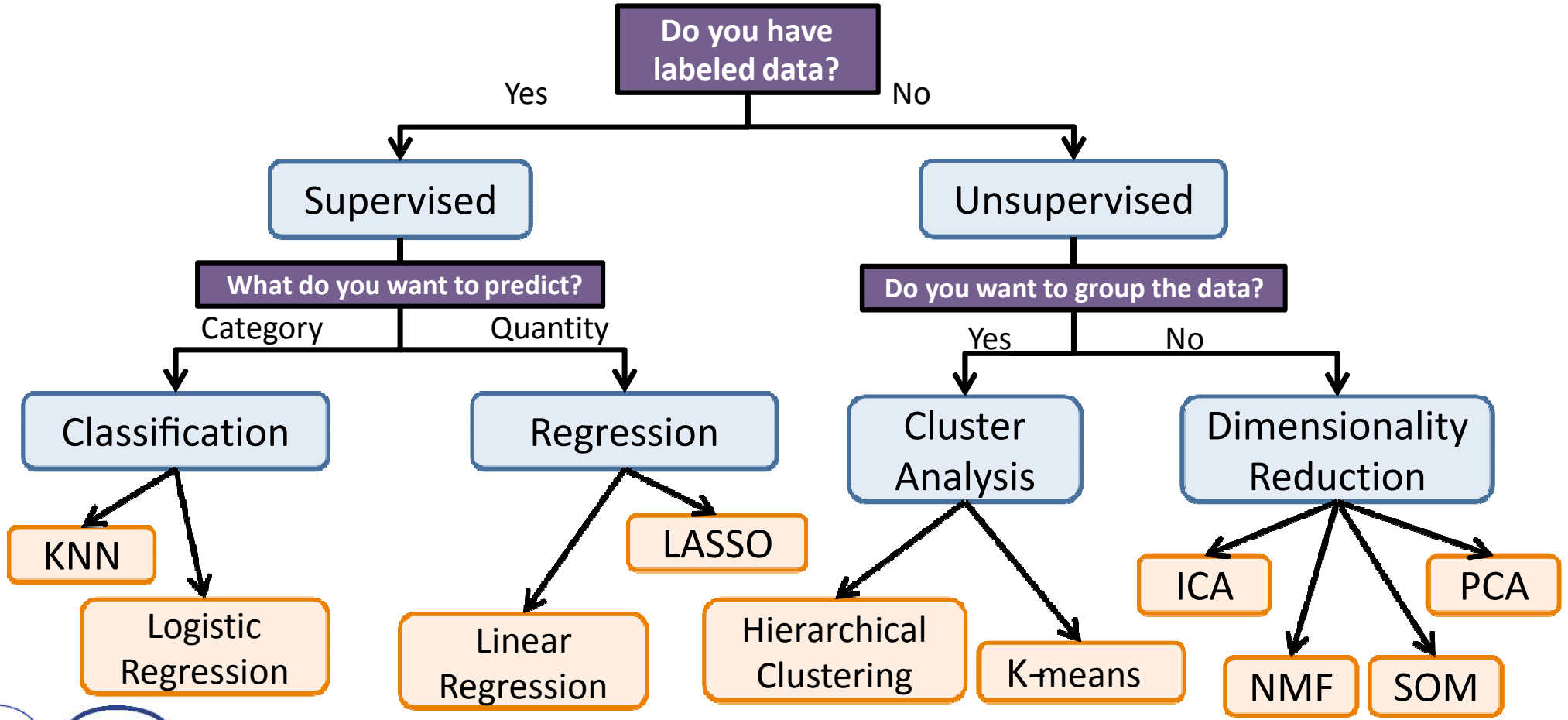
Cây phân loại và hồi quy

Nguyễn Thanh Tùng
Khoa Công nghệ thông tin – Đại học Thủy Lợi
tungnt@tlu.edu.vn

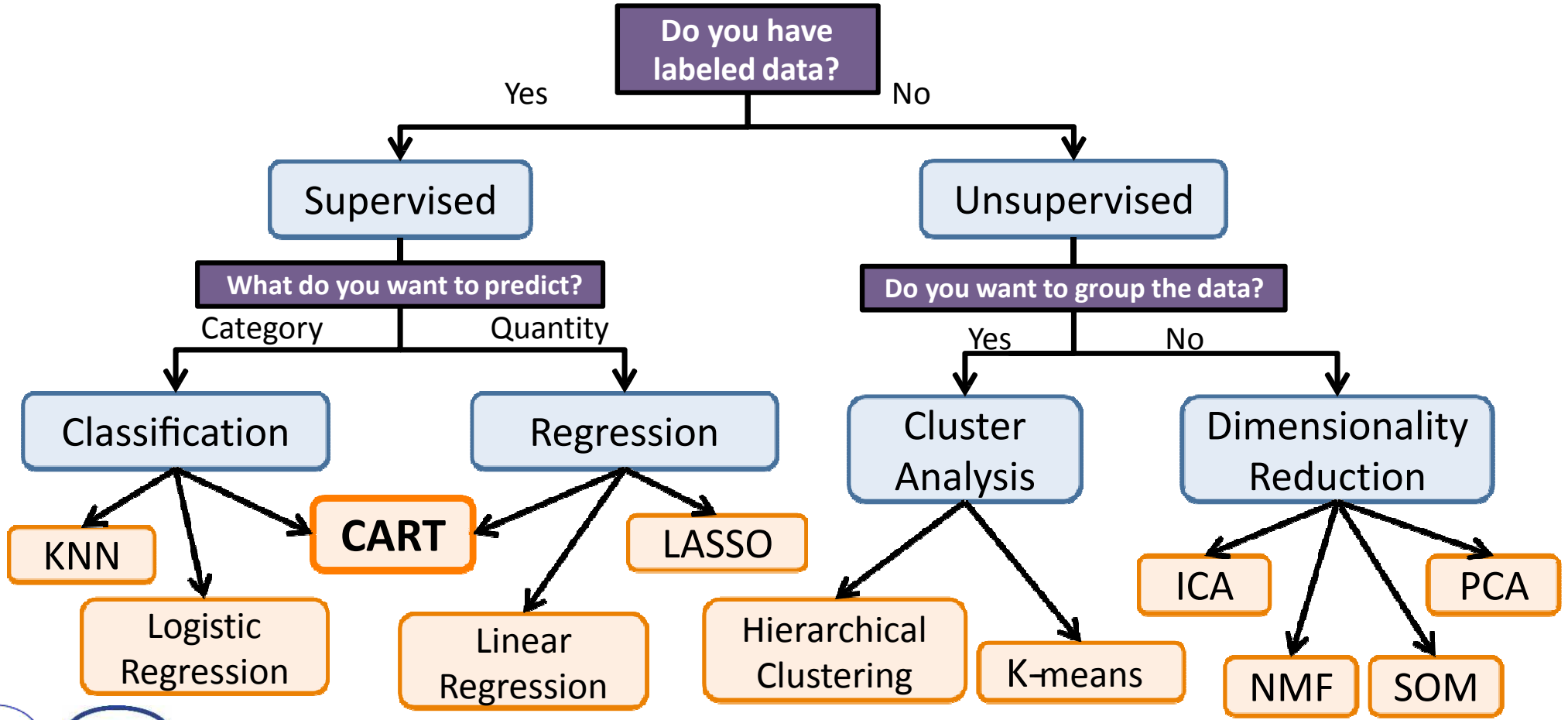
https://piazza.com/tlu.edu.vn/fall2017/cse445fall2017

Bài giảng có sử dụng hình vẽ trong cuốn sách "An Introduction to Statistical Learning with Applications in R" với sự cho phép của tác giả, có sử dụng slides các khóa học CME250 của ĐH Stanford và IOM530 của ĐH Southern California


Các giải thuật Học máy



Các giải thuật Học máy



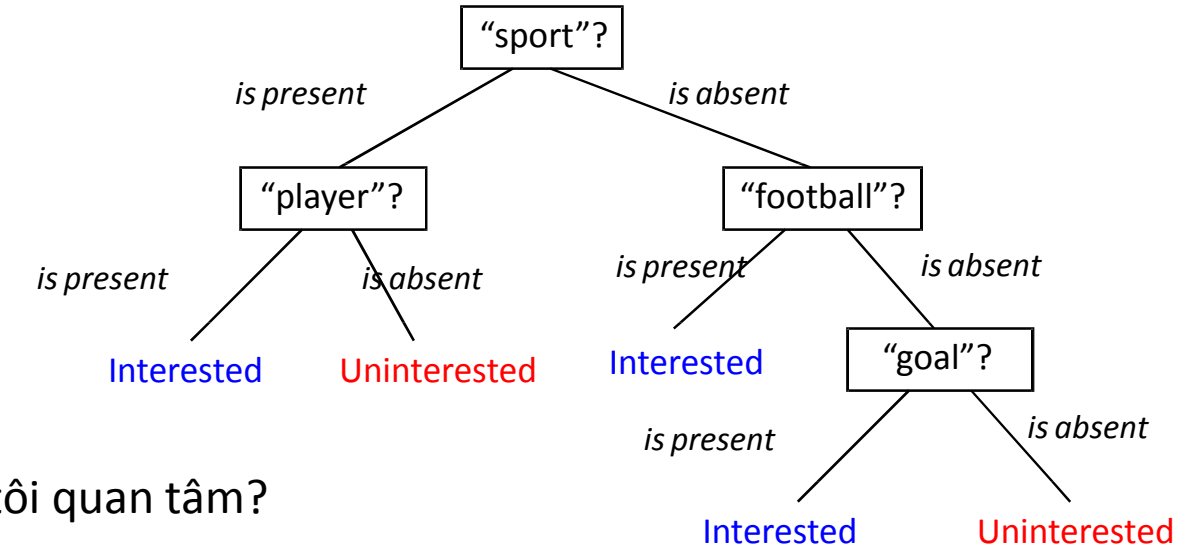
Cây quyết định
(Decision tree)

Cây quyết định là gì?

- Học cây quyết định (Decision tree –DT– learning)
  - Để học (xấp xỉ) một hàm mục tiêu có giá trị rời rạc (*discrete- valued target function*) – hàm phân lớp
  - Hàm phân lớp được biểu diễn bởi một cây quyết định
- Một cây quyết định có thể được biểu diễn (diễn giải) bằng một tập các luật IF-THEN (dễ đọc và dễ hiểu)
- Học cây quyết định có thể thực hiện ngay cả với các dữ liệu có chứa nhiễu/lỗi (noisy data)
- Được áp dụng thành công trong rất nhiều các bài toán ứng dụng thực tế

nguồn: Nguyễn Nhật Quang-Học máy

Cây quyết định là gì?

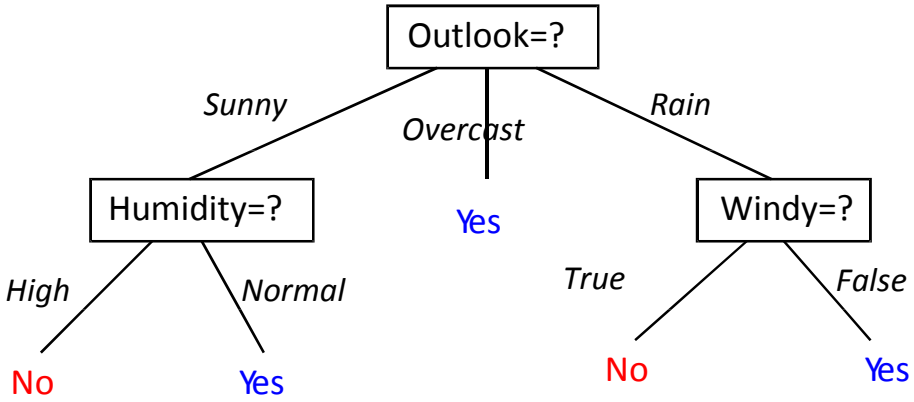


Ví dụ về DT: Những tin tức nào mà tôi quan tâm?

- (…,"sport",…,"player",…) → Interested
- (…,"goal",…) → Interested
- (…,"sport",…) → Uninterested

nguồn: Nguyễn Nhật Quang-Học máy

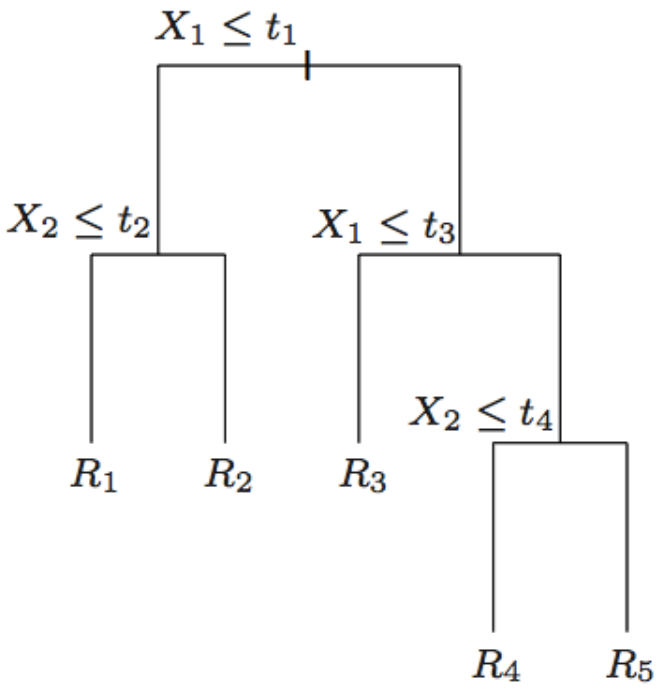
Cây quyết định là gì?



- (Outlook=Overcast, Temperature=Hot, Humidity=High, Windy=False)
  → Yes
- (Outlook=Rain, Temperature=Mild, Humidity=High, Windy=True)
  → No
- (Outlook=Sunny, Temperature=Hot, Humidity=High, Windy=True)
  → No

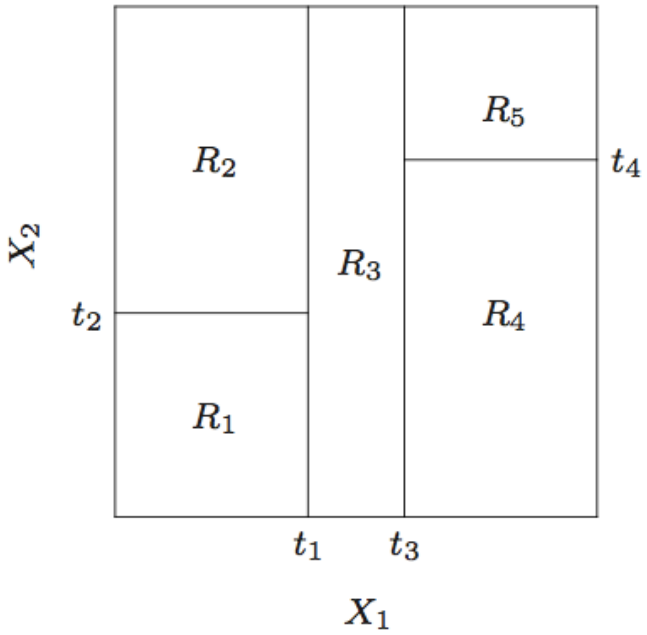
Ví dụ về DT: Một người có chơi tennis không?

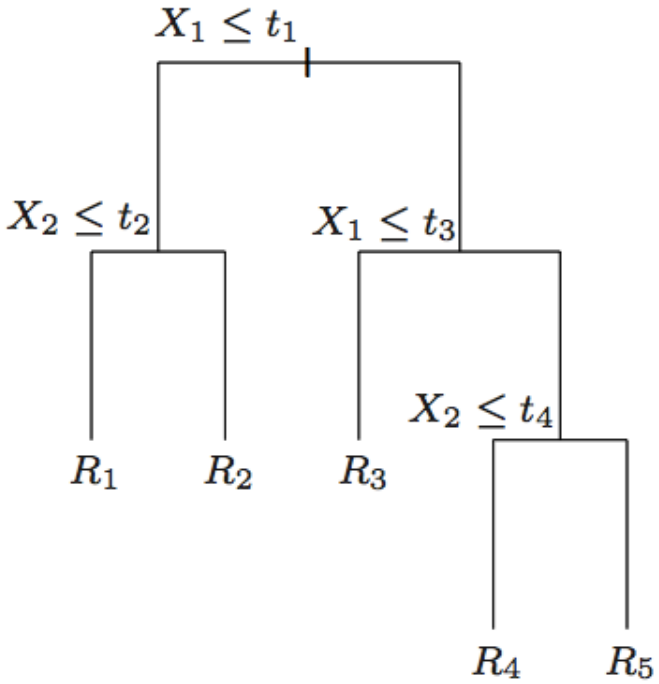
Cây quyết định là gì?



Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Cây quyết định là gì?

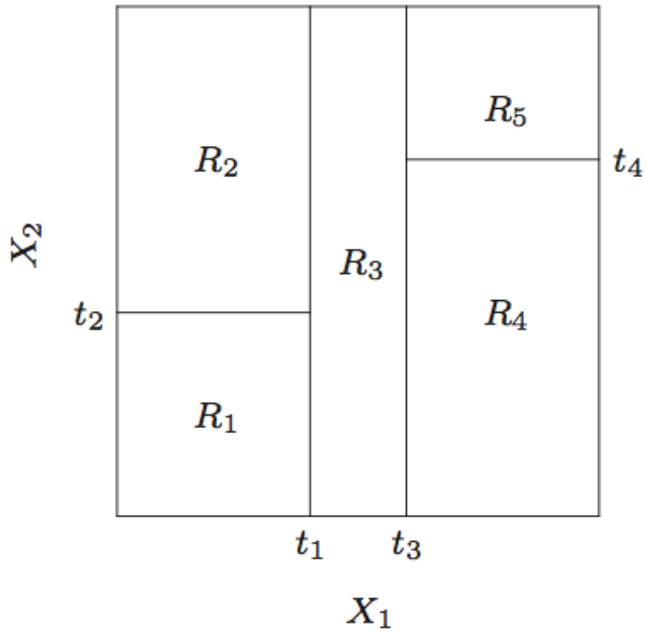




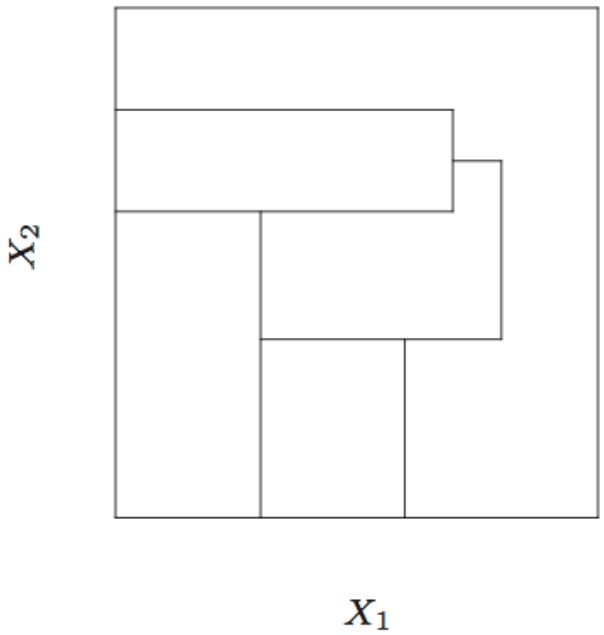
Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Cây quyết định là gì?

ĐÚNG



SAI



Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Dữ liệu đầu vào của cây quyết định

table (300 records)

Table Generate

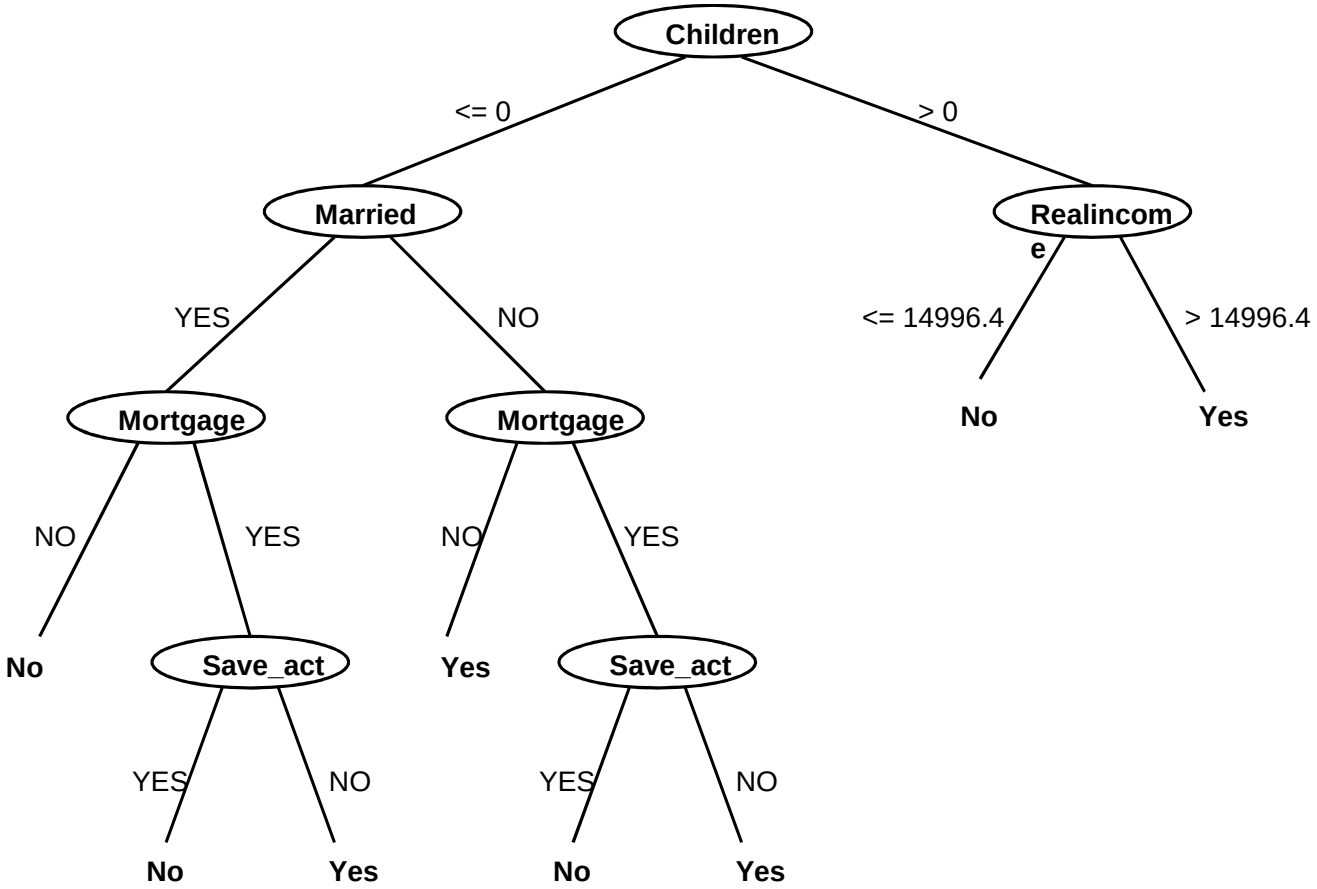
age	sex	region	income	married	children	car	save_act	current_act	mortgage	pep	realincome
48	FEMALE	INNER_CITY	17546.0	NO	1	NO	NO	NO	NO	YES	17546.0
40	MALE	TOWN	30085.1	YES	3	YES	NO	YES	YES	NO	10028.4
51	FEMALE	INNER_CITY	16575.4	YES	0	YES	YES	YES	NO	NO	16575.4
23	FEMALE	TOWN	20375.4	YES	3	NO	NO	YES	NO	NO	6791.8
57	FEMALE	RURAL	50576.3	YES	0	NO	YES	NO	NO	NO	50576.3
57	FEMALE	TOWN	37869.6	YES	2	NO	YES	YES	NO	YES	18934.8
22	MALE	RURAL	8877.07	NO	0	NO	NO	YES	NO	YES	8877.07
58	MALE	TOWN	24946.6	YES	0	YES	YES	YES	NO	NO	24946.6
37	FEMALE	SUBURBAN	25304.3	YES	2	YES	NO	NO	NO	NO	12652.1
54	MALE	TOWN	24212.1	YES	2	YES	YES	YES	NO	NO	12106.1
66	FEMALE	TOWN	59803.9	YES	0	NO	YES	YES	NO	NO	59803.9
52	FEMALE	INNER_CITY	26658.8	NO	0	YES	YES	YES	YES	NO	26658.8
44	FEMALE	TOWN	15735.8	YES	1	NO	YES	YES	YES	YES	15735.8
66	FEMALE	TOWN	55204.7	YES	1	YES	YES	YES	YES	YES	55204.7
36	MALE	RURAL	19474.6	YES	0	NO	YES	YES	YES	NO	19474.6
38	FEMALE	INNER_CITY	22342.1	YES	0	YES	YES	YES	YES	NO	22342.1
37	FEMALE	TOWN	17729.8	YES	2	NO	NO	NO	YES	NO	8864.9
46	FEMALE	SUBURBAN	41016.0	YES	0	NO	YES	NO	YES	NO	41016.0
62	FEMALE	INNER_CITY	26909.2	YES	0	NO	YES	NO	NO	YES	26909.2

Biểu diễn cây quyết định

- Mỗi nút trong (*internal node)* biểu diễn một biến cần kiểm tra giá trị (*a variable to be tested*) đối với các mẫu
- Mỗi nhánh (*branch*) từ một nút sẽ tương ứng với một giá trị có thể của biến gắn với nút đó
- Mỗi nút lá (leaf node) biểu diễn một phân lớp (a classification)
- Một cây quyết định học được sẽ phân lớp đối với một mẫu, bằng cách duyệt cây từ nút gốc đến một nút lá

  → Nhãn lớp gắn với nút lá đó sẽ được gán cho mẫu cần phân lớp

Minh họa cây quyết định



Tập luật từ cây quyết định

Rule #1
if children =< 0
and married == YES
and mortgage == YES
and save_act == NO
then -> YES (9.0, 0.889)

Rule #2
if children =< 0
and married == NO
and mortgage == NO
then -> YES (29.0, 0.931)

Rule #3
if children =< 0
and married == NO
and mortgage == YES
and save_act == NO
then -> YES (3.0, 1.0)

Rule #4
if children > 0
and realincome > 14996.4
then -> YES (85.0, 0.953)

Tập luật từ cây quyết định

Rule #1

if children =< 0
and married == YES
and mortgage == NO
then -> NO (59.0, 0.898)

Rule #2

if children =< 0
and married == YES
and mortgage == YES
and save_act == YES
then -> NO (16.0, 0.875

Rule #3

if children =< 0
and married == NO
and mortgage == YES
and save_act == YES
then -> NO (12.0, 1.0)

Rule #4

if children > 0
and realincome =< 14996.4
then -> NO (87.0, 0.908)

Biểu diễn cây quyết định

- Một cây quyết định biểu diễn một phép tuyển (disjunction) của các kết hợp (conjunctions) của các ràng buộc đối với các giá trị thuộc tính của các mẫu
- Mỗi đường đi (path) từ nút gốc đến một nút lá sẽ tương ứng với một kết hợp (conjunction) của các kiểm tra giá trị biến (variable tests)
- Cây quyết định (bản thân nó) chính là một phép tuyển của các kết hợp này

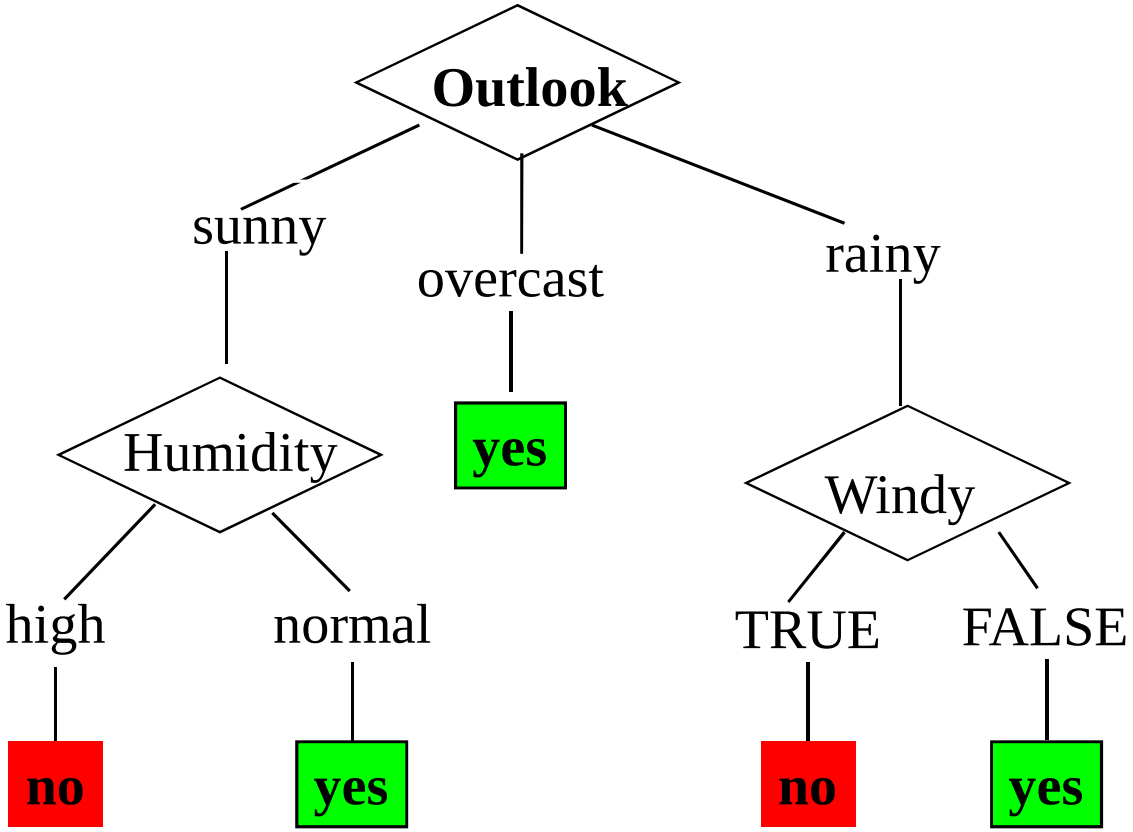
Tập dữ liệu Weather

Xét tập dữ liệu Weather ghi lại những ngày mà một người chơi (không chơi) tennis:

Day	Outlook	Temperature	Humidity	Windy	Play Tennis
D1	Sunny	Hot	High	FALSE	No
D2	Sunny	Hot	High	TRUE	No
D3	Overcast	Hot	High	FALSE	Yes
D4	Rain	Mild	High	FALSE	Yes
D5	Rain	Cool	Normal	FALSE	Yes
D6	Rain	Cool	Normal	TRUE	No
D7	Overcast	Cool	Normal	TRUE	Yes
D8	Sunny	Mild	High	FALSE	No
D9	Sunny	Cool	Normal	FALSE	Yes
D10	Rain	Mild	Normal	FALSE	Yes
D11	Sunny	Mild	Normal	TRUE	Yes
D12	Overcast	Mild	High	TRUE	Yes
D13	Overcast	Hot	Normal	FALSE	Yes
D14	Rain	Mild	High	TRUE	No

[Mitchell, 1997]

Mô hình cây QĐ có (không) chơi tennis



[(Outlook=Sunny) ∧ (Humidity=Normal)] ∨

(Outlook=Overcast) ∨

[(Outlook=Rain) ∧ (Windy=False)]

Xây dựng cây QĐ thế nào?

Phương pháp dựng cây theo Top-down

Ban đầu, tất cả các mẫu trong tập huấn luyện đều đặt tại nút gốc.

Tách các mẫu theo đệ quy bằng cách chọn 1 thuộc tính trong mỗi lần tách cho đến khi gặp điều kiện dừng.

Phương pháp tỉa cây theo Bottom-up

Ban đầu dựng cây lớn nhất có thể

Chuyển phần cây con hoặc nhánh từ phần đáy của cây lên nhằm cải thiện tính chính xác khi dự đoán mẫu mới

Giải thuật ID3

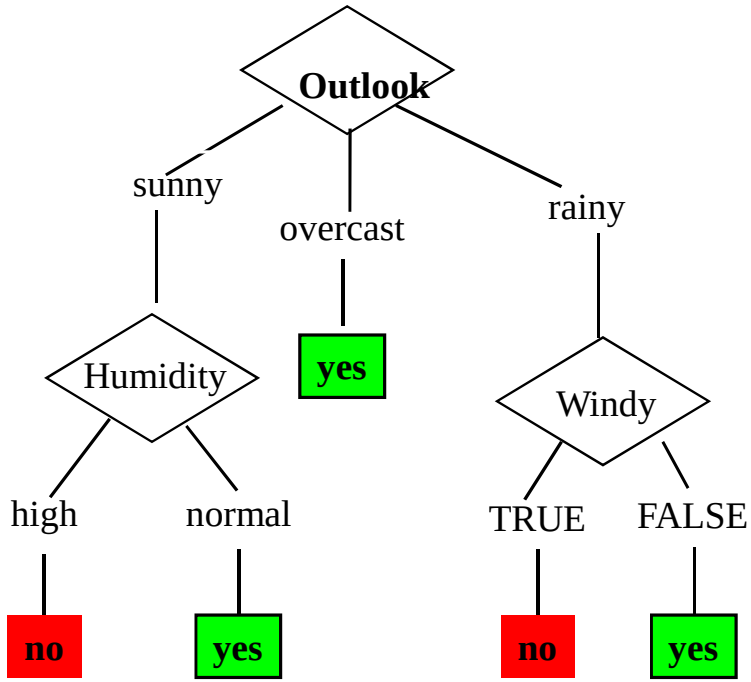
- Thực hiện giải thuật tìm kiếm tham lam (greedy search) đối với không gian các cây quyết định có thể
- Xây dựng (học) một cây quyết định theo chiến lược top-down, bắt đầu từ nút gốc
- Ở mỗi nút, biến kiểm tra (test variable) là biến có khả năng phân loại tốt nhất đối với các mẫu gắn với nút đó
- Tạo mới một cây con (sub-tree) của nút hiện tại cho mỗi giá trị có thể của biến kiểm tra, và tập huấn luyện sẽ được tách ra (thành các tập con) tương ứng với cây con vừa tạo
- Mỗi biến chỉ được phép xuất hiện tối đa 1 lần đối với bất kỳ một đường đi nào trong cây
- Quá trình phát triển (học) cây quyết định sẽ tiếp tục cho đến khi: Cây quyết định phân loại hoàn toàn (perfectly classifies) các mẫu, hoặc tất cả các thuộc tính đã được sử dụng


Lựa chọn biến kiểm tra

- Tại mỗi nút, chọn biến kiểm tra như thế nào?
- Chọn biến quan trọng nhất cho việc phân lớp các mẫu gắn với nút đó
- Làm thế nào để đánh giá khả năng của một biến đối với việc phân tách các mẫu theo nhãn lớp của chúng?

  → Sử dụng một đánh giá thống kê – *Information Gain*

  Với một số cây quyết định khác:
  - Information gain ratio (C4.5)
  - Gini index (CART)



Entropy

- Một đánh giá thường được sử dụng trong lĩnh vực lý thuyết thông tin (Information Theory)
- Để đánh giá mức độ hỗn tạp (impurity/inhomogeneity) của một tập
- Entropy của tập S đối với việc phân lớp có c lớp

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

trong đó p_i là tỷ lệ các mẫu trong tập S thuộc vào lớp i, và **$0.\log_2 0 = 0$**

- Entropy của tập S đối với việc phân lớp có 2 lớp

$$Entropy(S) = -p_1 . \log_2 p_1 - p_2 . \log_2 p_2$$

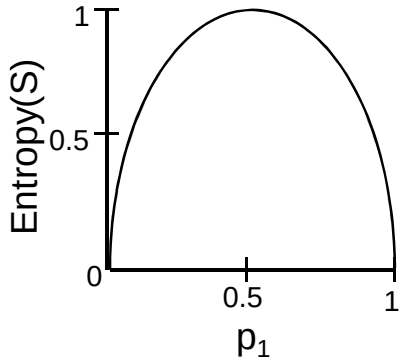
- Ý nghĩa của entropy trong lĩnh vực Information Theory

→ Entropy của tập S chỉ ra số lượng bits cần thiết để mã hóa lớp của một phần tử được lấy ra ngẫu nhiên từ tập S

Entropy – Ví dụ với 2 lớp

- S gồm 14 mẫu, trong đó 9 mẫu thuộc về lớp c_1 và 5 mẫu thuộc về lớp c_2
- Entropy của tập S đối với phân lớp có 2 lớp:

$$Entropy(S) = -(9/14).\log_2(9/14) - (5/14).\log_2(5/14) \approx 0.94$$



- Entropy =0, nếu tất cả các mẫu thuộc cùng một lớp (c_1 hoặc c_2)
- Entropy =1, số lượng các mẫu thuộc về lớp c_1 bằng số lượng các mẫu thuộc về lớp c_2
- Entropy = một giá trị trong khoảng (0,1), nếu như số lượng các mẫu thuộc về lớp c_1 khác với số lượng các mẫu thuộc về lớp c_2

Information gain

- Information Gain của một biến đối với một tập các mẫu:
  - Mức độ giảm về Entropy
  - Bởi việc phân tách (partitioning) các mẫu theo các giá trị của biến đó
- Information Gain của biến A đối với tập S

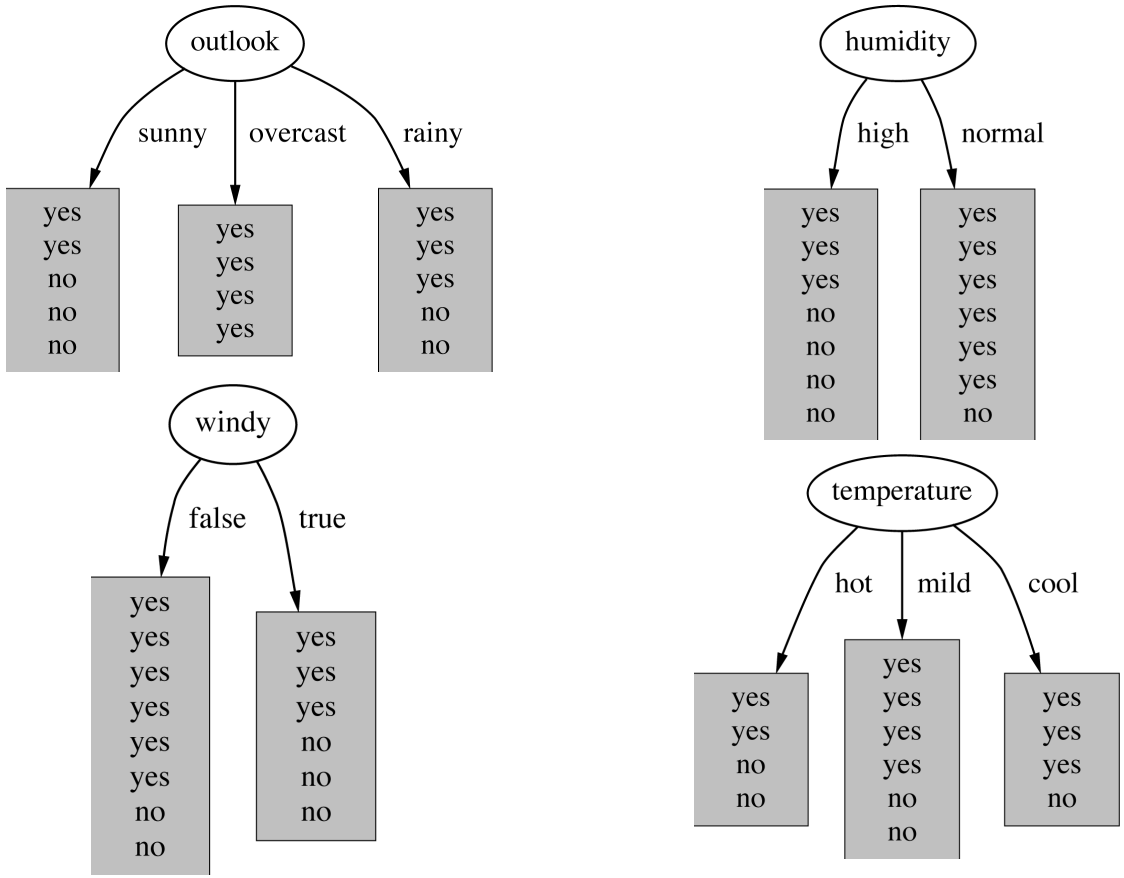
$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

trong đó Values(A) là tập các giá trị có thể của biến A, và

$S_v = \{x \mid x \in S, x_A = v\}$

- Trong công thức trên, thành phần thứ 2 thể hiện giá trị Entropy sau khi tập S được phân chia bởi các giá trị của biến A
- Ý nghĩa của Gain(S, A): Số lượng bits giảm được (reduced) đối với việc mã hóa lớp của một phần tử được lấy ra ngẫu nhiên từ tập S, khi biết giá trị của biến A

Weather-Tìm các khả năng tách



Biến Windy

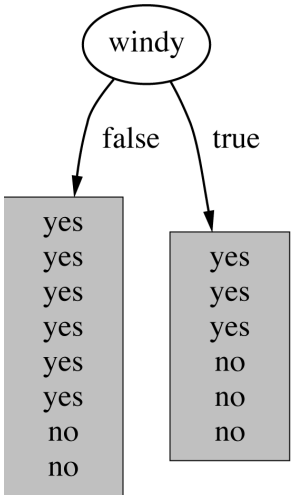
Hãy tính giá trị Information Gain của biến `Windy` đối với tập học S – `Gain(S,Windy)`?

Biến `Windy` có 2 giá trị có thể: `False` và `True`

S = {9 mẫu lớp `Yes` và 5 mẫu lớp `No`}

$S_{\texttt{False}}$ = {6 mẫu lớp `Yes` và 2 mẫu lớp `No` có giá trị `Windy=False`}

$S_{\texttt{True}}$ = {3 mẫu lớp `Yes` và 3 mẫu lớp `No` có giá trị `Windy=True`}



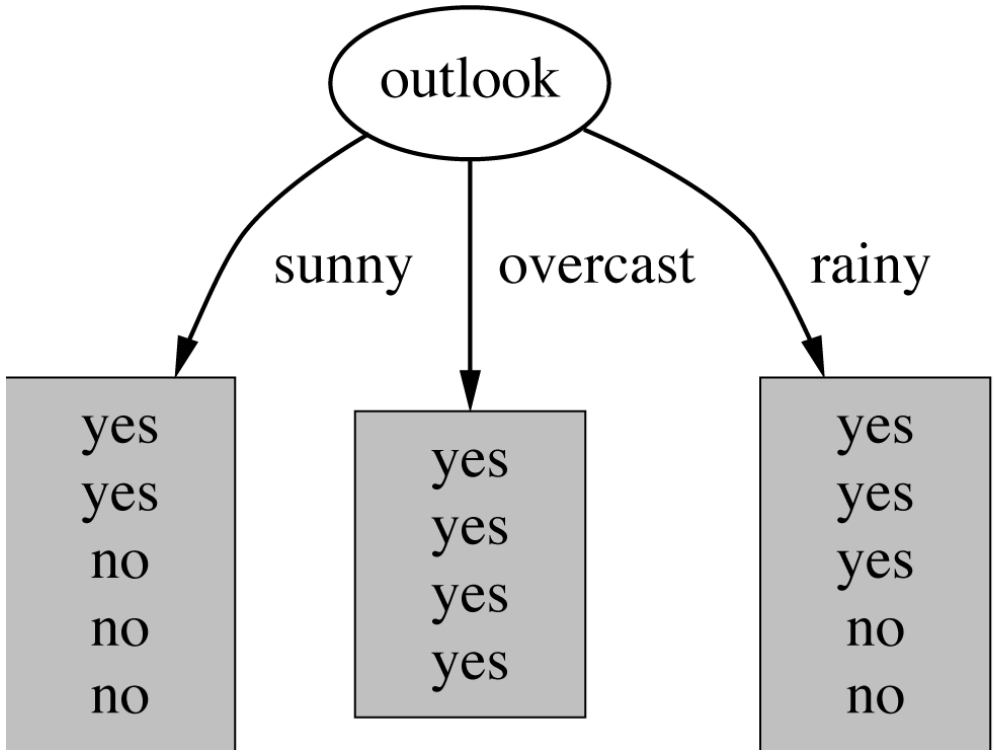
$$Gain(S,Windy) = Entropy(S) - \sum_{v \in \{False,True\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$= Entropy(S) - (8/14).Entropy(S_{False}) - (6/14).Entropy(S_{True})$$

$$= 0.94 - (8/14).(0.81) - (6/14).(1) = \underline{\mathbf{0.048}} \text{ bits}$$

$$\textbf{\textit{Entropy(S) = -(9/14).log}}_2 \textbf{\textit{(9/14)-(5/14).log}}_2\textbf{\textit{(5/14)} } \approx \textbf{\textit{0.94}}$$

Biến Outlook



Entropy của mỗi tập con bị tách do biến Outlook

- "Outlook" = "Sunny"

$$\text{info}([2,3]) = \text{entropy}(2/5,3/5) = -2/5\log(2/5) - 3/5\log(3/5) = 0.971\,\text{bits}$$

- "Outlook" = "Overcast"

$$\text{info}([4,0]) = \text{entropy}(1,0) = -1\log(1) - 0\log(0) = 0\,\text{bits}$$

Chú ý: log(0) không xác định, tuy nhiên ta tính quy ước 0*log(0) là 0

- "Outlook" = "Rainy"

$$\text{info}([3,2]) = \text{entropy}(3/5,2/5) = -3/5\log(3/5) - 2/5\log(2/5) = 0.971\,\text{bits}$$

- Thông tin kỳ vọng của biến Outlook:

$$\text{info}([3,2],[4,0],[3,2]) = (5/14)\times 0.971 + (4/14)\times 0 + (5/14)\times 0.971 = 0.693\,\text{bits}$$

Tính Information Gain

- Information gain=(thông tin trước khi tách) – (thông tin sau khi tách)

$$\text{Gain}(S, \text{Outlook}) = \text{info}([9,5]) - \text{info}([2,3],[4,0],[3,2]) = 0.940 - 0.693$$
$$= 0.247 \text{ bits}$$

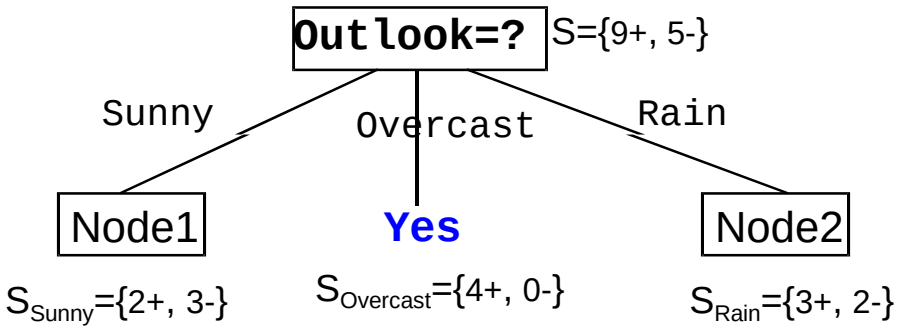
- Tương tự, ta tính được Information gain cho các biến trong tập dữ liệu weather:

$$\text{Gain}(S, \text{Outlook}) = 0.247 \text{ bits}$$
$$\text{Gain}(S, \text{Humidity}) = 0.152 \text{ bits}$$
$$\text{Gain}(S, \text{Temperature}) = 0.029 \text{ bits}$$
$$\text{Gain}(S, \text{Windy}) = 0.048 \text{ bits}$$

- Vậy Outlook là biến được chọn để kiểm tra cho nút gốc vì có Information Gain cao nhất

Tính Information Gain

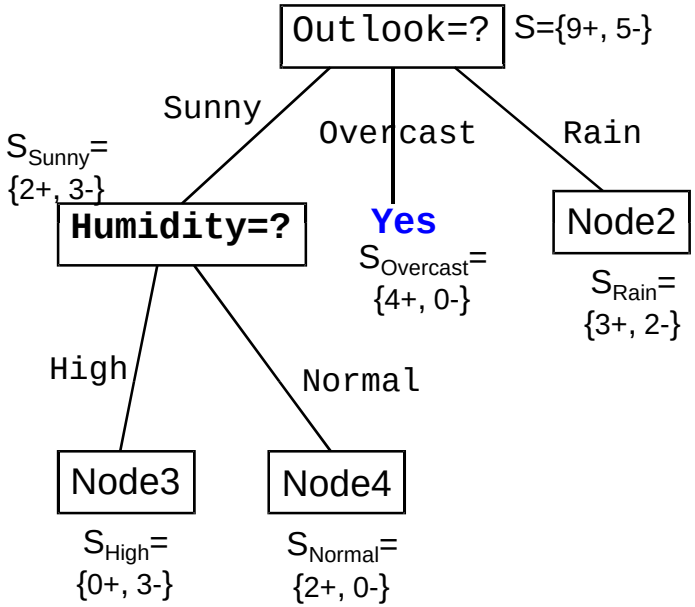
→`Outlook` được chọn là biến kiểm tra tại nút gốc!



Tiếp tục tách nút

- Tại nút Node1, biến nào trong số {Temperature, Humidity, Windy} nên được chọn là biến kiểm tra?
  - **Lưu ý!** Biến Outlook bị loại ra, bởi vì nó đã được sử dụng bởi cha của nút Node1(là nút gốc)
  - Gain(S_{Sunny}, Temperature) =...= 0.57
  - Gain(S_{Sunny}, **Humidity**) = ... = **0.97**
  - Gain(S_{Sunny}, Windy) = ... = 0.019

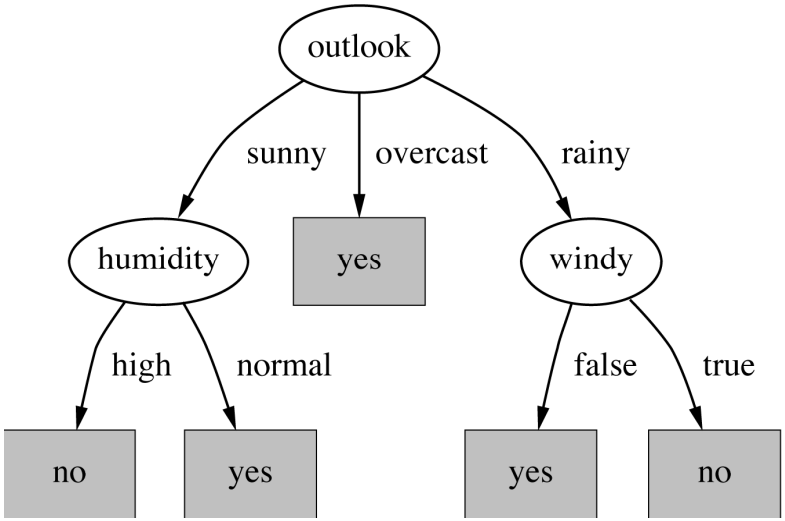
→Vì vậy, Humidity được chọn là biến kiểm tra cho nút Node1!



Điều kiện dừng

- Lượng dữ liệu của 1 nút được gán hầu hết vào 1 lớp
  vd: >90%
- Số lượng mẫu trong tập con tại nút nhỏ hơn 1 giá trị cho trước – ngưỡng (threshold)
- Giảm được Information gain
- Các biến đều đã được kiểm tra

Cây quyết định dựng được



Vấn đề trong ID3

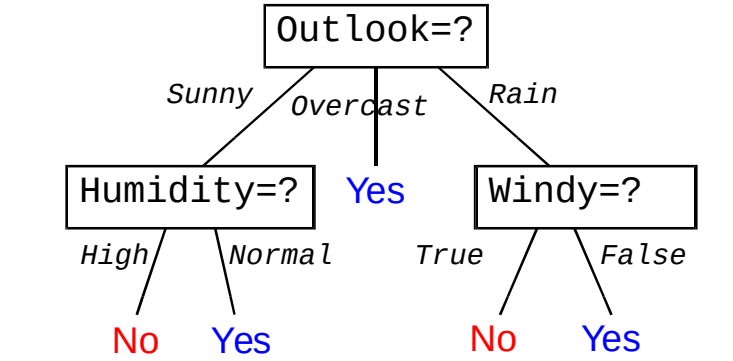
- Cây quyết định học được quá phù hợp (over-fit) với các mẫu
- Xử lý các biến có kiểu giá trị liên tục (kiểu số thực)

Over-fitting trong học cây quyết định

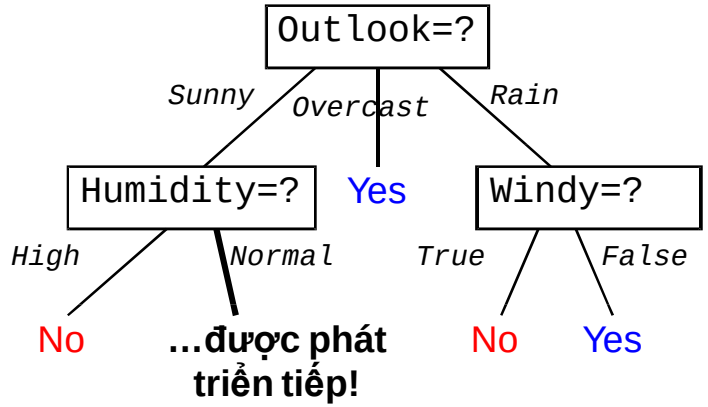
- Một cây quyết định phù hợp hoàn hảo đối với tập huấn luyện có phải là giải pháp tối ưu?
- Nếu như tập huấn luyện có nhiễu/lỗi…?

Vd: Một mẫu nhiễu/lỗi (Mẫu thực sự mang nhãn Yes, nhưng *bị gán nhãn nhầm là No*):

(Outlook=Sunny, Temperature=Hot, Humidity=Normal, Windy=True, PlayTennis=**No**)



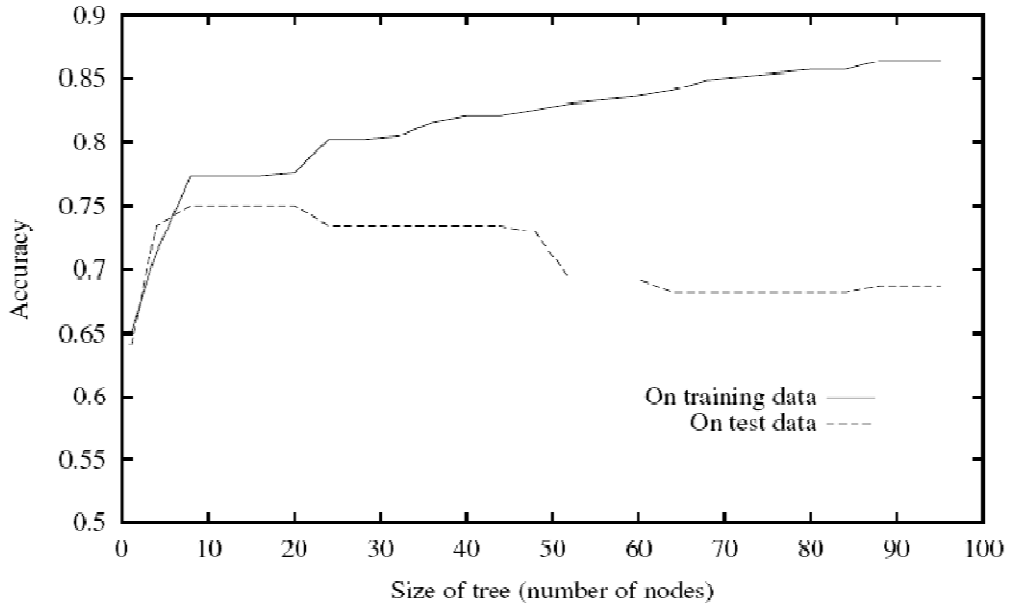
Học được một cây quyết định *phức tạp hơn*!
(chỉ bởi vì mẫu nhiễu/lỗi)



Nguyễn Nhật Quang-Học máy

Over-fitting trong học cây quyết định

Tiếp tục quá trình học cây quyết định sẽ làm giảm độ chính xác
đối với tập thử nghiệm mặc dù tăng độ chính xác đối với tập huấn luyện



[*Mitchell, 1997*]

Giải quyết vấn đề over-fitting

- Hai chiến lược
  - Ngừng việc học (phát triển) cây quyết định sớm hơn, trước khi nó đạt tới cấu trúc cây cho phép phân loại (khớp) hoàn hảo tập huấn luyện
  - Học (phát triển) cây đầy đủ (tương ứng với cấu trúc cây hoàn toàn phù hợp đối với tập huấn luyện), và sau đó thực hiện quá trình tỉa (to post-prune) cây
- Chiến lược tỉa cây đầy đủ (Post-pruning over-fit trees) thường cho hiệu quả tốt hơn trong thực tế

→ Lý do: Chiến lược "ngừng sớm" việc học cây cần phải đánh giá chính xác được *khi nào nên ngừng việc học* (phát triển) cây – Khó xác định!

Các thuộc tính có giá trị liên tục

- Cần xác định (chuyển đổi thành) các thuộc tính có giá trị rời rạc, bằng cách chia khoảng giá trị liên tục thành một tập các khoảng (intervals) không giao nhau
- Đối với thuộc tính (có giá trị liên tục) A, tạo một thuộc tính mới kiểu nhị phân A_v sao cho: A_v là đúng nếu A>v, và là sai nếu ngược lại
- Làm thế nào để xác định giá trị ngưỡng v "tốt nhất"?

  → Chọn giá trị ngưỡng v giúp sinh ra giá trị *Information Gain* cao nhất
- Ví dụ:
  - Sắp xếp các mẫu theo giá trị tăng dần đối với thuộc tính Temperature
  - Xác định các mẫu liền kề nhưng khác phân lớp
  - Có 2 giá trị ngưỡng có thể: $Temperature_{54}$ và $Temperature_{85}$
  - Thuộc tính mới kiểu nhị phân $Temperature_{54}$ được chọn, bởi vì:

  $Gain(S,Temperature_{54}) > Gain(S,Temperature_{85})$

Temperature	40	48	60	72	80	90
PlayTennis	No	No	Yes	Yes	Yes	No

Cây phân loại và hồi quy
Classification **a**nd **R**egression **T**rees (CART)

Xây dựng cây CART thế nào?

Có 2 dạng:

1. Hồi quy
2. Phân loại (lớp)

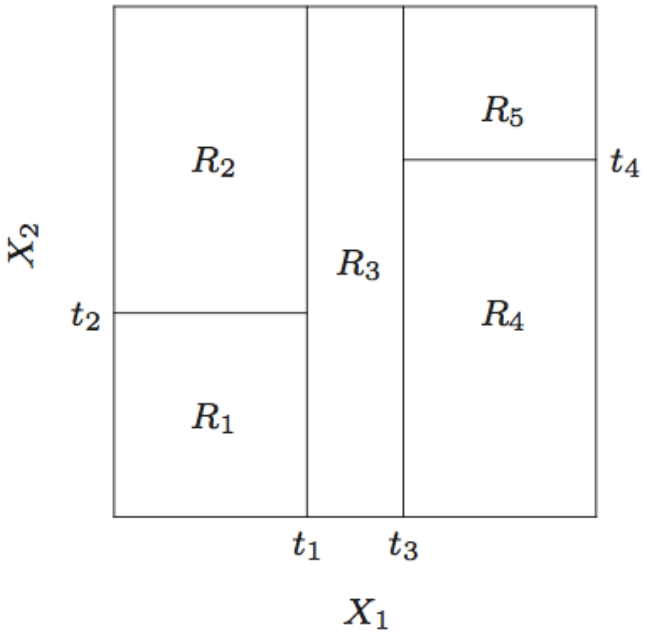
Mô hình liên tục từng đoạn(piecewise)

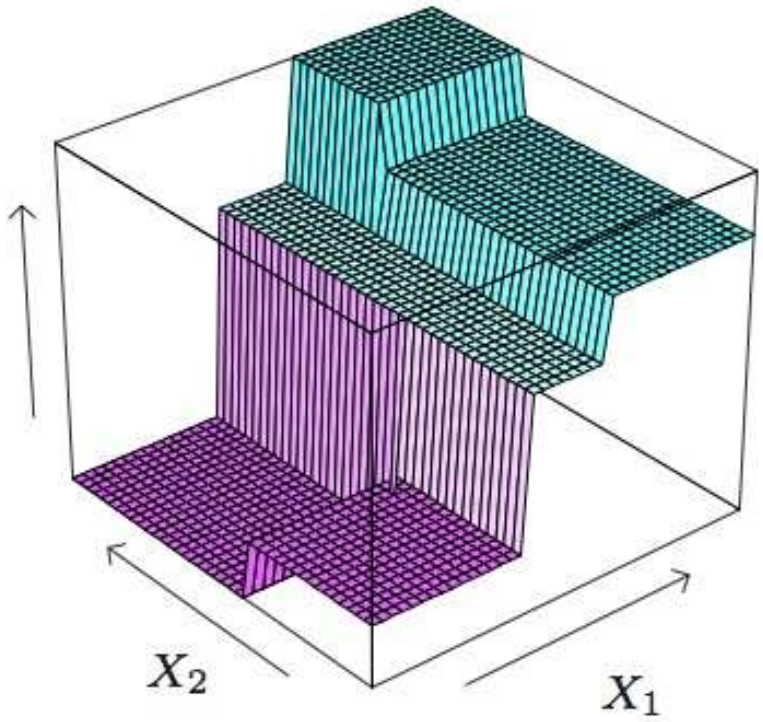
- Dự đoán liên tục trong mỗi vùng

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Mô hình liên tục từng đoạn





Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Minh họa cây CART

1 TYPE OF HOME
1. House
2. Condominium
3. Apartment
4. Mobile Home
5. Other

2 SEX
1. Male
2. Female

3 MARITAL STATUS
1. Married
2. Living together, not married
3. Divorced or separated
4. Widowed
5. Single, never married

4 AGE
1. 14 thru 17
2. 18 thru 24
3. 25 thru 34
4. 35 thru 44

5 EDUCATION
1. Grade 8 or less
2. Grades 9 to 11
3. Graduated high school
4. 1 to 3 years of college
5. College graduate
6. Grad Study

6 OCCUPATION
1. Professional/Managerial
2. Sales Worker
3. Factory Worker/Laborer/Driver
4. Clerical/Service Worker
5. Homemaker
6. Student, HS or College
7. Military
8. Retired
9. Unemployed

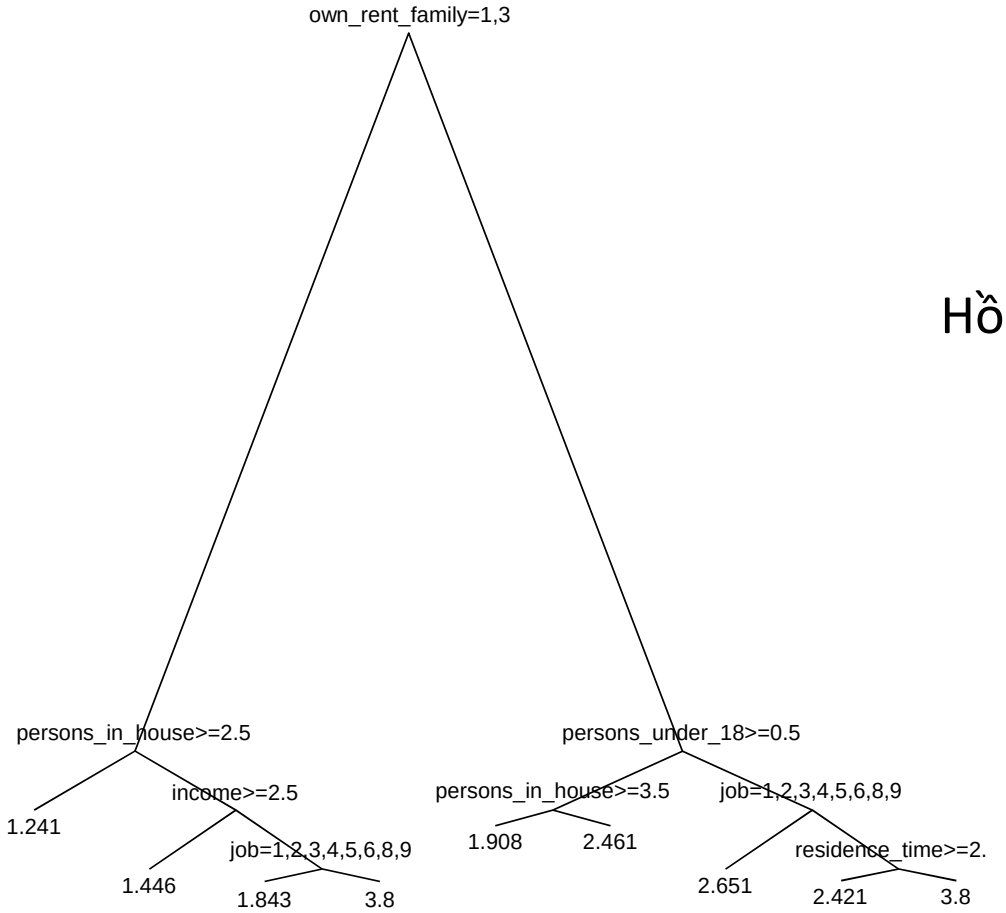
7 ANNUAL INCOME OF HOUSEHOLD (PERSONAL INCOME IF SINGLE
1. Less than $10,000
2. $10,000 to $14,999
3. $15,000 to $19,999
4. $20,000 to $24,999
5. $25,000 to $29,999

8 HOW LONG HAVE YOU LIVED IN THE SAN FRAN./OAKLAND/SAN JOSE AREA?
1. Less than one year
2. One to three years
3. Four to six years
4. Seven to ten years
5. More than ten years

9 DUAL INCOMES (IF MARRIED)
1. Not Married
2. Yes
3. No

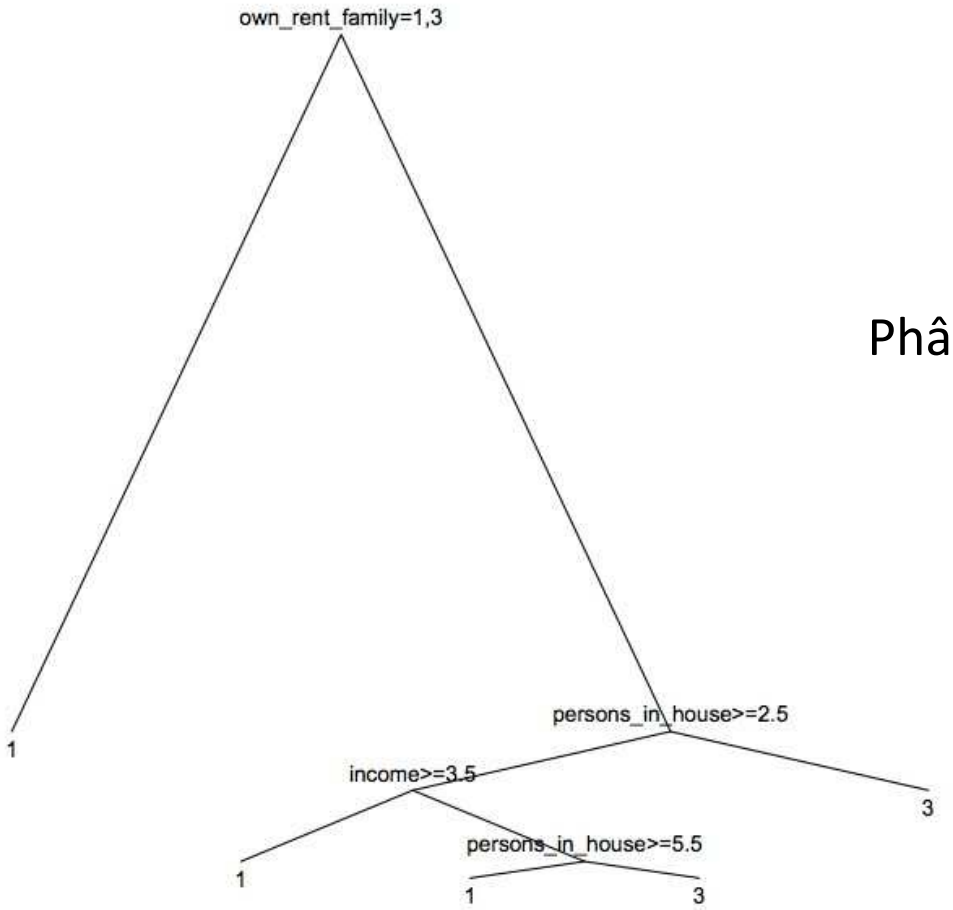
10 PERSONS IN YOUR HOUSEHOLD
1. One
2. Two
3. Three
4. Four
5. Five
6. Six
7. Seven
8. Eight
9. Nine or more

Minh họa cây CART



Hồi quy

Minh họa cây CART



Phân lớp

Cây hồi quy

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

$$\hat{c}_m = \text{ave}(y_i | x_i \in R_m)$$

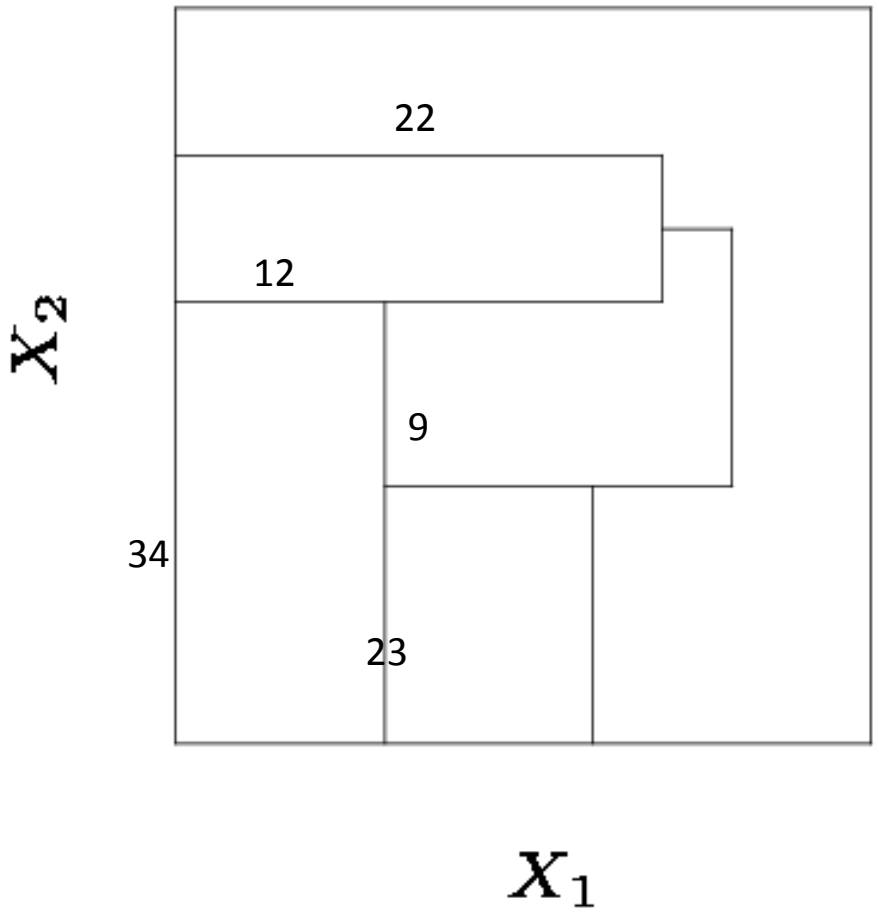
Giá trị dự đoán lưu tại lá của cây hồi quy. Nó được tính bằng giá trị trung bình của tất cả các mẫu (bản ghi) tại lá đó.

Cây hồi quy

- Giả sử ta có 2 vùng R_1 và R_2 với $\hat{Y}_1 = 10, \hat{Y}_2 = 20$
- Với các giá trị của X mà $X \in R_1$ ta sẽ có giá trị dự đoán là 10, ngược lại $X \in R_2$ ta có kết quả dự đoán là 20.

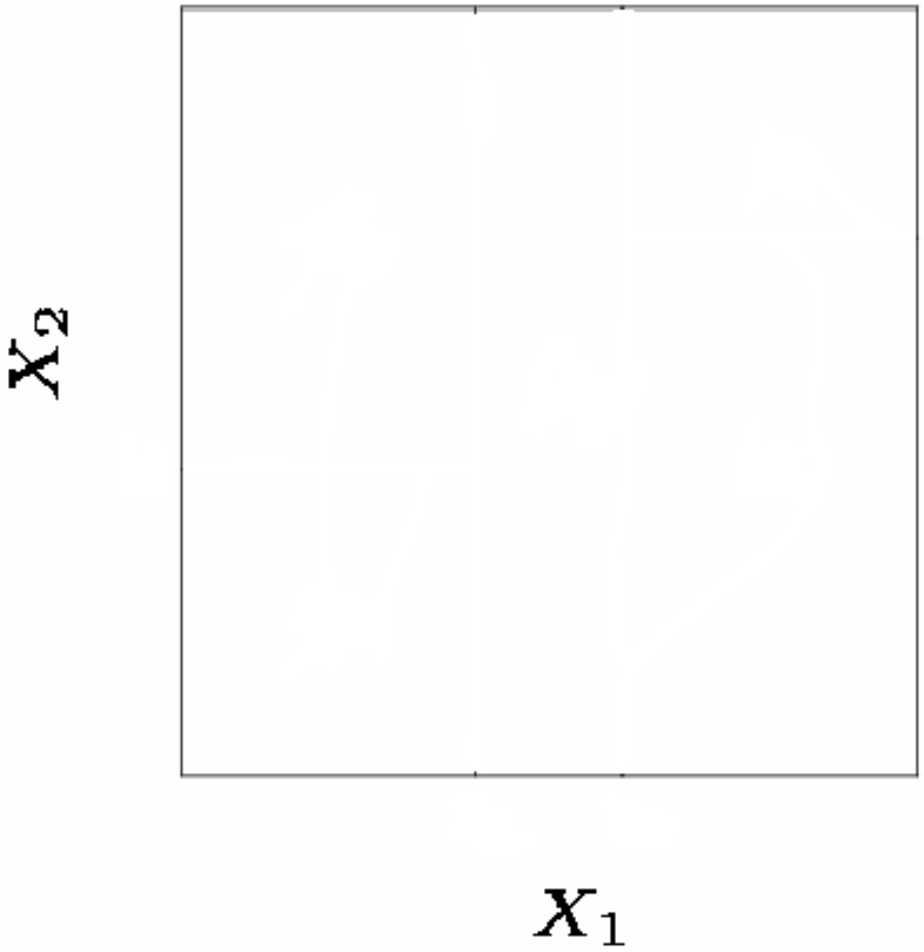
Cây hồi quy

- Cho 2 biến đầu vào và 5 vùng
- Tùy theo từng vùng của giá trị mới X ta sẽ có dự đoán 1 trong 5 giá trị cho Y.



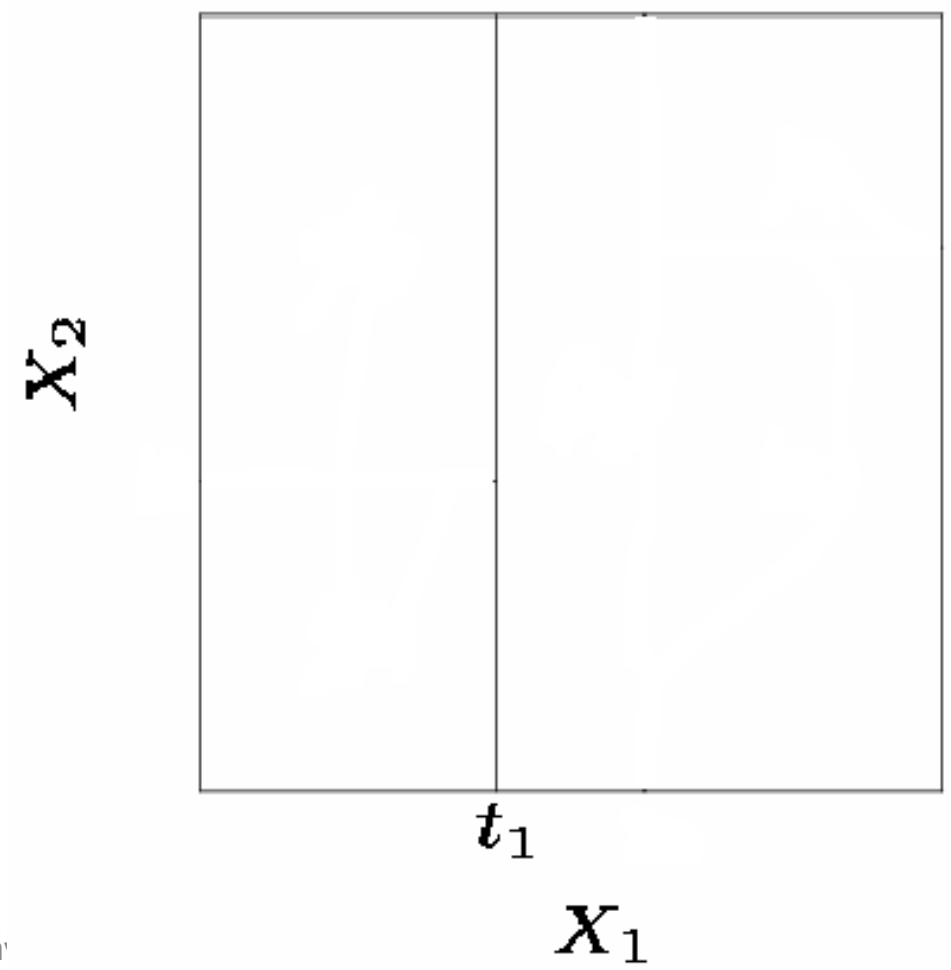
Tách các biến X

Ta tạo ra các phân vùng bằng cách tách lặp đi lặp lại một trong các biến X thành hai vùng



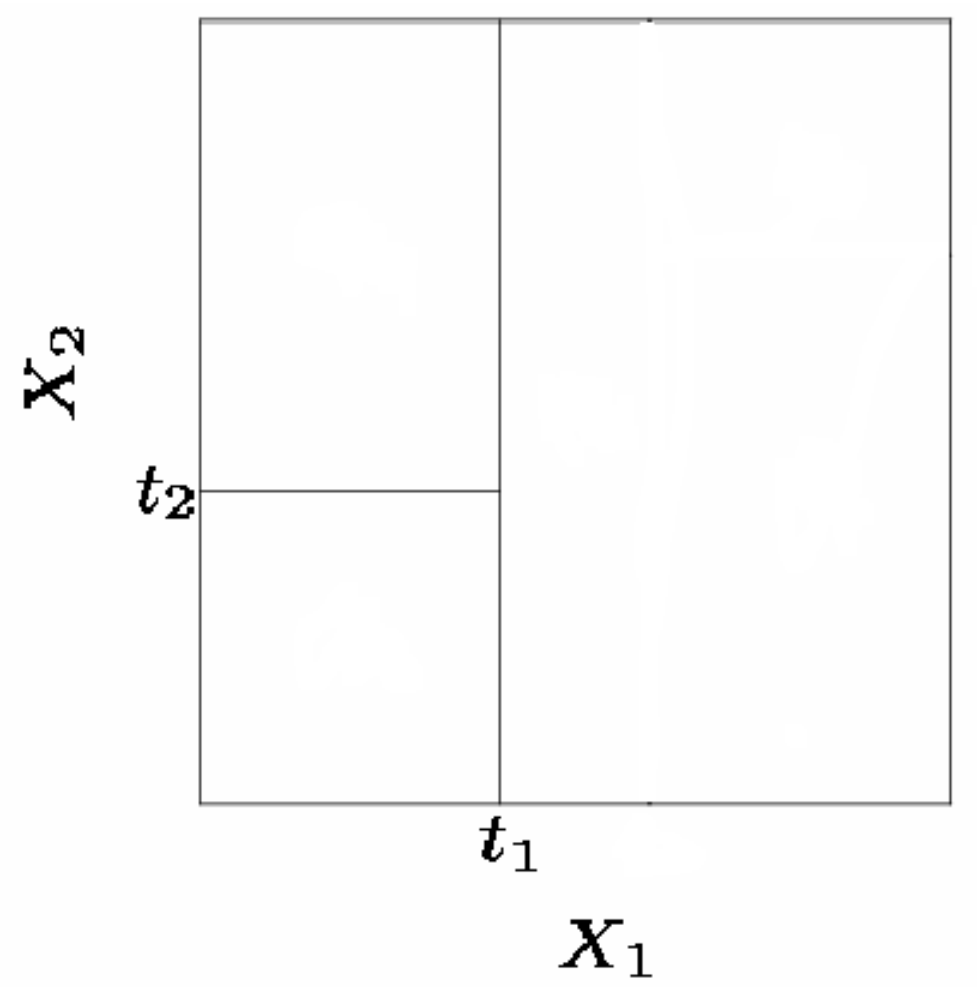
Tách các biến X

1. Đầu tiên tách trên $X_1=t_1$



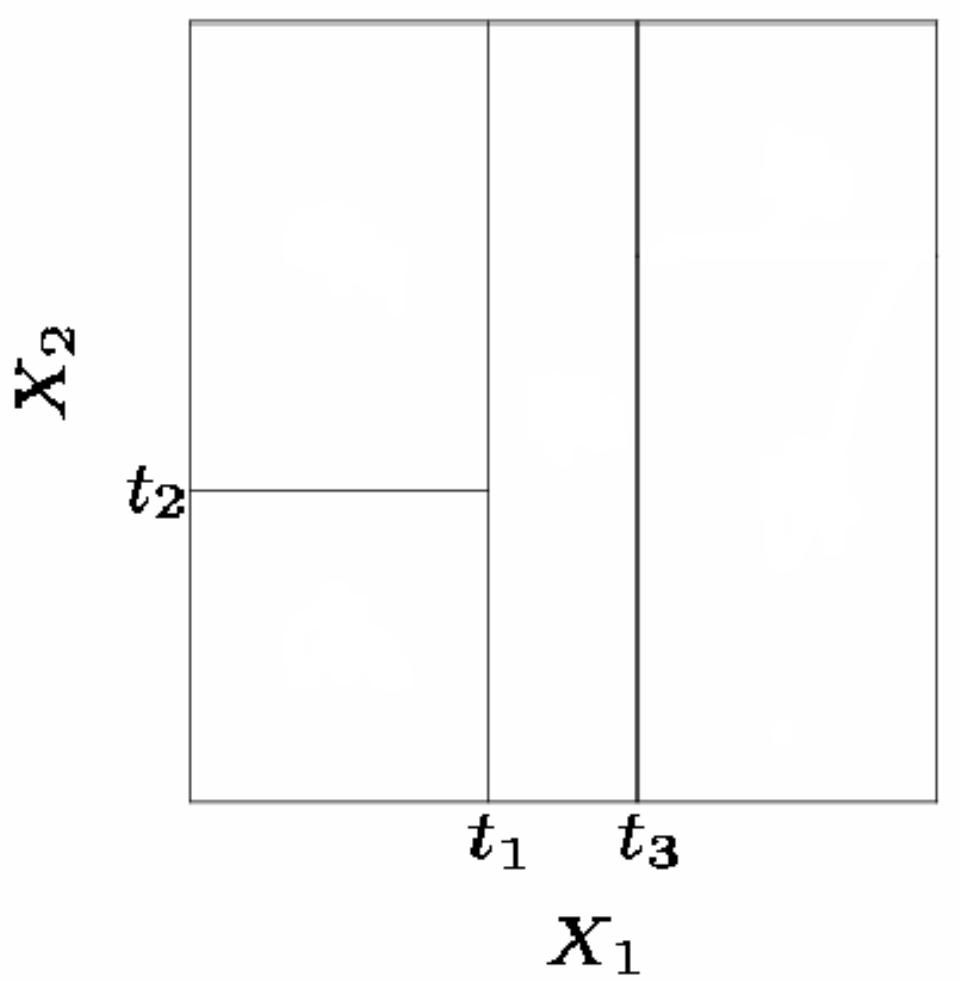
Tách các biến X

1. Đầu tiên tách trên $X_1=t_1$
2. Nếu $X_1<t_1$, tách trên $X_2=t_2$



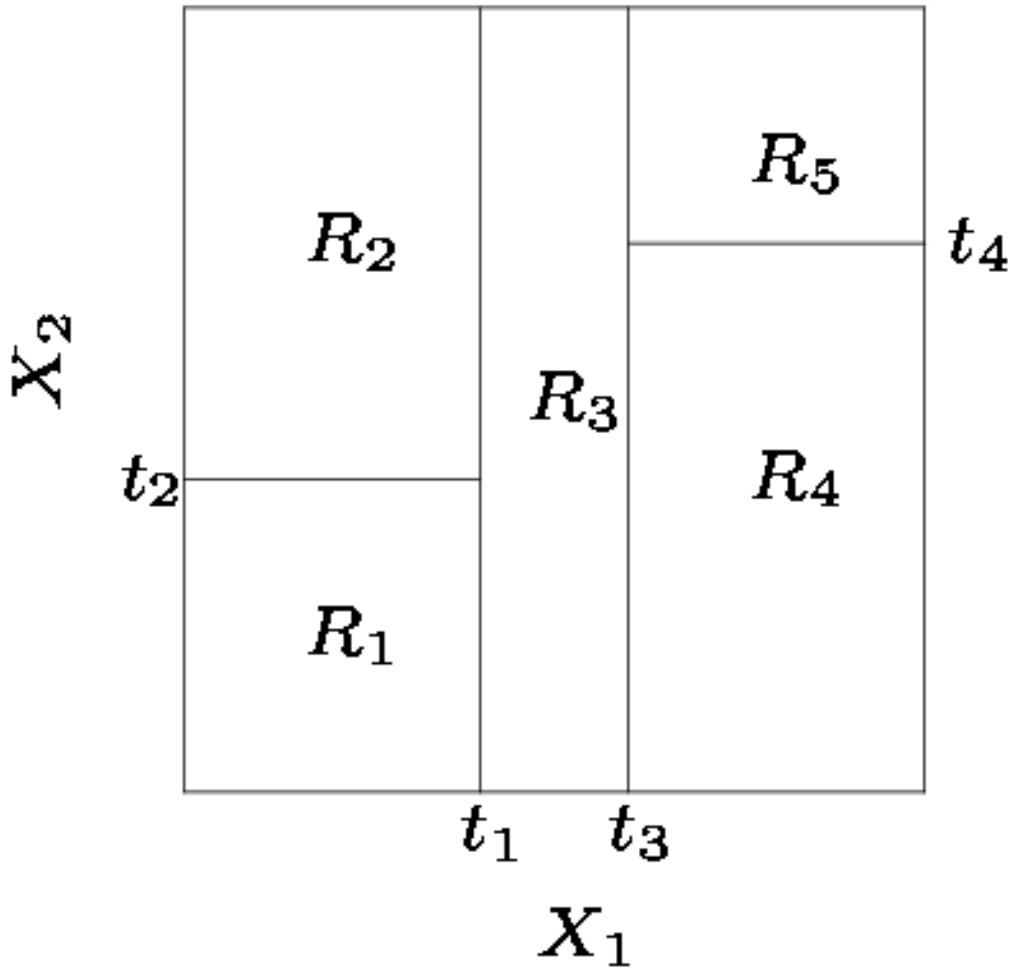
Tách các biến X

1. Đầu tiên tách trên $X_1=t_1$
2. Nếu $X_1<t_1$, tách trên $X_2=t_2$
3. Nếu $X_1>t_1$, tách trên $X_1=t_3$

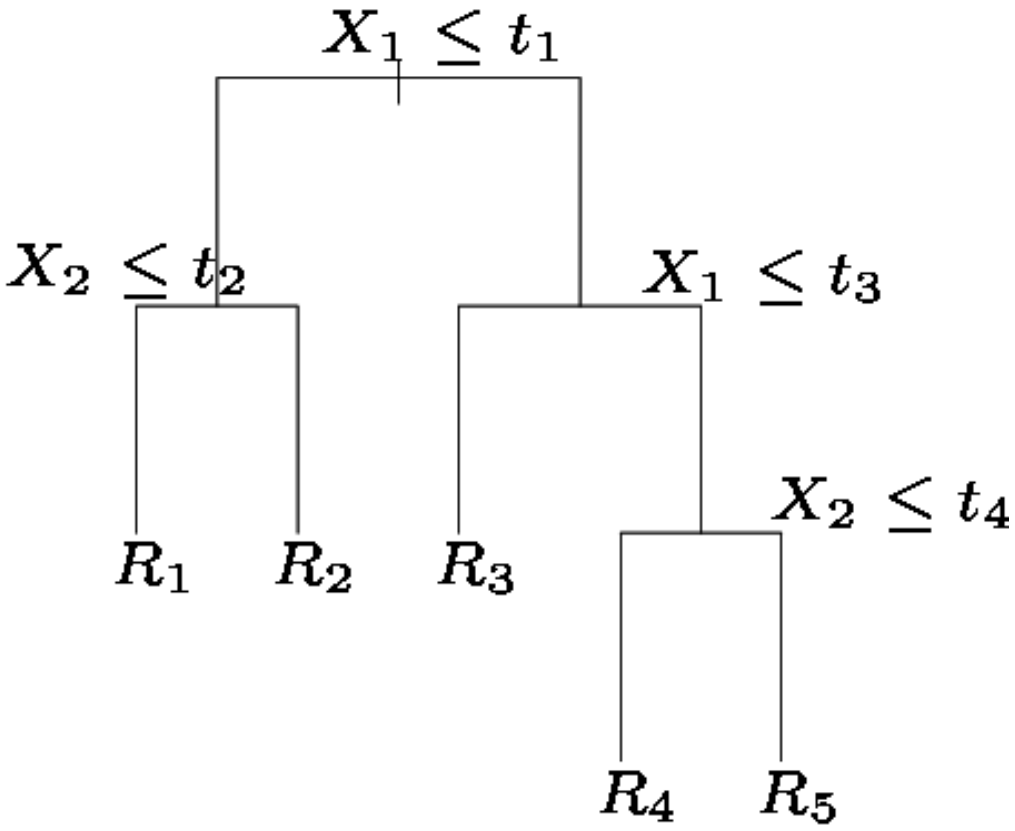


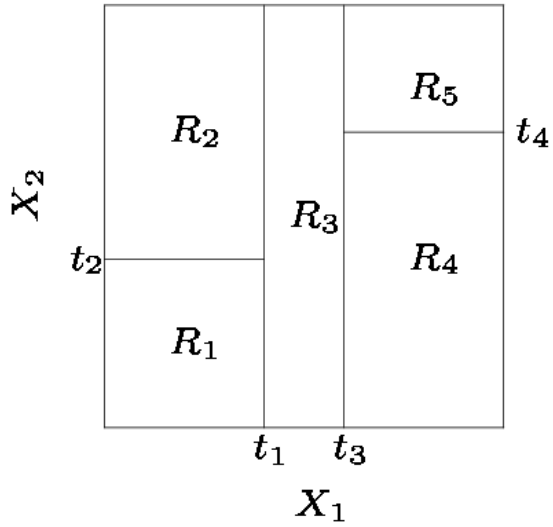
Tách các biến X

1. Đầu tiên tách trên $X_1=t_1$
2. Nếu $X_1<t_1$, tách trên $X_2=t_2$
3. Nếu $X_1>t_1$, tách trên $X_1=t_3$
4. Nếu $X_1>t_3$, tách $X_2=t_4$



Tách các biến X





- Khi ta tạo các vùng theo phương pháp này, ta có thể biểu diễn chúng dùng cấu trúc cây.
- Phương pháp này dễ diễn giải mô hình dự đoán, dễ diễn giải kết quả

Giải thuật tham lam: hồi quy

- Tìm thuộc tính tách j và điểm tách s mà nó cực tiểu lỗi dự đoán

$$\min_{j,\ s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right]$$

Cây phân lớp

$$\text{class } k(m) = \arg\max_k \hat{p}_{mk}$$

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

Giải thuật tham lam: phân lớp

- Nhiều độ đo cho lỗi dự đoán (độ hỗn tạp của nút-node impurity)

Misclassification error: $\frac{1}{N_m} \sum_{i \in R_m} I(y_i \neq k(m)) = 1 - \hat{p}_{mk(m)}$

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Giải thuật tham lam: phân lớp

- Nhiều độ đo cho lỗi dự đoán (độ hỗn tạp của nút-node impurity)

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

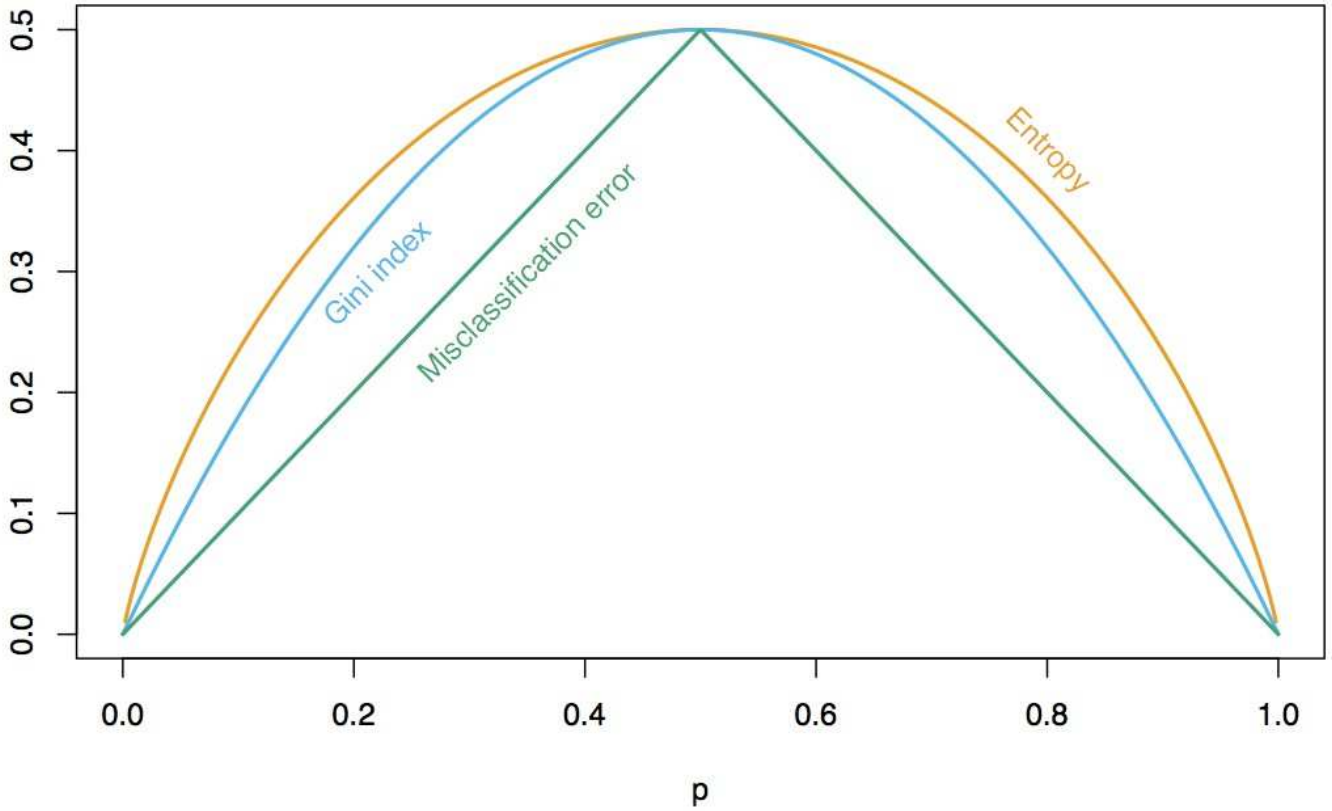
Gini index:

$$\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Độ hỗn tạp của nút khi phân lớp

Classification node impurity



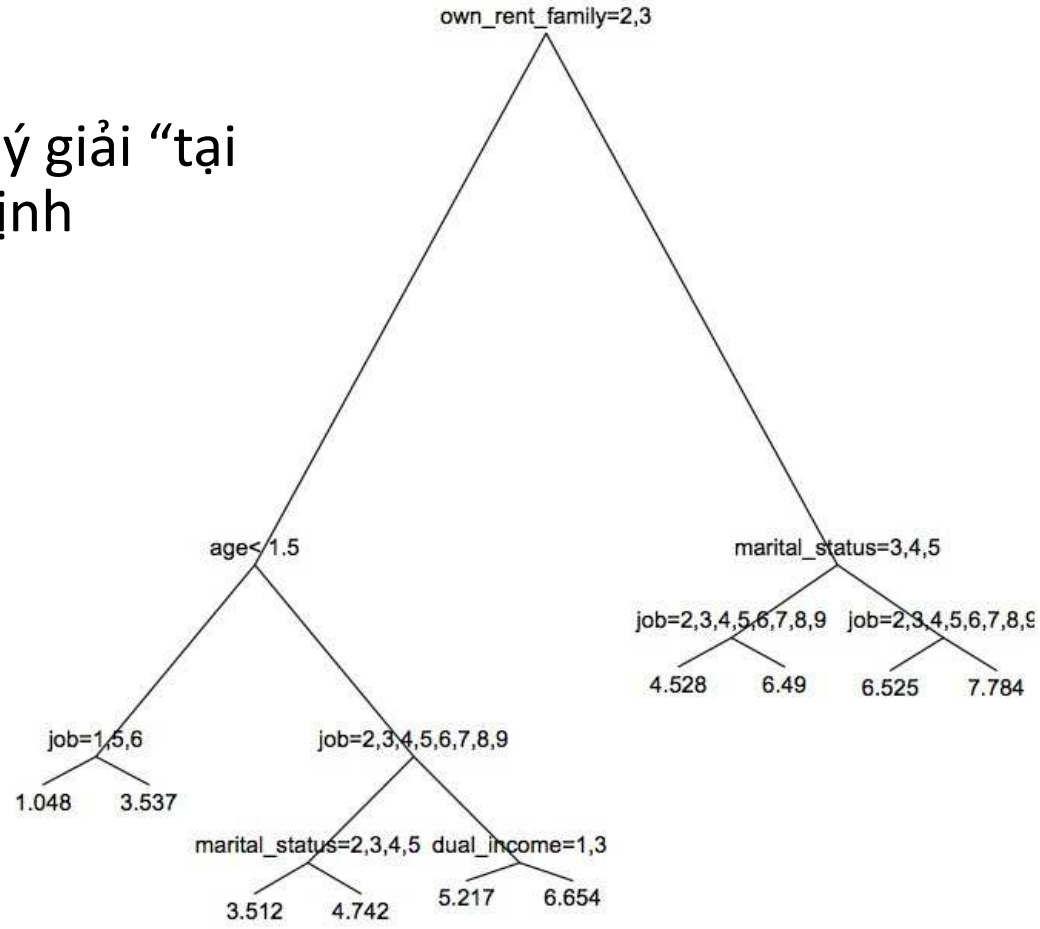
Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.

Ưu điểm của CART

- Dễ xử lý dữ liệu thiếu (surrogate splits)
- Mạnh trong xử lý dữ liệu chứa thông tin rác (non-informative data)
- Cho phép tự động lựa chọn thuộc tính (variable selection)
- Dễ giải thích, lý tưởng để giải thích "tại sao" đối với người ra quyết định
- Xử lý được tính tương tác cao giữa các thuộc tính

Ưu điểm của CART

Dễ giải thích, lý tưởng để lý giải "tại sao" cho người ra quyết định



Ưu điểm của CART

Xử lý được tính tương tác cao giữa các thuộc tính

$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \ \ldots$

$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \theta_1 x_1 x_2 + \theta_2 x_1 x_3 + \theta_3 x_2 x_3 + \lambda_1 x_1 x_2 x_3 \ldots$

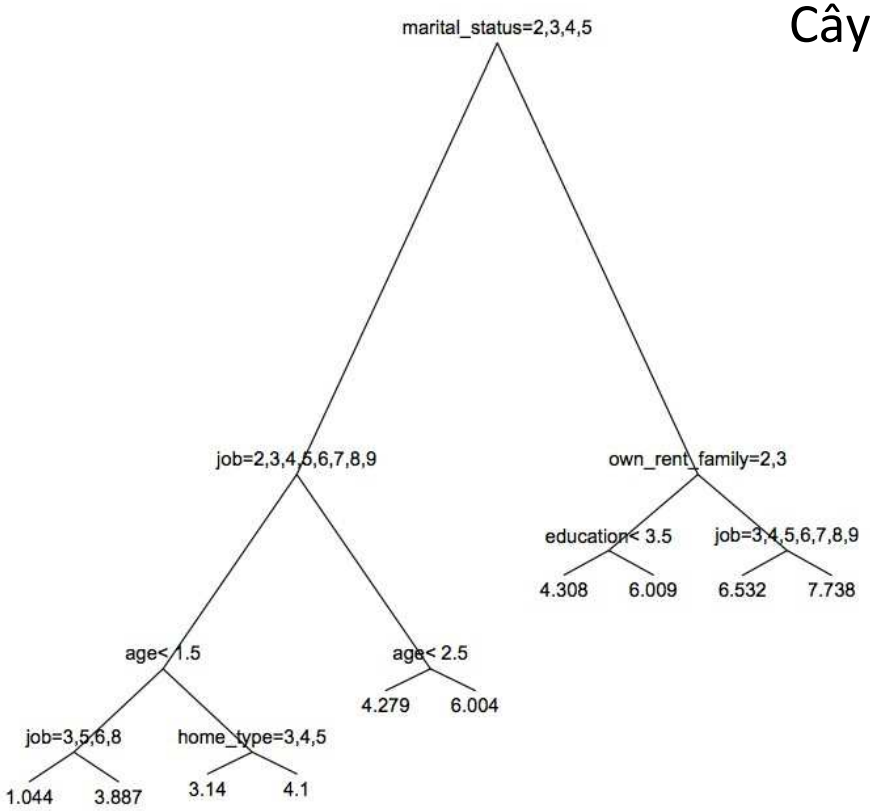
Y = 3.5 if ((1<marital_status<6) AND (1<job<9)) AND (age<1.5) OR …

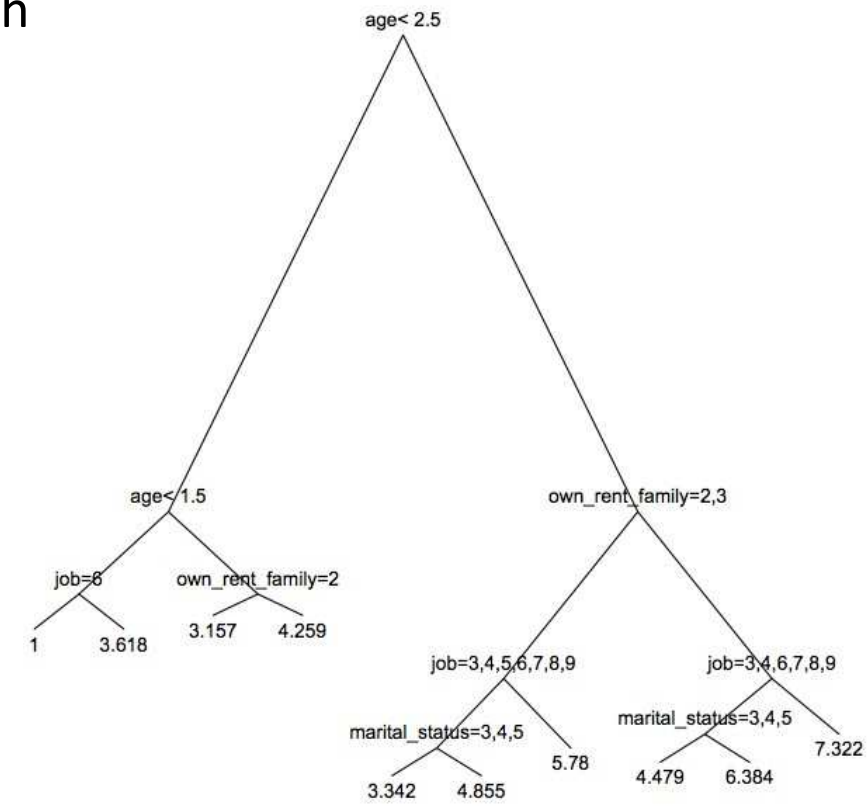
Nhược điểm của CART

- Cây không ổn định (Instability of trees)
- Thiếu tính trơn (Lack of smoothness)
- Khó nắm bắt độ cộng tính (Hard to capture additivity)

Nhược điểm của CART

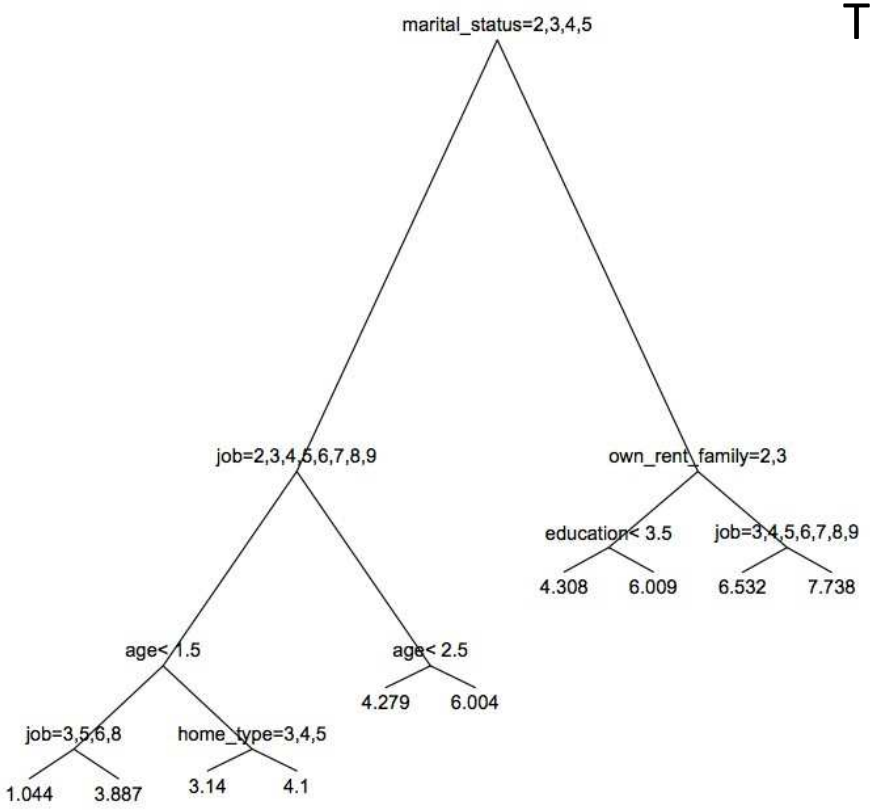
Cây không ổn định

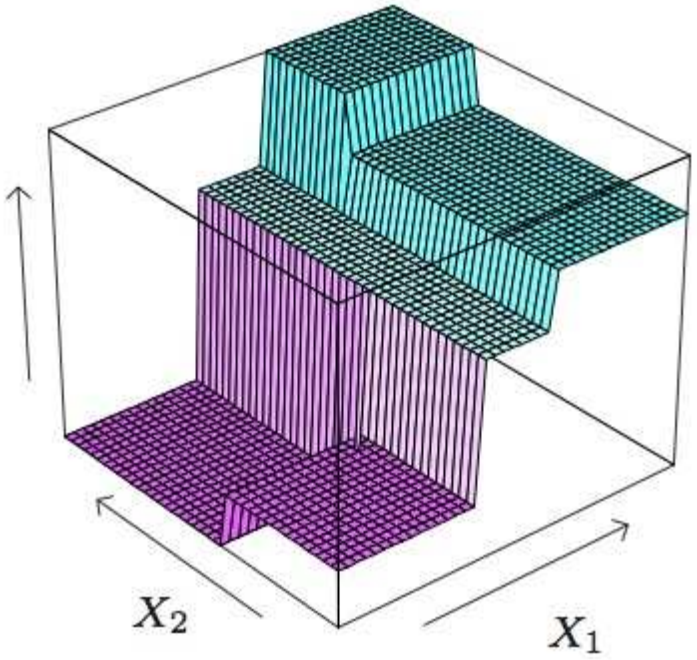




Nhược điểm của CART

Thiếu tính trơn (Smoothness)

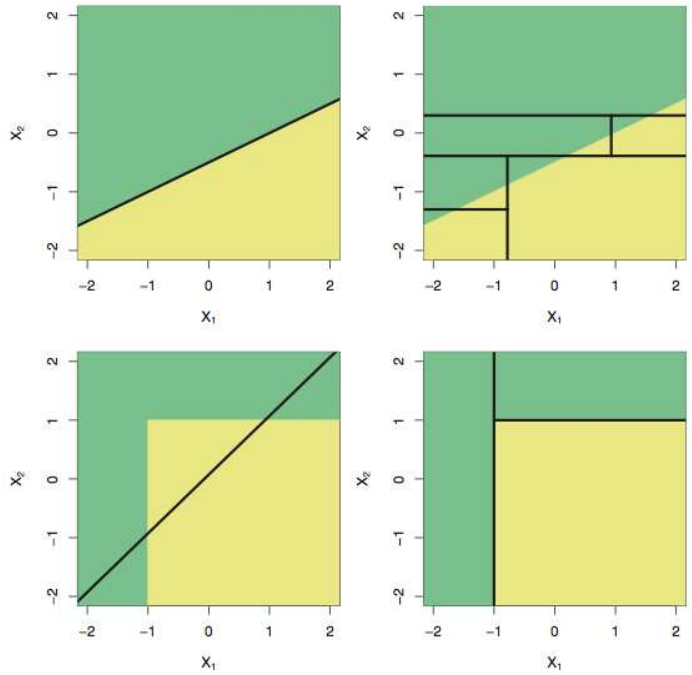




Nhược điểm của CART

Khó nắm bắt độ cộng tính (additivity)

$$Y = c_1 I(X_1 < t_1) + c_2 I(X_2 < t_2) + e$$



Hastie, Trevor, et al. Introduction to statistical learning.

Nhược điểm của CART

1. Cây không ổn định

Giải pháp – Random Forests

2. Thiếu tính trơn

Giải pháp – MARS

MARS – "Multivariate Adaptive Regression Splines"

3. Khó nắm bắt độ cộng tính (additivity)

Giải pháp – MART or *MARS*

MART – "Multiple Additive Regression Trees"

Câu hỏi?

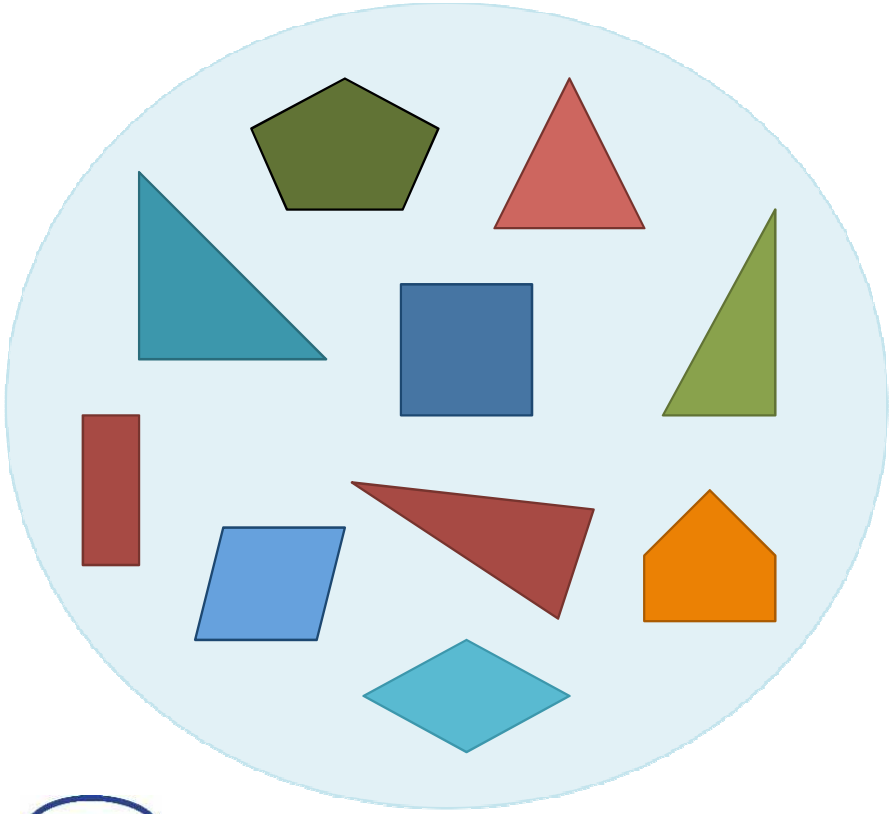
Đánh giá hiệu quả bộ phân lớp

Phân lớp

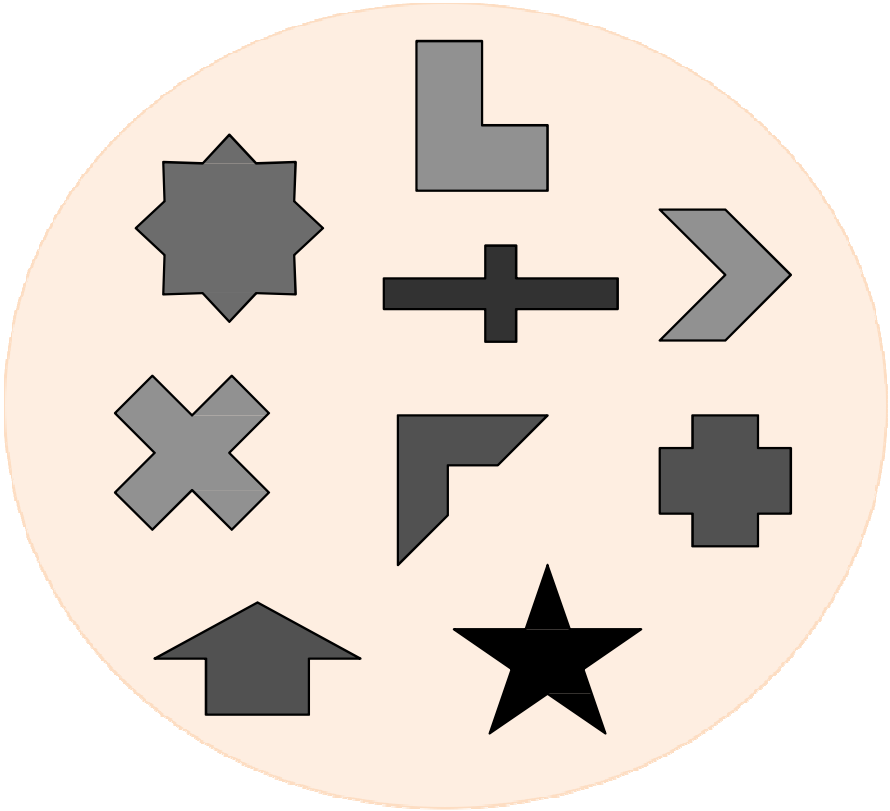
- Học có giám sát: Học từ các mẫu đã gán nhãn
- Biến đích có dạng rời rạc / hạng mục
- Mục tiêu: dự đoán biến đích có kiểu rời rạc
  - Gán mỗi mẫu cho 1 lớp
  - Các bài trước: K–NN, hồi quy logistic
  - Hôm nay: SVM

Học từ mẫu đã gán nhãn

Lớp "+"

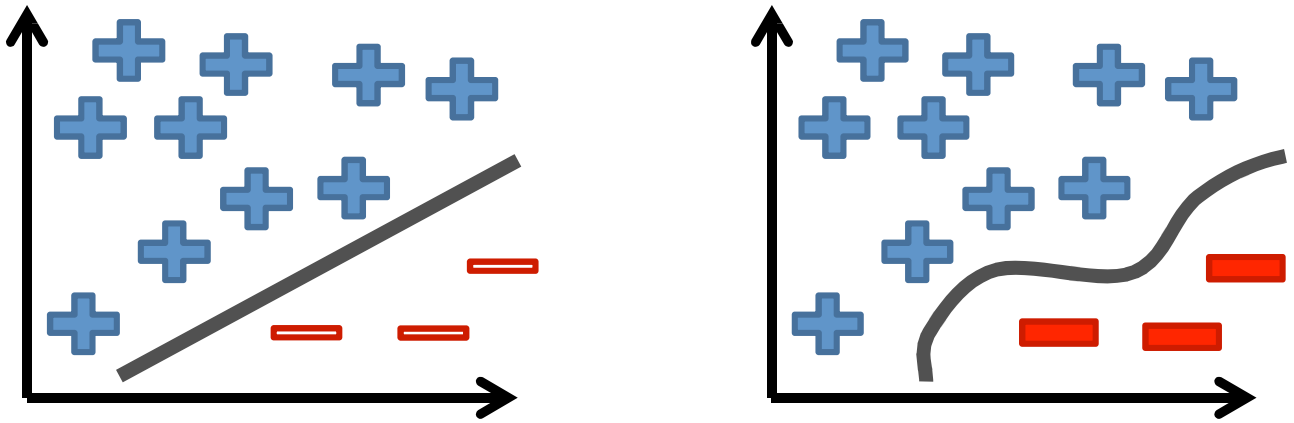


Lớp "–"



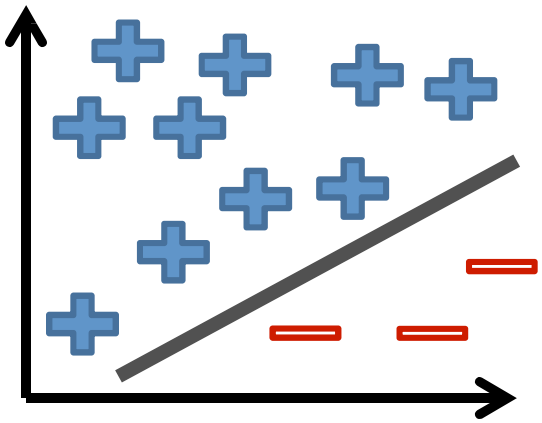
Nhãn mất cân bằng

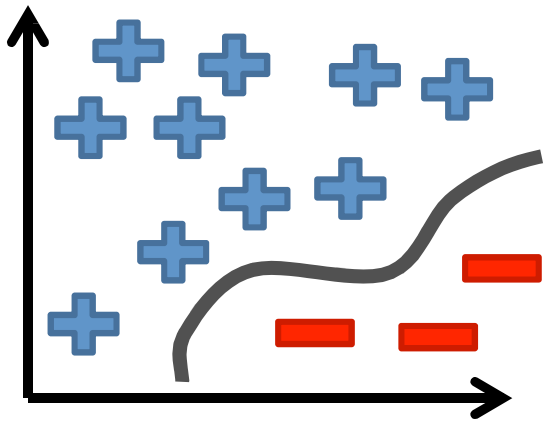
- Nhãn mất cân bằng (Imbalanced classes): lớp dương (+) xuất hiện với tần suất nhiều hơn lớp âm (–) trong tập dữ liệu huấn luyện
  - vd: phát hiện gian lận, dữ liệu y học



Nhãn mất cân bằng

- Tại sao đây là vấn đề?
  - Các thuật toán thực hiện tốt khi huấn luyện trên các mẫu trong mỗi lớp
  - Hiệu quả thấp trên các lớp có ít đại diện





Đánh giá hiệu quả bộ phân lớp

- Trong bài toán hồi quy, chúng ta dùng tổng phần dư bình phương đo lỗi để đánh giá hiệu quả thuật toán
- Với bài toán phân lớp, chúng ta cần độ đo để đánh giá hiệu quả của bộ phân lớp
  - Ví dụ: Ma trận nhầm lẫn (Confusion matrix), Độ chính xác/Hồi tưởng (Precision/Recall), Độ nhạy/Độ đặc hiệu (Sensitivity/ Specificity), Đường cong ROC (ROC curve)
- Xét bài toán phân lớp nhị phân: Có 2 lớp (+) và (–)

Đánh giá hiệu quả bộ phân lớp

- Ta có thể biểu thị tính hiệu quả của bộ phân lớp trong 1 bảng gọi là *ma trận nhầm lẫn* (*confusion matrix)*:
  – "Hiệu quả tốt": True Positive (TP), True Negative (TN) lớn và False Positive (FP), False Negative (FN) nhỏ
  - TP: Số lượng các mẫu thuộc lớp (+) được phân loại chính xác vào lớp (+)
  - FP: Số lượng các mẫu không thuộc lớp (+) bị phân loại nhầm vào lớp (+)

Đánh giá hiệu quả bộ phân lớp

- TN: Số lượng các mẫu không thuộc lớp (+) được phân loại đúng
- FN: Số lượng các mẫu thuộc lớp (+) bị phân loại nhầm (vào các lớp khác lớp (+))

		Lớp dự đoán (Predicted class)	
		+	**–**
Lớp thực (True class)	**+**	True Positive-TP	False Negative-FN Type II error
	–	(False Positive-FP) Type I error	True Negative-TN

Đánh giá hiệu quả bộ phân lớp

True positive rate (TPR)

(recall, sensitivity)

Predicted class

True class	+	−
+	**TP**	**FN**
−	FP	TN

$$TPR = \frac{TP}{TP + FN}$$

Positive predictive value (PPV) *(precision)*

Predicted class

True class	+	−
+	**TP**	FN
−	**FP**	TN

$$PPV = \frac{TP}{TP + FP}$$

False positive rate (FPR)

Predicted class

True class	+	−
+	TP	FN
−	**FP**	**TN**

$$FPR = \frac{FP}{FP + TN}$$

True negative rate (SPC)

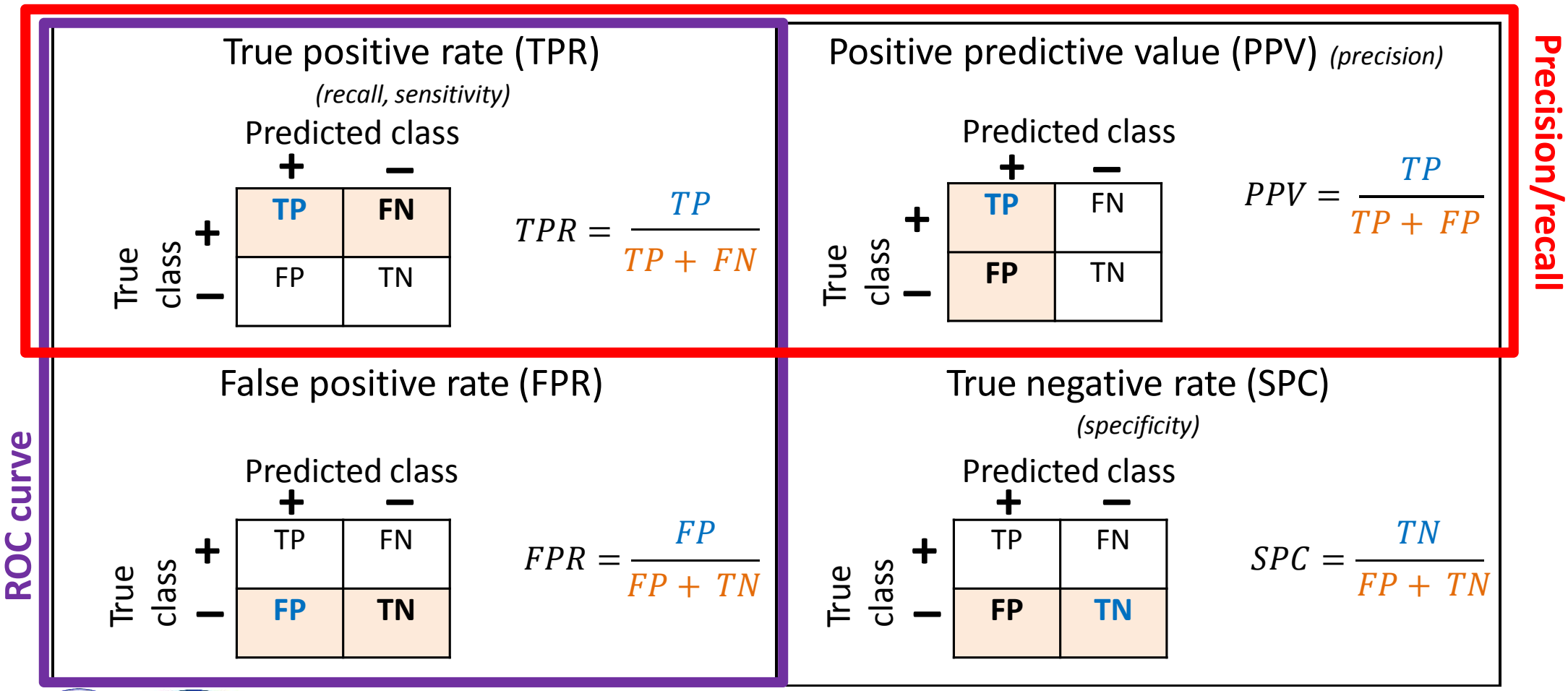
(specificity)

Predicted class

True class	+	−
+	TP	FN
−	**FP**	**TN**

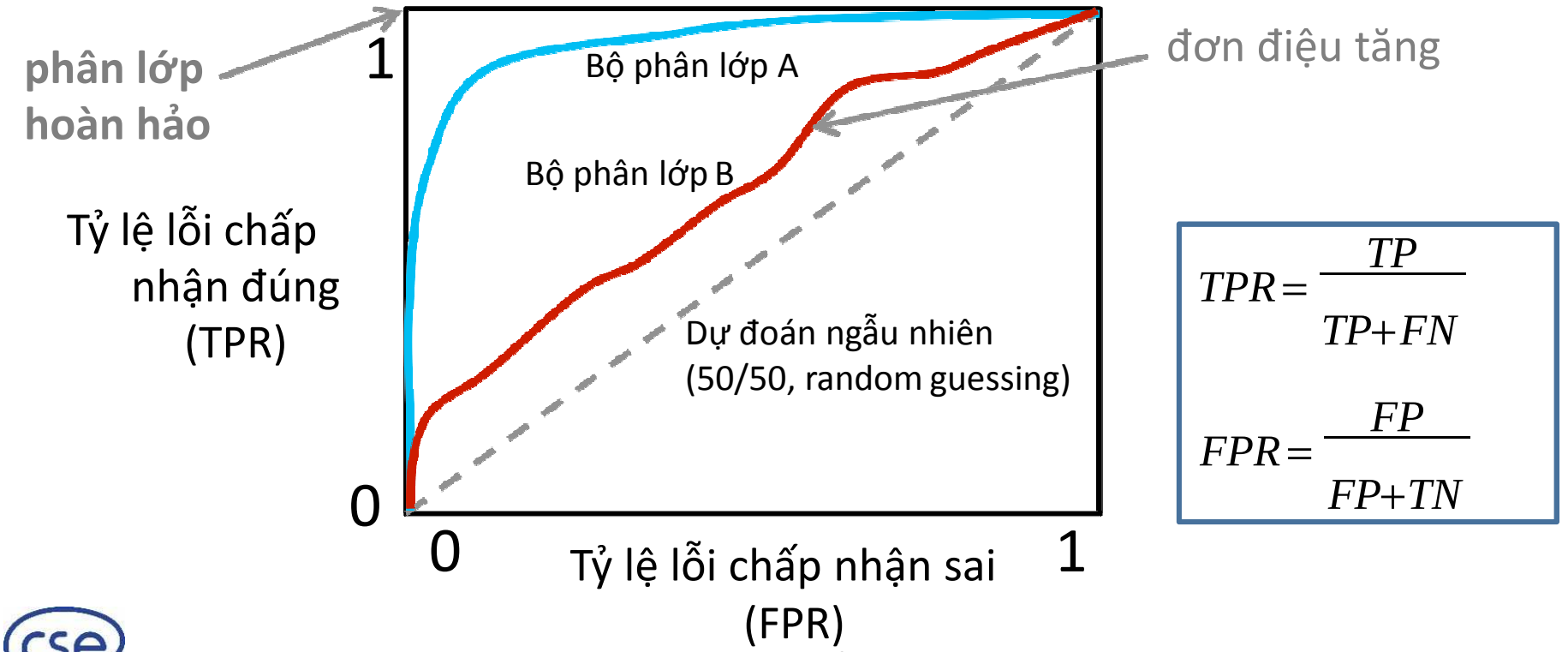
$$SPC = \frac{TN}{FP + TN}$$

Đánh giá hiệu quả bộ phân lớp



Đánh giá hiệu quả bộ phân lớp

- Đường cong ROC (receiver operating characteristic)



$$TPR = \frac{TP}{TP+FN}$$

$$FPR = \frac{FP}{FP+TN}$$

Đánh giá hiệu quả bộ phân lớp

- Nhược điểm của đường cong ROC
  - ROC không biểu thị đúng độ mất cân bằng các mẫu trong lớp thực
  - vd: Xét bộ dữ liệu có 1% mẫu thuộc lớp "+" và 99% mẫu thuộc lớp "–"
    - Giả sử ta nhận được kết quả phân lớp như sau:

      TPR = 0.9 và FPR = 0.12
    - TPR và FPR không hiểu thị được theo tính chất của đường cong ROC

		Predicted class +	Predicted class –
True class	+	90	10
True class	–	1188	8712

Đánh giá hiệu quả bộ phân lớp

- Độ chính xác/Triệu hồi (Precision/recall)
  - Độ chính xác (Positive predictive value): $PPV = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$
    - Tỷ lệ phần trăm của số mẫu thuộc lớp (+) được dự đoán đúng trên số mẫu thực là (+)
  - Recall (True positive rate): $TPR = \frac{\mathrm{TP}}{\mathrm{TP+FN}} = \frac{\mathrm{TP}}{\mathrm{P}}$
    - Tỷ lệ các mẫu (+) phân lớp chính xác lớp (+)
  - Recall và precision tỷ lệ nghịch với nhau
  - Với bộ phân lớp hoàn hảo, Recall = 1, Precision = 1
  - VD phân lớp mất cân bằng: Recall = 0.9, Precision = 0.07

		Predicted class +	Predicted class −
True class	+	90	10
True class	−	1188	8712