)

### Phân loại cho nút lá

Hàm \_set\_lable(self, node) truyền vào một node, nếu đó là node lá thì sẽ truyền giá trị phân lại cho node đó

def \_set\_label(self, node):

# Tìm nhãn(kết quả) cho 1 nút nếu nó là nút lá

target\_ids = [i + 1 for i in node.ids]

node.set\_label(self.target[target\_ids].mode()[0])

# Chọn giá trị phổ biến nhất trong nút lá làm nhãn

### Trả về danh sách node con

Hàm \_split truyền vào danh sách node, trả về các node con của node đó.

def \_split(self, node):

ids = node.ids

best\_gain = 0

best\_splits = []

best\_attribute = None

order = None

sub\_data = self.data.iloc[ids, :] # Lấy tất cả các mẫu ở các vị trí ids

for i, att in enumerate(self.attributes): # enumerate cài mảng có thêm giá trị unique 0 1 2... cho từng giá trị

values = self.data.iloc[ids, i].unique().tolist() # Lấy các mẫu vị trí ids, ở cột thứ i(ứng với từng thuộc tính điều kiện)

if len(values) == 1: continue # entropy = 0

splits = []

for val in values:

sub\_ids = sub\_data.index[sub\_data[att] == val].tolist() # Trả về 1 mảng chứa vị trí các mẫu có cùng giá trị thuộc tính điều kiện i

splits.append([sub\_id - 1 for sub\_id in sub\_ids]) # Giảm tất cả các giá trị của sub\_ids đi 1 và chèn sub\_ids vào splits

if min(map(len, splits)) < self.min\_samples\_split: continue # Không phân chia nếu 1 nếu splits có phần tử có chiều dài nhỏ hơn min\_samples\_split

# information gain

HxS = 0

for split in splits:

HxS += len(split) \* self.\_entropy(split) / len(ids) # Tính H(X,S)

gain = node.entropy - HxS # Định nghĩa information gain

if gain < self.min\_gain: continue # Dừng lại nếu gain < min\_gain

if gain > best\_gain: # Tìm ra gain lớn nhất trong i và chọn nó làm nút kế tiếp

best\_gain = gain

best\_splits = splits

best\_attribute = att

order = values

node.set\_properties(best\_attribute, order) # Đặt giá trị cho nút

print("test cuoi", best\_attribute, order)

# Tạo các child\_nodes với từng nút là các mẫu thuộc best\_splits

child\_nodes = [TreeNode(ids=split, entropy=self.\_entropy(split), depth=node.depth + 1) for split in best\_splits]

return child\_nodes

### Dự đoán kết quả phân loại

Hàm predict(self, new\_data): Truyền vào một tập dữ liệu gồm nhiều dòng theo câu trúc của tập dữ liệu huấn luyện, hàm sẽ trả về nhãn dữ đoán cho mỗi dòng.

def predict(self, new\_data):

"""

:new\_data la 1 tập dữ liệu khác, mỗi dòng là 1 mẫu tương tự như dữ liệu huấn luyện

:Hàm trả về nhãn dự đoán cho mỗi hàng

"""

npoints = new\_data.count()[0]

labels = [None] \* npoints # Tạo ra mảng None

for n in range(npoints):

x = new\_data.iloc[n, :] # one point

# start from root and recursively travel if not meet a leaf

node = self.root

while node.children:

node = node.children[node.order.index(

x[node.split\_attribute])] # Tìm ra nút lá dựa theo giá trị của các thuộc tính điều kiện

labels[n] = node.label # Trả về dự đoán dựa trên các thuộc tính điều kiện

return labels

### Tạo cấu trúc cây

Hàm fit(self, data, target) truyền vào các giá trị thuộc tính điều kiện và thuộc tính phân lớp. Trả về danh sách các node của thuộc tính điều kiện dựa vào hàm \_split.

def fit(self, data, target):

self.Ntrain = data.count()[0] # Hàm đếm số lượng mẫu có trong tập dữ liệu huấn luyện

self.data = data # Tương ứng với X

self.attributes = list(data) # Chứa các loại thuộc tính điều kiện :outlook,temperature,humidity,wind

self.target = target # Thuộc tính kết quả(play)

self.labels = target.unique() # Trả về số lần xuất hiện của các thuộc tính riêng biệt của thuộc tính kết quả

ids = range(self.Ntrain) # Hàm trả về 1 mảng các giá trị từ 0 đến self.Ntrain

self.root = TreeNode(ids=ids, entropy=self.\_entropy(ids), depth=0) # Hàm tạo nút

queue = [self.root]

while queue:

node = queue.pop() # Lấy giá trị sau cùng của queue

if node.depth < self.max\_depth or node.entropy < self.min\_gain: # Kiểm tra điều kiện tạo nút con

node.children = self.\_split(node) # Trả về các nút con sau khi đã phân chia dự trên thuộc tính điều kiện

if not node.children: # node.children là nút lá

self.\_set\_label(node) # Thiết lập nhãn

queue += node.children # Đẩy tất cả các node.children vào queue

else:

self.\_set\_label(node) # Thiết lập nhãn

## Naïve Bayes

### Một số thư viện cần sử dụng

import csv

import random

import math

import time

start = time.time()

### Tạo tập dữ liệu và tải dữ liệu

Dữ liệu của chúng ta được lưu dưới dạng file CSV nên chúng ta sẽ sử dụng thư viện csv của Python để đọc dữ liệu

# Load data từ CSV file

def load\_data(filename):

lines = csv.reader(open(filename, "rt"))

dataset = list(lines)

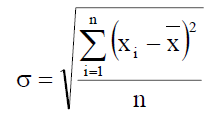
for i in range(len(dataset)):

dataset[i] = [float(x) for x in dataset[i]]

return dataset

### Tính độ lệch chuẩn

Như đã trình bày phần trên chúng ta cần một đoạn code để tính độ lệch chuẩn của biến ngẫu nhiên liên tục. Chúng ta có thể tham khảo công thức tính của nó như biểu thức sau:



Trong đó là giá trị trung bình của biến ngẫu nhiên trên toàn tập dữ liệu. Chúng ta sử dụng **Python** để thực hiện hàm tính **giá trị trung bình** và **độ lệch chuẩn** như sau:

# Tính toán giá trị trung bình của mỗi thuộc tính

def mean(numbers):

return sum(numbers) / float(len(numbers))

# Tính toán độ lệch chuẩn cho từng thuộc tính

def standard\_deviation(numbers):

avg = mean(numbers)

variance = sum([pow(x - avg, 2) for x in numbers]) / float(len(numbers) - 1)

return math.sqrt(variance)

### Tiền xử lý dữ liệu

Chúng tôi sẽ sử dụng ***Độ lệch chuẩn*** và ***giá trị trung bình*** để tính toán các xác suất cần thiết nên cần có một hàm để chuyển đổi dữ liệu ban đầu về dạng tập hợp của độ lệch chuẩn và trung bình nhằm phục vụ cho các phép tính xác suất sau này.

#Giá trị trung bình, độ lệch chuẩn

def summarize(dataset):

summaries = [(mean(attribute), standard\_deviation(attribute)) for attribute in zip(\*dataset)]

del summaries[-1]

return summaries

def summarize\_by\_class(dataset):

separated = separate\_data(dataset)

summaries = {}

for classValue, instances in separated.items():

summaries[classValue] = summarize(instances)

return summaries

### Tính xác suất của từng biến liên tục theo phân phối Gausian

Dựa vào cơ sở lý thuyết ở bên trên. Chúng ta tiến hành tính các xác suất phụ thuộc của biến ngẫu nhiên bao gồm p(x)*p*(*x*) của mỗi chỉ số sức khỏe và p(x|c)*p*(*x*∣*c*) của mỗi class tương ứng với chỉ số đó.

# Tính toán xác suất theo phân pjoois Gause của biến liên tục

def calculate\_prob(x, mean, stdev):

exponent = math.exp(-(math.pow(x - mean, 2) / (2 \* math.pow(stdev, 2))))

return (1 / (math.sqrt(2 \* math.pi) \* stdev)) \* exponent

# Tính xác suất cho mỗi thuộc tính phân chia theo class

def calculate\_class\_prob(summaries, inputVector):

probabilities = {}

for classValue, classSummaries in summaries.items():

probabilities[classValue] = 1

for i in range(len(classSummaries)):

mean, stdev = classSummaries[i]

x = inputVector[i]

probabilities[classValue] \*= calculate\_prob(x, mean, stdev)

return probabilities

### Dự đoán dựa vào xác suất

Đây là bước áp dụng định lý **Bayes** đã đước giới thiệu bên trên vào dự đoán các class thông qua các chỉ số trong tập dữ liệu

# Dự đoán vector thuộc phân lớp nào

def predict(summaries, inputVector):

probabilities = calculate\_class\_prob(summaries, inputVector)

bestLabel, bestProb = None, -1

for classValue, probability in probabilities.items():

if bestLabel is None or probability > bestProb:

bestProb = probability

bestLabel = classValue

return bestLabel

# Dự đoán tập dữ liệu testing thuộc vào phân lớp nào

def get\_predictions(summaries, testSet):

predictions = []

for i in range(len(testSet)):

result = predict(summaries, testSet[i])

predictions.append(result)

return predictions

## Kiểm tra tính chính xác của thuật toán

### Thông tin tập dữ liệu

* Tập dữ liệu bệnh tim ban đầu: <https://bitly.vn/2mtj>
* Truyền vào tập dữ liệu huấn luyện: <https://bitly.vn/2mti>
* Truyền vào tập dữ liệu thử nghiệm: <https://bitly.vn/2mth>

### Decision Tree

***Thử nghiệm kết quả với hàm main():***

Sau bước xử lý dữ liệu ban đầu chúng ta tiến hành learning như sau:

Gọi tree = DecisionTreeID3(max\_depth=3, min\_samples\_split=2) để khởi tạo cây quyết định. Trong đó

* max\_depth = 13 vì trong tập dữ liệu mẫu có 13 thuộc tính điều kiện : outlook,temperature,humidity,wind
* min\_samples\_split = 2 vì trong tập dữ liệu mẫu có 2 loại kết quả là 0, 1 (0: Chưa có dấu hiệu bệnh tim, 1: Có dấu hiệu bệnh tim)

Gọi tree.fit(X, y) để bắt đầu thực hiện cài đặt cây quyết địng trên cơ sở dữ liệu mẫu.

Tập dữ liệu huấn luyện và tập dữ liệu thử nghiệm được tác ra từ tập dữ liệu ban đầu.

if \_\_name\_\_ == "\_\_main\_\_":

# heart.csv = data3.csv

# Không thuộc tính age, sex..

filename = 'heart.csv'

dataset = load\_data(filename)

# Không thuộc tính age, sex..

testSet = load\_data('split\_data\_test.csv')

#Có thuộc tính age, sex

trainingSetData = pd.read\_csv('split\_data\_training\_co\_thuoc\_tinh.csv') # Hàm load file dữ liệu huấn luyện

# Có thuộc tính age, sex

testData = pd.read\_csv('split\_data\_test\_co\_thuoc\_tinh.csv') # Hàm load file dữ liệu huấn luyện

X = trainingSetData.iloc[:, :-1]

y = trainingSetData.iloc[:, -1]

A = testData.iloc[:, :-1]

# prepare model

tree = DecisionTreeID3(max\_depth=13, min\_samples\_split=2)

tree.fit(X, y)

print('Data size {0} \nTraining Size = {1} \nTest Size = {2}'.format(len(dataset), len(trainingSetData), len(testData)))

# test model

predictions = tree.predict(A)

# Kiểm tra độ chính xác

accuracy = get\_accuracy(testSet, predictions)

print('Accuracy of my implement: {0}%'.format(accuracy))

end = time.time()

print('Tine run', end - start)

#Thử nghiệm với sklearn

# Import Decision Tree Classifier

from sklearn.tree import DecisionTreeClassifier

training = pd.read\_csv('split\_data\_training.csv')

dataset = pd.read\_csv('split\_data\_test\_13\_thuoc\_tinh.csv')

dataTrain, labelTrain = get\_data\_label(training)

dataTest, labelTest = get\_data\_label(testSet) # testSet or testData

# Create Decision Tree classifer object

clf = DecisionTreeClassifier()

# Train Decision Tree Classifer

clf = clf.fit(X, y)

score = clf.score(dataset, labelTest)

print('Accuracy of sklearn: {0}%'.format(score \* 100))

***Kết quả thử nghiệm:***

Data size 303

Training Size = 242

Test Size = 61

Accuracy of my implement: 60.65573770491803%

Accuracy of sklearn: 78.68852459016394%

Tine run 2.9679927825927734

Process finished with exit code 0

### Naïve Bayes

***Thử nghiệm kết quả với hàm main():***

Sau bước xử lý dữ liệu ban đầu chúng ta tiến hành learning như sau:

Cho tập dữ liệu cần thử là tập testSet, có thể tự cung cấp để thử độ chính xác của thuật toán.

Các dòng dữ liệu này lấy từ model nên kết quả phân lớp hoàn toàn chính xác.

def main():

filename = 'heart.csv'

# splitRatio = 0.8

dataset = load\_data(filename)

# Dùng split\_data để chia tập dữ liệu ban đầu thành 2 tập trainingSet và testSet

# trainingSet, testSet = split\_data(dataset, splitRatio)

trainingSet = load\_data('split\_data\_training.csv')

testSet = load\_data('split\_data\_test.csv')

print('Data size {0} \nTraining Size={1} \nTest Size={2}'.format(len(dataset), len(trainingSet), len(testSet)))

# prepare model

summaries = summarize\_by\_class(trainingSet)

get\_data\_label(trainingSet)

# test model

predictions = get\_predictions(summaries, testSet)

# Kiểm tra độ chính xác

accuracy = get\_accuracy(testSet, predictions)

print('Accuracy of my implement: {0}%'.format(accuracy))

# So sánh với sklearn

dataTrain, labelTrain = get\_data\_label(trainingSet)

dataTest, labelTest = get\_data\_label(testSet)

from sklearn.naive\_bayes import GaussianNB

clf = GaussianNB()

clf.fit(dataTrain, labelTrain)

score = clf.score(dataTest, labelTest)

print('Accuracy of sklearn: {0}%'.format(score\*100))

if \_\_name\_\_ == "\_\_main\_\_":

main()

end = time.time()

print('Tine run', end - start)

***Kết quả thử nghiệm:***

Data size 303

Training Size=242

Test Size=61

Accuracy of my implement: 83.60655737704919%

Accuracy of sklearn: 100.0%

Tine run 0.749422550201416

Process finished with exit code 0

## Xây dựng mô hình cây quyết định bằng Sklearn

from sklearn.tree import export\_graphviz

from sklearn.externals.six import StringIO

from IPython.display import Image

import pydotplus

#split dataset in features and target variable

feature\_cols = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal']

dot\_data = StringIO()

export\_graphviz(clf, out\_file=dot\_data,

filled=True, rounded=True,

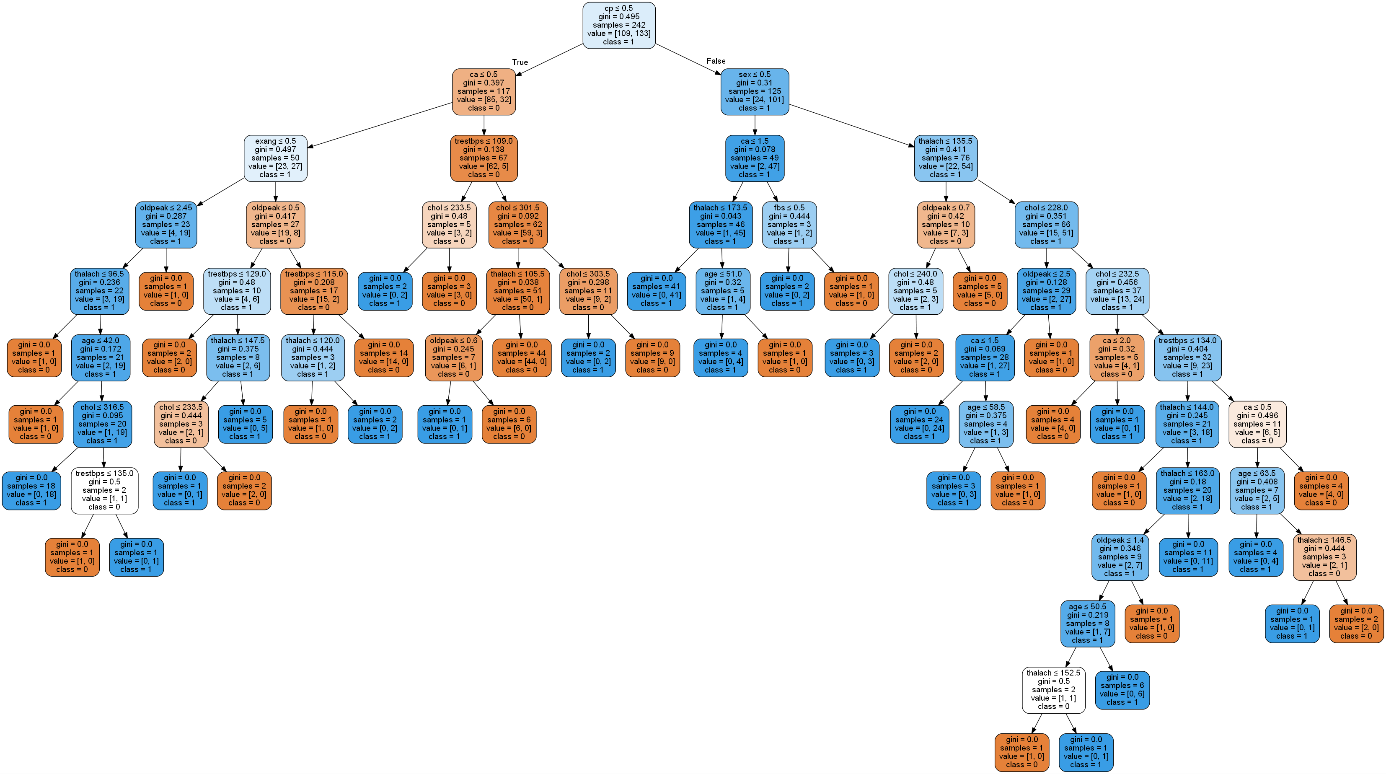
special\_characters=True,feature\_names = feature\_cols, class\_names=['0','1'])

graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())

graph.write\_png('heart\_decision\_tree.png')

Image(graph.create\_png())

Link ảnh ‘heart\_decision\_tree.png’: <https://bitly.vn/2mwt>



## Đánh giá

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thuật toán | Code | Độ chính xác | Độ sai lệch | Thời gian chạy |
| Decision Tree | Code tay | 60.66% | 39.34% | 2.97s |
| Thư viện | 77.05% | 29.25% |
| Naïve Bayes | Code tay | 83.61% | 16.39 | 0.75s |
| Thư viện | 100% | 0% |

Kết luận:

Nếu ta xét về khía cạnh thời gian thực thi và độ chính xác thì NaiveBayes là một sự lựa chọn tốt trong các trường hợp chạy thử với thời gian thực thi khoảng 0.75 seconds (ở đây nói là khoảng bởi lẽ các trường hợp chạy và kết quả được thực hiện trên các máy khác nhau nên tùy thuộc vào tốc độ xử lý của máy tính mà có sự khác biệt, tuy nhiên sự khác biệt ở đây là không đáng kể vì nhóm sử dụng máy tính có cấu hình cũng gần tương đương nhau nên ta có thể bỏ qua yếu tố này). Hơn nữa sự chênh lệch về độ chính xác của các thuật toán ở đây khá cao.

Với thuật toán cây quyết định (ID3) sau khi cài đặt ta nhận thấy thuật toán tính toán với tốc độ trung bình, không những thế độ chính xác chỉ trong khoảng 65% và với thư viện sklearn thì độ chính xác trong khoảng 78%, chúng tôi đã thực hiện hơn 20 lần với những loại tập dữ liệu khác nhau nhưng kết quả chưa khả quan. Tuy vậy, thử nghiệm này tùy thuộc vào cách phân chia dữ liệu nên chúng ta hoàn toàn có thể tin tưởng thuật toán này.

Bên cạch đó, với thuật toán Naïve Bayes sau khi cài đặt ta nhận thấy tốc độ rất nhanh với bộ dữ liệu thử nghiệm. Ngoài tốc độ nhanh, thuật toán còn cho chúng ta thấy độ chính xác khá cao với hơn 84%, tiếp tục thử nghiệm với thư viện sklearn chúng tôi đã nhìn thấy độ chính xác 100%.

Ta nhận thấy, với tập thuộc tính nhỏ chúng ta nhận thấy 2 thuận toán trên thì độ chính xác tương đối chấp nhận được. Tuy nhiên, đối với tập thuộc tính lớn, chúng tôi đang tiếp tục thử nghiệm nhưng kết quả thu lại được chưa tốt.

# **HIỆN THỰC HÓA MÔ HÌNH VỚI CHATBOT**

# **TỔNG KẾT**

## Tổng kết

Đồ án đưa ra một cách nhìn về sự kết hợp Công nghệ thông tin vào lĩnh vực Y tế. Kết quả đề tài là hệ hỗ trợ chẩn đoán mang tính chất cộng đồng, sẽ giúp rất nhiều về mặt chuyên môn đối với các tuyến y tế chưa có đội ngũ bác sĩ có trình độ chuyên môn cao, thiếu trang thiết bị y tế, cũng như các tuyến y tế vùng sâu vùng xa. Ngoài ra, đối với sinh viên y khoa và các bác sĩ trẻ, hệ hỗ trợ chẩn đoán giúp ôn lại các kiến thức về bệnh nhiễm ở vùng nhiệt đới cụ thể là cho biết sau khi bệnh nhân bị SXH sẽ biết được mức độ của bệnh để có hướng điều trị chính xác.

Kết quả đề tài vẫn chưa thật sự tốt, kết quả chẩn đoán của hệ hỗ trợ chẩn đoán vẫn còn nhiều trường hợp chưa đúng. Tuy nhiên sau khi triển khai thử nghiệm cho các bác sĩ chuyên khoa tại Bệnh viện Bệnh Nhiệt Đới, kết quả đề tài được đánh giá cao. Theo nhận định của các bác sĩ, nếu tiếp tục được đầu tư và phát triển, hệ hỗ trợ chẩn đoán sẽ giúp ích rất nhiều cho bác sĩ trong việc chẩn đoán và điều trị bệnh cho bệnh nhân.

Ngoài ra đề tài có thể được phát triển để chẩn đoán nhanh và chính xác hơn khi hệ hỗ trợ chẩn đoán được kết nối trực tiếp với các hệ thống thông tin quản lý bệnh án tại cơ sở y tế. Hơn nữa, hệ hỗ trợ chẩn đoán có thể áp dụng cho nhiều loại bệnh khác nhau.

## Ưu điểm

Triển khai thành công các thuật toán phân lớp cây quyết định (Decision Tree ID3) và Naïve Bayes đối với dữ liệu hiện có.

Nâng cao khả năng làm việc nhóm.

## Hạn chế của đề tài

Về mặt công nghệ, đề tài sử dụng thuật toán phân lớp có một số hạn chế về vấn đề xử lý dữ liệu. Trong trường hợp dữ liệu có quá nhiều lớp thuật toán sẽ dễ gây ra lỗi và dữ liệu càng nhiều, thời gian huấn luyện càng lâu.

Về vấn đề thực tiễn, dữ liệu của đề tài nghiên cứu thu thập tại một thời điểm cắt ngang chính vì vậy không quan sát rõ được diễn tiến bệnh của bệnh nhân. Để đạt được kết quả tốt hơn ta cần thu thập dữ liệu từ khi bệnh nhân có những triệu chứng ban đầu đến khi phát bệnh và khỏi bệnh. Đồng thời cần lấy thêm dữ liệu của quá trình điều trị bệnh của bệnh nhân, điều đó sẽ cho ta hiểu hơn quá trình sinh bệnh và giúp cho chương trình đạt được hiệu quả thực tiễn cao.

Chưa hiểu rõ hết toàn bộ ý nghĩa một số thuộc tính của thuật toán.

## Hướng phát triển

Khai phá dữ liệu là bài toán được nhiều nhà nghiên cứu quan tâm bởi nó được ứng dụng rộng rãi trong các lĩnh vực cũng như chứa đựng nhiều hướng mở rộng khác nhau. Tuy nhiên để mở rộng ứng dụng và được đưa vào thực tiễn thì ta cần làm thêm một số công việc sau:

* Số liệu bệnh nhân phải được thu thập nhiều hơn.
* Thu thập số liệu diễn tiến bệnh về lâm sàng và cận lâm sàng.
* Thu thập diễn tiến phác đồ điều trị.
* Xử lý dữ liệu tốt hơn để tăng hiệu suất thực thi chương trình.
* Xây dựng hệ thống chẩn đoán bệnh cho nhiều loại bệnh khác nhau.
* Cần sự hợp tác chuyên môn của các chuyên gia công nghệ thông tin và y tế.
* Tìm hiểu thuật toán khác như C5.0, mạng Bayesian hoặc mạng Neuron nếu có hiệu quả hơn.
* Tìm hiểu thuật toán ILA để so sánh các kết quả đạt được đối với thuật toán ID3 và NaiveBayes

# **Bảng đánh giá chéo các thành viên trong nhóm**

(Người nào đánh giá thì cột của người đó sẽ đánh dấu X, còn người khác sẽ là điểm)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nội dung đánh giá | Thang điểm | | Quang | Nhí |
| Tham gia các buổi họp nhóm | Đầy đủ | 15 | 15 | X |
| Thường xuyên | 10 |
| Một vài buổi | 5 |
| Không buổi nào | 0 |
| Hoàn thành nhiệm vụ của nhóm giao đúng thời hạn | Luôn luôn | 15 | 15 | X |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Tham gia đóng góp ý kiến nhóm | Tích cực | 15 | 15 | X |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Có ý tưởng mới, hay, sáng tạo đóng góp cho nhóm | Luôn luôn | 15 | 15 | X |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Vai trò trong nhóm | Nhóm trưởng | 15 | 15 | X |
| Thư kí | 10 |
| Tổng điểm | | | 75 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nội dung đánh giá | Thang điểm | | Đoàn | Tín |
| Tham gia các buổi họp nhóm | Đầy đủ | 15 | X | 15 |
| Thường xuyên | 10 |
| Một vài buổi | 5 |
| Không buổi nào | 0 |
| Hoàn thành nhiệm vụ của nhóm giao đúng thời hạn | Luôn luôn | 15 | X | 15 |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Tham gia đóng góp ý kiến nhóm | Tích cực | 15 | X | 15 |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Có ý tưởng mới, hay, sáng tạo đóng góp cho nhóm | Luôn luôn | 15 | X | 15 |
| Thường xuyên | 10 |
| Thỉnh thoảng | 5 |
| Không bao giờ | 0 |
| Vai trò trong nhóm | Nhóm trưởng | 15 | X | 10 |
| Thư kí | 10 |
| Tổng điểm | | |  | 75 |

**Tổng kết**

|  |  |
| --- | --- |
| Thành viên | Điểm trung bình |
| Trần Văn Quang | 75 |
| Ngô Minh Nhí | 70 |

# **TÀI LIỆU THAM KHẢO**