CHƯƠNG 1: Tổng quan về học máy

1.1. Tổng quan

Học máy (Machine Learning) là một ngành khoa học nghiên cứu các thuật toán cho phép máy tính có thể học được các khái niệm (concept).

Phân loại: Có hai loại phương pháp học máy chính

- Phương pháp quy nạp: Máy học/phân biệt các khái niệm dựa trên dữ liệu đã thu thập được trước đó. Phương pháp này cho phép tận dụng được nguồn dữ liệu rất nhiều và sẵn có.
- Phương pháp suy diễn: Máy học/phân biệt các khái niệm dựa vào các luật. Phương pháp này cho phép tận dụng được các kiến thức chuyên ngành để hỗ trợ máy tính.

Hiện nay, các thuật toán đều cố gắng tận dụng được ưu điểm của hai phương pháp này.

Các ngành khoa học liên quan:

- Lý thuyết thống kê: các kết quả trong xác suất thống kê là tiền đề cho rất nhiều phương pháp học máy. Đặc biệt, lý thuyết thống kê cho phép ước lượng sai số của các phương pháp học máy.
- Các phương pháp tính: các thuật toán học máy thường sử dụng các tính toán số thực/số nguyên trên dữ liệu rất lớn. Trong đó, các bài toán như: tối ưu có/không ràng buộc, giải phương trình tuyến tính v.v… được sử dụng rất phổ biến.
- Khoa học máy tính: là cơ sở để thiết kế các thuật toán, đồng thời đánh giá thời gian chạy, bộ nhớ của các thuật toán học máy.

Các nhóm giải thuật học máy:

- Học có giám sát: Máy tính được xem một số mẫu gồm đầu vào (input) và đầu ra (output) tương ứng trước. Sau khi học xong các mẫu này, máy tính quan sát một đầu vào mới và cho ra kết quả.
- Học không giám sát: Máy tính chỉ được xem các mẫu không có đầu ra, sau đó máy tính phải tự tìm cách phân loại các mẫu này và các mẫu mới.
- Học nửa giám sát: Một dạng lai giữa hai nhóm giải thuật trên.
- Học tăng cường: Máy tính đưa ra quyết định hành động (action) và nhận kết quả phản hồi (response/reward) từ môi trường (environment). Sau đó máy tính tìm cách chỉnh sửa cách ra quyết định hành động của mình.

1.2Các ứng dụng của học máy

Ứng dụng: Học máy có ứng dụng rộng khắp trong các ngành khoa học/sản xuất, đặc biệt những ngành cần phân tích khối lượng dữ liệu khổng lồ. Một số ứng dụng thường thấy

- Xử lý ngôn ngữ tự nhiên (Natural Language Processing): xử lý văn bản, giao tiếp người – máy, …
- Nhận dạng (Pattern Recognition): nhận dạng tiếng nói, chữ viết tay, vân tay, thị giác máy (Computer Vision) …
- Tìm kiếm (Search Engine)
- Chẩn đoán trong y tế: phân tích ảnh X-quang, các hệ chuyên gia chẩn đoán tự động.
- Tin sinh học: phân loại chuỗi gene, quá trình hình thành gene/protein
- Vật lý: phân tích ảnh thiên văn, tác động giữa các hạt …
- Phát hiện gian lận tài chính (financial fraud): gian lận thẻ tín dụng
- Phân tích thị trường chứng khoán (stock market analysis)
- Chơi trò chơi: tự động chơi cờ, hành động của các nhân vật ảo

Rôbốt: là tổng hợp của rất nhiều ngành khoa học, trong đó học máy tạo nên hệ thần kinh/bộ não của người máy.

CHƯƠNG 3: Phương pháp học theo cây quyết định

3.1 Phương pháp học theo cây quyết định

3.1.1 Giới thiệu chung

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 một nút trong (internal node) tương ứng với một biến; đường nối giữa nó với nút con của nó thể hiện một 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 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.

Học bằng cây quyết định cũng là một phương pháp thông dụng trong khai phá dữ liệu. Khi đó, cây quyết định mô tả một cấu trúc cây, trong đó, các lá đại diện cho các phân loại còn cành đại diện cho các kết hợp của các thuộc tính dẫn tới phân loại đó. Một cây quyết định có thể được học bằng cách chia tập hợp nguồn thành các tập con dựa theo một kiểm tra giá trị thuộc tính . Quá trình này được lặp lại một cách đệ qui cho mỗi tập con dẫn xuất. Quá trình đệ qui hoàn thành khi không thể tiếp tục thực hiện việc chia tách được nữa, hay khi một phân loại đơn có thể áp dụng cho từng phần tử của tập con dẫn xuất. Một bộ phân loại rừng ngẫu nhiên (random forest) sử dụng một số cây quyết định để có thể cải thiện tỉ lệ phân loại.

Cây quyết định cũng là một phương tiện có tính mô tả dành cho việc tính toán các xác suất có điều kiện.

Cây quyết định có thể được mô tả như là sự kết hợp của các kỹ thuật toán học và tính toán nhằm hỗ trợ việc mô tả, phân loại và tổng quát hóa một tập dữ liệu cho trước.

Dữ liệu được cho dưới dạng các bản ghi có dạng:

$(x, y) = (x_1, x_2, x_3, ..., x_k, y)$

Biến phụ thuộc (dependant variable) y là biến mà chúng ta cần tìm hiểu, phân loại hay tổng quát hóa. $x_1, x_2, x_3...$ là các biến sẽ giúp ta thực hiện công việc đó.

3.1.2Các kiểu cây quyết định

Cây quyết định còn có hai tên khác:

- 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. (ví dụ: ước tính giá một ngôi nhà hoặc khoảng thời gian một bệnh nhân nằm viện)
- 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).

3.1.3 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. Nếu có thể quan sát một tình huống cho trước trong một mô hình, thì có thể dễ dàng giải thích điều kiện đó bằng logic Boolean. 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.
- Cây quyết định có thể xử lý tốt một lượng dữ liệu lớn trong thời gian ngắn. Có thể dùng máy tính cá nhân để phân tích các lượng dữ liệu lớn trong một thời gian đủ ngắn để cho phép các nhà chiến lược đưa ra quyết định dựa trên phân tích của cây quyết định.

3.2 Các thuật toán

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.

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).

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 với việc cho phép xử lý trên tập dữ liệu có các thuộc tính số (numeric atributes) và và làm việc được với tập dữ liệu bị thiếu và bị nhiễu. Nó thực hiện phân lớp tập mẫu dữ liệu theo chiến lược ưu tiên theo chiều sâu (*Depth - First*). 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.

Thuật toán SLIQ[5]

Thuật toán SLIQ (Supervised Learning In Quest) được gọi là thuật toán phân lớp leo thang nhanh. Thuật toán này có thể áp dụng cho cả hai kiểu thuộc liên tục và thuộc tính rời rạc.

Thuật toán này có sử dụng kỹ thuật tiền xử lý phân loại(Pre sorting) trước khi xây dựng cây, do đó giải quyết được vấn đề bộ nhớ cho thuật toán ID3.

Thuật toán SLIQ có sử dụng giải thuật cắt tỉa cây hữu hiệu.

Thuật toán SLIQ có thể phân lớp rất hiệu quả đối với các tập dữ liệu lớn và không phụ thuộc vào số lượng lớp, số lượng thuộc tính và số lượng mẫu trong tập dữ liệu.

3.3 Thuật toán ID3

3.3.1 Giới thiệu về thuật toán ID3

Giải thuật quy nạp cây ID3 (gọi tắt là ID3) là một giải thuật học đơn giản nhưng tỏ ra thành công trong nhiều lĩnh vực. ID3 là một giải thuật hay vì cách biểu diễn tri thức học được của nó, tiếp cận của nó trong việc quản lý tính phức tạp, heuristic 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.

ID3 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.

Như vậy, nhiệm vụ của giải thuật ID3 là học cây quyết định từ một tập các ví dụ rèn luyện (training example) hay còn gọi là dữ liệu rèn 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 rèn luyện, và hy vọng là phân loại đúng cho cả các ví dụ chưa gặp trong tương lai.

Ví dụ, chúng ta hãy xét bài toán phân loại xem ta 'có đi chơi tennis' ứng với thời tiết nào đó không. Giải thuật ID3 sẽ học cây quyết định từ tập hợp các ví dụ sau:

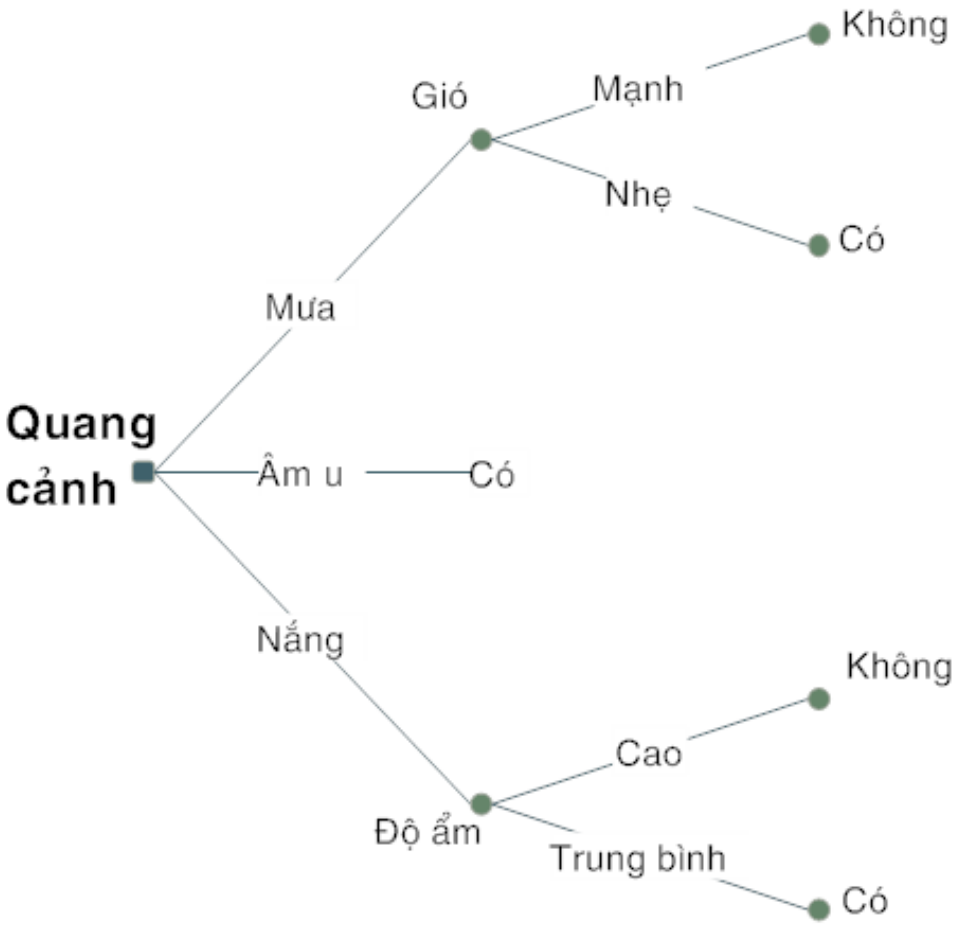
Ngày	Quang cảnh	Nhiệt độ	Độ ẩm	Gió	Chơi Tennis
D1	Nắng	Nóng	Cao	nhẹ	Không
D2	Nắng	Nóng	Cao	Mạnh	Không
D3	Âm u	Nóng	Cao	Nhẹ	Có
D4	Mưa	ấm áp	Cao	nhẹ	Có
D5	Mưa	Mát	TB	nhẹ	Có
D6	Mưa	Mát	TB	Mạnh	Không
D7	Âm u	Mát	TB	Mạnh	Có
D8	Nắng	ấm áp	Cao	nhẹ	Không
D9	Nắng	Mát	TB	nhẹ	Có
D10	Mưa	ấm áp	TB	nhẹ	Có
D11	Nắng	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

Tập dữ liệu này bao gồm 14 ví dụ. Mỗi ví dụ biểu diễn cho tình trạng thời tiết gồm các thuộc tính quang cảnh, nhiệt độ, độ ẩm và gió; và đều có một thuộc tính phân loại 'chơi Tennis' (có, không). 'Không' nghĩa là không đi chơi tennis ứng với thời tiết đó, 'Có' nghĩa là ngược lại. Giá trị phân loại ở đây chỉ có hai loại (có, không), hay còn ta nói phân loại của tập ví dụ của khái

niệm này thành hai lớp (classes). Thuộc tính 'Chơi tennis' còn được gọi là thuộc tính đích (target attribute).

Mỗi thuộc tính đều có một tập các giá trị hữu hạn. Thuộc tính quang cảnh có ba giá trị (âm u, mưa, nắng), nhiệt độ có ba giá trị (nóng, mát, ấm áp), độ ẩm có hai giá trị (cao, TB) và gió có hai giá trị (mạnh, nhẹ). Các giá trị này chính là ký hiệu (symbol) dùng để biểu diễn bài toán.

Từ tập dữ liệu rèn luyện này, giải thuật ID3 sẽ học một cây quyết định có khả năng phân loại đúng đắn các ví dụ trong tập này, đồng thời hy vọng trong tương lai, nó cũng sẽ phân loại đúng các ví dụ không nằm trong tập này. Một cây quyết định ví dụ mà giải thuật ID3 có thể quy nạp được là:



Các nút trong cây quyết định biểu diễn cho một sự kiểm tra trên một thuộc tính nào đó, mỗi giá trị có thể có của thuộc tính đó tương ứng với một nhánh của cây. Các nút lá thể hiện sự phân loại của các ví dụ thuộc nhánh đó, hay chính là giá trị của thuộc tính phân loại.

Sau khi giải thuật đã quy nạp được cây quyết định, thì cây này sẽ được sử dụng để phân loại tất cả các ví dụ hay thể hiện (instance) trong tương lai.

Và cây quyết định sẽ không thay đổi cho đến khi ta cho thực hiện lại giải thuật ID3 trên một tập dữ liệu rèn luyện khác.

Ứng với một tập dữ liệu rèn luyện sẽ có nhiều cây quyết định có thể phân loại đúng tất cả các ví dụ trong tập dữ liệu rèn luyện. Kích cỡ của các cây quyết định khác nhau tùy thuộc vào thứ tự của các kiểm tra trên thuộc tính.

Vậy làm sao để học được cây quyết định có thể phân loại đúng tất cả các ví dụ trong tập rèn luyện? Một cách tiếp cận đơn giản là học thuộc lòng tất cả các ví dụ bằng cách xây dựng một cây mà có một lá cho mỗi ví dụ. Với cách tiếp cận này thì có thể cây quyết định sẽ không phân loại đúng cho các ví dụ chưa gặp trong tương lai. Vì phương pháp này cũng giống như hình thức 'học vẹt', mà cây không hề học được một khái quát nào của khái niệm cần học. Vậy, ta nên học một cây quyết định như thế nào là tốt?

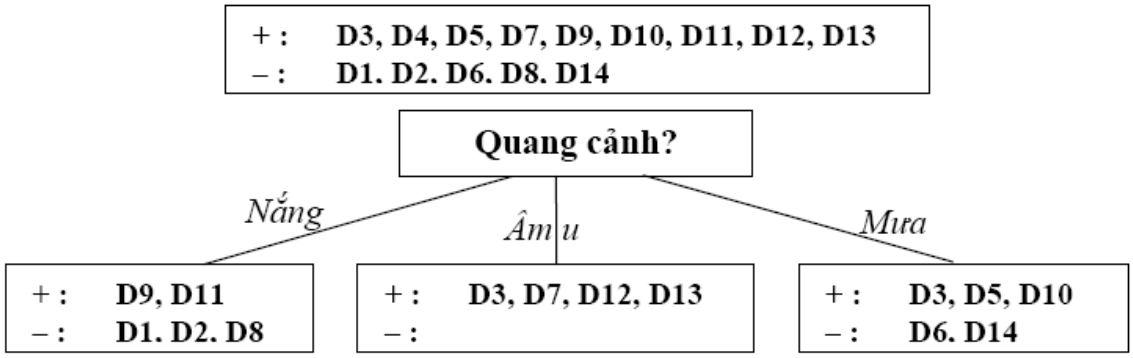
Occam's razor và một số lập luận khác đều cho rằng 'giả thuyết có khả năng nhất là giả thuyết đơn giản nhất thống nhất với tất cả các quan sát', ta nên luôn luôn chấp nhận những câu trả lời đơn giản nhất đáp ứng một cách đúng đắn dữ liệu của chúng ta. Trong trường hợp này là các giải thuật học cố gắng tạo ra cây quyết định nhỏ nhất phân loại một cách đúng đắn tất cả các ví dụ đã cho. Trong phần kế tiếp, chúng ta sẽ đi vào giải thuật ID3, là một giải thuật quy nạp cây quyết định đơn giản thỏa mãn các vấn đề vừa nêu.

3.3.2 Giải thuật ID3 xây dựng cây quyết định từ trên xuống

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.

Vì thứ tự của các trắc nghiệm là rất quan trọng đối với việc xây dựng một cây QĐ đơn giản, ID3 phụ thuộc rất nhiều vào tiêu chuẩn chọn lựa trắc nghiệm để làm gốc của cây. Để đơn giản, phần này chỉ mô tả giải thuật dùng để xây dựng cây QĐ, với việc giả định một hàm chọn trắc nghiệm thích hợp. Phần kế tiếp sẽ trình bày heuristic chọn lựa của ID3.

Ví dụ, hãy xem xét cách xây dựng cây QĐ của ID3 từ ví dụ trước đó

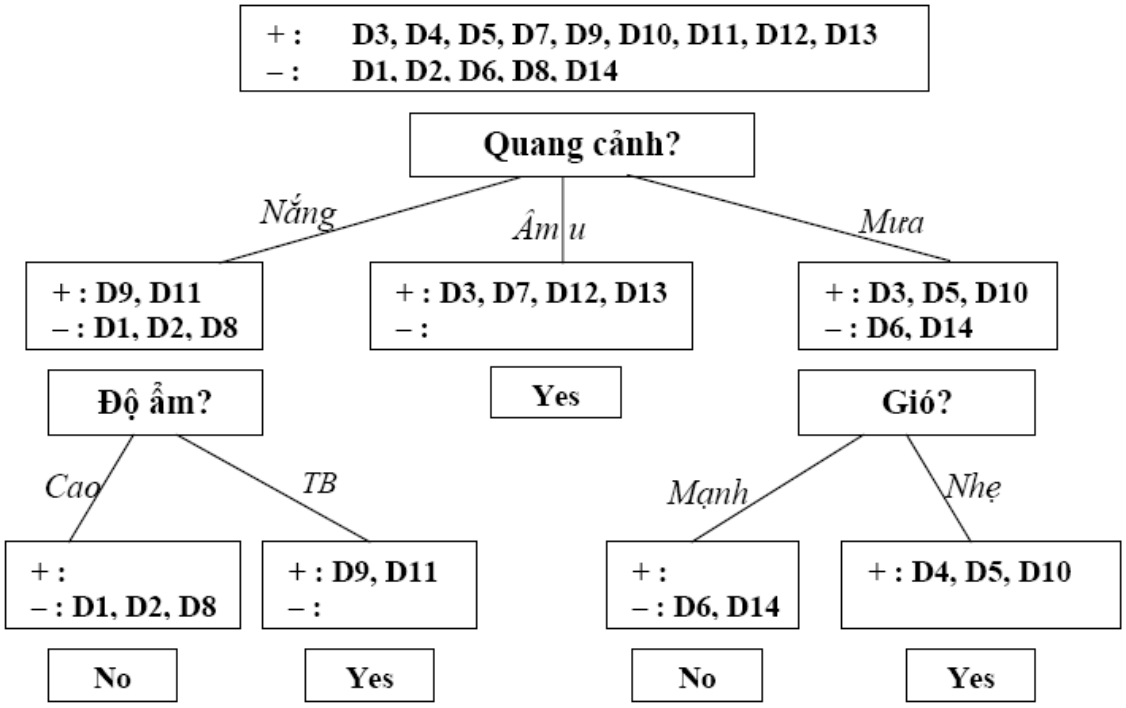


Bắt đầu với bảng đầy đủ gồm 14 ví dụ rèn luyện, ID3 chọn thuộc tính quang cảnh để làm thuộc tính gốc sử dụng hàm chọn lựa thuộc tính mô tả trong phần kế tiếp. Trắc nghiệm này phân chia tập ví dụ như cho thấy trong hình 9.2 với phần tử của mỗi phân vùng được liệt kê bởi số thứ tự của chúng trong bảng.

```
* ID3 xây dựng cây quyết định theo giải thuật sau:
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
```

ID3 áp dụng hàm induce_tree một cách đệ quy cho từng phân vùng. Ví dụ, phân vùng của nhánh "Âm u" có các ví dụ toàn dương, hay thuộc lớp 'Có', nên ID3 tạo một nút lá với nhãn là lớp 'Có'. Còn phân vùng của hai nhánh còn lại vừa có ví dụ âm, vừa có ví dụ dương. Nên tiếp tục chọn thuộc tính "Độ ẩm" để làm trắc nghiệm cho nhánh Nắng, và thuộc tính Gió cho nhánh

Mưa, vì các ví dụ trong các phân vùng con của các nhánh cây này đều thuộc cùng một lớp, nên giải thuật ID3 kết thúc và ta có được cây QĐ như sau



Lưu ý, để phân loại một ví dụ, có khi cây QĐ không cần sử dụng tất cả các thuộc tính đã cho, mặc dù nó vẫn phân loại đúng tất cả các ví dụ.

* Các khả năng có thể có của các phân vùng (partition):

Trong quá trình xây dựng cây QĐ, phân vùng của một nhánh mới có thể có các dạng sau:

- Có các ví dụ thuộc các lớp khác nhau, chẳng hạn như có cả ví dụ âm và dương như phân vùng "Quang cảnh = Nắng" của ví dụ trên => giải thuật phải tiếp tục tách một lần nữa.
- Tất cả các ví dụ đều thuộc cùng một lớp, chẳng hạn như toàn âm hoặc toàn dương như phân vùng "Quang cảnh = Âm u" của ví dụ trên => giải thuật trả về nút lá với nhãn là lớp đó.
- Không còn ví dụ nào => giải thuật trả về mặc nhiên
- Không còn thuộc tính nào => nghĩa là dữ liệu bị nhiễu, khi đó giải thuật phải sử dụng một luật nào đó để xử lý, chẳng hạn như luật đa số (lớp nào có nhiều ví dụ hơn sẽ được dùng để gán nhãn cho nút lá trả về).

Từ các nhận xét này, ta thấy rằng để có một cây QĐ đơn giản, hay một cây có chiều cao là thấp, ta nên chọn một thuộc tính sao cho tạo ra càng nhiều các phân vùng chỉ chứa các ví dụ thuộc cùng một lớp càng tốt. Một phân vùng chỉ

có ví dụ thuộc cùng một lớp, ta nói phân vùng đó có tính thuần nhất. Vậy, để chọn thuộc tính kiểm tra có thể giảm thiểu chiều sâu của cây QĐ, ta cần một phép đo để đo tính thuần nhất của các phân vùng, và chọn thuộc tính kiểm tra tạo ra càng nhiều phân vùng thuần nhất càng tốt. ID3 sử dụng lý thuyết thông tin để thực hiện điều này.

3.3.3 Thuộc tính nào là thuộc tính dùng để phân loại tốt nhất?

Quinlan (1983) là người đầu tiên đề xuất việc sử dụng lý thuyết thông tin để tạo ra các cây quyết định và công trình của ông là cơ sở cho phần trình bày ở đây. Lý thuyết thông tin của Shannon (1948) cung cấp khái niệm entropy để đo tính thuần nhất (hay ngược lại là độ pha trộn) của một tập hợp. Một tập hợp là thuần nhất nếu như tất cả các phần tử của tập hợp đều thuộc cùng một loại, và khi đó ta nói tập hợp này có độ pha trộn là thấp nhất. Trong trường hợp của tập ví dụ, thì tập ví dụ là thuần nhất nếu như tất cả các ví dụ đều có cùng giá trị phân loại.

Khi tập ví dụ là thuần nhất thì có thể nói: ta biết chắc chắn về giá trị phân loại của một ví dụ thuộc tập này, hay ta có lượng thông tin về tập đó là cao nhất. Khi tập ví dụ có độ pha trộn cao nhất, nghĩa là số lượng các ví dụ có cùng giá trị phân loại cho mỗi loại là tương đương nhau, thì khi đó ta không thể đoán chính xác được một ví dụ có thể có giá trị phân loại gì, hay nói khác hơn, lượng thông tin ta có được về tập này là ít nhất. Vậy, điều ta mong muốn ở đây là làm sao chọn thuộc tính để hỏi sao cho có thể chia tập ví dụ ban đầu thành các tập ví dụ thuần nhất càng nhanh càng tốt. Vậy trước hết, ta cần có một phép đo để đo độ thuần nhất của một tập hợp, từ đó mới có thể so sánh tập ví dụ nào thì tốt hơn. Phần kế tiếp sẽ trình bày công thức tính entropy của một tập hợp.

3.3.3.1 Entropy đo tính thuần nhất của tập ví dụ

Khái niệm entropy của một tập S được định nghĩa trong Lý thuyết thông tin là số lượng mong đợi các bít cần thiết để mã hóa thông tin về lớp của một thành viên rút ra một cách ngẫu nhiên từ tập S. Trong trường hợp tối ưu, mã có độ dài ngắn nhất. Theo lý thuyết thông tin, mã có độ dài tối ưu là mã gán –log2p bits cho thông điệp có xác suất là p.

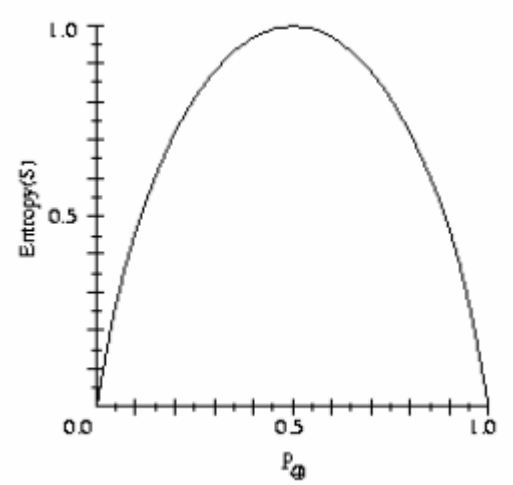
Trong trường hợp S là tập ví dụ, thì thành viên của S là một ví dụ, mỗi ví dụ thuộc một lớp hay có một giá trị phân loại.

- Entropy có giá trị nằm trong khoảng [0..1],
- Entropy(S) = 0 ⇔ tập ví dụ S chỉ toàn ví dụ thuộc cùng một loại, hay S là thuần nhất.

- Entropy(S) = 1 ⇔ tập ví dụ S có các ví dụ thuộc các loại khác nhau với độ pha trộn là cao nhất.

- 0 < Entropy(S) < 1 ⇔ tập ví dụ S có số lượng ví dụ thuộc các loại khác nhau là không bằng nhau.

Để đơn giản ta xét trường hợp các ví dụ của S chỉ thuộc loại âm (-) hoặc dương (+).



Cho trước:

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

• 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 pha trộn 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 sử là có c giá trị phân loại thì công thức entropy tổng quát là:

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

3.3.3.2Lượng thông tin thu được đo mức độ giảm entropy mong đợi

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, 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:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Trong đó Values(A) là tập hợp có thể có các giá trị của thuộc tính A, và Sv là tập con của S chứa các ví dụ có thuộc tính A mang giá trị v.

Trở lại ví dụ ban đầu, nếu không sử dụng Entropy để xác định độ thuần nhất của ví dụ thì có thể xảy ra trường hợp cây quyết định có chiều cao lớn. Ta áp dụng phương thức tính Entropy để xác định chắc chắn thuộc tính nào được chọn trong quá trình tạo cây quyết định

Đầu tiên ta tính độ thuần nhất của tập dữ liệu:

Entropy(S) = - (9/14) Log_2 (9/14) - (5/14) Log_2 (5/14) = 0.940

Từ đó ta tính tiếp Gain cho từng thuộc tính để suy ra thuộc tính nào được chọn làm nút gốc

Gain(S, Quang cảnh) = Entropy(S) – (5/14)Entropy($S_{Nắng}$) – (4/14)Entropy($S_{Âm\ u}$) – (5/14) Entropy($S_{Mưa}$) = 0.246

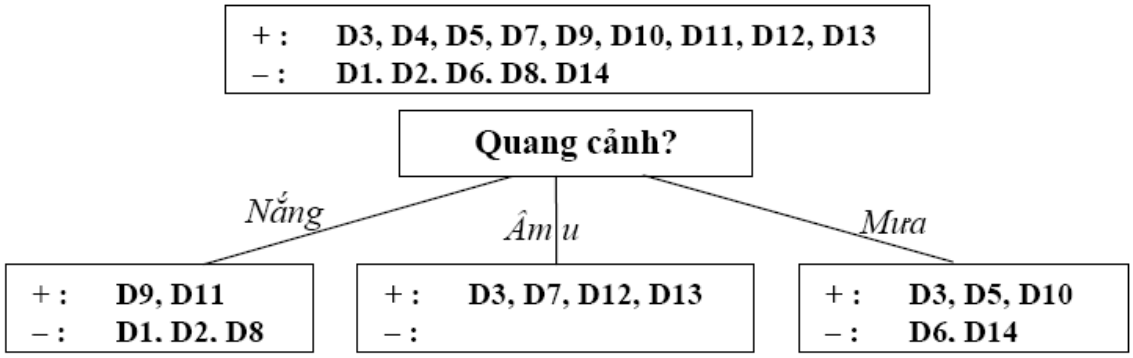
Tương tự cho các Gain khác:

Gain(S, Nhiệt độ) = 0.029

Gain(S, Độ ẩm) = 0.151

Gain(S, Gió) = 0.048

Ta thấy Gain(S, Quang cảnh) là lớn nhất → lấy thuộc tính quang cảnh làm nút gốc



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:

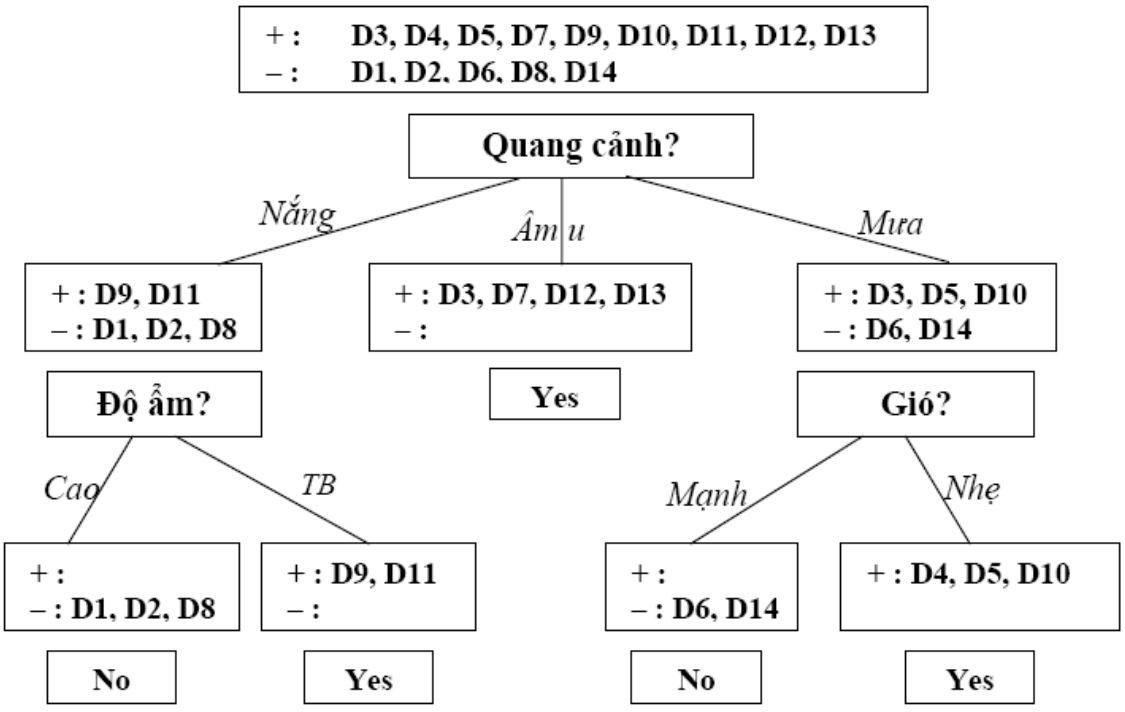
Gain($S_{Nắng}$, Độ ẩm) = 0.970

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 → ta chọn thuộc tính Độ ẩm làm nút kế tiếp ….

Tương tự như vậy đối với nhánh còn lại của cây quyết định ta được cây quyết định hoàn chỉnh như sau



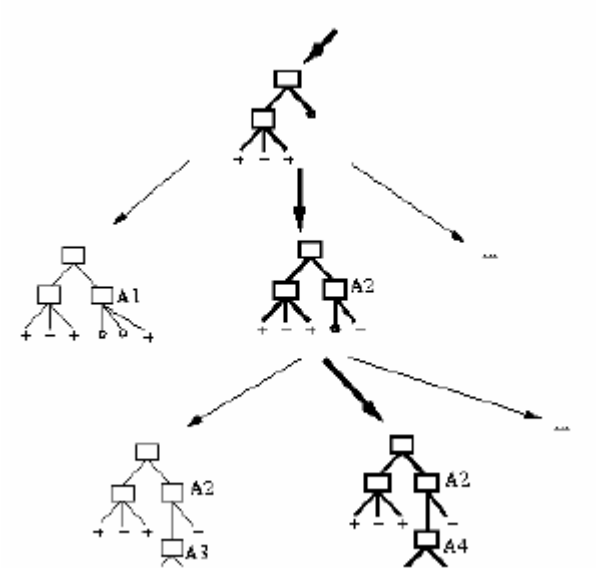
3.3.3.4 Tìm kiếm không gian giả thuyết trong ID3

Cũng như các phương pháp học quy nạp khác, ID3 cũng tìm kiếm trong một không gian các giả thuyết một giả thuyết phù hợp với tập dữ liệu rèn luyện. Không gian giả thuyết mà ID3 tìm kiếm là một tập hợp các cây quyết định có thể có. ID3 thực hiện một phép tìm kiếm từ đơn giản đến phức tạp, theo giải thuật leo-núi (hill climbing), bắt đầu từ cây rỗng, sau đó dần dần xem xét các giả thuyết phức tạp hơn mà có thể phân loại đúng các ví dụ rèn luyện. Hàm đánh giá được dùng để hướng dẫn tìm kiếm leo núi ở đây là phép đo lượng thông tin thu được.

Từ cách nhìn ID3 như là một giải thuật tìm kiếm trong không gian các giả thuyết, ta có một số nhận xét như sau:

- Không gian giả thuyết các cây quyết định của ID3 là một không gian đầy đủ các cây quyết định trên các thuộc tính đã cho trong tập rèn luyện. Điều này có nghĩa là không gian mà ID3 tìm kiếm chắc chắn có chứa cây quyết định cần tìm.

- Trong khi tìm kiếm, ID3 chỉ duy trì một giả thuyết hiện tại. Vì vậy, giải thuật này không có khả năng biểu diễn được tất cả các cây quyết định khác nhau có khả năng phân loại đúng dữ liệu hiện có.



- Giải thuật thuần ID3 không có khả năng quay lui trong khi tìm kiếm. Vì vậy, nó có thể gặp phải những hạn chế giống như giải thuật leo núi, đó là hội tụ về cực tiểu địa phương.

- Vì ID3 sử dụng tất cả các ví dụ ở mỗi bước để đưa ra các quyết định dựa trên thống kê, nên kết quả tìm kiếm của ID3 rất ít bị ảnh hưởng bởi một vài dữ liệu sai (hay dữ liệu nhiễu).

- Trong quá trình tìm kiếm, giải thuật ID3 có xu hướng chọn cây quyết định ngắn hơn là những cây quyết định dài. Đây là tính chất thiên lệch quy nạp của ID3.

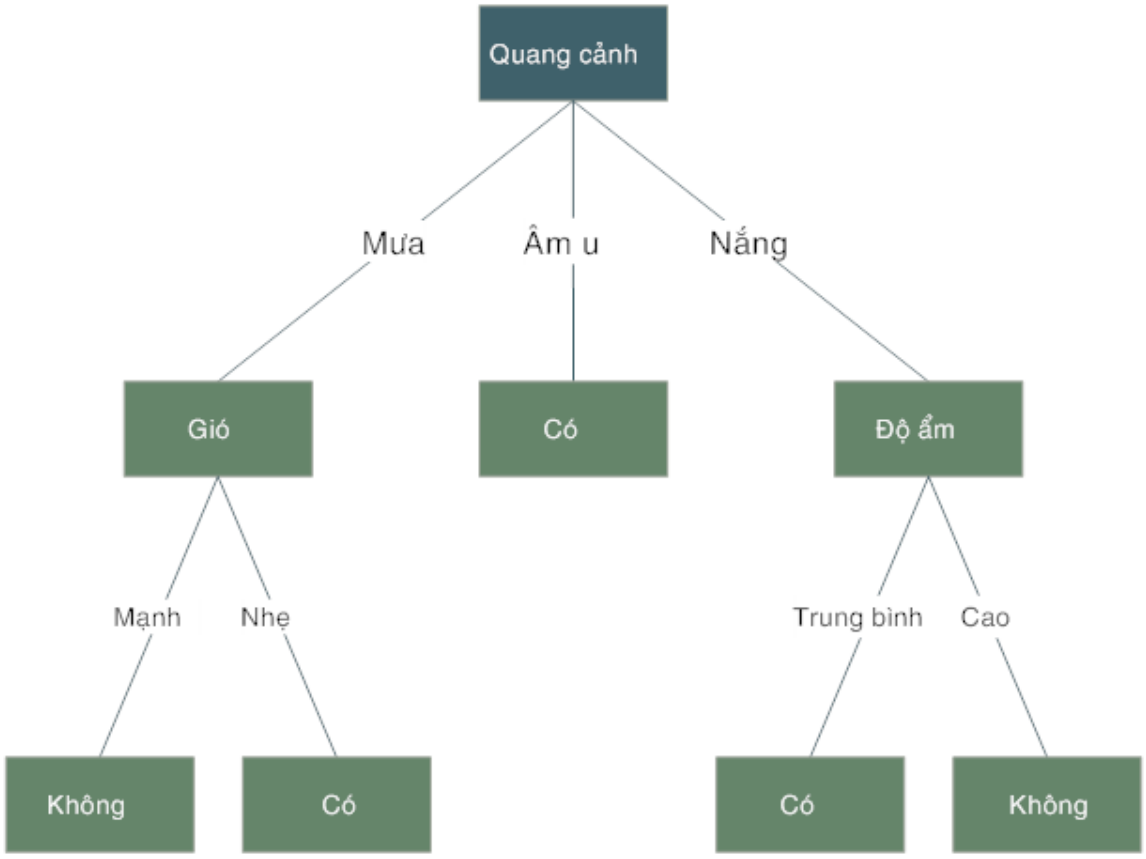
3.3.3.5 Đánh giá hiệu suất của cây quyết định

Một cây quyết định sinh ra bởi ID3 được đánh giá là tốt nếu như cây này có khả năng phân loại đúng được các trường hợp hay ví dụ sẽ gặp trong tương lai, hay cụ thể hơn là có khả năng phân loại đúng các ví dụ không nằm trong tập dữ liệu rèn luyện.

Để đánh giá hiệu suất của một cây quyết định người ta thường sử dụng một tập ví dụ tách rời, tập này khác với tập dữ liệu rèn luyện, để đánh giá khả năng phân loại của cây trên các ví dụ của tập này. Tập dữ liệu này gọi là tập kiểm tra (validation set). Thông thường, tập dữ liệu sẵn có sẽ được chia thành hai tập: tập rèn luyện thường chiếm 2/3 số ví dụ và tập kiểm tra chiếm 1/3.

3.3.3.6 Chuyển cây về các luật

Thông thường, cây quyết định sẽ được chuyển về dạng các luật để thuận tiện cho việc cài đặt và sử dụng. Ví dụ cây quyết định cho tập dữ liệu rèn luyện có thể được chuyển thành một số luật như sau :



If (Quang-cảnh =nắng) ^ (Độ ẩm = Cao) Then Chơi-Tennis = No

If (Quang-cảnh =nắng) ^ (Độ ẩm = TB) Then Chơi-Tennis = Yes

If (Quang-cảnh =Âm u) Then Chơi-Tennis = Yes

…

3.3.3.7 Khi nào nên sử dụng ID3

Giải thuật ID3 là một giải thuật học đơn giản nhưng nó chỉ phù hợp với một lớp các bài toán hay vấn đề có thể biểu diễn bằng ký hiệu. Chính vì vậy, giải thuật này thuộc tiếp cận giải quyết vấn đề dựa trên ký hiệu (symbol – based approach).

Tập dữ liệu rèn luyện ở đây bao gồm các ví dụ được mô tả bằng các cặp "Thuộc tính – giá trị", như trong ví dụ 'Chơi tennis' trình bày trong suốt chương này, đó là 'Gió – mạnh', hay 'Gió – nhẹ',… và mỗi ví dụ đều có một thuộc tính phân loại, ví dụ như 'chơi_tennis', thuộc tính này phải có giá trị rời rạc, như có, không.

Tuy nhiên, khác với một số giải thuật khác cũng thuộc tiếp cận này, ID3 sử dụng các ví dụ rèn luyện ở dạng xác suất nên nó có ưu điểm là ít bị ảnh hưởng bởi một vài dữ liệu nhiễu. Vì vậy, tập dữ liệu rèn luyện ở đây có thể chứa lỗi hoặc có thể thiếu một vài giá trị ở một số thuộc tính nào đó. Một giải pháp thường được áp dụng đối với các dữ liệu bị thiếu là sử dụng luật đa số, chương trình tiền xử lý dữ liệu sẽ điền vào các vị trí còn trống giá trị có tần số xuất hiện cao nhất của thuộc tính đó.

Bên cạnh các vấn đề cơ bản được trình bày trong phần này, ID3 còn được thảo luận nhiều vấn đề liên quan như làm sao để tránh cho cây quyết định không bị ảnh hưởng quá nhiều (overfitting) vào dữ liệu rèn luyện, để nó có thể tổng quát hơn, phân loại đúng được cho các trường hợp chưa gặp. Có nhiều giải pháp đã được đưa ra như cắt tỉa lại cây quyết định sau khi học, hoặc cắt tỉa các luật sau khi chuyển cây về dạng luật. Một vấn đề khác nữa đó là nếu như một vài thuộc tính nào đó có giá trị liên tục thì sao. Giải quyết các vấn đề này dẫn đến việc sinh ra nhiều thế hệ sau của ID3, một giải thuật nổi bật trong số đó là C4.5 (Quinlan 1996). Ngoài ra, một số kỹ thuật được tạo ra để thao tác trên dữ liệu nhằm tạo ra các cây quyết định khác nhau trên cùng tập dữ liệu rèn luyện đã cho như kỹ thuật bagging and boosting.

3.4 Ví dụ minh họa

3.4.1 Phát biểu bài toán

David là quản lý của một câu lạc bộ đánh golf nổi tiếng. Anh ta đang có rắc rối chuyện các thành viên đến hay không đến. Có ngày ai cũng muốn chơi golf nhưng số nhân viên câu lạc bộ lại không đủ phục vụ. Có hôm, không hiểu vì lý do gì mà chẳng ai đến chơi, và câu lạc bộ lại thừa nhân viên.

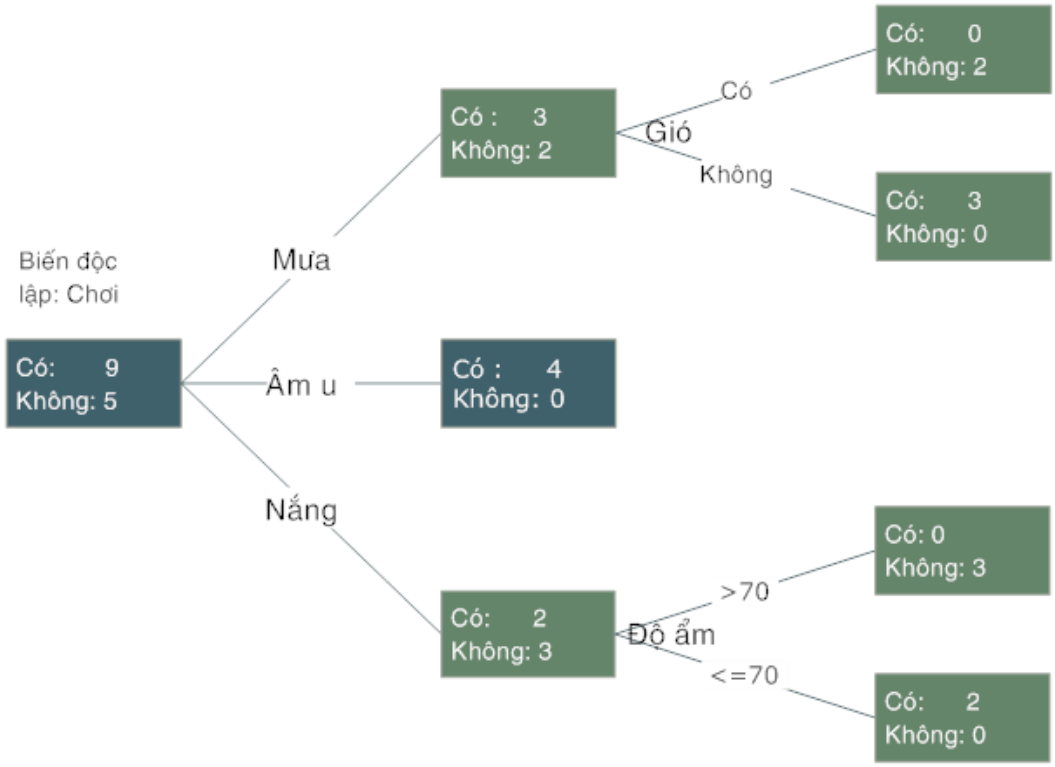
Mục tiêu của David là tối ưu hóa số nhân viên phục vụ mỗi ngày bằng cách dựa theo thông tin dự báo thời tiết để đoán xem khi nào người ta sẽ đến chơi golf. Để thực hiện điều đó, anh cần hiểu được tại sao khách hàng quyết định chơi và tìm hiểu xem có cách giải thích nào cho việc đó hay không.

Vậy là trong hai tuần, anh ta thu thập thông tin về: Trời (outlook) (nắng (sunny), nhiều mây (overcast) hoặc mưa (raining). Nhiệt độ (temperature) bằng độ F. Độ ẩm (humidity). Có gió mạnh (windy) hay không. Và là số người đến chơi golf vào hôm đó. David thu được một bộ dữ liệu gồm 14 dòng và 5 cột.

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

Bảng 1.1. Dữ liệu chơi golf

Sau đó, để giải quyết bài toán của David, người ta đã đưa ra một mô hình cây quyết định.



Hình 1.1. Mô hình cây quyết định chơi golf

Cây quyết định là một mô hình dữ liệu mã hóa phân bố của nhãn lớp (cũng là y) theo các thuộc tính dùng để dự đoán. Đây là một đồ thị có hướng phi chu trình dưới dạng một cây. Nút gốc (nút nằm trên đỉnh) đại diện cho toàn bộ dữ liệu. Thuật toán cây phân loại phát hiện ra rằng cách tốt nhất để giải thích biến phụ thuộc, play (chơi), là sử dụng biến Outlook. Phân loại theo các giá trị của biến Outlook, ta có ba nhóm khác nhau: Nhóm người chơi golf khi trời nắng, nhóm chơi khi trời nhiều mây, và nhóm chơi khi trời mưa.

3.4.2 Minh họa xây dựng cây

Ta có: **S = [9+, 5-]**

Entropy(S) = entropy(9+,5-)

$$= - \mathbf{p_+ log_2 p_+} - \mathbf{p_- log_2 p_-} = \mathbf{-(9/14)log_2(9/14) - (5/14)log_2(5/14)}$$

$$= 0.940$$

➢ Values(Outlook) = **Sunny, Overcast, Rain**

$\mathbf{S_{Sunny}}$ **= [2+, 3-]**

$\mathbf{S_{Overcast}}$ **= [4+, 0-]**

S_{Rain} = [3+, 2-]

$$Gain(S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

= Entropy(S) – (5/14)Entropy(S_{Sunny}) – (4/14)Entropy($S_{Overcast}$) – (5/14)Entropy(S_{Rain})

Trong đó:

Entropy(S) = 0.940

Entropy(S_{Sunny}) = – (2/5)$\log_2$(2/5) – (3/5)$\log_2$(3/5)

= 0.5288 + 0.4422 = 0.971

Entropy($S_{Overcast}$) = – (4/4)$\log_2$(4/4) – (0/4)$\log_2$(0/4) = 0 + 0 = 0

Entropy(S_{Rain}) = – (3/5)$\log_2$(3/5) – (2/5)$\log_2$(2/5)

= 0.4422 + 0.5288 = 0.971

Suy ra:

Gain(S, Outlook) = 0.940 – (5/14)* 0.971 – (4/14)* 0 – (5/14)* 0.971 = 0.246

➢ Values(Temperature) = **Hot, Mild, Cool**

S_{Hot} = [2+, 2-]

S_{Mild} = [4+, 2-]

S_{Cool} = [3+, 1-]

$$Gain(S, Temperature) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

= Entropy(S) – (4/14)Entropy(S_{Hot}) – (6/14)Entropy(S_{Mild}) – (4/14)Entropy(S_{Cool})

Trong đó:

Entropy(S) = 0.940

Entropy(S_{Hot}) = – (2/4)$\log_2$(2/4) – (2/4)$\log_2$(2/4)

= 0.5 + 0.5 = 1

Entropy(S_{Mild}) = – (4/6)$\log_2$(4/6) – (2/6)$\log_2$(2/6)

= 0.3896 + 0.5282 = 0.9178

Entropy(S_{Cool}) = – (3/4)$\log_2$(3/4) – (1/4)$\log_2$(1/4)

= 0.3112781 + 0.5 = 0.81128

Suy ra:

Gain(S, Temperature) = 0.940 – (4/14)*1 – (6/14)*0.9178 – (4/14)*0.81128 = 0.029

➢ Values(Humidity) = **High, Normal**

S_{High} = [3+, 4-]

S_{Normal} = [6+,1-]

$$Gain(S, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

= Entropy(S) – (7/14)Entropy(S_{High}) – (7/14)Entropy(S_{Normal})

Trong đó:

Entropy(S) = 0.940

Entropy(S_{High}) = – (3/7)$\log_2$(3/7) – (4/7)$\log_2$(4/7)

= 0.5238 + 0.4613 = 0.9851

Entropy(S_{Normal}) = – (6/7)$\log_2$(6/7) – (1/7)$\log_2$(1/7)

= 0.1966 + 0.4010 = 0.5976

Suy ra:

Gain(S, Humidity) = 0.940 – (7/14)*0.9851 – (7/14)*0.5976 = 0.151

➢ Values(Wind) = **Weak, Strong**

S_{Weak} = [6+, 2-]

S_{Strong} = [3+, 3-]

$$Gain(S,Wind) = Entropy(S) - \sum_{v\in\{Weak,Strong\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

= Entropy(S) – (8/14)Entropy(S_{Weak}) – (6/14)Entropy(S_{Strong})

Trong đó:

Entropy(S) = 0.940

Entropy(S_{Weak}) = – (6/8)$\log_2$(6/8) – (2/8)$\log_2$(2/8)

= 0.3112 + 0.5 = 0.8112

Entropy(S_{Strong}) = – (3/6)$\log_2$(3/6) – (3/6)$\log_2$(3/6)

= 0.5 + 0.5 = 1

Suy ra:

Gain(S, Wind) = 0.940 – (8/14)*0.811 – (6/14)*1 = 0.048

Ta thu được kết quả:

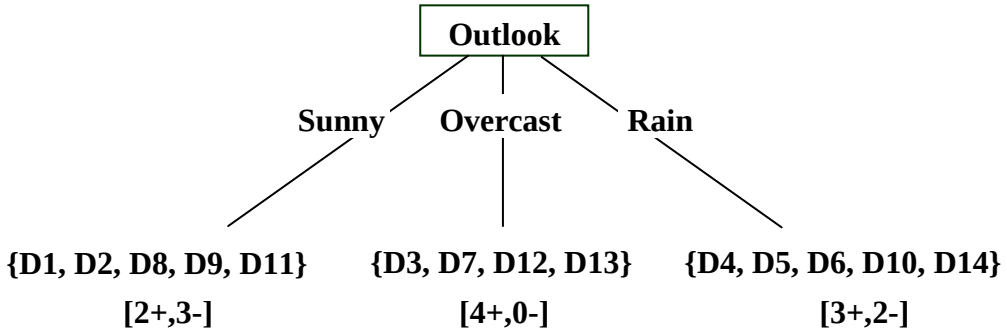
Gain(S, Outlook) = 0.246

Gain(S, Temperature) = 0.029

Gain(S, Humidity) = 0.151

Gain(S, Wind) = 0.048

Ta thấy giá trị Gain(S, Outlook) lớn nhất nên Outlook được chọn làm thuộc tính kiểm tra



❖ Xác định thuộc tính tt1

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
Dl	**Sunny**	**Hot**	**High**	**Weak**	**No**
D2	**Sunny**	**Hot**	**High**	**Strong**	**No**
D8	**Sunny**	**Mild**	**High**	**Weak**	**No**
D9	**Sunny**	**Cool**	**Normal**	**Weak**	**Yes**
Dl1	**Sunny**	**Mild**	**Normal**	**Strong**	**Yes**

Ta có:

S_{Sunny} = {D1, D2, D8, D9, D11} = [2+, 3-]

Entropy(S_{Sunny}) = entropy(2+,3-)

= – $p_+ \log_2 p_+$ – $p_- \log_2 p_-$ = – (2/5)$\log_2$(2/5) – (3/5)$\log_2$(3/5)

= 0.5288 + 0.4421 = 0.971

➢ Values(Temperature) = **Hot, Mild, Cool**

S_{Hot} = [0+, 2-]

S_{Mild} = [1+, 1-]

S_{Cool} = [1+, 0-]

$$Gain(S_{Sunny}, Temperature) = Entropy(S_{Sunny}) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S_{Sunny}|} Entropy(S_v)$$

= Entropy(S_{Sunny}) – (2/5)Entropy(S_{Hot}) – (2/5)Entropy(S_{Mild}) – (1/5)Entropy(S_{Cool})

Trong đó:

Entropy(S_{Sunny}) = 0.971

Entropy(S_{Hot}) = – (0/2)$\log_2$(0/2) – (2/2)$\log_2$(2/2) =

= 0 + 0 = 0

Entropy(S_{Mild}) = – (1/2)$\log_2$(1/2) – (1/2)$\log_2$(1/2)

= 0.5 + 0.5 = 1

Entropy(S_{Cool}) = – (1/1)$\log_2$(1/1) – (0/1)$\log_2$(0/1)

= 0 + 0 = 0

Suy ra:

Gain(S_{Sunny}, Temperature) = 0.971 – (2/5)*0 – (2/5)*1 – (1/5)*0 = 0.571

➢ Values(Humidity) = **High, Normal**

S_{High} = [0+, 3-]

S_{Normal} = [2+,0-]

$$Gain(S_{Sunny}, Humidity) = Entropy(S_{Sunny}) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S_{Sunny}|} Entropy(S_v)$$

= Entropy(S_{Sunny}) – (3/5)Entropy(S_{High}) – (2/5)Entropy(S_{Normal})

Trong đó:

Entropy(S_{Sunny}) = 0.971

Entropy(S_{High}) = – (0/3)$\log_2$(0/3) – (3/3)$\log_2$(3/3)

= 0 + 0 = 0

Entropy(S_{Normal}) = – (2/2)$\log_2$(2/2) – (0/2)$\log_2$(0/2)

= 0 + 0 = 0

Suy ra:

Gain(S_{Sunny}, Humidity) = 0.971 – (3/5)*0 – (2/5)*0 = 0.971

➢ Values(Wind) = **Weak, Strong**

S_{Weak} = [1+, 2-]

S_{Strong} = [1+, 1-]

$$Gain(S_{Sunny}, Wind) = Entropy(S_{Sunny}) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S_{Sunny}|} Entropy(S_v)$$

= Entropy(S_{Sunny}) – (3/5)Entropy(S_{Weak}) – (2/5)Entropy(S_{Strong})

Trong đó:

Entropy(S_{Sunny}) = 0.971

Entropy(S_{Weak}) = – (1/3)$\log_2$(1/3) – (2/3)$\log_2$(2/3)

= 0.5278 + 0.3897 = 0.918

Entropy(S_{Strong}) = – (1/2)$\log_2$(1/2) – (1/2)$\log_2$(1/2)

= 0.5 + 0.5 = 1

Suy ra:

Gain(S_{Sunny}, Wind) = 0.971 – (3/5)*0.918 – (2/5)*1 = 0.020

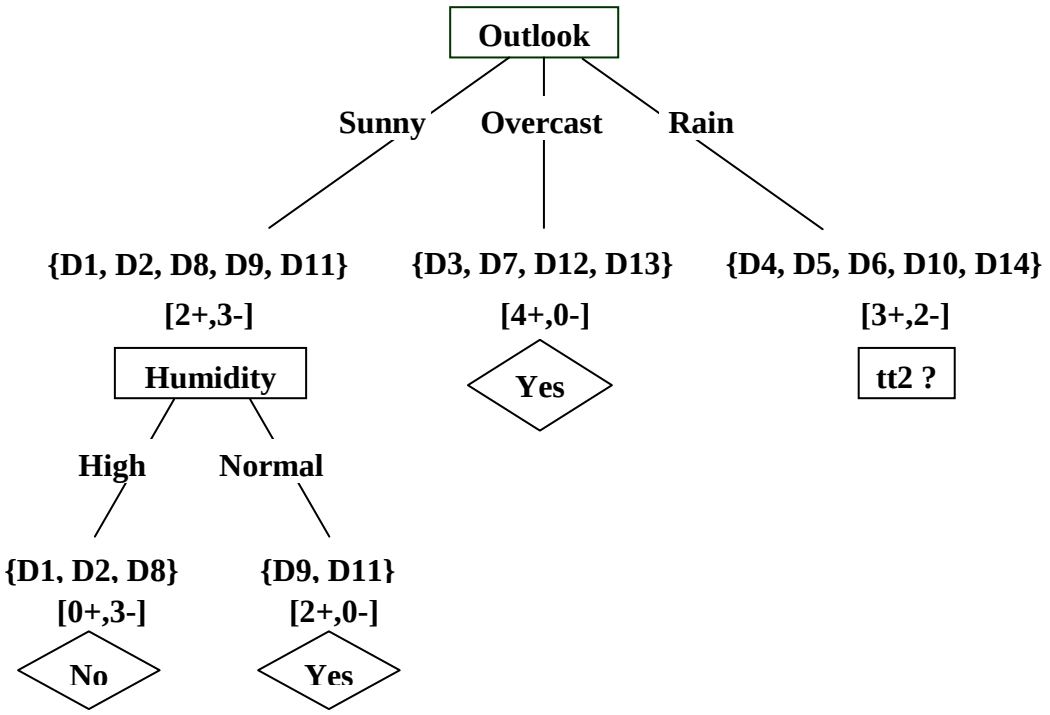
Ta thu được kết quả:

Gain(S_{Sunny}, Temperature) = 0.571

Gain(S_{Sunny}, Humidity) = 0.971

Gain(S_{Sunny}, Wind) = 0.020

Ta thấy giá trị Gain(S_{Sunny}, Humidity) lớn nhất nên Humidity được chọn làm thuộc tính kiểm tra



Hình 2.3. Một phần cây quyết định sinh ra từ bước đầu ID3

❖ Xác định thuộc tính tt2

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D4	**Rain**	**Mild**	**High**	**Weak**	**Yes**
D5	**Rain**	**Cool**	**Normal**	**Weak**	**Yes**
D6	**Rain**	**Cool**	**Normal**	**Strong**	**No**
D10	**Rain**	**Mild**	**Normal**	**Weak**	**Yes**
Dl4	**Rain**	**Mild**	**High**	**Strong**	**No**

Bảng 3.2. Xác định thuộc tính tt2

Ta có:

S_{Rain} = {D4, D5, D6, D10, D14} = [3+, 2-]

Entropy(S_{Rain}) = Entropy(3+,2-)

= – $p_+ \log_2 p_+ – p_- \log_2 p_-$ = – (3/5)$\log_2$(3/5) – (2/5)$\log_2$(2/5)

= 0.4421+ 0.5288 = 0.971

❖ Values(Temperature) = **Mild, Cool**

S_{Mild} = [2+, 1-]

S_{Cool} = [1+, 1-]

$$Gain(S_{Rain}, Temperature) = Entropy(S_{Rain}) - \sum_{v \in \{Mild, Cool\}} \frac{|S_v|}{|S_{Rain}|} Entropy(S_v)$$

= Entropy(S_{Rain}) – (3/5)Entropy(S_{Mild}) – (2/5)Entropy(S_{Cool})

Trong đó:

Entropy(S_{Rain}) = 0.971

Entropy(S_{Mild}) = – (2/3)$\log_2$(2/3) – (1/3)$\log_2$(1/3)

= 0.3897 + 0.5278 = 0.918

Entropy(S_{Cool}) = – (1/2)$\log_2$(1/2) – (1/2)$\log_2$(1/2)

= 0.5 + 0.5 = 1

Suy ra:

Gain(S_{Rain}, Temperature) = 0.971 – (3/5)*0.918 – (2/5)*1 = 0.020

❖ Values(Humidity) = **High, Normal**

S_{High} = [1+, 1-]

S_{Normal} = [2+,1-]

$$Gain(S_{Rain}, Humidity) = Entropy(S_{Rain}) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S_{Rain}|} Entropy(S_v)$$

= Entropy(S_{Rain}) – (2/5)Entropy(S_{High}) – (3/5)Entropy(S_{Normal})

Trong đó:

Entropy(S_{Rain}) = 0.971

Entropy(S_{High}) = $-(1/2)\log_2(1/2) - (1/2)\log_2(1/2)$

$= 0.5 + 0.5 = 1$

Entropy(S_{Normal}) = $-(2/3)\log_2(2/3) - (1/3)\log_2(1/3)$

$= 0.3897 + 0.5278 = 0.918$

Suy ra:

Gain(S_{Rain},Humidity) = 0.971 – (2/5)*1 – (3/5)*0.918 = 0.020

❖ Values(Wind) = **Weak, Strong**

S_{Weak} = [3+, 0-]

S_{Strong} = [0+, 2-]

$$Gain(S_{Rain}, Wind) = Entropy(S_{Rain}) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S_{Rain}|} Entropy(S_v)$$

= Entropy(S_{Rain}) – (3/5)Entropy(S_{Weak}) – (2/5)Entropy(S_{Strong})

Trong đó:

Entropy(S_{Rain}) = 0.971

Entropy(S_{Weak}) = $-(3/3)\log_2(3/3) - (0/3)\log_2(0/3)$

$= 0 + 0 = 0$

Entropy(S_{Strong}) = $-(0/2)\log_2(0/2) - (2/2)\log_2(2/2)$

$= 0 + 0 = 0$

Suy ra:

Gain(S_{Rain}, Wind) = 0.971 – (3/5)*0 – (2/5)*0 = 0.971

Ta thu được kết quả:

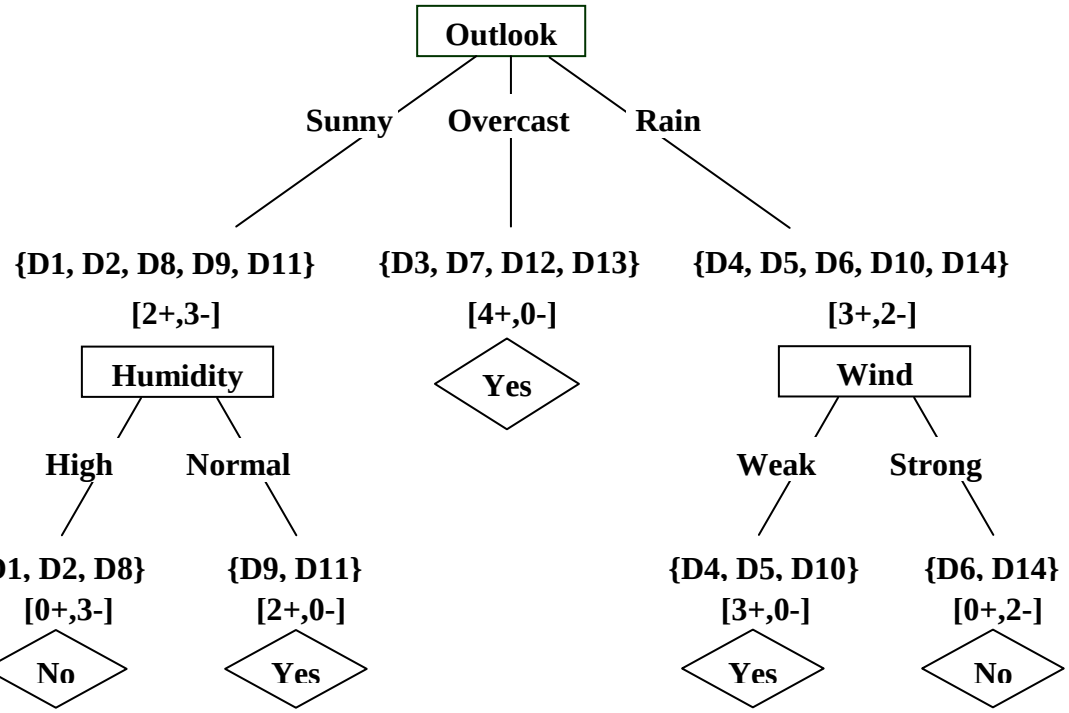
Gain(S_{Rain}, Temperature) = 0.020

Gain(S_{Rain}, Humidity) = 0.020

Gain(S_{Rain}, Wind) = 0.971

Ta thấy giá trị Gain(S_{Rain}, Wind) lớn nhất nên Wind được chọn làm thuộc tính kiểm tra.

Ta được cây quyết định hoàn chỉnh như sau



Hình 3.4. Cây quyết định cần tìm.