TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN MÔN HỌC MÁY HỌC

Lớp Cao Học - Chuyên Ngành KHMT & HTTT

MÔ HÌNH CÂY QUYẾT ĐỊNH

DECISION TREE

GVHD: TS. Trần Thái Sơn

Thành viên nhóm:

1112016 – Hồ Sơn Lâm

1112023 – Bùi Tuấn Phụng

1112042 – Đỗ Minh Tuấn

1112044 – Trần Thị Tuyết Vân

1112046 – Phan Hoàn Vũ

TP.HCM – 4-5-6/2012

MỤC LỤC

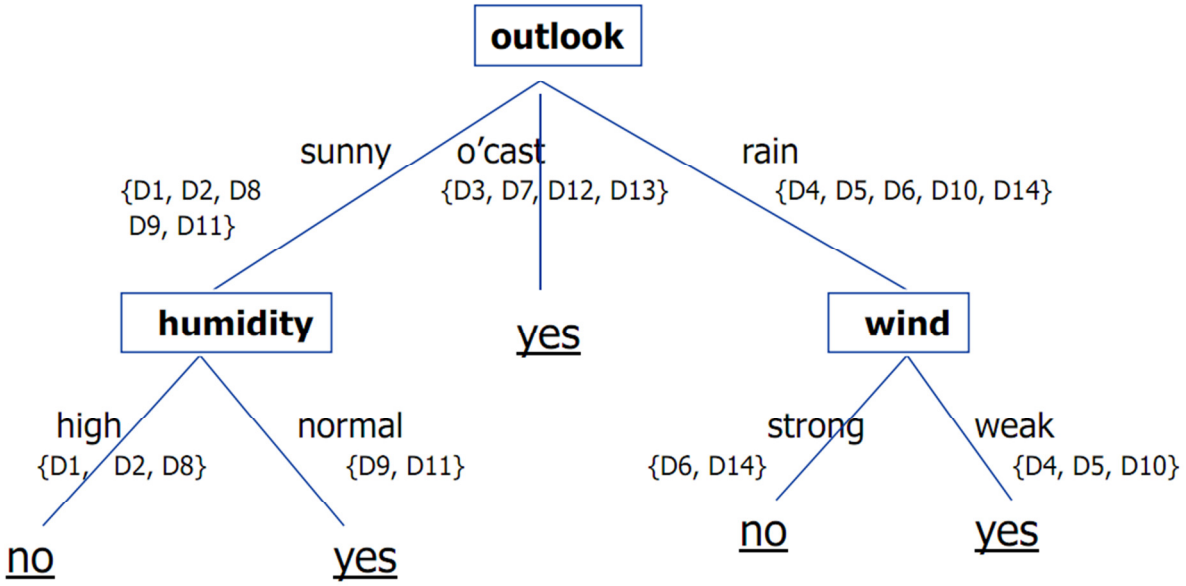
1. Giới thiệu (Đỗ Minh Tuấn)

1.1 Mô hình cây quyết định

Cây quyết định (decision tree) là một trong những hình thức mô tả dữ liệu trực quan nhất, dễ hiểu nhất đối với người dùng. Cấu trúc của một cây quyết định bao gồm các nút và các nhánh. Nút dưới cùng được gọi là nút lá, trong mô hình phân lớp dữ liệu chính là các giá trị của các nhãn lớp (gọi tắt là nhãn). Các nút khác nút lá được gọi là các nút con, đây còn là các thuộc tính của tập dữ liệu, hiển nhiên các thuộc tính này phải khác thuộc tính phân lớp. Mỗi một nhánh của cây xuất phát từ một nút p nào đó ứng với một phép so sánh dựa trên miền giá trị của nút đó. Nút đầu tiên được gọi là nút gốc của cây. Xem xét một ví dụ về một cây quyết định như sau[1]:

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Từ bảng dữ liệu trên, ta xây dựng được cây quyết định như sau:



Cây quyết định của ví dụ trên có thể được giải thích như sau: các nút lá chứa các giá trị của thuộc tính phân lớp (thuộc tính "Play"). Các nút con tương ứng với các thuộc tính khác thuộc tính phân lớp; nút gốc cũng được xem như một nút con đặc biệt, ở đây chính là thuộc tính "Outlook". Các nhánh của cây từ một nút bất kỳ tương đương một phép so sánh có thể là so sánh bằng, so sánh khác, lớn hơn nhỏ hơn… nhưng kết quả các phép so sánh này bắt buộc phải thể hiện một giá trị logic (Đúng hoặc Sai) dựa trên một giá trị nào đó của thuộc tính của nút. Lưu ý cây quyết định trên không có sự tham gia của thuộc tính "thu nhập" trong thành phần cây, các thuộc tính như vậy được gọi chung là các thuộc tính dư thừa bởi vì các thuộc tính này không ảnh hưởng đến quá trình xây dựng mô hình của cây.

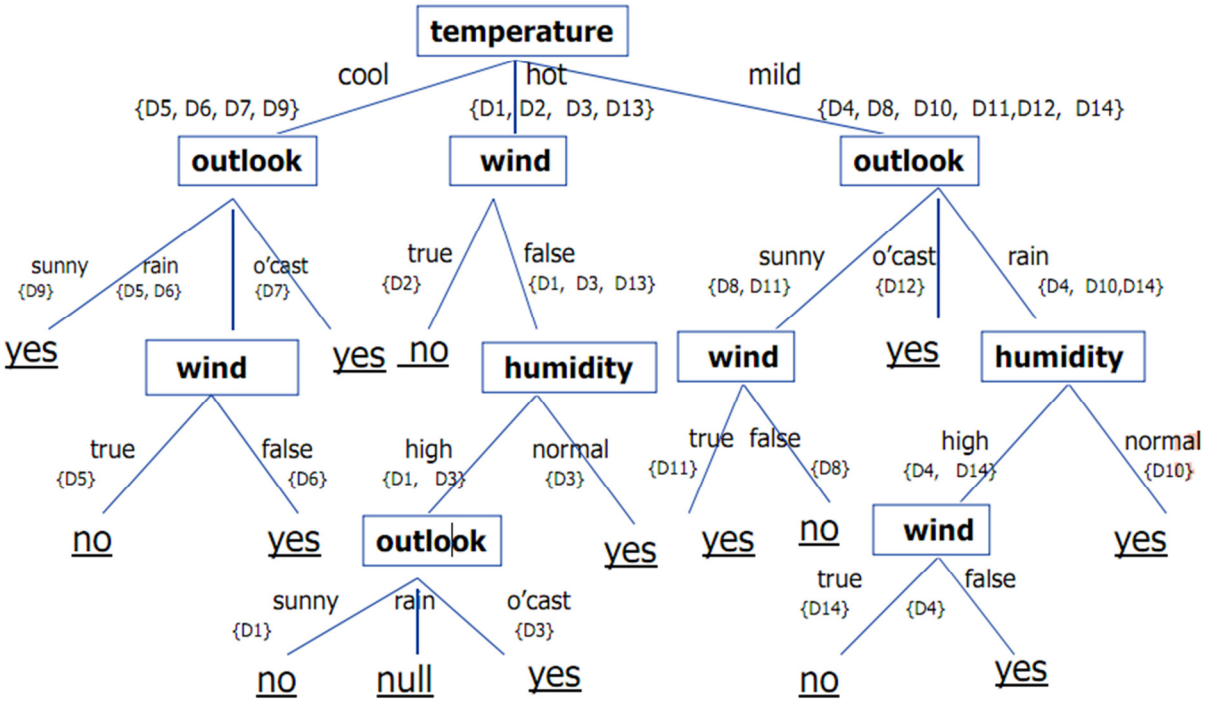
Các thuộc tính tham gia vào quá trình phân lớp thông thường có các giá trị liên tục hay còn gọi là kiểu số (ordered or numeric values) hoặc kiểu rời rạc hay còn gọi là kiểu dữ liệu phân loại (unordered or category values). Ví dụ kiểu dữ liệu lương biểu diễn bằng số thực là kiểu dữ liệu liên tục, kiểu dữ liệu giới tính là kiểu dữ liệu rời rạc (có thể rời rạc hóa thuộc tính giới tính một cách dễ dàng).

1.2 Chiến lược cơ bản để xây dựng cây quyết định

- Bắt đầu từ nút đơn biểu diễn tất cả các mẫu
- Nếu các mẫu thuộc về cùng một lớp, nút trở thành nút lá và được gán nhãn bằng lớp đó
- Ngược lại, dùng độ đo thuộc tính để chọn thuộc tính sẽ phân tách tốt nhất các mẫu vào các lớp

- Một nhánh được tạo cho từng giá trị của thuộc tính được chọn và các mẫu được phân hoạch theo
- Dùng đệ quy cùng một quá trình để tạo cây quyết định
- Tiến trình kết thúc chỉ khi bất kỳ điều kiện nào sau đây là đúng
  - Tất cả các mẫu cho một nút cho trước đều thuộc về cùng một lớp.
  - Không còn thuộc tính nào mà mẫu có thể dựa vào để phân hoạch xa hơn.
  - Không còn mẫu nào cho nhánh test_attribute = a_i

Tuy nhiên, nếu không chọn được thuộc tính phân lớp hợp lý tại mỗi nút, ta sẽ tạo ca cây rất phức tạp, ví dụ như cây dưới đây:



Như vậy, vấn đề đặt ra là phải chọn được thuộc tính phân lớp tốt nhất. Phần tiếp theo sẽ giới thiệu các tiêu chuẩn, dựa vào các tiêu chuẩn này, ta sẽ chọn ra thuộc tính phân lớp tốt nhất tại mỗi nút.

1.3 Thuận lợi và hạn chế của mô hình cây quyết định

❖ ***Một số thuận lợi sau đây của cây quyết định được xem như là một công cụ phân loại mà đã chỉ ra trong tài liệu này:***

1. Cây quyết định tự giải thích và khi được gắn kết lại, chúng có thể dễ dàng tự sinh ra. Nói cách khác, nếu cây quyết định mà có số lượng nút lá vừa phải thì người

không chuyên cũng dễ dàng hiểu được nó. Hơn nữa, cây quyết định cũng có thể chuyển sang tập luật. Vì vậy, cây quyết định được xem như là dễ hiểu.

2. Cây quyết định có thể xử lý cả thuộc tính tên và số đầu vào.
3. Thể hiện của cây quyết định là đủ đa dạng để biểu diễn cho bất kỳ giá trị rời rạc nào.
4. Cây quyết định có khả năng xử lý các bộ dữ liệu mà có thể gây ra lỗi.
5. Cây quyết định có khả năng xử lý các bộ dữ liệu mà có giá trị rỗng.
6. Cây quyết định được xem như là một phương pháp phi tham số. Điều này có nghĩa là cây quyết định không có giả định về sự phân chia bộ nhớ và cấu trúc phân lớp.

❖ ***Bên cạnh đó, cây quyết định cũng có những bất lợi sau đây:***

1. Hầu hết các thuật toán (như ID3 hoặc C4.5) bắt buộc các thuộc tính mục tiêu phải là các giá trị rời rạc.
2. Khi cây quyết định sử dụng phương pháp "chia để trị", chúng có thể thực hiện tốt nếu tồn tại một số thuộc tính liên quan chặt chẽ với nhau, nhưng sẽ khó khăn nếu một số tương tác phức tạp xuất hiện. Một trong những nguyên nhân gây ra điều này là những sự phân lớp mà có mô tả rất mạch lạc về việc phân lớp cũng có thể gặp khó khăn trong việc biểu diễn bằng cây quyết định. Một minh họa đơn giản của hiện tượng này là vấn đề tái tạo cây quyết định (Pagallo và Huassler, 1990). Khi mà hầu hết các cây quyết định phân chia không gian thể hiện thành những khu vực loại trừ lẫn nhau để biểu diễn một khái niệm, trong một số trường hợp, cây nên chứa một vài cây con giống nhau trong thứ tự thể hiện của việc phân lớp. Ví dụ, nếu khái niệm sau mà thể hiện theo hàm nhị phân: $y = (A_1 \cap A_2) \cup (A_3 \cap A_4)$ thì cây quyết định đơn biến tối tiểu mà biểu diễn hàm này đã được biểu diễn trong phần 9.3. Lưu ý là cây có chứa 2 bản sao của cùng một cây con.
3. Các đặc tính liên quan của cây quyết định dẫn đến những khó khăn khác như là độ nhạy với tập huấn luyện, các thuộc tính không phù hợp, nhiễu. (Quinlan, 1993).

2. Các tiêu chuẩn tạo cây quyết định (Đỗ Minh Tuấn)

Việc tìm các tiêu chí để đánh giá tìm điểm chia là rất quan trọng, chúng được xem là một tiêu chuẩn "heuristic" để phân chia dữ liệu. Ý tưởng chính trong việc đưa ra các tiêu chí trên là làm sao cho các tập con được phân chia càng trở nên "trong suốt" (tất cả các bộ thuộc về cùng một nhãn) càng tốt. Cho một tập dữ liệu D, một tập các nhãn Ci (i>=1 và i<=m với m là số nhãn), định nghĩa các khái niệm sau:

C_i,D : là tất cả các bộ dữ liệu có nhãn lớp C_i trong D.

|D| : là tổng số bộ dữ liệu của tập dữ liệu D.

| Ci,D | : là tổng số bộ dữ liệu của tập dữ liệu D có nhãn lớp Ci.[1]

2.1 Tiêu chuẩn tách 1 chiều (Univariate Splitting Criteria):

Nghĩa là tách chỉ dựa trên 1 thuộc tính. Xét theo cấu trúc của mẫu dữ liệu thì có 3 tiêu chuẩn

2.1.1 Impurity-based Criteria:

Khi tất cả các mẫu dữ liệu thuộc về 1 phân lớp, ta gọi đó là Purity. Ngược lại, khi các mẫu dữ liệu tạo ra nhiều phân lớp thì đó gọi là Impurity. Xét theo tiêu chuẩn Impurity-based thì có các độ đo sau:

2.1.1.1 Information Gain

Các thuật toán cũ trước đây thường dùng độ đo Gain để xác định điểm chia. Độ đo này dựa trên cơ sở lý thuyết thông tin của nhà toán học Claude Shannon, độ đo này xác định giá trị của nội dung mà các thông tin sở hữu trong một loạt các thông điệp. Giả sử tại nút hiện hành N, tập D là tập dữ liệu cần được xác định điểm chia, lặp qua tất cả các thuộc tính và chọn lựa thuộc tính nào có độ đo Gain lớn nhất làm ứng cử viên để phân chia. Công thức tính độ đo Gain như sau [1]:

$$\mathrm{Info}(D) = -\sum_{i=1}^{m} P_i \times \log_2(P_i)$$

Với p_i là xác suất của một bộ bất kỳ trên D thuộc về nhãn Ci.

$$P_i = \frac{|Ci, D|}{|D|}$$

Có thể xem công thức Info(D) như một hàm tính giá trị trung bình trên lượng thông tin sử dụng nhằm xác định nhãn của một bộ bất kỳ trong tập D, Info(D) còn được gọi là độ đo sự hỗn loạn (entropy) của D. Giả sử phân chia các bộ trong D trên một thuộc tính A bất kỳ, để không mất tính tổng quát có thể xem như A có các giá trị phân biệt {a_1, a_2, a_3, ….a_v}. Nếu thuộc tính A được sử dụng để chia thành v tập con,

những tập con này sẽ tương ứng với các nhánh con của nút hiện tại, độ đo thông tin có được sau khi phân lớp theo v tập con trên sẽ được tính như sau [1]:

$$\text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

Với |Dj| là tổng số bộ dữ liệu được phân chia vào tập con thứ j.

Độ đo Gain được xác định là sự khác biệt giữa thông tin gốc (thông tin khi chưa phân lớp) và thông tin mới (thông tin sau khi đã phân lớp) và được tính theo công thức bên dưới như sau [1] :

$$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$$

Nói một cách khác, độ đo Gain cho biết được lượng thông tin thu được khi phân lớp, thuộc tính nào có độ đo Gain lớn nhất sẽ được chọn làm ứng cử viên để phân chia. Việc chọn thuộc tính theo tiêu chí độ đo Gain lớn nhất tương đương với việc muốn tìm được một phân hoạch sao cho việc phân lớp là tốt nhất hay nói cách khác lượng thông tin cần thiết để hoàn thành việc phân lớp (thể hiện qua giá trị $\text{Info}_A(D)$) là nhỏ nhất [1].

RID	Age	Income	Student	Credit_rating	Class (buys computer)
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_age	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_age	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_age	medium	no	excellent	yes
13	middle_age	high	yes	fair	yes
14	senior	medium	no	excellent	no

Giải thích cơ sở dữ liệu ở bảng dữ liệu trên: để tiện lợi ta xem tất cả các thuộc tính đều có kiểu dữ liệu rời rạc. Thuộc tính nhãn lớp tức thuộc tính "buys_computer" chỉ có hai giá trị là C1="yes" và C2="no", như vậy có chín bộ dữ liệu có nhãn lớp là giá trị C1 và năm bộ giá trị C2. Để tìm điểm chia tốt nhất, phải tính toán chỉ số Gain của tất cả các thuộc tính trên. Đầu tiên sẽ tính cho toàn bộ tập huấn luyện D [1]:

$$\text{Info}(D) = \frac{9}{14}\log_2(\frac{9}{14}) + \frac{5}{14}\log_2(\frac{5}{14}) \approx 0.94$$

Kế tiếp tính cho từng thuộc tính, bắt đầu với thuộc tính "Age". Thuộc tính này có ba giá trị là "youth", "middle_aged" và "senior". Nhìn vào bảng dữ liệu, với giá trị "youth" có hai bộ có giá trị thuộc tính nhãn là "yes" và ba bộ giá trị thuộc tính nhãn là "no". Tương tự giá trị "middle_aged" có bốn bộ có nhãn lớp là "yes" và không có bộ nào có nhãn lớp là "no"; với giá trị "senior" có ba bộ nhãn lớp "yes" và hai bộ có nhãn lớp "no". Theo công thức trên, độ đo của thuộc tính A xét trên tập huấn luyện D là [1]:

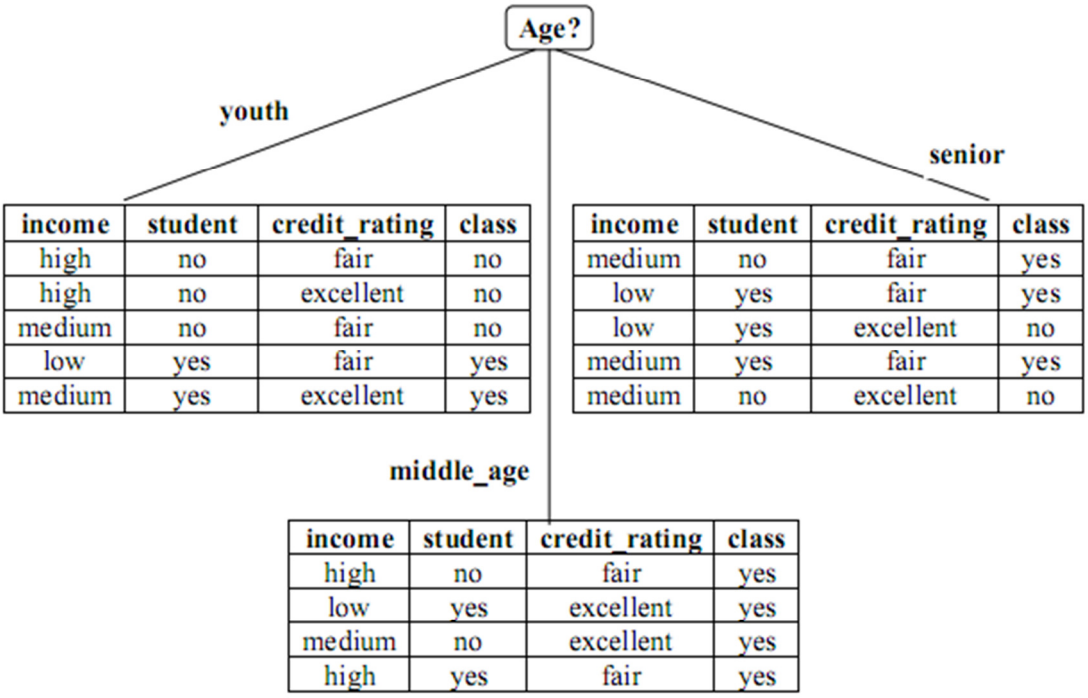
$$\text{Info}_A(D) =$$

$$\frac{5}{14}\times(\frac{2}{5}\log_2\frac{2}{5}+\frac{3}{5}\log_2\frac{3}{5})+\frac{4}{14}\times(\frac{4}{4}\log_2\frac{4}{4}+\frac{0}{4}\log_2\frac{0}{4})+\frac{5}{14}\times(\frac{3}{5}\log_2\frac{3}{5}+\frac{2}{5}\log_2\frac{2}{5}) \approx 0.649$$

Vậy theo công thức tính chỉ số Gain:

$$\text{Gain}(\text{"Age"}) = \text{Info}(D) - \text{Info}_A(D) = 0.940 - 0.649 = 0.246.$$

Theo cách tính tương tự như trên, tính chỉ số Gain cho lần lượt các thuộc tính "income", "student" và "credit_rating". Kết quả sẽ là Gain("income") = 0.029; Gain("student") = 0.151 và Gain("credit_rating") = 0.048. Như vậy, thuộc tính "Age" là thuộc tính có chỉ số Gain lớn nhất nên sẽ được chọn là thuộc tính phân chia. Kết quả phân chia sẽ là cây quyết định như sau [1]:



income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

2.1.1.2 Gini index

Chỉ số Gini (Gini index): Chỉ số Gini được sử dụng trong thuật toán CART. Trái ngược với độ đo Gain, chỉ số Gini là độ đo về tính "không trong suốt" của tập dữ liệu. Chỉ số Gini của một tập dữ liệu D được định nghĩa như sau [1]:

$$Gini(D) = 1 - \sum_{i=1}^{m} (p_i)^2$$

Với m là tổng số nhãn lớp, pi là xác suất để một bộ bất kỳ trong D thuộc về một nhãn C_i, được tính như sau:

$$p_i = \frac{|C_{i,D}|}{|D|}.$$

Chỉ số Gini thường sẽ được tính toán dựa trên giả định một tập dữ liệu D được phân chia nhị phân thành hai tập con. Đầu tiên xét trường hợp thuộc tính A bất kỳ trong D có kiểu dữ liệu rời rạc, khi dùng phép chiếu sẽ thu được v = $\{a_1, a_2 \ldots .a_v\}$ giá trị khác nhau. Để xác định điểm chia tốt nhất của A, kiểm tra tất cả tập con có thể tạo được từ v giá trị phân biệt trên, mỗi tập con tạm gọi là S_A là một điều kiện kiểm tra nhị phân dạng $A \in S_A$. Như vậy với v giá trị khác nhau ta sẽ có $2^v - 2$ tập con, trong đó tập rỗng và tập toàn phần v = $\{a_1, a_2 \ldots .a_v\}$ sẽ không được xét đến. Như vậy tiến hành lặp qua tất cả các tập con này, mỗi lần lặp sẽ phân chia tập giá trị v thành hai tập con v_1 và v_2 riêng biệt thoả điều kiện rời rạc toàn phần (hội v_1 và v_2 chính là tập v và phần giao là tập rỗng). Với hai tập con v_1 và v_2 này tương ứng tập con D cũng được phân

chia thành hai tập con D_1 (các bộ có giá trị thuộc tính A ∈ v_1) và D_2 (các bộ có giá trị thuộc tính A ∈ v_2) theo , Gini(D) sẽ được tính như sau [1]:

$$Gini_A(D) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2).$$

Khác với độ đo Gain, người ta chọn chỉ số Gini nhỏ nhất với mong muốn sau khi phân chia dữ liệu sẽ làm giảm tính không trong suốt của tập D nhiều nhất. Đối với các giá trị liên tục có một lưu ý là đầu tiên phải sắp xếp các giá trị này, sau đó tất cả các giá trị cũng sẽ được tính toán chỉ số Gini và cũng chọn ra giá trị nào có thuộc tính Gini nhỏ nhất. Cũng giống như độ đo Gain, chỉ số Gini thông thường cũng được tính cho điểm giữa của hai giá trị liên tục nằm liền kề nhau. Lúc này tập D sẽ được chia làm hai tập D_1 là các bộ dữ liệu thoả điều kiện giá trị thuộc tính A nhỏ hơn hoặc bằng giá trị điểm giữa và D_2 thoả điều kiện giá trị thuộc tính A lớn hơn giá trị điểm giữa. Mục tiêu của chí số Gini là càng làm giảm tính không trong suốt của dữ liệu càng nhiều càng tốt, giá trị giảm trừ này thể hiện qua công thức [1]:

$$\Delta gini(A) = Gini(D) - Gini_A(D)$$

Lưu ý Gini(D) là một con số cố định, chính vì mục đích chọn điểm chia sao cho Δgini(A) là lớn nhất nên bắt buộc chọn thuộc tính A sao cho GiniA(D) là nhỏ nhất. Ví dụ bên dưới sẽ tính chỉ số Gini cho tập dữ liệu từ bảng dữ liệu ở trên, lưu ý có chín bộ dữ liệu có nhãn lớp "buys_computer" = yes và năm bộ dữ liệu có nhãn lớp "buys_computer" = no [1]:

$$Gini(D) = 1 - (\frac{9}{14})^2 - (\frac{5}{14})^2 \approx 0.459.$$

Để tìm điểm chia tốt nhất, tiến hành lặp qua tất cả tập con (trừ tập rỗng và tập toàn bộ) của từng thuộc tính. Giả sử xét thuộc tính "income" bao gồm ba giá trị: {low, medium, high}. Xét tập con {low, medium}, như vậy có mười bộ dữ liệu thuộc tập con này, trong đó có bốn bộ có giá trị low và sáu bộ có giá trị medium:

$$Gini_{income \in \{low, medium\}}(D) = \frac{10}{14}Gini(D_1) + \frac{4}{14}Gini(D_2)$$

$$= \frac{10}{14}(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2) + \frac{4}{14}(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2) = 0.45 = Gini_{income \in \{high\}}(D)$$

Tương tự, các tập con còn lại ({low, high} và {medium}) có Gini = 0.315 và ({medium, high} và {low}) có Gini = 0.3. Như vậy, nếu xét trên thuộc tính "income", tập con ({medium, high} và {low}) có Gini = 0.3 sẽ được chọn (lưu ý chỉ xét riêng trên thuộc tính này). Lần lượt thực hiện cho các thuộc tính còn lại và chọn ra thụộc tính nào có Gini nhỏ nhất, đó chính là thuộc tính sẽ được chọn để phân chia. [1]

2.1.2 Normalized impurity based criteria:

Ta dùng các tiêu chuẩn này khi thuộc tính có nhiều giá trị. Các tiêu chuẩn thuộc loại này là Gain Ratio, Distance Measure. Phần dưới đây sẽ giới thiệu về tiêu chuẩn Gain Ratio.

Theo các nghiên cứu thì độ đo Gain thích hợp trong trường hợp các thuộc tính có nhiều giá trị hiện hành (dĩ nhiên các giá trị này phải thuộc miền giá trị, ví dụ với 100 mẫu tin có 80 giá trị khác nhau của thuộc tính khi sử dụng phép chiếu lên thuộc tính). Xem xét trường hợp thuộc tính "Client_ID", trong đó mỗi khách hàng sẽ có một mã số riêng biệt, như vậy khi áp dụng phép chia trên thuộc tính này sẽ có một số rất lớn các tập con phát sinh, thậm chí mỗi khách hàng thuộc một tập con. Điều trên xảy ra là do mỗi khách hàng khi xét trên duy nhất một thuộc tính "Client_ID" được xem như là "trong suốt" (InfoClient_ID(D)=0). Như vậy việc phân chia theo thuộc tính này được xem như vô ích. Thuật toán C4.5 (một thuật toán cải tiến từ ID3) sử dụng độ đo tỷ lệ Gain (Gain ratio) được mở rộng từ độ đo Gain, được định nghĩa như sau [1]:

$$\text{SplitInfo}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

Công thức $\text{SplitInfo}_A(D)$ cho biết thông tin tiềm ẩn được tạo ra bằng cách chia tập D trong v tập con. Với mỗi tập con được tạo ra, tính toán tỷ lệ của số bộ trong tập con này so với tổng số bộ dữ liệu trong tập D. Khi đó, độ đo tỷ lệ Gain sẽ được tính toán theo công thức sau [1]:

$$\text{GainRatio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}(A)}$$

Tất cả thuộc tính sẽ được tính toán độ đo tỷ lệ Gain, thuộc tính nào có độ đo tỷ lệ Gain lớn nhất sẽ được chọn làm thuộc tính phân chia. Tuy nhiên, khi sử dụng độ đo tỷ lệ Gain, cần phải lưu ý một điều về mẫu số trong công thức SplitInfo(A) vì mẫu số này có thể đạt giá trị bằng 0. Xét vì dụ được nêu trong bảng dữ liệu trên, để tính độ đo tỷ lệ Gain cho thuộc tính "income", lưu ý thuộc tính này khi chiếu lên có ba giá trị riêng biệt: "low" (bốn bộ dữ liệu), "medium" (sáu bộ dữ liệu) và "high" (bốn bộ dữ liệu). Theo công thức [1]:

$$\text{SplitInfo}_{\text{income}}(D) = \frac{4}{14} \times \log_2 \frac{4}{14} + \frac{6}{14} \times \log_2 \frac{6}{14} + \frac{4}{14} \log_2 \frac{4}{14} \approx 0.926$$

Xem lại ví dụ phần độ đo Gain, tính được Gain("income") = 0.029. Như vậy, tỷ lệ độ đo Gain của thuộc tính "income":

$$\text{GainRatio}(\text{income}) = \frac{0.029}{0.926} \approx 0.031$$

2.1.3 Binary criteria

Dùng để tạo cây quyết định nhị phân. Các tiêu chuẩn thường được sử dụng đối với tiêu chuẩn này là:

- Twoing Criterion
- Orthogonal (ORT) Criterion
- Kolmogorov–Smirnov Criterion
- AUC–Splitting Criteria

2.2 Tiêu chuẩn tách đa chiều:

Khác với tách 1 chiều nghĩa là tách theo 1 thuộc tính, tiêu chuẩn tách đa chiều sử dụng kết hợp nhiều thuộc tính cùng lúc để phân tách. Tuy nhiên, điều này sẽ ảnh hưởng tới performance nên ít được sử dụng.

2.3 Tiêu chuẩn dừng (Stopping Criteria):

Dưới đây là một số tiêu chuẩn dừng thường được sử dụng:

- Từng thuộc tính đã được đưa vào dọc theo con đường trên cây
- Các mẫu huấn luyện ứng với nút lá có cùng giá trị thuộc tính đích (chẳng hạn, chúng có entropy bằng 0)
- Tất cả các mẫu dữ liệu E thuộc về cùng một lớp duy nhất
- Tất cả các mẫu có cùng giá trị thuộc tính

3. Một số thuật toán (Trần Thị Tuyết Vân)

Với tiêu chí xây dựng cây quyết định ngày càng đơn giản, cho độ chính xác phân lớp cao, chi phí thấp, có khả năng mở rộng,… thì có rất nhiều tác giả đã cho ra đời các thuật toán ngày càng tối ưu hơn. Một số thuật toán tiêu biểu sau:

Algorithms	References
CLS(Concept learning System)	C. I. Hovland và E. B. Hunt
CART(Classification And Regression Tree)	Breiman et al.(1984)
ID3(Interactive Dichotomizer 3)	Quinlan(1986)
C4.5	Quinlan(1993)
CHAID (CHi-squared Automatic Interaction Detecor)	Kass(1980)
QUEST	LohandShih(1997)
CAL5	Muller and Wysotzki(1994)
FACT	Loh and Vanichsetakul(1988)
LMDT	Brodley and Utgoff(1995)
T1	Holte(1993)
PUBLIC	Rastogi and Shim(2000)
MARS	Friedman(1991)
SLIQ (Supervised Learning in Quest)	Mehta(1996)
SPRINT(A Scalable Parallel Classifier for DataMining)	Shafer, Agrawal, Mehta
….	….

Trong phạm vi đồ án môn học này chúng tôi xin trình bày cụ thể 4 thuật toán gồm thuật toán CLS, ID3, C4.5, SPRINT.

3.1 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 [2]:

1. Tạo một nút T, nút này gồm tất cả các mẫu của tập huấn luyện.
2. Nếu tất cả các mẫu trong T ***thuộc cùng một lớp*** và có thuộc tính quyết định mang giá trị :

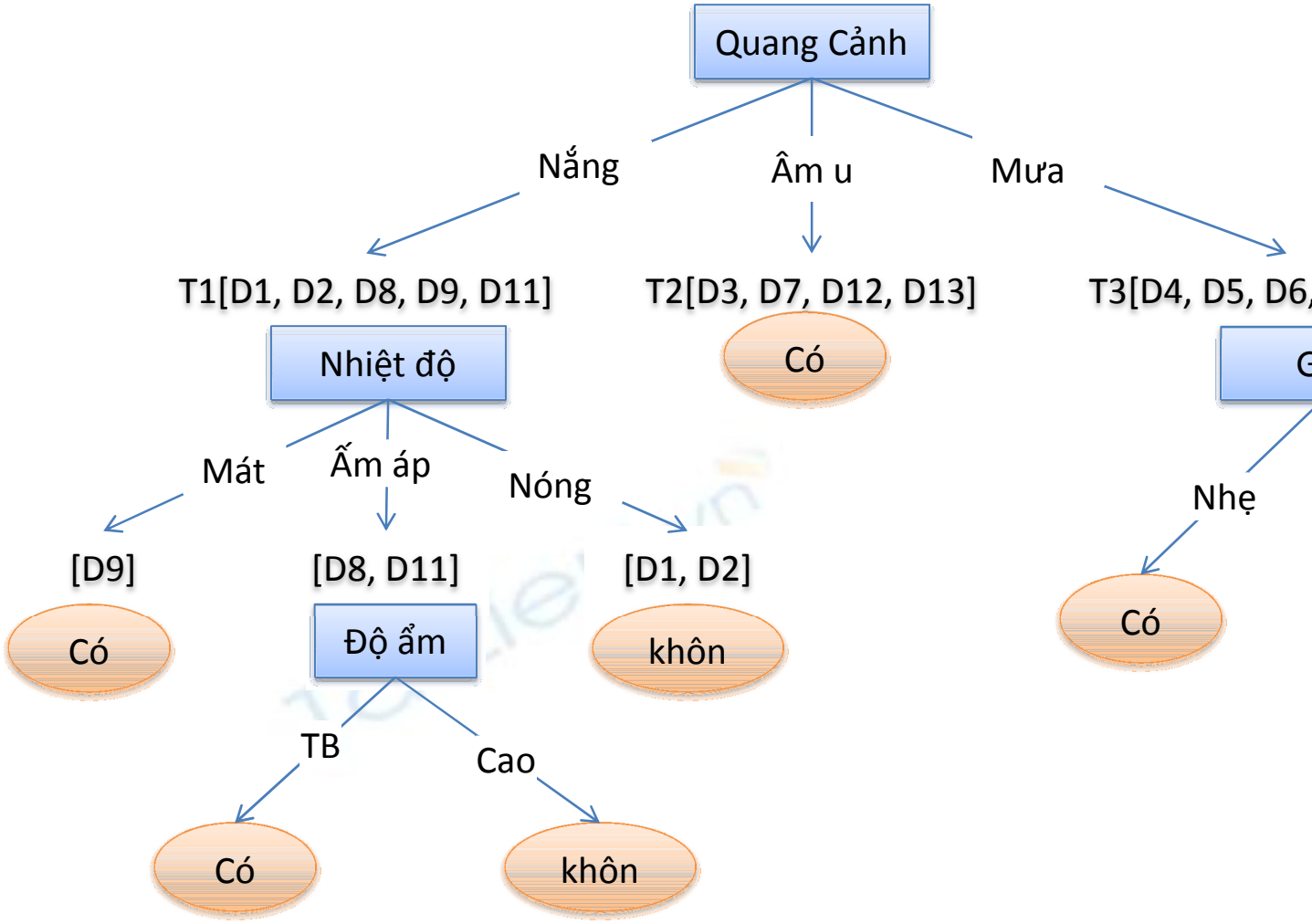
- **"yes"** thì gán nhãn cho nút T là "yes" và dừng lại. T lúc này là nút lá.
- **"no"** thì gán nhãn cho nút T là "no" và dừng lại. T lúc này là nút lá.

3. 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.
4. Thực hiện lặp cho các nút con T_i(i =1,2..n) và quay lại bước 2.

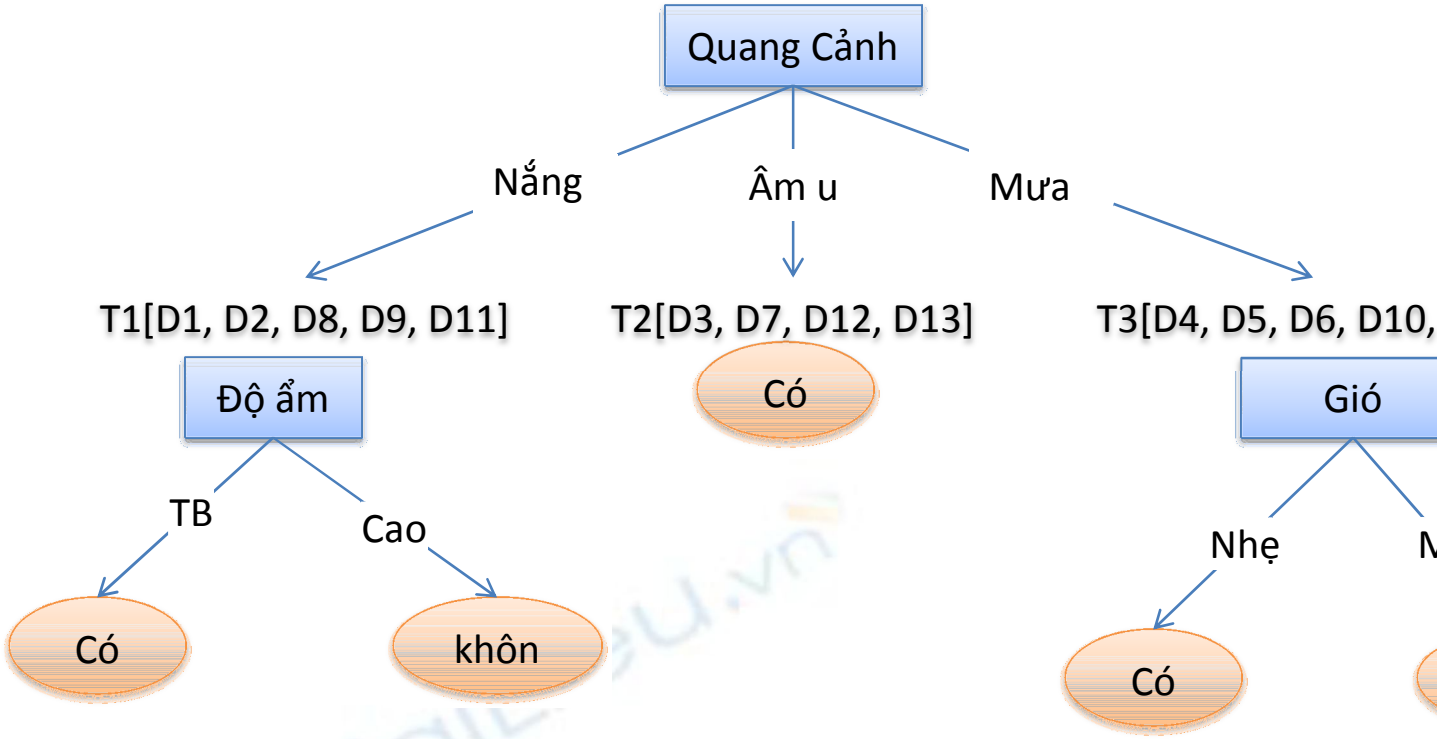
Ví dụ 3.1: Cho tập huấn luyện gồm 14 mẫu, dựa vào thời tiết để xác định người đó có đi chơi Tennis hay không?

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

Theo các bước của thuật toán ta có cây quyết định như sau:



Ta nhận thấy trong bước 3 của thuật toán, thuộc tính được chọn để triển khai cây là tuỳ ý. Nếu ta chọn thuộc tính "Độ ẩm" làm thuộc tính để triển khai T1 thì ta có 1 cây khác:



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.

3.2 Thuật toán ID3

Thuật toán ID3 được phát biểu bởi tác giả Quinlan (trường đại học Syney, Australia) và được công bố vào cuối thập niên 70 của thế kỷ 20. Sau đó, thuật toán này 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). ID3 sử dụng độ đo ***Information Gain*** (trình bày ở 2.1.1.1)để đo tính hiệu quả của các thuộc tính phân lớp. Trong quá trình xây dựng cây quyết định theo thuật toán ID3 tại mỗi bước phát triển cây, thuộc tính được chọn để triển khai là thuộc tính có giá trị Gain lớn nhất. Hàm xây dựng cây quyết định trong thuật toán ID3 [2]

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_vùng_V các ví dụ trong tập_ví_dụ có giá trị V
tại thuộc tính P;
    Gọi induce_tree(phân_vùng_V, tập_thuộc_tính), gắn kết quả
vào nhánh V
end
  end
end
```

Xét **ví dụ 3.1** cho thuật toán ID3:

- Gọi tập huấn luyện là S, số mẫu thuộc lớp **Có** ký hiệu là (+) và số mẫu thuộc lớp **Không** ký hiệu là (-), ta có S[9+,5-] tức tập huấn luyện S có 14 mẫu trong đó có 9 mẫu thuộc lớp **Có** và 5 mẫu thuộc lớp **Không**.
- Để xác định thuộc tính phân lớp ta cần tính Information Gain cho từng thuộc tính của mẫu huấn luyện:
  - Thuộc tính *Quang Cảnh*

    Value(QC)={Nắng, Mưa, Âm u}

    Gọi $S_{Nắng}$ là tập các mẫu có QC=Nắng ta có $S_{Nắng}$=[2+,3-]

    Tương tự ta có $S_{Mưa}$=[3+,2-], $S_{Âm\ u}$=[4+,0-]

$$Gain(S, QC) = Entropy(S) - \sum_{V \epsilon Value(QC)} \frac{|S_v|}{|S|} Entropy(S_v) = Entropy(S) - \frac{5}{14} Entropy(S_{Nắng}) - \frac{4}{14} Entropy(S_{Âm\ u}) - \frac{5}{14} Entropy(S_{Mưa}) = 0.94 - \frac{5}{14} \times 0.971 - \frac{4}{14} \times 0 - \frac{5}{14} \times 0.971 \approx 0.246$$

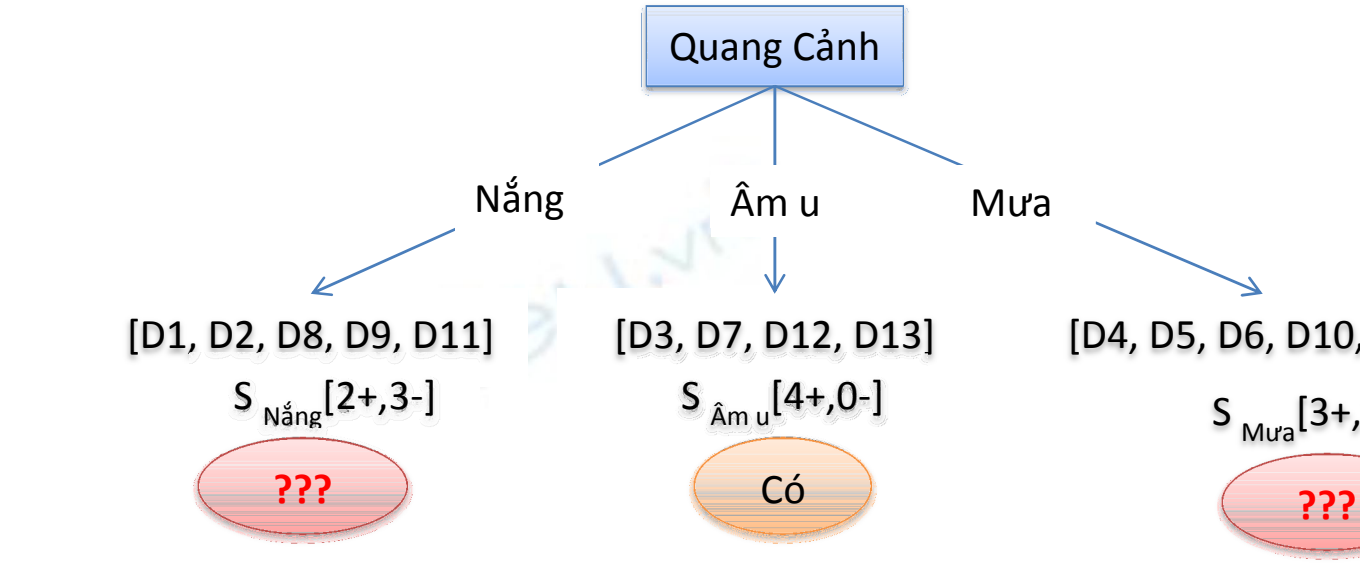
Tư tượng đối với các thuộc tính *Nhiệt độ, Độ ẩm, Gió* ta có Gain tương ứng như sau:

- *Gain(S,ND)= 0.029*

- *Gain(S,DA)= 0.151*
- *Gain(S,G)= 0.048*

➔*Chọn Quang cảnh làm thuộc tính phân lớp vì có Gain lớn nhất*

- Vẽ cây quyết định:



Ngày	Nhiệt độ	Độ ẩm	Gió	Chơi Tennis
D1	Nóng	Cao	Nhẹ	Không
D2	Nóng	Cao	Mạnh	Không
D8	Ấm áp	Cao	Nhẹ	Không
D9	Mát	TB	Nhẹ	Có
D11	Ấm áp	TB	Mạnh	Có

Ngày	Nhiệt độ	Độ ẩm
D4	Ấm áp	Cao
D5	Mát	TB
D6	Mát	TB
D10	Ấm áp	TB
D14	Ấm áp	Cao

Do *Quang cảnh=Nắng và Quang cảnh=Mưa* chưa xác định được thuộc tính phân lớp nên ta chia tập huấn liệu thành 2 bảng như hình trên và tiếp tục tìm thuộc tính phân lớp cho 2 bảng mẫu huấn luyện. Kết quả cuối cùng ta có cây quyết định sau: