TRƯỜNG ĐẠI HỌC ĐIỆN LỰC

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO CHUYÊN ĐỀ HỌC PHẦN

KHAI PHÁ DỮ LIỆU

<u>Đề tài</u> : Sử dụng thuật toán cây quyết phân lớp để dự đoán bệnh tim

Sinh viên thực hiện : BÙI ĐÌNH SƠN

Giảng viên hướng dẫn : ĐÀO NAM ANH

Ngành : CÔNG NGHỆ THÔNG TIN

Lớp : D14QTANM2

Khóa : 2019-2014

Hà Nội , Tháng 6 năm 2022

PHIẾU CHẤM ĐIỂM

STT	Họ tên sinh viên	Nội Dung	Điểm	Chữ ký
1	Bùi Đình Sơn			
2	Nguyễn Lam Trường			
3	Phạm Thanh Hải			

STT	Họ và tên giáo viên	Chữ kí	Ghi chú
1			
2			

LỜI CẢM ƠN

Nhóm chúng em xin chân thành cảm ơn các thầy, cô giáo trong Khoa Công nghệ thông tin, trường Đại học Điện Lực, đã tạo điều kiện cho em thực hiện đề tài này.

Để có thể hoàn thành báo cáo đề tài "Sử dụng thuật toán phân lớp cây quyết định để dự đoán bệnh tim", nhóm em xin gửi lời cảm ơn chân thành nhất tới thầy ***Đào Nam Anh*** , đã truyền đạt, giảng dạy cho chúng em những kiến thức, những kinh nghiệm qúy báu trong thời gian học tập và rèn luyện, tận tình hướng dẫn chúng em trong quá trình làm báo cáo này.

Nhóm em cũng gửi lời cảm ơn tới bạn bè đã đóng góp những ý kiến quý báu để nhóm em có thể hoàn thành báo cáo tốt hơn.

Tuy nhiên, do thời gian và trình độ có hạn nên báo cáo này chắc chắn không tránh khỏi những thiếu sót, nhóm em rất mong được sự đóng góp ý kiến của các thầy và toàn thể các bạn.

Nhóm sinh viên thực hiện

Bùi Đình Sơn

Nguyễn Lam Trường

Phạm Thanh Hải

LỜI MỞ ĐẦU

Trong thời buổi hiện đại ngày nay, công nghệ thông tin cũng như những ứng dụng của nó không ngừng phát triển, lượng thông tin và cơ sở dữ liệu được thu thập và lưu trữ cũng tích lũy ngày một nhiều lên. Con người cũng vì thế mà cần có thông tin với tốc độ nhanh nhất để đưa ra quyết định dựa trên lượng dữ liệu khổng lồ đã có. Các phương pháp quản trị và khai thác cơ sở dữ liệu truyền thống ngày càng không đáp ứng được thực tế. Vì thế, một khuynh hướng kỹ thuật mới là kỹ thuật phát hiện tri thức và khai phá dữ liệu nhanh chóng được phát triển.

Khai phá dữ liệu đã và đang được nghiên cứu, ứng dụng trong nhiều lĩnh vực khác nhau ở các nước trên thế giới. Ở Việt Nam, kỹ thuật này đang được nghiên cứu và dần đưa vào ứng dụng. Khai phá dữ liệu là một bước trong quy trình phát hiện tri thức. Hiện nay, mọi người không ngừng tìm tòi các kỹ thuật để thực hiện khai phá dữ liệu một cách nhanh nhất và có được kết quả tốt nhất.

Trong bài tập lớn này, chúng em tìm hiểu và trình bày về một kỹ thuật trong khai phá dữ liệu để phân lớp dữ liệu cũng như tổng quan về khai phá dữ liệu, với đề tài "Sử dụng thuật toán cây quyết định để dự đoán bệnh tim ". Trong quá trình làm bài tập lớn này, chúng em xin gửi lời cảm ơn đến thầy ***Đào Nam Anh***. Thầy đã rất tận tình hướng dẫn chi tiết cho chúng em, những kiến thức thầy cung cấp rất hữu ích. Chúng em rất mong nhận được những góp ý từ các thầy cô.

Chúng em xin chân thành cảm ơn!

CHƯƠNG 1: TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU

1.1. Khái niệm chung:

1.1.1. Khái niệm cơ bản:

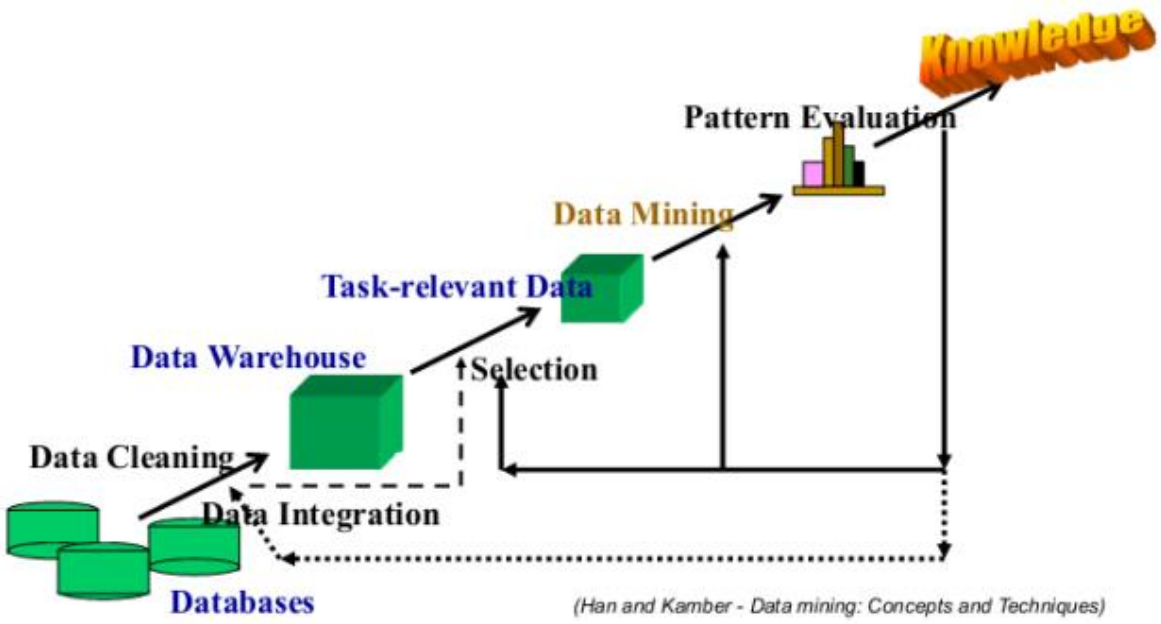
Khai phá dữ liệu (*data mining*) Là quá trình tính toán để tìm ra các mẫu trong các bộ dữ liệu lớn liên quan đến các phương pháp tại giao điểm của máy học, thống kê và các hệ thống cơ sở dữ liệu. Đây là một lĩnh vực liên ngành của khoa học máy tính. Mục tiêu tổng thể của quá trình khai thác dữ liệu là 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.

1.1.2. Các bước trong quá trình khai phá

Quá trình được thực hiện qua 7 bước:

1. Làm sạch dữ liệu (data cleaning & preprocessing)s: Loại bỏ nhiễu và các dữ liệu không cần thiết.
2. Tích hợp dữ liệu: (data integration): quá trình hợp nhất dữ liệu thành những kho dữ liệu (data warehouses & data marts) sau khi đã làm sạch và tiền xử lý (data cleaning & preprocessing).
3. Trích chọn dữ liệu (data selection): trích chọn dữ liệu từ những kho dữ liệu và sau đó chuyển đổi về dạng thích hợp cho quá trình khai thác tri thức. Quá trình này bao gồm cả việc xử lý với dữ liệu nhiễu (noisy data), dữ liệu không đầy đủ (incomplete data), .v.v.
4. Chuyển đổi dữ liệu: Các dữ liệu được chuyển đổi sang các dạng phù hợp cho quá trình xử lý
5. Khai phá dữ liệu(data mining): Là một trong các bước quan trọng nhất, trong đó sử dụng những phương pháp thông minh để chắt lọc ra những mẫu dữ liệu.
6. Ước lượng mẫu (knowledge evaluation): Quá trình đánh giá các kết quả tìm được thông qua các độ đo nào đó.
7. Biểu diễn tri thức (knowledge presentation): Quá trình này sử dụng các kỹ thuật để biểu diễn và thể hiện trực quan cho người dùng.

Quá trình khám phá tri thức theo cách nhìn của giới nghiên cứu về các hệ thống dữ liệu và kho dữ liệu về quá trình khám phá tri thức.



Chuẩn bị dữ liệu (*data preparation*), bao gồm các quá trình làm sạch dữ liệu (*data cleaning*), tích hợp dữ liệu (*data integration*), chọn dữ liệu (*data selection*), biến đổi dữ liệu (*data transformation*). Khai thác dữ liệu (*data mining*): xác định nhiệm vụ khai thác dữ liệu và lựa chọn kỹ thuật khai thác dữ liệu. Kết quả cho ta một nguồn tri thức thô. Đánh giá (*evaluation*): dựa trên một số tiêu chí tiến hành kiểm tra và lọc nguồn tri thức thu được. Triển khai (*deployment*).

Quá trình khai thác tri thức không chỉ là một quá trình tuần tự từ bước đầu tiên đến bước cuối cùng mà là một quá trình lặp và có quay trở lại các bước đã qua.

1.1.3 Ứng dụng của khai phá dữ liệu

Kinh tế - ứng dụng trong kinh doanh, tài chính, tiếp thị bán hàng, bảo hiểm, thương mại, ngân hàng, … Đưa ra các bản báo cáo giàu thông tin; phân tích rủi ro trước khi đưa ra các chiến lược kinh doanh, sản xuất; phân loại khách hàng từ đó phân định thị trường, thị phần; …

Khoa học: Thiên văn học – dự đoán đường đi các thiên thể, hành tinh, …; Công nghệ sinh học – tìm ra các gen mới, cây con giống mới, …

Web: các công cụ tìm kiếm.

1.2. Tiền xử lý dữ liệu

Quá trình tiền xử lý dữ liệu, đầu tiên phải nắm được dạng dữ liệu, thuộc tính, mô tả của dữ liệu thao tác. Sau đó tiếp hành 4 giai đoạn chính: làm sạch, tích hợp, biến đổi, thu giảm dữ liệu.

1.2.1. Dữ liệu

- Một tập dữ liệu (dataset) là một tập các đối tượng (object) và các thuộc tính của chúng.

- Mỗi thuộc tính (attribute) mô tả một đặc điểm của một đối tượng.

1.2.2. Làm sạch dữ liệu (data cleaning)

Đối với dữ liệu thu thập được, cần xác định các vấn đề ảnh hưởng là cho nó không sạch. Bởi vì, dữ liệu không sạch (có chứa lỗi, nhiễu, không đầy đủ, có mâu thuẫn) thì các tri thức khám phá được sẽ bị ảnh hưởng và không đáng tin cậy, sẽ dẫn đến các quyết định không chính xác. Do đó, cần gán các giá trị thuộc tính còn thiếu; sửa chữa các dữ liệu nhiễu/lỗi; xác định hoặc loại bỏ các ngoại lai (outliers); giải quyết các mâu thuẫn dữ liệu.

Trên thực thế dữ liệu thu có thể chứa nhiễu, lỗi, không hoàn chỉnh, có mâu thuẫn.

- Không hoàn chỉnh (incomplete): Thiếu các giá trị thuộc tính hoặc thiếu một số thuộc tính. Ví dụ: salary = .

- Nhiễu/lỗi (noise/error): Chứa đựng những lỗi hoặc các mang các giá trị bất thường. Ví dụ: salary = “-525” , giá trị của thuộc tính không thể là một số âm.

- Mâu thuẫn (inconsistent): Chứa đựng các mâu thuẫn (không thống nhất). Ví dụ: salary = “abc” , không phù hợp với kiểu dữ liệu số của thuộc tính salary.

Giải pháp khi dữ liệu chứa nhiễu/lỗi - Phân khoảng (binning): Sắp xếp dữ liệu và phân chia thành các khoảng (bins) có tần số xuất hiện giá trị như nhau. Sau đó, mỗi khoảng dữ liệu có thể được biểu diễn bằng trung bình, trung vị, hoặc các giới hạn … của các giá trị trong khoảng đó. - Hồi quy (regression): Gắn dữ liệu với một hàm hồi quy.

- Phân cụm (clustering): Phát hiện và loại bỏ các ngoại lai (sau khi đã xác định các cụm).

- Kết hợp giữa máy tính và kiểm tra của con người: Máy tính sẽ tự động phát hiện ra các giá trị nghi ngờ. Các giá trị này sẽ được con người kiểm tra lại.

1.2.3. Tích hợp dữ liệu (data integration)

Tích hợp dữ liệu là quá trình trộn dữ liệu từ các nguồn khác nhau vào một kho dữ liệu có sẵn cho quá trình khai phá dữ liệu. Khi tích hợp cần xác định thực thể từ nhiều nguồn dữ liệu để tránh dư thừa dữ liệu.Ví dụ: Bill Clinton ≡ B.Clinton.

Việc dư thừa dữ liệu là thường xuyên xảy ra, khi tích hợp nhiều nguồn. Bởi cùng một thuộc tính (hay cùng một đối tượng) có thể mang các tên khác nhau trong các nguồn (cơ sở dữ liệu) khác nhau. Hay các dữ liệu suy ra được như một thuộc tính trong một bảng có thể được suy ra từ các thuộc tính trong bảng khác. Hay sự trùng lắp các dữ liệu. Các thuộc tính dư thừa có thể bị phát hiện bằng phân tích tương quan giữa chúng.

Phát hiện và xử lý các mâu thuẫn đối với giá trị dữ liệu: Đối với cùng một thực thể trên thực tế, nhưng các giá trị thuộc tính từ nhiều nguồn khác nhau lại khác nhau. Có thể cách biểu diễn khác nhau, hay mức đánh giá, độ do khác nhau.

Yêu cầu chung đối với quá trình tích hợp là giảm thiểu (tránh được là tốt nhất) các dư thừa và các mâu thuẫn. Giúp cải thiện tốc độ của quá trình khai phá dữ liệu và nâng cao chất lượng của các kết quả tri thức thu được.

1.2.4. Biến đổi dữ liệu (data transformation)

Biến đổi dữ liệu là việc chuyển toàn bộ tập giá trị của một thuộc tính sang một tập các giá trị thay thế, sao cho mỗi giá trị cũ tương ứng với một trong các giá trị mới.

1.2.5. Thu giảm dữ liệu (data reduction)

Một kho dữ liệu lớn có thể chứa lượng dữ liệu lên đến terabytes sẽ làm cho quá trình khai phá dữ liệu chạy rất mất thời gian, do đó nên thu giảm dữ liệu. Việc thu giảm dữ liệu sẽ thu được một biểu diễn thu gọn, mà nó vẫn sinh ra cùng (hoặc xấp xỉ) các kết quả khai phá như tập dữ liệu ban đầu.

1.3. Phương pháp khai phá

1.3.1. Phân loại

Phân loại dữ liệu là dạng phân tích dữ liệu nhằm rút trích các mô hình mô tả các lớp dữ liệu hoặc dự đoán xu hướng dữ liệu. Quá trình gồm hai bước:

- Bước học (giai đoạn huấn luyện): xây dựng bộ phân loại (classifier) bằng việc phân tích/học tập huấn luyện.

- Bước phân loại (classification): phân loại dữ liệu/đối tượng mới nếu độ chính xác của bộ phân loại được đánh giá là có thể chấp nhận được (acceptable).

Các giải thuật phân loại dữ liệu:

- Phân loại dữ liệu với cây quyết định (decision tree).

- Phân loại dữ liệu với mạng Bayesian.

- Phân loại dữ liệu với mạng neural.

- Phân loại dữ liệu với k phần tử gần nhất (k-nearest neighbor).

- Phân loại dữ liệu với suy diễn dữa trên tình huống (case-based reasoning).

1.3.2. Hồi quy

Phân tích hồi quy *(regression analysis)* là kỹ thuật thống kê dùng để ước lượng phương trình phù hợp nhất với các tập hợp kết quả quan sát của biến phụ thuộc và biến độc lập. Nó cho phép đạt được kết quả ước lượng tốt nhất về mối quan hệ chân thực giữa các biến số. Từ phương trình ước lượng được này, người ta có thể dự báo về biến phụ thuộc (chưa biết) dựa vào giá trị cho trước của biến độc lập (đã biết).

1.3.2. Luật kết hợp

Là quá trình khám phá các tập giá trị thuộc tính xuất hiện phổ biến trong các đối tượng dữ liệu. Từ tập phổ biến có thể tạo ra các luật kết hợp giữa các giá trị thuộc tính trong tập các đối tượng.

1.3.3. Phân cụm

Clustering hay phân cụm được coi là phương pháp quan trọng và ứng dụng phổ biến, không những mang lại lợi ích phân tích mà nó còn hỗ trợ những thuật toán khác. Clustering cho phép chúng ta hiểu được rất nhanh chóng khi chưa cần đi sâu vào phân tích, giúp xác định được các “patern” là các mẫu dữ liệu mà ở đó những đơn vị quan sát bên trong là gần giống nhau, khai phá các quy luật, các mối quan hệ tự nhiên tiềm ẩn trong dữ liệu.

Tóm lại, Clustering hay phân cụm là phương pháp phân tích, qua đó tập dữ liệu sẽ phân thành nhiều cụm hoặc nhóm khác nhau, trong mỗi cụm hoặc nhóm các điểm dữ liệu quan sát sẽ giống nhau, và giữa các cụm hoặc nhóm sẽ có những khác biệt.

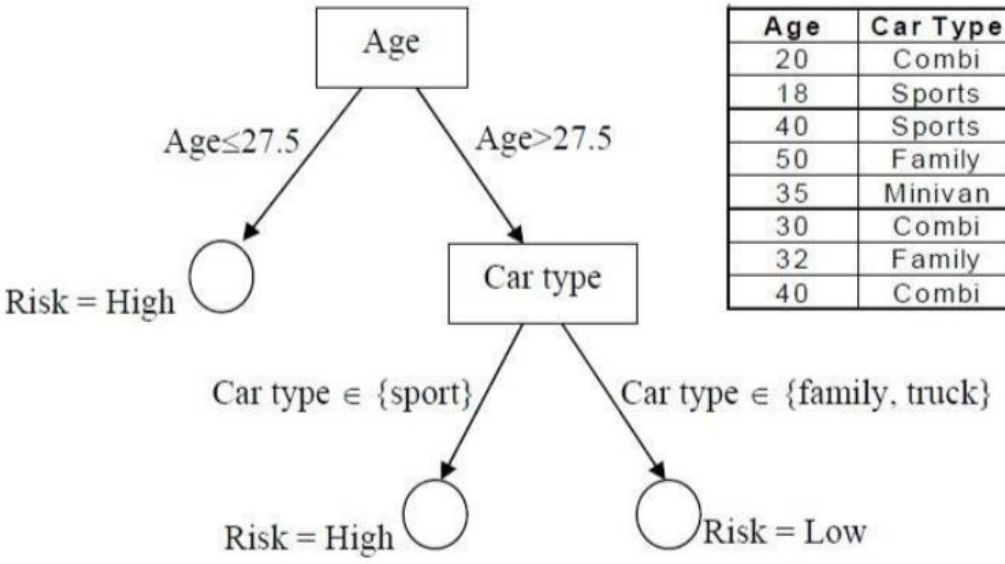
CHƯƠNG 2: GIẢI QUYẾT BÀI TOÁN CÂY QUYẾT ĐỊNH

2.1 Cây quyết định

2.1.1 Giới thiệu

Trong lĩnh vực học máy, cây quyết định là một kiểu mô hình dự báo (*predictive model*), nghĩa là một ánh xạ từ các quan sát về một sự vật/hiện tượng tới các kết luận về giá trị mục tiêu của sự vật/hiện tượng. Mỗi nút trong (*internal node*) tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện giá trị cụ thể cho biến đó. Mỗi nút lá đại diện cho giá trị dự đoán của biến mục tiêu, cho trước các giá trị dự đoán của các biến được biểu diễn bởi đường đi từ nút gốc tới nút lá đó. Kỹ thuật học máy dùng trong cây quyết định được gọi là học bằng cây quyết định, hay chỉ gọi với cái tên ngắn gọn là cây quyết định.

Ví dụ: Cây quyết định phân lớp mức lương



Age	Car Type	Risk
20	Combi	High
18	Sports	High
40	Sports	High
50	Family	Low
35	Minivan	Low
30	Combi	High
32	Family	Low
40	Combi	Low

2.1.2 Các kiểu cây quyết định

Cây quyết định còn có hai loại:

- Cây hồi quy (Regression tree): ước lượng các hàm giá có giá trị là số thực thay vì được sử dụng cho các nhiệm vụ phân loại.
- Cây phân loại (Classification tree): nếu y là một biến phân loại như: giới tính (nam hay nữ), kết quả của một trận đấu (thắng hay thua).

2.1.3 Ưu điểm của cây quyết định

So với các phương pháp khai phá dữ liệu khác, cây quyết định là phương pháp có một số ưu điểm:

- Cây quyết định dễ hiểu. Người ta có thể hiểu mô hình cây quyết định sau khi được giải thích ngắn.
- Việc chuẩn bị dữ liệu cho một cây quyết định là cơ bản hoặc không cần thiết. Các kỹ thuật khác thường đòi hỏi chuẩn hóa dữ liệu, cần tạo các biến phụ (dummy variable) và loại bỏ các giá trị rỗng.
- Cây quyết định có thể xử lý cả dữ liệu có giá trị bằng số và dữ liệu có giá trị là tên thể loại. Các kỹ thuật khác thường chuyên để phân tích các bộ dữ liệu chỉ gồm một loại biến. Chẳng hạn, các luật quan hệ chỉ có thể dùng cho các biến tên, trong khi mạng nơ-ron chỉ có thể dùng cho các biến có giá trị bằng số.
- Cây quyết định là một mô hình hộp trắng. Mạng nơ-ron là một ví dụ về mô hình hộp đen, do lời giải thích cho kết quả quá phức tạp để có thể hiểu được.

Có thể thẩm định một mô hình bằng các kiểm tra thống kê. Điều này làm cho ta có thể tin tưởng vào mô hình.

2.1.4 Phân lớp dữ liệu bằng cây quyết định

Phân lớp dựa trên cây quyết định rất thích hợp cho việc khai phá dữ liệu vì cây quyết định có cấu trúc đơn giản, dễ hiểu và có thể được xây dựng khá nhanh, từ cây quyết định có thể dễ dàng rút ra các luật.

Quy nạp cây quyết định là một quá trình học tập của cây quyết định từ các nhãn lớp của bộ dữ liệu huấn luyện (training tuple). Một cây quyết định là một biểu đồ dòng dữ liệu như cấu trúc cây, mỗi nút trong (không phải lá) tượng trưng cho một thuộc tính kiểm tra, mỗi nhánh đại diện cho kết quả của việc kiểm tra, và mỗi nút lá (hay nút giới hạn) giữ một lớp nhãn. Nút đầu tiên trên cây là nút gốc.

Quá trình phân lớp dữ liệu thông qua 2 bước cơ bản như sau:

- Bước 1: Xây dựng mô hình từ tập huấn luyện
- Bước 2: Sử dụng mô hình, kiểm tra tính đúng đắn của mô hình và dùng nó để phân lớp dữ liệu mới.

2.1.5 Xây dựng cây quyết định.

Có nhiều thuật toán khác nhau để xây dựng cây quyết định như: CLS, ID3, C4.5, SLIQ, SPRINT, EC4.5, C5.0…Nhưng nói chung quá trình xây dựng cây quyết định đều được chia ra làm 3 giai đoạn cơ bản:

a. Xây dựng cây: Thực hiện chia một cách đệ quy tập mẫu dữ liệu huấn luyện cho đến khi các mẫu ở mỗi nút lá thuộc cùng một lớp

b. Cắt tỉa cây: Là việc làm dùng để tối ưu hoá cây. Cắt tỉa cây chính là việc trộn một cây con vào trong một nút lá.

c. Đánh giá cây: Dùng để đánh giá độ chính xác của cây kết quả. Tiêu chí đánh giá là tổng số mẫu được phân lớp chính xác trên tổng số mẫu đưa vào.

2.1.6 Rút ra luật từ cây quyết định

Có thể chuyển đổi qua lại giữa mô hình cây quyết định và mô hình dạng luật (IF …THEN…). Hai mô hình này là tương đương nhau.

2.2. Các thuật toán xây dựng cây quyết định

2.2.1 Thuật toán CLS

Thuật toán này được Hovland và Hint giới thiệu trong Concept learning System (CLS) vào những năm 50 của thế kỷ 20. Sau đó gọi tắt là thuật toán CLS. Thuật toán CLS được thiết kế theo chiến lược chia để trị từ trên xuống. Nó gồm các bước sau:

1. Tạo một nút T, nút này gồm tất cả các mẫu của tập huấn luyện.
2. Nếu tất cả các mẫu trong T có thuộc tính quyết định mang giá trị "yes" (hay thuộc cùng một lớp), thì gán nhãn cho nút T là "yes" và dừng lại. T lúc này là nút lá.
3. Nếu tất cả các mẫu trong T có thuộc tính quyết định mang giá trị "no" (hay thuộc cùng một lớp), thì gán nhãn cho nút T là "no" và dừng lại. T lúc này là nút lá.
4. Trường hợp ngược lại các mẫu của tập huấn luyện thuộc cả hai lớp "yes" và "no" thì:

   + Chọn một thuộc tính X trong tập thuộc tính của tập mẫu dữ liệu, X có các giá trị $v_1, v_2, \ldots v_n$.

   + Chia tập mẫu trong T thành các tập con $T_1, T_2, \ldots, T_n$. chia theo giá trị của X.

   + Tạo n nút con T_i (i=1,2…n) với nút cha là nút T.

   + Tạo các nhánh nối từ nút T đến các nút T_i (i=1,2…n) là các thuộc tính của X.

5. Thực hiện lặp cho các nút con T_i(i =1,2..n) và quay lại bước 2.

Ta nhận thấy trong bước 4 của thuật toán, thuộc tính được chọn để triển khai cây là tuỳ ý. Do vậy cùng với một tập mẫu dữ liệu huấn luyện nếu áp dụng thuật toán CLS với thứ tự chọn thuộc tính triển khai cây khác nhau, sẽ cho ra các cây có hình dạng khác nhau. Việc lựa chọn thuộc tính sẽ ảnh hưởng tới độ rộng, độ sâu, độ phức tạp của cây. Vì vậy một câu hỏi đặt ra là thứ tự thuộc tính nào được chọn để triển khai cây sẽ là tốt nhất. Vấn đề này sẽ được giải quyết trong thuật toán ID3 dưới đây.

2.2.2 Thuật toán ID3

Thuật toán ID3 được phát biểu bởi Quinlan (trường đại học Syney, Australia) và được công bố vào cuối thập niên 70 của thế kỷ 20. Sau đó, thuật toán ID3 được giới thiệu

và trình bày trong mục Induction on decision trees, machine learning năm 1986. ID3 được xem như là một cải tiến của CLS với khả năng lựa chọn thuộc tính tốt nhất để tiếp tục triển khai cây tại mỗi bước. ID3 xây dựng cây quyết định từ trên- xuống (top -down).

Nhiệm vụ của giải thuật ID3 là học cây quyết định từ một tập các ví dụ huấn luyện (training example) hay còn gọi là dữ liệu huấn luyện (training data). Hay nói khác hơn, giải thuật có:

- Đầu vào: Một tập hợp các ví dụ. Mỗi ví dụ bao gồm các thuộc tính mô tả một tình huống, hay một đối tượng nào đó, và một giá trị phân loại của nó.

- Đầu ra: Cây quyết định có khả năng phân loại đúng đắn các ví dụ trong tập dữ liệu huấn luyện, và hy vọng là phân loại đúng cho cả các ví dụ chưa gặp trong tương lai.

ID3 xây dựng cây quyết định (cây QĐ) theo cách từ trên xuống. Lưu ý rằng đối với bất kỳ thuộc tính nào, chúng ta cũng có thể phân vùng tập hợp các ví dụ rèn luyện thành những tập con tách rời, mà ở đó mọi ví dụ trong một phân vùng (partition) có một giá trị chung cho thuộc tính đó. ID3 chọn một thuộc tính để kiểm tra tại nút hiện tại của cây và dùng trắc nghiệm này để phân vùng tập hợp các ví dụ; thuật toán khi đó xây dựng theo cách đệ quy một cây con cho từng phân vùng. Việc này tiếp tục cho đến khi mọi thành viên của phân vùng đều nằm trong cùng một lớp; lớp đó trở thành nút lá của cây.

2.2.3. Các thuộc tính phân loại:

***Entropy*:** dùng để đo tính thuần nhất của một tập dữ liệu. Entropy của một tập S được tính theo công thức

$$\text{Entropy}(S) = - P^{+} \log (P^{+}) - P^{-} \log (P^{-})$$

Trong trường hợp các mẫu dữ liệu có hai thuộc tính phân lớp "yes" (+), "no" (-). Ký hiệu p^{+} là để chỉ tỷ lệ các mẫu có giá trị của thuộc tính quyết định là "yes", và kí hiệu p^{-} là tỷ lệ các mẫu có giá trị của thuộc tính quyết định là "no" trong tập S.

Trường hợp tổng quát, đối với tập con S có n phân lớp thì ta có công thức sau:

$$\text{Entropy}(S) = \sum_{i=1} (- \mathbf{Pi} \ \mathbf{log2} \ (\boldsymbol{Pi}))$$

Trong đó P_i là tỷ lệ các mẫu thuộc lớp i trên tập hợp S các mẫu kiểm tra. Các trường hợp đặc biệt:

- Nếu tất cả các mẫu thành viên trong tập S đều thuộc cùng một lớp thì Entropy(S) =0

- Nếu trong tập S có số mẫu phân bổ đều nhau vào các lớp thì Entropy(S) =1

- Các trường hợp còn lại là 0< Entropy(S)<1

Information Gain (viết tắt là *Gain*): Gain là đại lượng dùng để đo tính hiệu quả của một thuộc tính được lựa chọn cho việc phân lớp. Đại lượng này được tính thông qua hai giá trị Information và Entropy.

Cho tập dữ liệu S gồm có n thuộc tính A_i(i=1,2…n) giá trị Information của thuộc tính A_i ký hiệu là Information(A_i) được xác định bởi công thức .

$$\text{Information}(A_i) = -\sum_{i=1}^{n} log_2(P_i) = Entropy(S)$$

Giá trị Gain của thuộc tính A trong tập S ký hiệu là Gain(S,A) và được tính theo công thức sau:

$$Gain(S, A) \square Entropy(S) \square \square_{V \square Value(A)} \frac{S_v}{S} Entropy(S_v)$$

Trong đó :

- S là tập hợp ban đầu với thuộc tính A. Các giá trị của v tương ứng là các giá trị của thuộc tính A.
- S_v bằng tập hợp con của tập S mà có thuộc tính A mang giá trị v.
- $|S_v|$ là số phần tử của tập S_v.
- |S| là số phần tử của tập S.

Trong quá trình xây dựng cây quyết định theo thuật toán ID3 tại mỗi bước triển khai cây, thuộc tính được chọn để triển khai là thuộc tính có giá trị Gain lớn nhất.

Với việc tính toán giá trị Gain để lựa chọn thuộc tính tối ưu cho việc triển khai cây, thuật toán ID3 được xem là một cải tiến của thuật toán CLS. Tuy nhiên thuật toán ID3 không có khả năng xử lý đối với những dữ liệu có chứa thuộc tính số - thuộc tính liên tục (*numeric attribute*) và khó khăn trong việc xử lý các dữ liệu thiếu (*missing data*)và dữ liệu nhiễu (*noisy data*). Vấn đề này sẽ được giải quyết trong thuật toán C4.5 sau đây.

2.2.3 Thuật toán C4.5

Thuật toán C4.5 được phát triển và công bố bởi Quinlan vào năm 1996. Thuật toán C4.5 là một thuật toán được cải tiến từ thuật toán ID3. Giải thuật C4.5 là một giải thuật học đơn giản nhưng tỏ ra thành công trong nhiều lĩnh vực. C4.5 là một thuật toán hay vì cách biểu diễn tri thức học được của nó, tiếp cận của nó trong việc quản lý tính phức tạp, kinh

nghiệm của nó dùng cho việc chọn lựa các khái niệm ứng viên, và tiềm năng của nó đối với việc xử lý dữ liệu nhiễu.

Thuật toán C4.5 biểu diễn các khái niệm (concept) ở dạng các cây quyết định (decision tree). Biểu diễn này cho phép chúng ta xác định phân loại của một đối tượng bằng cách kiểm tra các giá trị của nó trên một số thuộc tính nào đó.

Các thuộc tính:

a. Entropy:

Tập S là tập dữ liệu huấn luyện, trong đó thuộc tính phân loại có hai giá trị, giả sử là âm (-) và dương (+). Trong đó:

- p+ là phần các ví dụ dương trong tập S.
- p_ là phần các ví dụ âm trong tập S.

Khi đó, *entropy* đo độ hỗn loạn của tập S theo công thức sau:

$$Entropy(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

Một cách tổng quát hơn, nếu các ví dụ của tập S thuộc nhiều hơn hai loại, giả ử là có ***c*** giá trị phân loại thì công thức entropy tổng quát là:

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

Ví dụ: Xét bài toán "có chơi tennis" ứng với thời tiết hay không? Thuật toán C4.5 sẽ học cây quyết định từ tập dữ liệu huấn luyện sau:

Ngày	Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi tennis
D1	Nắng	Nóng	Cao	Nhẹ	Không
D2	Nắng	Nóng	Cao	Mạnh	Không
D3	Âm u	Nóng	Cao	Nhẹ	Có
D4	Mưa	Ấm áp	Cao	Nhẹ	Có
D5	Mưa	Mát	TB	Nhẹ	Có
D6	Mưa	Mát	TB	Mạnh	Không
D7	Âm u	Mát	TB	Mạnh	Có
D8	Nắng	Ấm áp	Cao	Nhẹ	Không
D9	Nắng	Mát	TB	Nhẹ	Có
D10	Mưa	Ấm áp	TB	Nhẹ	Có

D11	Nắng	Ấm áp	TB	Mạnh	Có
D12	Âm u	Ấm áp	Cao	Mạnh	Có
D13	Âm u	Nóng	TB	Nhẹ	Có
D14	Mưa	Ấm áp	Cao	Mạnh	Không

Từ 14 mẫu của bảng dữ liệu "Chơi tennis", ta nhận thấy trong tập thuộc tính đích S có 9 mẫu thuộc lớp dương và 5 mẫu thuộc lớp âm (ký hiệu là [9+, 5-]). Do đó:

$$\text{Entropy}(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0{,}940$$

b. Gain:

Entropy là một số đo đo độ pha trộn của một tập ví dụ, bây giờ chúng ta sẽ định nghĩa một phép đo hiệu suất phân loại các ví dụ của một thuộc tính. Phép đo này gọi là lượng thông tin thu được (hay độ lợi thông tin), nó đơn giản là lượng giảm entropy mong đợi gây ra bởi việc phân chia các ví dụ theo thuộc tính này.

Một cách chính xác hơn, Gain(S, A) của thuộc tính A, trên tập S, được định nghĩa như sau:

Giá trị Value (A) là tập các giá trị có thể cho thuộc tính A, và S_v là tập con của S mà A nhận giá trị v.

Ví dụ: Trở lại với bảng dữ liệu "Chơi tennis", áp dụng công thức trên ta có:

$$\text{Gain}(S, \text{Quang cảnh}) = \text{Entropy}(S) - (5/14)\text{Entropy}(S_{\text{nắng}}) - (4/14)\text{Entropy}(S_{\text{âm u}}) - (5/14)\text{Entropy}(S_{\text{mưa}}) = \mathbf{0{,}246}$$

Một cách tương tự:

Gain(S, Độ ẩm) = 0,151

Gain(S, Nhiệt độ) = 0,029

Gain(S, Gió) = 0,048

Ta thấy, Gain(S, Quang cảnh) lớn nhất nên thuộc tính *Quang cảnh* được chọn làm nút phân tách cây.

Sau khi lập được cấp đầu tiên của cây quyết định ta lại xét nhánh

Nắng Tiếp tục lấy Entropy và Gain cho nhánh *Nắng* ta được hiệu

suất như sau: Gain($S_{Nắng}$, Độ ẩm) = 0,970

Gain($S_{Nắng}$, Nhiệt độ) =

0,570 Gain($S_{Nắng}$, Gió)

= 0,019

Như vậy thuộc tính độ ẩm có hiệu suất phân loại cao nhất trong nhánh *Nắng* nên ta chọn thuộc tính *Độ ẩm* làm nút kế tiếp ….

b. SplitInfo và Gain Ratio

Khái niệm độ lợi thông tin Gain có xu hướng ưu tiên các thuộc tính có số lượng lớn các giá trị. Nếu thuộc tính D có giá trị riêng biệt cho mỗi bảng ghi (thuộc tính *Ngày* ở bảng dữ liệu trên), thì Entropy(S, D) = 0, như vậy Gain(S, D) sẽ đạt giá trị cực đại. Rõ ràng, một phân vùng như vậy thì việc phân loại là vô ích.

Thuật toán xét tất cả các phép thử có thể để phân chia tập dữ liệu đã cho và chọn ra một phép thử có giá trị GainRatio tốt nhất. GainRatio là một đại lượng để đánh giá độ hiệu quả của thuộc tính dùng để thực hiện phép tách trong thuật toán để phát triển cây quyết định. GainRatio được tính dựa trên kết quả tính toán đại lượng Information Gain theo công thức sau:

Với:

$$SplitInfo\ (X) = -\sum_{i=1}^{n} \frac{T_i}{T} \log_2\left(\frac{T_i}{T}\right)$$

$$GainRatio(X) = \frac{Gain(X)}{SplitInfo(X)}$$

Trong đó:

Value(X) là tập các giá trị của thuộc tính X

T_i là tập con của tập T ứng với thuộc tính X = giá trị là v_i.

Ví dụ: Tính Gain Ratio cho các thuộc tính ở bảng dữ liệu 1.7

SplitInformation(S, Quang cảnh) = $-(5/14)\log_2(5/14)-(4/14)\log_2(4/14)$

$-(5/14)\log_2(5/14) = 1,57$

Gain(S, Quang cảnh) = 0,246

GainRatio(S, Quang cảnh) = 0,246/1,57 = 0,157 Tương tự, ta cũng tính được:

GainRatio(S, Nhiệt độ) = 0,0187

GainRatio(S, Độ ẩm) = 0,151

GainRatio(S, Gió) = 0,049

Ta nhận thấy, GainRatio(S, Quang cảnh) có giá trị lớn nhất nên thuộc tính *Quang cảnh* được chọn làm nút phân tách cây.

Đối với các thuộc tính liên tục, chúng ta tiến hành phép thử nhị phân cho mọi giá trị của thuộc tính đó. Để thu thập được giá trị Entropy gain của tất cả các phép thử nhị phân một cách hữu hiệu ta tiến hành xắp xếp các dữ liệu theo giá trị của thuộc tính liên tục đó bằng thuật toán Quicksort, (2.8) được sử dụng làm tiêu chuẩn để lựa chọn thuộc tính khi phân lớp. Thuộc tính được chọn là thuộc tính có giá trị GainRatio tính theo (2.8) đạt giá trị lớn nhất.

Một số cài tiến của thuật toán C4.5:

1. *Làm việc với thuộc tính đa trị*

Tiêu chuẩn Gain có một khuyết điểm là không chấp nhận các thuộc tính đa trị. Vì vậy thuật toán C4.5 đã đưa ra các đại lượng GainRatio và SplitInfo (SplitInformation), chúng được xác định theo các công thức (2.7), (2.8)

Giá trị *SplitInfo* là đại lượng đánh giá thông tin tiềm năng thu thập được khi phân chia tập T thành n tập hợp con.

GainRatio là tiêu chuẩn để đánh giá việc lựa chọn thuộc tính phân loại.

2. *Làm việc với dữ liệu bị thiếu*

Thuật toán vừa xây dựng dựa vào giả thuyết tất cả các mẫu dữ liệu có đủ các thuộc tính. Nhưng trong thực tế, xẩy ra hiện tượng dữ liệu bị thiếu, tức là ở một số mẫu dữ liệu có những thuộc tính không được xác định,hoặc mâu thuẫn, hoặc không bình thường. Ta

xem xét kỹ hơn với trường hợp dữ liệu bị thiếu. Đơn giản nhất là không đưa các mẫu với các giá trị bị thiếu vào, nếu làm như vậy thì có thể dẫn đến tình trạng thiếu các mẫu học. Giả sử T là một tập hợp gồm các mẫu cần được phân loại, X là phép kiểm tra theo thuộc tính L, U là số lượng các giá trị bị thiếu của thuộc tính L. Khi đó ta có:

$$\text{Info(T)} = -\sum_{j=1}^{k} \frac{\text{freq}(C_j,\text{T})}{|\text{T}|-\text{U}} * \log_2\left(\frac{freq(C_j,T)}{|T|-U}\right)$$

$$\text{Info}_x(\text{T}) = -\sum_{j=1}^{n} \frac{|\text{T}|}{|\text{T}|-\text{U}} * \log_2(T_i)$$

Trong trường hợp này, khi tính tần số freq (C_i , T) ta chỉ tính riêng các mẫu với giá trị trên thuộc tính L đã xác định. Khi đó tiêu chuẩn (2.8) được viết lại bằng công thức (2.13) như sau:

$$Gain(X) = \frac{|T|-U}{|T|}(\text{Info(T)}-\text{Info}_x(T))$$

Tương tự thay đổi tiêu chuẩn . Nếu phép kiểm tra có N giá trị đầu vào thì tiêu chuẩn được tính như trong trường hợp chia N tập hợp ban đầu thành (N+1) tập hợp con.

Giả sử phép thử X có các giá trị $O_1,O_2,\ldots O_n$ được lựa chọn theo tiểu chuẩn (2.13), ta cần xử lý như thế nào với các dữ liệu bị thiếu. Giả sử mẫu từ tập hợp T với đầu ra là Oi có liên quan đến tập hợp Ti thì khả năng mẫu đó thuộc tập hợp Ti là 1.

Giả sử mỗi mẫu trong T_i có một chỉ số xác định xác suất thuộc tập hợp T_i. Nếu mẫu có giá trị thuộc tính L thì có trọng số bằng 1. Nếu trong trường hợp ngược lại, thì mẫu này liên quan đến tập con $T_1,T_2,\ldots T_n$ với xác xuất tương ứng là :

$$\sum_{i=1}^{n} \frac{|T_i|}{|T|-U} = 1$$

Ta có thể dễ dàng thấy được rằng tổng các xác xuất này bằng 1.

$$\sum_{i=1} = 1$$

Tóm lại giải pháp này được phát biểu như sau: xác suất xuất hiện của các giá trị bị thiếu tỷ lệ thuận với xác suất xuất hiện của các giá trị không thiếu.

2.2.4 Cắt tỉa cây quyết định

Qua tìm hiểu các thuật toán xây dựng cây quyết định ở trên, ta thấy việc xây dựng cây bằng cách phát triển nhánh cây đầy đủ theo chiều sâu để phân lớp hoàn toàn các mẫu huấn luyện; như thuật toán CLS và thuật toán ID3 đôi khi gặp khó khăn trong các trường hợp dữ liệu bị nhiễu (*Noisy Data*) hoặc dữ liệu bị thiếu (*Missing Data*) không đủ để đại diện cho một quy luật; tức là tạo ra các nút có số mẫu rất nhỏ. Trong trường hợp này, nếu thuật toán vẫn cứ phát triển cây thì ta sẽ dẫn đến một tình huống mà ta gọi là tình trạng "*Over fitting*" trong cây quyết định.

Vấn đề *Over fitting* là một khó khăn trong việc nghiên cứu và ứng dụng cây quyết định. Để giải quyết tình trạng này người ta sử dụng phương pháp cắt tỉa cây quyết định. Có hai phương pháp cắt tỉa cây quyết định.

a, Tiền cắt tỉa (Prepruning):

Chiến thuật tiến cắt tỉa nghĩa là sẽ dừng sớm việc phát triển cây trước khi nó vươn đến điểm mà việc phân lớp các mẫu huấn luyện được hoàn thành. Nghĩa là trong quá trình xây dựng cây, một nút có thể sẽ không được tách thêm bước nữa nếu như kết quả của phép tách đó rơi vào một ngưỡng gần như chắc chắn. Nút đó trở thành nút lá và được gán nhãn là nhãn của lớp phổ biến nhất của tập các mẫu tại nút đó.

b, Hậu cắt tỉa (Postpruning):

Chiến thuật này ngược với chiến thuật tiền cắt tỉa. Nó cho phép phát triển cây đầy đủ sau đó mới cắt tỉa. Nghĩa là xây dựng cây sau đó mới thực hiện cắt bỏ các nhánh không hợp lý. Trong quá trình xây dựng cây theo chiến thuật hậu cắt tỉa thì cho phép tình trạng Over fitting xảy ra. Nếu một nút mà các cây con của nó bị cắt thì nó sẽ trở thành nút lá và nhãn của lá được gán là nhãn của lớp phổ biến nhất của các con trước đó của nó.

Trong thực tế, phương pháp hậu cắt tỉa là một phương pháp khá thành công cho việc tìm ra các giả thuyết chính xác cao. Chiến thuật hậu cắt tỉa được tiến hành thông qua việc tính toán lỗi như sau:

Giả sử ta gọi: E(S) là lỗi tĩnh (Static error hay expected error) của một nút S; BackUpError(S) là lỗi từ các nút con của S (Back Up Error); Error(S) là lỗi của nút S. Các giá trị này được tính như sau:

$$Error(S) = Min(E(S), BackUpError(S))$$

$$E(S) = (N - n + 1) / (N + 2)$$

Trong đó: N là tổng số mẫu ở nút S

n là số mẫu của lớp phổ biến nhất trong S.

Trong trường hợp tổng quát, nếu thuộc tính lớp có K giá trị (K lớp) thì:

$$E(S) = (N \quad n+K-1)/ (N+K)$$

$$BackUpError(S) = \sum_i P_i \ Error(S_i)$$

Trong đó: S_i là nút con của S

P_i là tỷ lệ số mẫu trong S_i trên số mẫu trong S

Như vậy các nút lá sẽ có lỗi $Error(S_i) = E(S_i)$ do nút lá không có nút con dẫn đến không có lỗi BackUpError. Nếu BackUpError(S) >= E(S) thì chiến thuật hậu cắt tỉa cây quyết định sẽ cắt tại nút S (tức là cắt bỏ các cây con của S).

Tóm lại, việc cắt tỉa cây nhằm: tối ưu hoá cây kết quả. Tối ưu về kích cỡ cây và về độ chính xác của việc phân lớp bằng cách cắt bỏ các nhánh không phù hợp (*over fitted branches*). Để thực hiện việc cắt tỉa cây thì có các kỹ thuật cơ bản sau đây:

- Sử dụng tập hợp tách rời của mẫu học để đánh giá tính hữu dụng của việc hậu cắt tỉa những nút trong cây. Sử dụng kỹ thuật cắt tỉa cây này có thuật toán CART, gọi tắt là chi phí phức tạp *(Cost - Complexity prunning).*
- Áp dụng phương pháp thống kê để đánh giá và cắt bỏ các nhánh có độ tin cậy kém hoặc mở rộng tiếp các nhánh có độ chính xác cao. Kỹ thuật cắt tỉa này được gọi là cắt tỉa bi quan và thường được sử dụng để cắt tỉa các cây được xây dựng theo thuật toán ID3 và C4.5.
- Kỹ thuật mô tả độ dài tối thiểu - MDL (*Minimum Description Length)* (với kỹ thuật này không cần kiểm tra các mẫu). Kỹ thuật này không cần thiết phải kiểm tra các mẫu và nó thường được sử dụng trong các thuật toán SLIQ, SPRINT.

2.2.5 Đánh giá và kết luận về các thuật toán xây dựng cây quyết định

Các thuật toán xây dựng cây quyết định vừa được trình bày ở trên đều có những điểm mạnh và điểm yếu riêng của nó.

- Đầu tiên ta xét đến thuật toán CLS đây là một trong những thuật toán ra đời sớm nhất. Nó chỉ áp dụng cho các CSDL có các thuộc tính nhỏ, giá trị các thuộc tính dạng phân loại hay rời rạc. Còn đối với các CSDL lớn và có chứa các thuộc tính mà giá trị của nó là liên tục thì CLS làm việc không hiệu quả.Thuật toán có thể cho các kết quả khác nhau với cùng một tập dữ liệu đầu vào. Bởi vì, thuật toán này chưa có tiêu chí để lựa chọn thuộc tính trong quá trình xây dựng cây. Nhưng đây là thuật toán đơn giản, dễ cài đặt, phù hợp trong việc hình thành ý tưởng và giải quyết những nhiệm vụ đơn giản.
- Thuật toán ID3: trong thuật toán ID3, Quinlan đã khắc phục được hạn chế của thuật toán CLS (ID3 được xem là phiên bản cải tiến của CLS). Thuật toán này làm việc rất có hiệu quả, nó cho kết quả tối ưu hơn thuật toán CLS. Khi áp dụng thuật toán ID3 cho cùng một tập dữ liệu đầu vào và thử nhiều lần thì cho cùng một kết quả. Bởi vì, thuộc tính ứng viên được lựa chọn ở mỗi bước trong quá trình xây dựng cây được lựa chọn trước. Tuy nhiên thuật toán này cũng chưa giải quyết được về vấn đề thuộc tính số, liên tục, số lượng các thuộc tính còn bị hạn chế và giải quyết hạn chế với vấn đề dữ liệu bị thiếu hoặc bị nhiễu.

- Thuật toán C4.5: Để tiếp tục khắc phục những nhược điểm của thuật toán ID3, Quinlan đã đưa ra thuật toán C4.5(C4.5 là sự cải tiến cho thuật toán ID3 và cọi là phiên bản sau của ID3). Trong thuật toán này đã giải quyết được vấn đề làm việc với thuộc tính số(liên tục), thuộc tính có nhiều giá trị, và vấn đề dữ liệu bị thiếu hoặc bị nhiễu. Trong C4.5 thực hiện việc phân ngưỡng với thuộc tính số bằng phép tách nhị phân đưa vào đại lượng GainRatio thay thế cho đại lượng Gain của ID3. Để giải quyết được vấn đề thuộc tính có nhiều giá trị. Ngoài ra C4.5 còn có bước cắt tỉa nhánh không phù hợp. Tuy nhiên yếu điểm của thuật toán này là làm việc không hiệu quả với những CSDL lơn vì chưa giải quyết được vấn đề bộ nhớ.

CHƯƠNG 3: ỨNG DỤNG THUẬT TOÁN

3.1. Sử dụng thuật toán cây quyết để dự đoán bệnh tim

3.1.1. Phát biểu bài toán:

Bài toán sử dụng toàn bộ thông tin từ file dữ liệu bệnh tim sau đó sử dụng thuật toán cây quyết định để dự đoán bệnh timm cho bệnh nhân

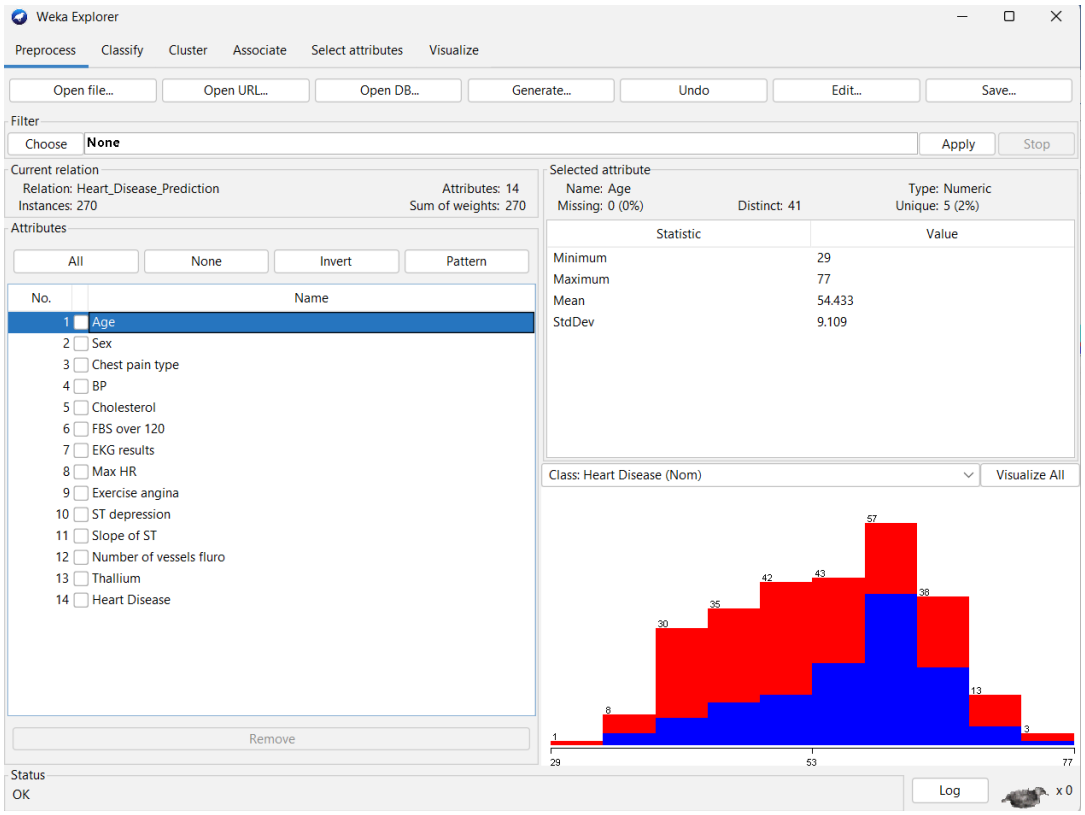
- Input: File dữ liệu thông tin về bệnh tim
- Output: Dự đoán bệnh nhân có bệnh tim hay không.

3.1.3. Cách thức thực hiện

- **Link bộ dữ liệu**: https://www.kaggle.com/datasets/andrewmvd/heart-failure-clinical-data

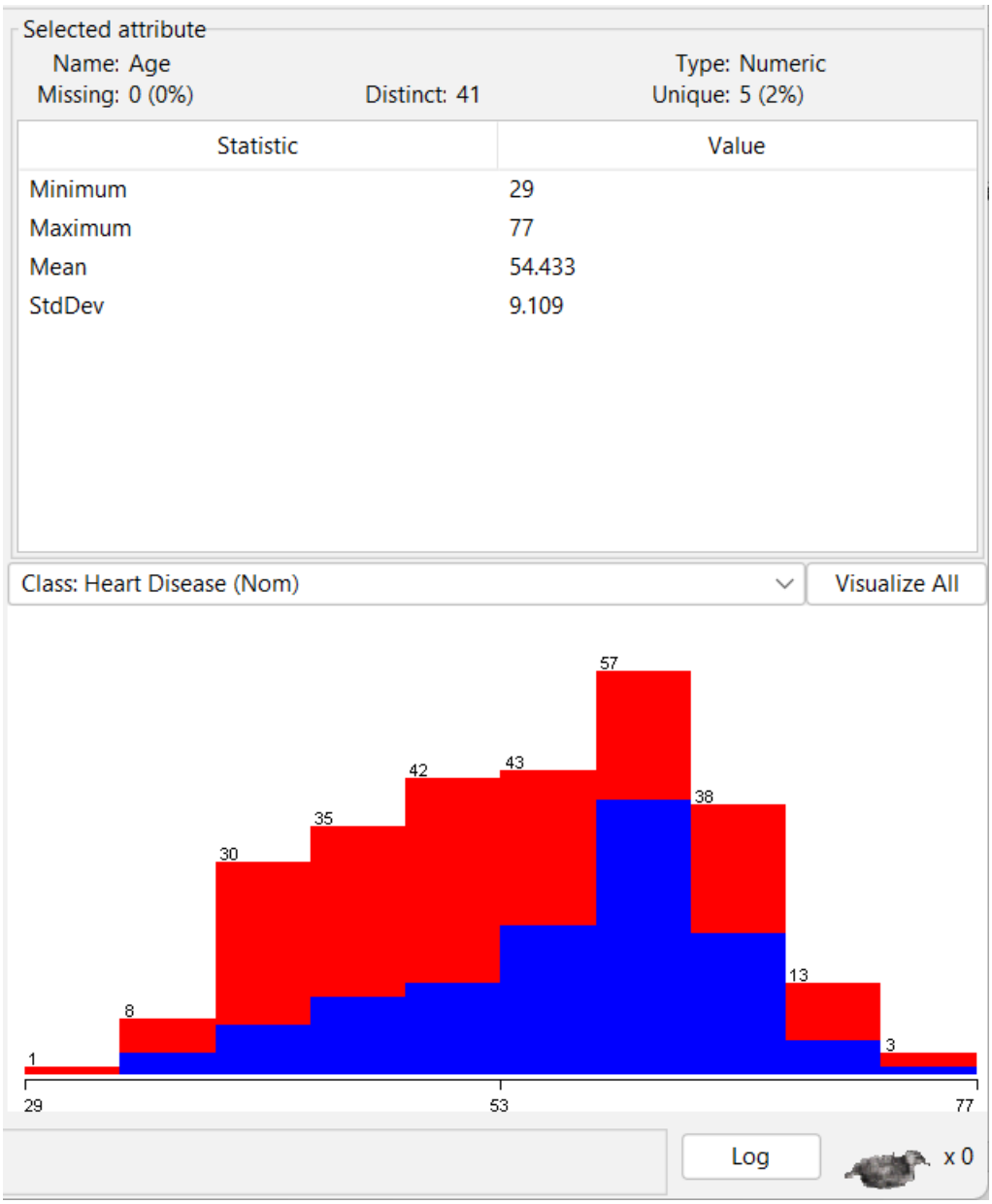
Age	Sex	Chest pain	BP	Cholesterc	FBS over 1	EKG result	Max HR	Exercise a	ST depress	Slope of S	Number of	Thallium	Heart Disease
70	1	4	130	322	0	2	109	0	2.4	2	3	3	Presence
67	0	3	115	564	0	2	160	0	1.6	2	0	7	Absence
57	1	2	124	261	0	0	141	0	0.3	1	0	7	Presence
64	1	4	128	263	0	0	105	1	0.2	2	1	7	Absence
74	0	2	120	269	0	2	121	1	0.2	1	1	3	Absence
65	1	4	120	177	0	0	140	0	0.4	1	0	7	Absence
56	1	3	130	256	1	2	142	1	0.6	2	1	6	Presence
59	1	4	110	239	0	2	142	1	1.2	2	1	7	Presence
60	1	4	140	293	0	2	170	0	1.2	2	2	7	Presence
63	0	4	150	407	0	2	154	0	4	2	3	7	Presence
59	1	4	135	234	0	0	161	0	0.5	2	0	7	Absence
53	1	4	142	226	0	2	111	1	0	1	0	7	Absence
44	1	3	140	235	0	2	180	0	0	1	0	3	Absence
61	1	1	134	234	0	0	145	0	2.6	2	2	3	Presence
57	0	4	128	303	0	2	159	0	0	1	1	3	Absence
71	0	4	112	149	0	0	125	0	1.6	2	0	3	Absence
46	1	4	140	311	0	0	120	1	1.8	2	2	7	Presence
53	1	4	140	203	1	2	155	1	3.1	3	0	7	Presence
64	1	1	110	211	0	2	144	1	1.8	2	0	3	Absence
40	1	1	140	199	0	0	178	1	1.4	1	0	7	Absence
67	1	4	120	229	0	2	129	1	2.6	2	2	7	Presence
48	1	2	130	245	0	2	180	0	0.2	2	0	3	Absence
43	1	4	115	303	0	0	181	0	1.2	2	0	3	Absence
47	1	4	112	204	0	0	143	0	0.1	1	0	3	Absence
54	0	2	132	288	1	2	159	1	0	1	1	3	Absence
48	0	3	130	275	0	0	139	0	0.2	1	0	3	Absence

- **Upload dữ liệu cars.csv lên phần mềm weka:**



Ta thấy tên dữ liệu là Heart_Disease_Prediction với 270 mẫu dữ liệu và 14 thuộc tính

- **Biểu đồ thuộc tính age**



Ta thấy các thông số như

Minimum : 29 (tức là giá trị nhỏ nhất của age là 29)

Maximum : 77 (giá trị lớn nhất age là 77)

Mean : 54.433(giá trị trung bình age là 54.433)

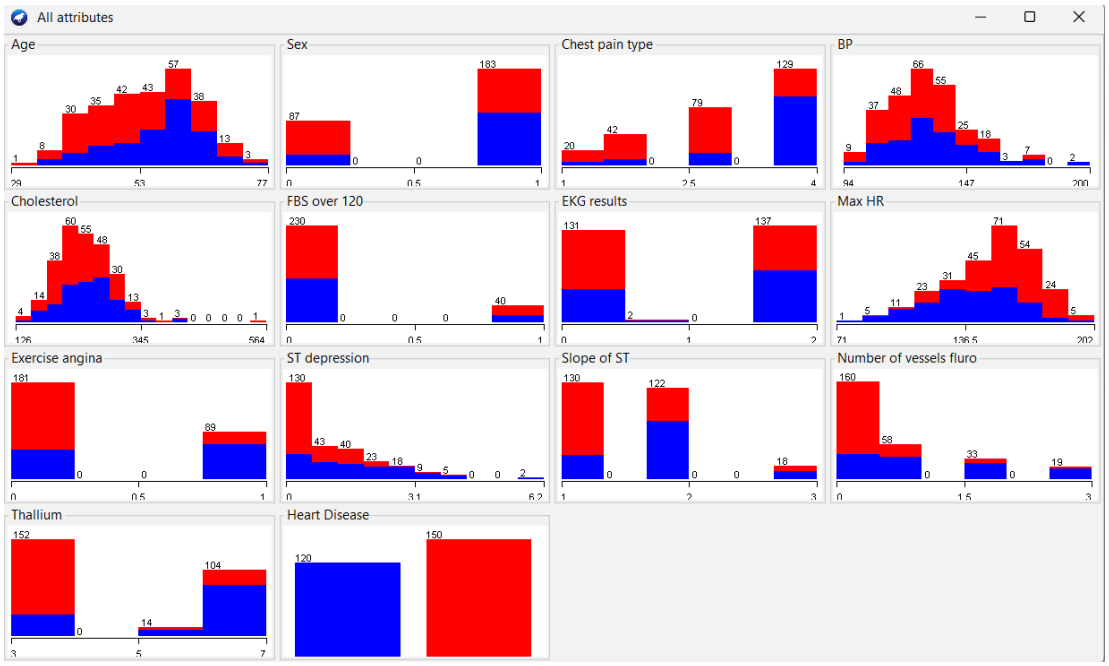
StdDev : 9.109 (độ lệch chuẩn age là 9.109)

- **Biểu đồ thuộc tính brand**

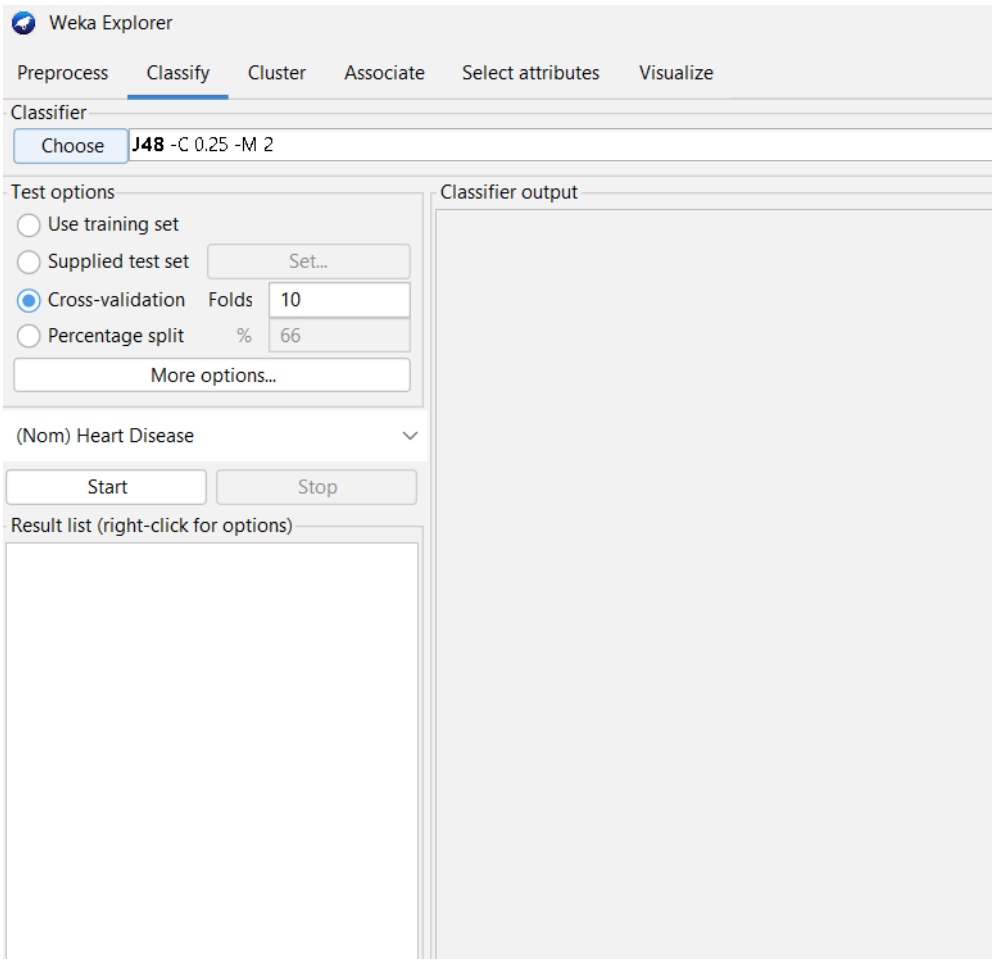
Ta thấy có tới 120 mẫu dữ liệu có heart disease là Presence

150 mẫu dữ liệu có heart disease là Absence

- **Biểu đồ các thuộc tính trong file dữ liệu**



- **Lựa chọn thuật toán giải quyết bài toán**



- **Click vào nút start để thực thi hương trình**

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     Heart_Disease_Prediction
Instances:    270
Attributes:   14
              Age
              Sex
              Chest pain type
              BP
              Cholesterol
              FBS over 120
              EKG results
              Max HR
              Exercise angina
              ST depression
              Slope of ST
              Number of vessels fluro
              Thallium
              Heart Disease
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

Thallium <= 3
|   Chest pain type <= 3: Absence (101.0/10.0)
|   Chest pain type > 3
|   |   Number of vessels fluro <= 0
|   |   |   Age <= 54: Absence (17.0)
|   |   |   Age > 54
```

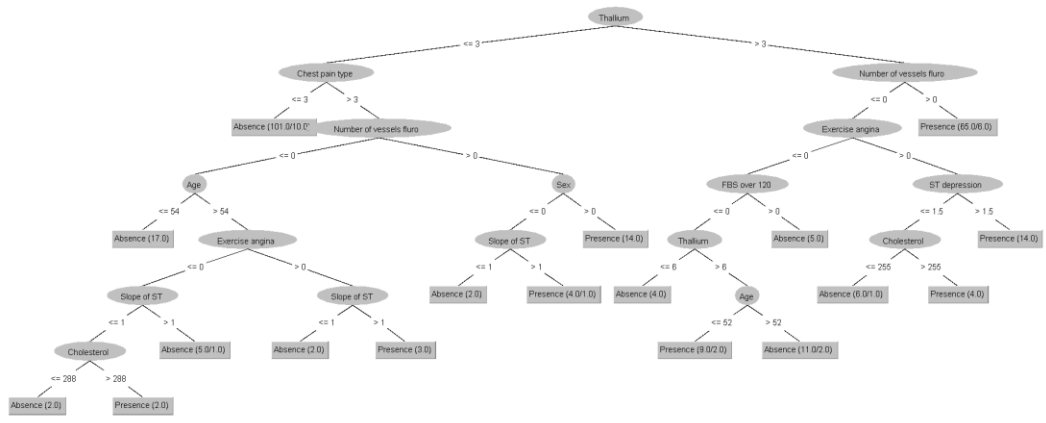
Classifier output

```
Thallium <= 3
|   Chest pain type <= 3: Absence (101.0/10.0)
|   Chest pain type > 3
|   |   Number of vessels fluro <= 0
|   |   |   Age <= 54: Absence (17.0)
|   |   |   Age > 54
|   |   |   |   Exercise angina <= 0
|   |   |   |   |   Slope of ST <= 1
|   |   |   |   |   |   Cholesterol <= 288: Absence (2.0)
|   |   |   |   |   |   Cholesterol > 288: Presence (2.0)
|   |   |   |   |   Slope of ST > 1: Absence (5.0/1.0)
|   |   |   |   Exercise angina > 0
|   |   |   |   |   Slope of ST <= 1: Absence (2.0)
|   |   |   |   |   Slope of ST > 1: Presence (3.0)
|   |   Number of vessels fluro > 0
|   |   |   Sex <= 0
|   |   |   |   Slope of ST <= 1: Absence (2.0)
|   |   |   |   Slope of ST > 1: Presence (4.0/1.0)
|   |   |   Sex > 0: Presence (14.0)
Thallium > 3
|   Number of vessels fluro <= 0
|   |   Exercise angina <= 0
|   |   |   FBS over 120 <= 0
|   |   |   |   Thallium <= 6: Absence (4.0)
|   |   |   |   Thallium > 6
|   |   |   |   |   Age <= 52: Presence (9.0/2.0)
|   |   |   |   |   Age > 52: Absence (11.0/2.0)
|   |   |   FBS over 120 > 0: Absence (5.0)
|   |   Exercise angina > 0
|   |   |   ST depression <= 1.5
|   |   |   |   Cholesterol <= 255: Absence (6.0/1.0)
|   |   |   |   Cholesterol > 255: Presence (4.0)
|   |   |   ST depression > 1.5: Presence (14.0)
```

Classifier output

```
Number of Leaves  :     18

Size of the tree :      35


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         207               76.6667 %
Incorrectly Classified Instances        63               23.3333 %
Kappa statistic                          0.5271
Mean absolute error                      0.274
Root mean squared error                  0.4601
Relative absolute error                 55.4778 %
Root relative squared error             92.5962 %
Total Number of Instances              270

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.733     0.207     0.739      0.733     0.736      0.527    0.744     0.641     Presence
               0.793     0.267     0.788      0.793     0.791      0.527    0.744     0.737     Absence
Weighted Avg.  0.767     0.240     0.766      0.767     0.767      0.527    0.744     0.694

=== Confusion Matrix ===

   a   b   <-- classified as
  88  32 |   a = Presence
  31 119 |   b = Absence
```

Tree View



KẾT LUẬN

Những kết quả đạt được của đề tài:

- Đã tìm hiểu kiến thức về khai phá dữ liệu
- Đã tìm hiểu thuật toán cây quyết định
- Đã áp dụng thuật toán cây quyết định để dự đoán bệnh tim

Hạn chế: Trong quá trình làm bài chúng em còn gặp nhiều vấn đề như tìm hiểu thuật toán còn gặp nhiều khó khăn, kiến thức còn hạn chế, thời gian có hạn với môn học.

Trong quá trình thực hiện chúng em đã cố gắng hết sức tìm hiểu, học hỏi vì khả năng có giới hạn, không tránh khỏi những sai sót, nên chưa giải quyết các vấn đề đặt ra. Chúng em mong nhận được sự thông cảm của các quý thầy cô và các bạn.

Chúng em xin chân thành cảm ơn!

TÀI LIỆU THAM KHẢO

[1] Slide giáo trình Khai Phá Dữ Liệu
[2] Bài giảng giáo trình Khai Phá Dữ Liệu
[3] Giáo trình khai phá dữ liệu