DECISION TREE

DECISION TREE

Decision tree

- Là một thuật toán học máy dùng để phân loại hoặc dự đoán kết quả dựa trên một loạt các đặc trưng.
- Được áp dụng để giải quyết:
  - Phân lớp
  - Hồi quy
- Có thể xử lý đồng thời ba dạng dữ liệu:
  - Rời rạc: Categorical data
  - Liên tục: Continuous/Numeric data
  - Hỗn hợp: Mixed data

Decision tree

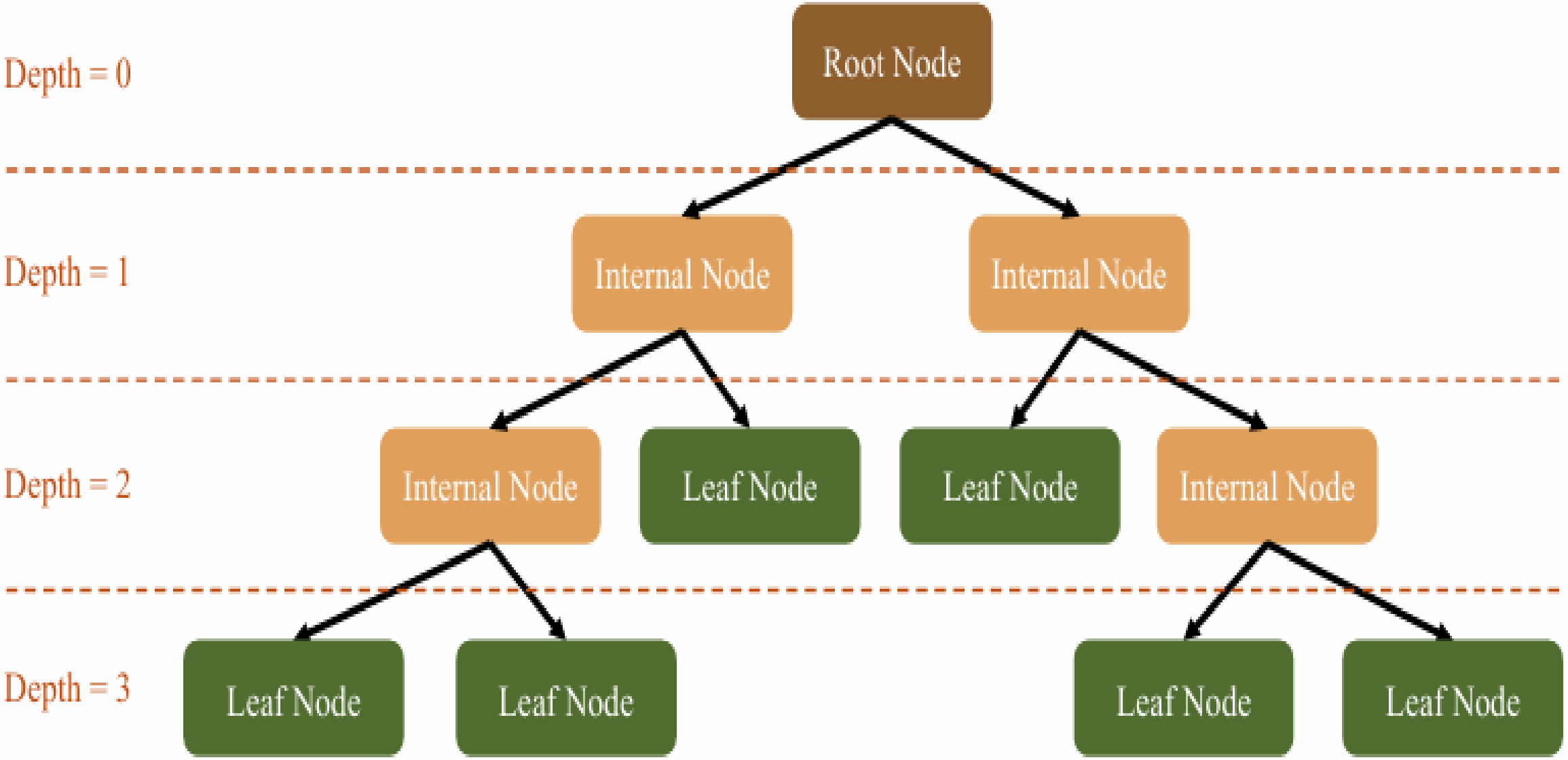
- Có thể xử lý đồng thời ba dạng dữ liệu:
  - Rời rạc: Categorical data
  - Liên tục: Continuous/Numeric data
  - Hỗn hợp: Mixed data

Giới tính	Nghề nghiệp	Mua
Nữ	Bác sĩ	Có
Nữ	Giáo viên	Không
Nam	Giáo viên	Có
Nam	Bác sĩ	Không

Chiều cao	Cân nặng	Mua
160	55	Có
169	53	Không
150	60	Không
152	49	Có

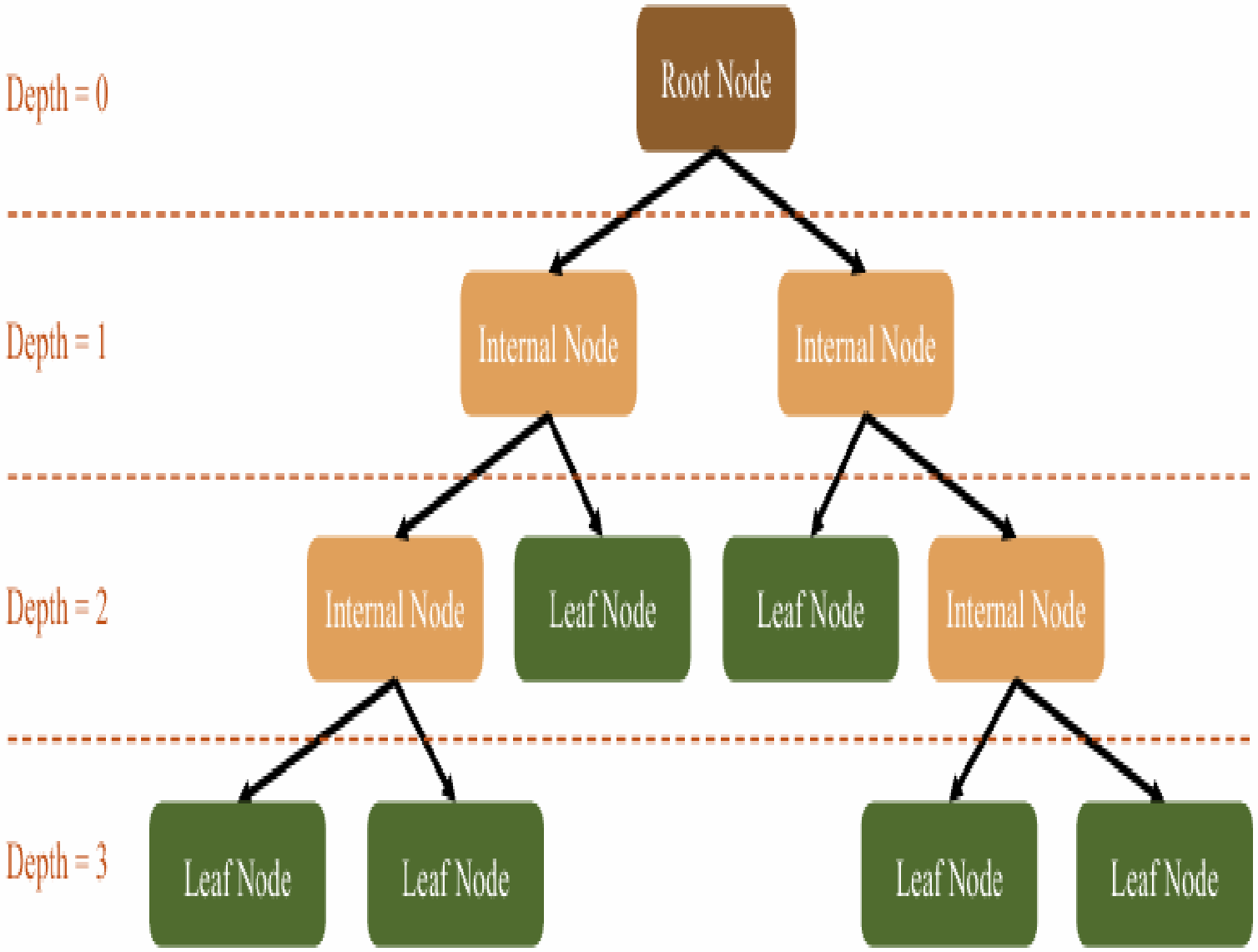
Chiều cao	Nghề nghiệp	Mua
160	Bác sĩ	Có
169	Giáo viên	Không
150	Giáo viên	Không
152	Bác sĩ	Có

Cấu trúc Decision tree



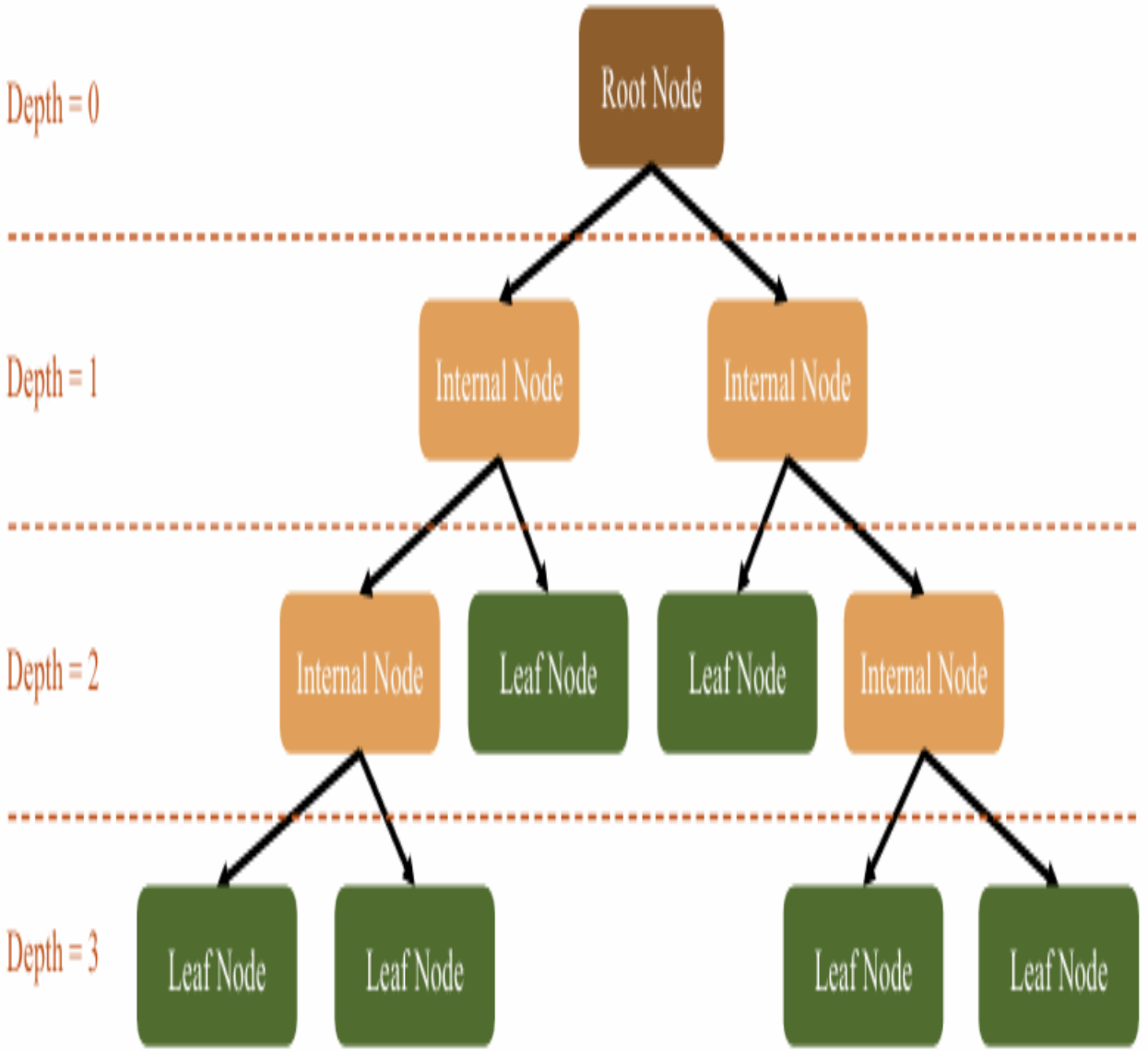
Cấu trúc Decision tree

- Root Node (Nút gốc)
  - Là nút đầu tiên của cây đại diện cho:
    - Một thuộc tính
    - Một điều kiện rẽ nhánh
  - Phân tách toàn bộ dữ liệu ban đầu thành các tập con
  - Có ít nhất 2 nhánh con



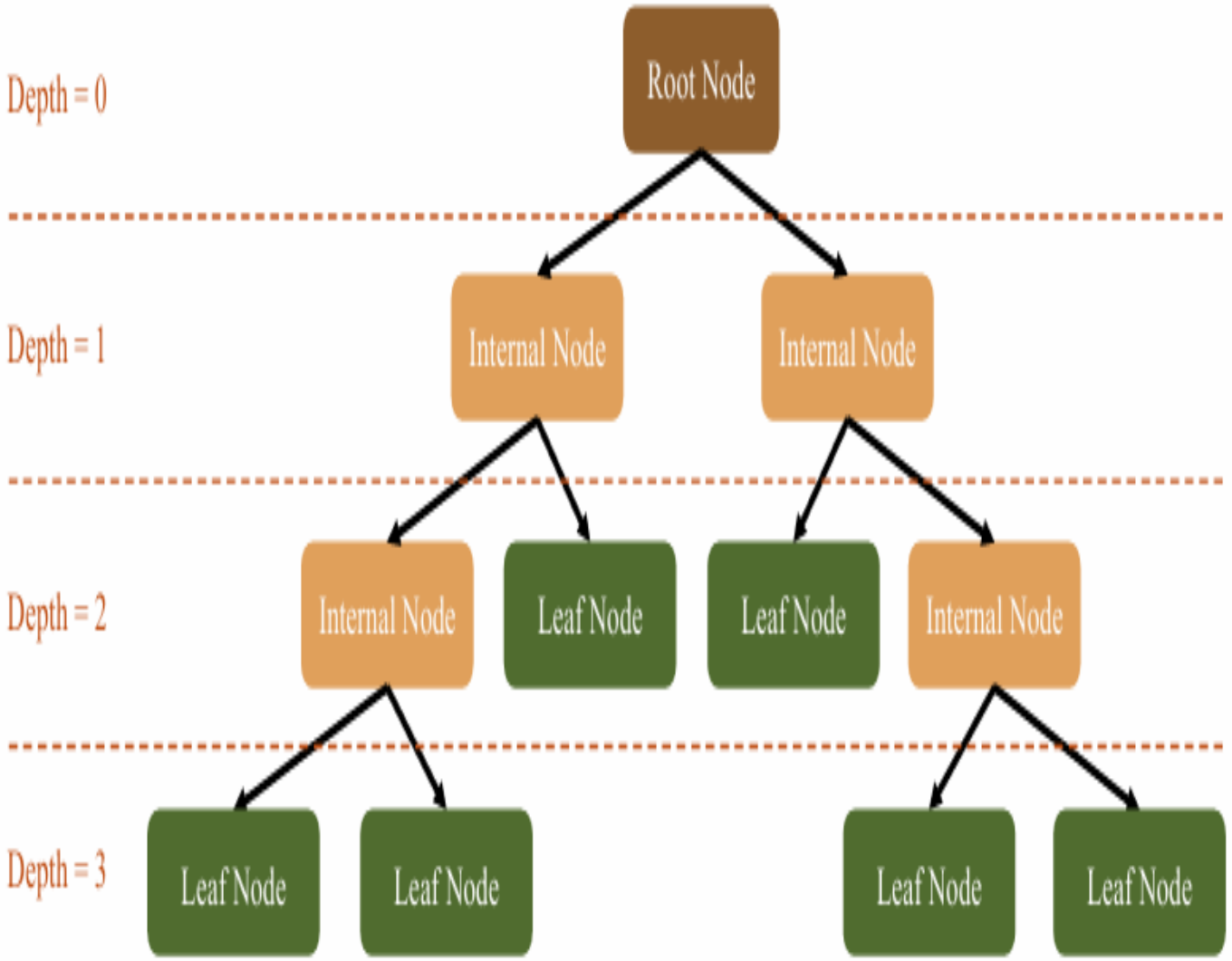
Cấu trúc Decision tree

- Internal Node (Nút trong)
  - Là các nút không phải gốc, không phải lá
  - Mỗi node đại diện cho:
    - Một thuộc tính
    - Một điều kiện rẽ nhánh
  - Có ít nhất 2 nhánh con
  - Phân tách các tập con



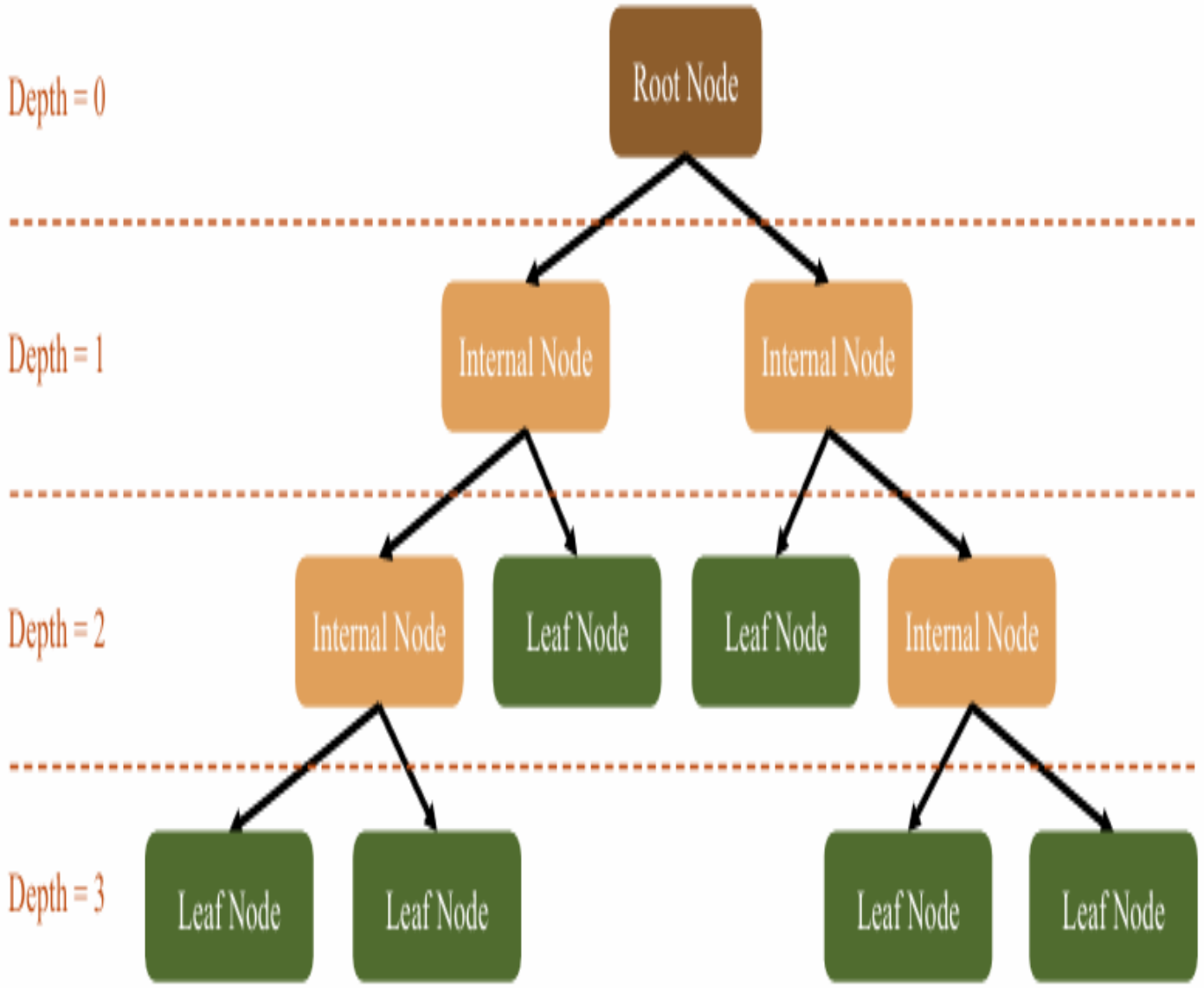
Cấu trúc Decision tree

- Branch (Nhánh)
  - Biểu diễn kết quả của điều kiện
  - Có thể là:
    - True / False
    - ≤ / >
    - Các giá trị rời rạc



Cấu trúc Decision tree

- Leaf Node (Nút lá)
  - Nút kết thúc
  - Chứa kết quả dự đoán cuối cùng:
    - Nhãn lớp (classification)
    - Giá trị dự đoán (regression)



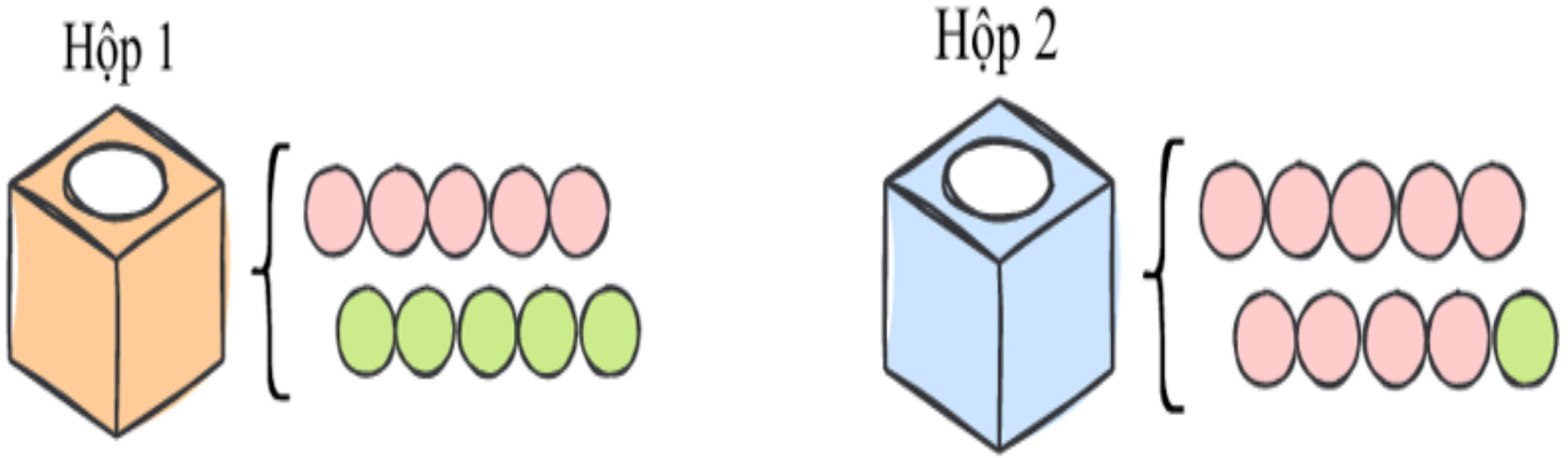
Decision tree

- Mục tiêu của Decision tree:
  - Xây dựng một mô hình dạng cây để dự đoán nhãn (classification) hoặc giá trị (regression) của dữ liệu mới bằng cách chia dữ liệu thành các vùng thuần khiết nhất có thể.
  - Thuần khiết tức là chỉ chứa các mẫu thuộc về một lớp duy nhất.
- Câu hỏi đặt ra:
  - Chia dữ liệu như thế nào thì sẽ tạo thành các vùng thuần khiết nhất?

Decision tree

- Mục tiêu của Decision tree:
  - Xây dựng một mô hình dạng cây để dự đoán nhãn (classification) hoặc giá trị (regression) của dữ liệu mới bằng cách chia không gian dữ liệu thành các vùng thuần khiết nhất có thể.
- Câu hỏi đặt ra:
  - Chia dữ liệu như thế nào thì sẽ tạo thành các vùng thuần khiết nhất?
  - Đối với Classification
    - Entropy – Information Gain
    - Gini Impurity – Gini Gain
  - Đôi với Regression
    - MSE
    - Variance

Entropy



- Ta cần một hàm để đo mức độ bất định/không chắc chắn của một phân phối xác suất.

Chứng minh công thức entropy

- Hàm I phải thỏa các yêu cầu:
  - (A1) Tính chất liên tục theo p.
  - (A2) Giảm khi p tăng.
  - (A3) Nếu hai biến độc lập xảy ra liên tiếp: $I(pq)=I(p)+I(q)$

Chứng minh công thức entropy

- Định lý (Shannon, 1948): Hàm duy nhất (ngoại trừ hằng số nhân dương) thỏa các tiên đề (A1)–(A3) là:

$$H(Y) = -\sum_{i=1}^{k} p_i \log p_i$$

Entropy – Information Gain

- Entropy là gì?
- Entropy đo mức độ hỗn loạn / không chắc chắn của nhãn lớp trong S.

$$Entropy(S) = -\sum_{c \in C} p_c log_2 p_c$$

- p_c: Xác suất có nhãn c trong S
- C: Số lớp
- Vai trò: Dùng để tính Information Gain

Tốc độ gió	Khả năng mưa	Đi đánh cầu lông
Cao	Có	No
Bình thường	Có	Yes
Thấp	Không	Yes
Cao	Không	No
Bình thường	Không	Yes

Entropy – Information Gain

- Information Gain là gì?
- Information Gain đo lượng thông tin thu được khi ta chia tập dữ liệu S theo một thuộc tính/đặc trưng A.

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

- $Entropy(S)$: Độ hỗn loạn ban đầu của tập dữ liệu
- S_v: tập con của S gồm các mẫu có A = v
- $\frac{|S_v|}{|S|}$ là trọng số theo kích thước tập con

=> Thuộc tính có giá trị IG càng lớn thì khả năng phân tách càng tốt.

Entropy – Information Gain

- Information Gain là gì?
- Information Gain đo lượng thông tin thu được khi ta chia tập dữ liệu S theo một thuộc tính/đặc trưng A.

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

- $Entropy(S)$: Độ hỗn loạn ban đầu của tập dữ liệu
- S_v: tập con của S gồm các mẫu có A = v
- $\frac{|S_v|}{|S|}$ là trọng số theo kích thước tập con

=> Information Gain chính là độ giảm trung bình của entropy sau khi tách theo thuộc tính A. Thuộc tính có giá trị IG càng lớn thì khả năng phân tách càng tốt.

Tốc độ gió	Khả năng mưa	Đi đánh cầu lông
Cao	Có	No
Cao	Có	Yes
Thấp	Không	Yes
Cao	Không	No
Thấp	Không	Yes

Gini Impurity & Gini Gain

- Gini Impurity là gì?
- Gini Impurity đo mức độ hỗn loạn / không chắc chắn của nhãn lớp trong node.

$$Gini(S) = \sum_{c \in C} p_c(1 - p_c) = 1 - \sum_{c \in C} p_c^2$$

- p_c: Xác suất có nhãn c trong S
- C: Số lớp (Tập các nhãn)
- G(S) đo mức "không tinh khiết" hoặc "không thuần" (impurity) trong phân phối nhãn.
- Vai trò: Dùng để tính Gini Gain

Tốc độ gió	Khả năng mưa	Đi đánh cầu lông
Cao	Có	No
Cao	Có	Yes
Thấp	Không	Yes
Cao	Không	No
Thấp	Không	Yes

Gini Impurity & Gini Gain

- Gini Gain là gì?
- Gini Gain đo **lượng thông tin thu được** khi ta chia tập dữ liệu S theo một thuộc tính/đặc trưng A.

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

Tốc độ gió	Khả năng mưa	Đi đánh cầu lông
Cao	Có	No
Cao	Có	Yes
Thấp	Không	Yes
Cao	Không	No
Thấp	Không	Yes

- $G(S)$: Độ hỗn loạn ban đầu của tập dữ liệu
- S_v: tập con của S gồm các mẫu có A = v
- $\frac{|S_v|}{|S|}$ là trọng số theo kích thước tập con
- GG(S,A) đo mức giảm impurity thu được nhờ phân tách theo A.

Quy trình xây dựng một decision tree cho classification

Quy trình Xây dựng Cây Quyết định

Bước 1: Tính Thống kê Gốc

Tính Root Entropy H(S) (Entropy của cột nhãn) cho toàn bộ mẫu.

Tính Root Gini G(S) tương tự cho toàn bộ mẫu.

Bước 2: Tính Lợi ích (Gain) cho từng thuộc tính

1. Với Entropy:

$$IG(S,A) = H(S) - \sum_{v \in Vals(A)} \frac{||S_v|}{|S|} H(S_v).$$

2. Với Gini:

$$G_G(S,A) = G(S) - \sum_{v \in Vals(A)} \frac{|S_v|}{|S|} G(S_v).$$

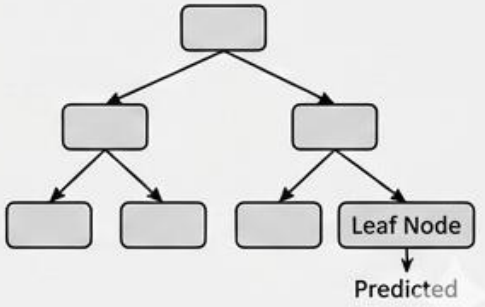
Bước 3: Chọn thuộc tính phân nhánh

Chọn thuộc tính có IG hoặc GG cao nhất để phân tách.

Phân tách dữ liệu, lặp lại Bước 1-3 cho mỗi node con đến khi các node thuần/tinh khiết (Pure) hoặc hết thuộc tính.

Bước 4: Ứng dụng cây để dự đoán

Với dữ liệu mới, từ root node đi theo nhánh phù hợp đến leaf node, lấy nhãn dự đoán.



THỰC HÀNH

- Duyệt vay ngân hàng

Khách hàng	**Thu nhập**	**Lịch sử tín dụng**	**Công việc ổn định**	**Duyệt vay**
A	Cao	Tốt	Có	Yes
B	Thấp	Xấu	Không	No
C	Trung bình	Tốt	Có	Yes
D	Cao	Xấu	Có	No
E	Trung bình	Tốt	Không	Yes
F	Cao	Xấu	Không	?????

THỰC HÀNH

- Bước 1: Tính thống kê gốc
- S={3 yes, 2 no}, như vậy:
- $p_{yes} = \frac{3}{5}$
- $p_{no} = \frac{2}{5}$
- $Entropy(S) = H(S) = -\sum_{i=1}^{C} p_i log_2 p_i$
- $= -\left(\frac{3}{5} log_2 \frac{3}{5} + \frac{2}{5} log_2 \frac{2}{5}\right) = 0.971$
- $Gini(S) = G(S) = \sum_{c \in C} p_c(1 - p_c) = 1 - \sum_{c \in C} p_c^2$
- $= 1 - ((\frac{3}{5})^2 + (\frac{2}{5})^2)$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập
- Có 3 giá trị là Cao, Thấp, Trung bình
- S_{cao}={1 yes, 1 no}, như vậy: $p_{yes} = \frac{1}{2}; \ p_{no} = \frac{1}{2}$
- $S_{thấp}$={0 yes, 1 no}, như vậy: $p_{yes} = 0; \ p_{no} = 1$
- S_{tb}={2 yes, 0 no}, như vậy: $p_{yes} = 1; \ p_{no} = 0$
- $H(S_{cao}) = -(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2})$=1
- $H(S_{thấp}) = -(0 log_2 0 + 1 log_2 1)$=0
- $H(S_{tb}) = -(0 log_2 0 + 1 log_2 1)$=0
- $IG(S, Thu\ nhập) = 0.971 - (\frac{2}{5} \times 1 + \frac{1}{5} \times 0 + \frac{2}{5} \times 0)$=0.571

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 2. Lịch sử tính dụng
- Có 2 giá trị là Tốt và Xấu
- $S_{tốt}$={3 yes, 0 no}, như vậy: $p_{yes} = 1;\ p_{no} = 0$
- $S_{xấu}$={0 yes, 2 no}, như vậy: $p_{yes} = 0;\ p_{no} = 1$
- $H(S_{tốt}) = -(1log_2 1 + 0log_2 0)$=0
- $H(S_{xấu}) = -(0log_2 0 + 1log_2 1)$=0
- $IG(S, Lịch\ sử\ tín\ dụng) = 0.971 - (\frac{3}{5} \times 0 + \frac{2}{5} \times 0) = 0.971$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 3. Công việc ổn định
- Có 2 giá trị là Có và Không
- $S_{có}$={2 yes, 1 no}, như vậy: $p_{yes} = \frac{2}{3};\ p_{no} = \frac{1}{3}$
- $S_{không}$={1 yes, 1 no}, như vậy:$p_{yes} = \frac{1}{2};\ p_{no} = \frac{1}{2}$
- $H(S_{có}) = -(\frac{2}{3} log_2 \frac{2}{3} + \frac{1}{3} log_2 \frac{1}{3})$=0.918
- $H\left(S_{không}\right) = -(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2})$=1
- $IG(S, Công\ việc\ ổn\ định) = 0.971 - (\frac{3}{5} \times 0.918 + \frac{2}{5} \times 1) =$

THỰC HÀNH

- Bước 3: Chọn thuộc tính phân nhánh
- So sánh
- $IG(S, Thu\ nhập) = 0.571$

- $IG(S, Lịch\ sử\ tính\ dụng) = 0.971$

- $IG(S, Công\ việc\ ổn\ định) = 0.971 - (\frac{3}{5} \times \cdots + \frac{2}{5} \times \cdots =$
- => Chọn “Lịch sử tín dụng” làm node gốc

THỰC HÀNH

- Khách hàng có mua thiết bị điện hay không

KH	**Thu nhập (triệu)**	**Loại KH**	**Bảo hành**	**Mua**
1	8	Cá nhân	Không	No
2	15	Cá nhân	Có	Yes
3	22	Doanh nghiệp	Có	Yes
4	10	Cá nhân	Không	No
5	18	Doanh nghiệp	Không	Yes
6	12	Cá nhân	Có	Yes
7	9	Cá nhân	Không	????

THỰC HÀNH

- Bước 1: Tính thống kê gốc
- S={4 yes, 2 no}, như vậy: $p_{yes} = \frac{4}{6}; \; p_{no} = \frac{2}{6}$
- $Entropy(S) = H(S) = -\left(\frac{2}{3} log_2 \frac{2}{3} + \frac{1}{3} log_2 \frac{1}{3}\right) = 0.918$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập

KH	**Thu nhập (triệu)**	**Mua**
1	8	No
4	10	No
6	12	Yes
2	15	Yes
5	18	Yes
3	22	Yes

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập

KH	**Ngưỡng điểm**	**Thu nhập (triệu)**	**Mua**
1	9	8	No
4	11	10	No
6	13,5	12	Yes
2	16,5	15	Yes
5	20	18	Yes
3		22	Yes

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập

Ngưỡng điểm	**Information Gain**
9	0.286
11	**0.918**
13,5	0.459
16,5	0.251
20	0.109

KH	**Thu nhập (triệu)**	**Mua**
1	8	No
4	10	No
6	12	Yes
2	15	Yes
5	18	Yes
3	22	Yes

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 2. Loại khách hàng
- Có 2 giá trị là Cá nhân và Doanh nghiệp
- $S_{\text{cá\_nhân}}$={2 yes, 2 no}, như vậy: $p_{yes} = \frac{1}{2};\ p_{no} = \frac{1}{2}$
- $S_{\text{doanh\_nghiệp}}$={2 yes, 0 no}, như vậy: $p_{yes} = 1;\ p_{no} = 0$
- $H(S_{\text{cá\_nhân}}) = -(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2})$=1
- $H(S_{\text{doanh\_nghiệp}}) = 0$
- $IG(S, Loại\ khách\ hàng) = 0.918 - (\frac{4}{6} \times 1 + \frac{2}{6} \times 0) = 0.251$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 3. Bảo hành
- Có 2 giá trị là Có và Không
- $S_{có}$={3 yes, 0 no}
- $S_{không}$={1 yes, 2 no}, như vậy:$p_{yes} = \frac{1}{3}$; $p_{no} = \frac{2}{3}$
- $H(S_{có}) = 0$
- $H(S_{không}) = -\left(\frac{1}{3} log_2 \frac{1}{3} + \frac{2}{3} log_2 \frac{2}{3}\right)$ =0.918
- $IG(S, Bảo\ hành) = 0.918 - (\frac{1}{2} \times 0.918 + \frac{1}{2} \times 0) = 0.459$

THỰC HÀNH

- Bước 3: Chọn thuộc tính phân nhánh
- So sánh
- $IG(S, Thu\ nhập) = 0.918$
- $IG(S, Loại\ khách\ hàng) = 0.251$
- $IG(S, Bảo\ hành) = 0.459$
- => Chọn “Thu nhập <11” làm node gốc

THỰC HÀNH

- Bước 4: Chia tiếp
- Nhánh 1: Thu nhập >=11
  - $S_{thu\_nhập>=11}$={4 yes, 0 no} => Thuần => dừng
- Nhánh 1: Thu nhập <11
  - $S_{thu\_nhập<11}$={0 yes, 2 no} => Thuần => dừng

THỰC HÀNH

- Khách hàng có mua thiết bị điện hay không

KH	**Thu nhập (triệu)**	**Loại KH**	**Bảo hành**	**Mua**
1	8	Cá nhân	Không	No
2	15	Cá nhân	Có	Yes
3	22	Doanh nghiệp	Có	Yes
4	10	Cá nhân	Không	No
5	18	Doanh nghiệp	Không	Yes
6	12	Cá nhân	Có	Yes

THỰC HÀNH

- Bước 1: Tính thống kê gốc
- S={4 yes, 2 no}, như vậy: $p_{yes} = \frac{4}{6}; \; p_{no} = \frac{2}{6}$
- $Entropy(S) = H(S) = -\left(\frac{2}{3} log_2 \frac{2}{3} + \frac{1}{3} log_2 \frac{1}{3}\right) = 0.918$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập

KH	**Thu nhập (triệu)**	**Mua**
1	8	No
4	10	No
6	12	Yes
2	15	Yes
5	18	Yes
3	22	Yes

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Thu nhập
- Có 2 giá trị là <14và >=14
- $S_{<14}$={1 yes, 2 no}, như vậy: $p_{yes} = \frac{1}{3};\ p_{no} = \frac{2}{3}$
- $S_{>=14}$={3 yes, 0 no}
- $H(S_{<14}) = -(\frac{1}{3} log_2 \frac{1}{3} + \frac{2}{3} log_2 \frac{2}{3})$=0.918
- $H(S_{\geq 14}) = 0$
- $IG(S, Loại\ khách\ hàng) = 0.918 - (\frac{3}{6} \times 0.918 + \frac{3}{6} \times 0) = 0.459$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 2. Loại khách hàng
- Có 2 giá trị là Cá nhân và Doanh nghiệp
- $S_{\text{cá\_nhân}}$={2 yes, 2 no}, như vậy: $p_{yes} = \frac{1}{2};\ p_{no} = \frac{1}{2}$
- $S_{\text{doanh\_nghiệp}}$={2 yes, 0 no}, như vậy: $p_{yes} = 1;\ p_{no} = 0$
- $H(S_{\text{cá\_nhân}}) = -(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2})$=1
- $H(S_{\text{doanh\_nghiệp}}) = 0$
- $IG(S, Loại\ khách\ hàng) = 0.918 - (\frac{4}{6} \times 1 + \frac{2}{6} \times 0) = 0.251$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 3. Bảo hành
- Có 2 giá trị là Có và Không
- $S_{có}$={3 yes, 0 no}
- $S_{không}$={1 yes, 2 no}, như vậy:$p_{yes} = \frac{1}{3}$; $p_{no} = \frac{2}{3}$
- $H(S_{có}) = 0$
- $H(S_{không}) = -\left(\frac{1}{3} log_2 \frac{1}{3} + \frac{2}{3} log_2 \frac{2}{3}\right)$ =0.918
- $IG(S, Bảo\ hành) = 0.918 - (\frac{1}{2} \times 0.918 + \frac{1}{2} \times 0) = 0.459$

THỰC HÀNH

- Bước 3: Chọn thuộc tính phân nhánh
- So sánh
- $IG(S, Thu\ nhập) = 0.459$
- $IG(S, Loại\ khách\ hàng) = 0.251$
- $IG(S, Bảo\ hành) = 0.459$
- => Chọn “Thu nhập <=14” làm node gốc

THỰC HÀNH

- Bước 4: Chia tiếp
- Nhánh 1: Thu nhập >14
  - $S_{thu\_nhập>11}$={4 yes, 0 no} => Thuần => dừng
- Nhánh 1: Thu nhập <=14
  - Dữ liệu con:

KH	**Loại KH**	**Bảo hành**	**Mua**
1	Cá nhân	Không	No
4	Cá nhân	Không	No
6	Cá nhân	Có	Yes

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 1. Loại khách hàng
- Có 1 giá trị là Cá nhân
- $S_{\text{cá\_nhân}}$={1 yes, 2 no}, như vậy: $p_{yes} = \frac{1}{3}; \; p_{no} = \frac{2}{3}$
- $H(S_{\text{cá\_nhân}}) = -(\frac{1}{3} log_2 \frac{1}{3} + \frac{2}{3} log_2 \frac{2}{3}) = 0.918$
- $IG(S, Loại\ khách\ hàng) = 0.918 - (1 \times 0.918) = 0$

THỰC HÀNH

- Bước 2: Tính Gain cho từng đặc trưng
- 2. Bảo hành
- Có 2 giá trị là Có và Không
- $S_{có}$={1 yes, 0 no}
- $S_{không}$={1 yes, 1 no}, như vậy:$p_{yes} = \frac{1}{2}; \; p_{no} = \frac{1}{2}$
- $H(S_{có}) = 0$
- $H\left(S_{không}\right) = -\left(\frac{1}{2} log_2 \frac{1}{2} + \frac{1}{2} log_2 \frac{1}{2}\right) = 1$
- $IG(S, Bảo \; hành) = 0.918 - (\frac{1}{3} \times 0 + \frac{2}{3} \times 1) = 0.251$

THỰC HÀNH

- Bước 3: Chọn thuộc tính phân nhánh
- So sánh
- $IG(S, Loại\ khách\ hàng) = 0$
- $IG(S, Bảo\ hành) = 0.251$
- => Chọn “Bảo hành” làm node tiếp theo

THỰC HÀNH

- Bước 4: Chia tiếp
- Nhánh 1: Bảo hành = Có
  - $S_{bảo\_hành=Có}$={1 yes, 0 no} => Thuần => dừng
- Nhánh 1: Bảo hành=Không
  - $S_{bảo\_hành=Không}$={0 yes, 2 no} => Thuần => dừng

KH	**Loại KH**	**Bảo hành**	**Mua**
1	Cá nhân	Không	No
4	Cá nhân	Không	No
6	Cá nhân	Có	Yes

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def entropy(data):
    labels = [row["Mua"] for row in data]
    total = len(labels)

    counts = {}
    for label in labels:
        counts[label] = counts.get(label, 0) + 1

    ent = 0
    for count in counts.values():
        p = count / total
        ent -= p * math.log2(p)

    return ent
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def split_categorical(data, feature):
    splits = {}
    for row in data:
        key = row[feature]
        splits.setdefault(key, []).append(row)
    return splits
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def split_categorical(data, feature):
    splits = {}
    for row in data:
        key = row[feature]
        splits.setdefault(key, []).append(row)
    return splits


def split_numeric(data, feature, threshold):
    left = [row for row in data if row[feature] <= threshold]
    right = [row for row in data if row[feature] > threshold]
    return left, right
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def information_gain_categorical(data, feature):
    total_entropy = entropy(data)
    splits = split_categorical(data, feature)

    weighted_entropy = 0
    for subset in splits.values():
        weighted_entropy += (len(subset) / len(data)) * entropy(subset)

    return total_entropy - weighted_entropy
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def information_gain_numeric(data, feature):
    values = sorted(set(row[feature] for row in data))
    best_ig = -1
    best_threshold = None

    for threshold in values:
        left, right = split_numeric(data, feature, threshold)
        if not left or not right:
            continue

        weighted_entropy = (
            len(left)/len(data) * entropy(left) +
            len(right)/len(data) * entropy(right)
        )

        ig = entropy(data) - weighted_entropy

        if ig > best_ig:
            best_ig = ig
            best_threshold = threshold

    return best_ig, best_threshold
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
features = ["Thu_nhap", "Loai_KH", "Bao_hanh"]

print("Entropy ban đầu:", entropy(data))
print()

# Thu nhập (định lượng)
ig_income, threshold = information_gain_numeric(data, "Thu_nhap")
print(f"IG(Thu_nhap) = {ig_income:.3f}, threshold = {threshold}")

# Loại KH
ig_type = information_gain_categorical(data, "Loai_KH")
print(f"IG(Loai_KH) = {ig_type:.3f}")

# Bảo hành
ig_warranty = information_gain_categorical(data, "Bao_hanh")
print(f"IG(Bao_hanh) = {ig_warranty:.3f}")
```

CODE

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

```
def build_tree(data, features, depth, max_depth):
    labels = [row["Mua"] for row in data]

    # Điều kiện dừng
    if labels.count(labels[0]) == len(labels):
        return labels[0]

    if depth == max_depth or not features:
        return max(set(labels), key=labels.count)

    # Chọn thuộc tính tốt nhất
    best_feature = None
    best_threshold = None
    best_ig = -1

    for feature in features:
        if isinstance(data[0][feature], (int, float)):
            ig, threshold = information_gain_numeric(data, feature)
        else:
            ig = information_gain_categorical(data, feature)
            threshold = None

        if ig > best_ig:
            best_ig = ig
            best_feature = feature
            best_threshold = threshold

    tree = {best_feature: {}}
```

```
    # Chia dữ liệu
    if best_threshold is not None:
        left, right = split_numeric(data, best_feature, best_threshold)
        tree[best_feature][f"<= {best_threshold}"] = build_tree(
            left, features, depth+1, max_depth
        )
        tree[best_feature][f"> {best_threshold}"] = build_tree(
            right, features, depth+1, max_depth
        )
    else:
        splits = split_categorical(data, best_feature)
        for key, subset in splits.items():
            tree[best_feature][key] = build_tree(
                subset,
                [f for f in features if f != best_feature],
                depth+1,
                max_depth
            )

    return tree
```

CODE

```
tree = build_tree(data, features, depth=0, max_depth=2)
print(tree)
```

```
def predict(tree, sample):
    """
    tree   : cây decision tree (dict hoặc nhãn)
    sample : 1 mẫu dữ liệu (dict)
    """
    # Nếu đã là lá → trả về nhãn
    if not isinstance(tree, dict):
        return tree

    # Lấy thuộc tính tại nút hiện tại
    feature = next(iter(tree))
    branches = tree[feature]

    value = sample[feature]

    # Trường hợp thuộc tính định lượng
    for key in branches:
        if key.startswith("<="):
            threshold = float(key.split()[-1])
            if value <= threshold:
                return predict(branches[key], sample)
        elif key.startswith(">"):
            threshold = float(key.split()[-1])
            if value > threshold:
                return predict(branches[key], sample)

    # Trường hợp thuộc tính định tính
    if value in branches:
        return predict(branches[value], sample)

    # Nếu rơi vào giá trị chưa từng thấy → dự đoán mặc định
    return None
```

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

CODE

```
new_customer = {
    "Thu_nhap": 11,
    "Loai_KH": "Ca_nhan",
    "Bao_hanh": "Co"
}

result = predict(tree, new_customer)
print("Dự đoán:", result)
```

Đọc và hoàn thiện các đoạn code sau để có một chương trình huấn luyện mô hình hoàn chỉnh

DECISION TREE CHO REGRESSION

i	Diện tích (x_1)	Số phòng (x_2)	Giá (y)
1	40	1	1.5
2	45	1	1.6
3	50	2	2.0
4	60	2	2.4
5	70	3	3.0
6	80	3	3.6

DECISION TREE CHO REGRESSION

- **Bước 1. Trung bình**
  - $\bar{y}$
- **Bước 2. Tính MSE cho toàn bộ dữ liệu**
  - $MSE(S) = \frac{1}{|S|}\sum_{i \in S}(y_i - \bar{y})^2$
- **Bước 3. Tính MSE cho từng đặc trưng**
- **Bước 4. Tính mức giảm MSE**
- $\nabla MSE(S, ngưỡng) = MSE(S) - \sum_{v \in ngưỡng} \frac{|S_v|}{|S|} MSE(S_v)$

Random Forest

Giới thiệu

- Random Forest là một tập hợp của nhiều Decision Tree:
  - Mỗi Decision tree sẽ hoạt động độc lập để đưa ra quyết định
  - Quyết định của các Decision tree sẽ được tổng hợp thông qua một cơ chế bỏ phiếu => quyết định cuối cùng

Nhắc lại

1. Entropy & Information Gain

- **Entropy** $H(S)$ đo lường mức độ hỗn loạn của một tập dữ liệu S:

$$H(S) = -\sum_{c} p_c \log_2(p_c),$$

- **Information Gain** $\mathrm{IG}(S, A)$ đo lường mức độ giảm Entropy khi phân tách tập S theo thuộc tính A.

$$\mathrm{IG}(S, A) = H(S) - \sum_{v} \frac{|S_v|}{|S|} H(S_v).$$

2. Gini Impurity & Gini Gain

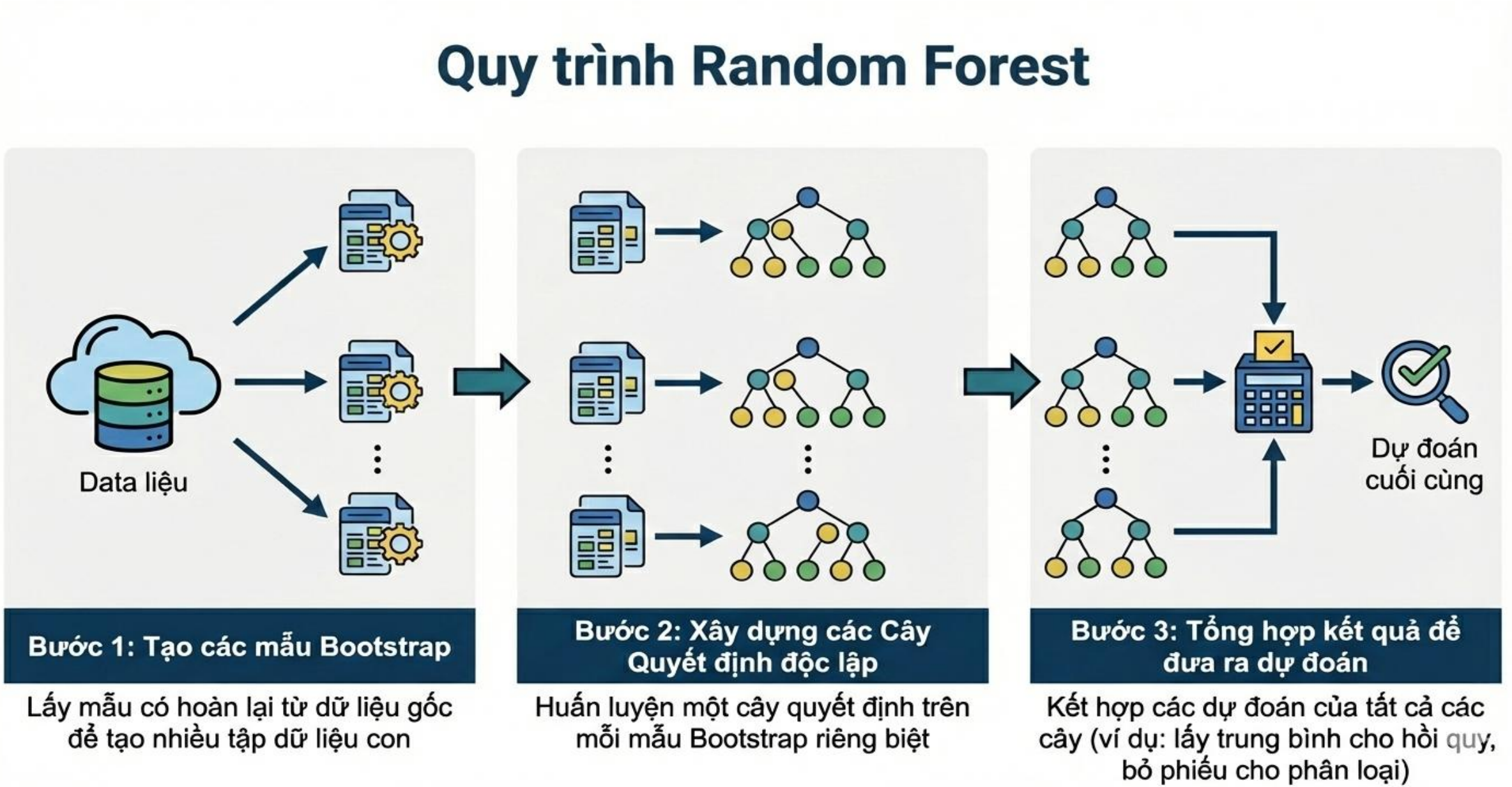
- **Gini Impurity** $G(S)$ đo xác suất phân loại sai một mẫu nếu gán nhãn ngẫu nhiên:

$$G(S) = 1 - \sum_{c} p_c^2,$$

- **Gini Gain** $\mathrm{GG}(S, A)$ đo lường mức độ giảm Gini Impurity khi phân tách tập S theo thuộc tính A.

$$\mathrm{GG}(S, A) = G(S) - \sum_{v} \frac{|S_v|}{|S|} G(S_v).$$

Quy trình xây dựng mô hình Random Forest

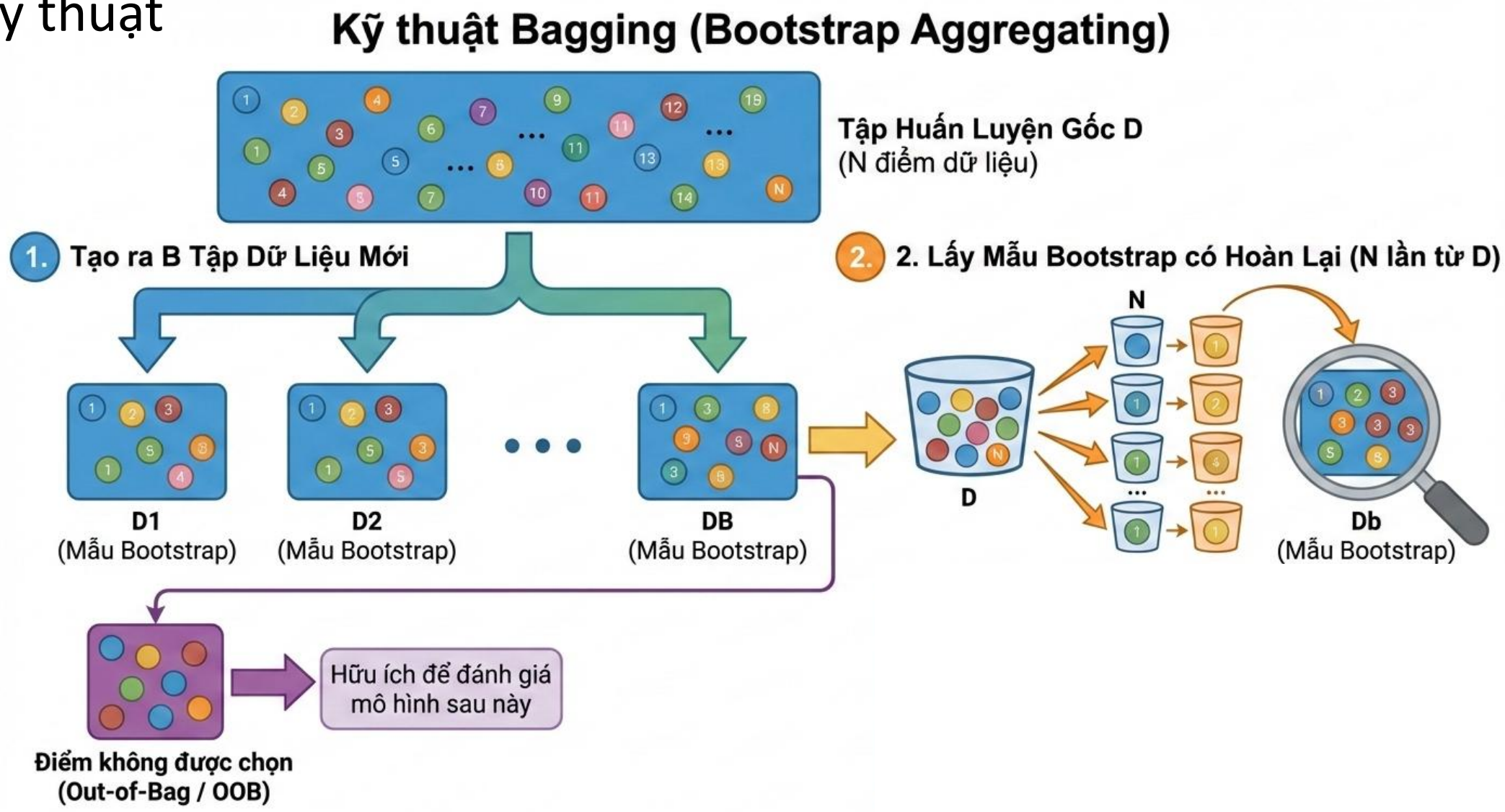


Bước 1: Tạo các mẫu Bootstrap (Bootstrap Aggregating)

- Ý tưởng:
  - Thay vì dùng chung một tập dữ liệu để huấn luyện tất cả các cây, ta tạo ra nhiều tập dữ liệu huấn luyện khác nhau bằng cách lấy mẫu ngẫu nhiên có hoàn lại từ tập dữ liệu ban đầu.
- Mục đích:
  - Nhờ mỗi cây được học trên một tập dữ liệu con khác nhau, chúng sẽ nắm bắt những đặc trưng và quy luật không hoàn toàn giống nhau, đóng vai trò như các “chuyên gia” với góc nhìn riêng. Sự khác biệt này giúp hình thành một tập hợp các cây đa dạng, góp phần nâng cao hiệu quả của mô hình tổng thể.

Bước 1: Tạo các mẫu Bootstrap (Bootstrap Aggregating)

- Kỹ thuật



Bước 2: Xây dựng các Cây Quyết định độc lập

- Ý tưởng:
  - Ở mỗi nút phân chia của cây, thay vì đánh giá toàn bộ các đặc trưng, thuật toán chỉ chọn ngẫu nhiên một tập con gồm m đặc trưng và tìm phương án chia tốt nhất trong tập con đó.
- Mục đích:
  - Cách làm này giúp hạn chế sự chi phối của những đặc trưng quá mạnh lên tất cả các cây, buộc mỗi cây phải khai thác những đặc trưng khác nhau. Nhờ đó, các cây trở nên đa dạng hơn và ít phụ thuộc lẫn nhau, góp phần nâng cao hiệu quả của mô hình tập hợp.

  => Chọn m như thế nào?

Bước 2: Xây dựng các Cây Quyết định độc lập

- Chọn m:
- Việc chọn m là một quyết định quan trọng.
- Các giá trị thường được đề xuất[a] là:
  - Cho bài toán phân loại: $m \approx \sqrt{n}$.
  - Cho bài toán hồi quy: $m \approx n/3$.

[a]Leo Breiman, “Random Forests,” Machine Learning, 45(1), 5–32, 2001.

Bước 3: Tổng hợp kết quả để đưa ra dự đoán

- Đối với bài toán Phân loại- Lấy phiếu theo đa số (Majority Voting): Nhãn dự đoán cuối cùng là nhãn được nhiều cây “bỏ phiếu” nhất.
- Ví dụ: Giả sử một khu rừng có B = 5 cây cần phân loại một email là “Spam” hay “Không Spam”.
- Kết quả dự đoán của 5 cây lần lượt là:
  - Cây 1: “Spam”.
  - Cây 2: “Không Spam”.
  - Cây 3: “Spam”.
  - Cây 4: “Spam”.
  - Cây 5: “Không Spam”.

Tổng kết phiếu bầu: 3 phiếu cho “Spam” và 2 phiếu cho “Không Spam”. Do đó, dự đoán cuối cùng của mô hình là “Spam”.

Bước 3: Tổng hợp kết quả để đưa ra dự đoán

- Đối với bài toán Hồi quy- Lấy giá trị trung bình (Averaging): Giá trị dự đoán cuối cùng là trung bình cộng của tất cả các dự đoán từ các cây.
- Ví dụ: Một khu rừng có B = 4 cây được dùng để dự đoán giá của một căn nhà (đơn vị: tỷ VNĐ).
  - Kết quả dự đoán của 4 cây lần lượt là:
    - Cây 1: 2.1
    - Cây 2: 2.3
    - Cây 3: 2.0
    - Cây 4: 2.2

    => Dự đoán cuối cùng là trung bình cộng của các kết quả trên: (2.1+2.3+2.0+2.2)/4=2.15

THỰC HÀNH

STT	Màu da	Tầm vóc	Thể trạng	Dùng kem chống nắng	Tình trạng da
1	Sẫm	Trung bình	Nhẹ	Không	Cháy nắng
2	Sẫm	Cao	Trung bình	Có	Không
3	Sáng	Thấp	Trung bình	Có	Không
4	Ngăm	Thấp	Trung bình	Không	Cháy nắng
5	Ngăm	Trung bình	Nặng	Không	Cháy nắng
6	Sáng	Trung bình	Nặng	Có	Không
7	Nâu	Trung bình	Nhẹ	Có	????
8	Sáng	Cao	Trung Bình	Có	?????

Bước 1: Tạo các mẫu Bootstrap

- Mẫu D1 (giả định): Lấy ngẫu nhiên 6 mẫu có hoàn lại, ta được tập dữ liệu

STT	Màu da	Tầm vóc	Thể trạng	Dùng kem chống nắng	Tình trạng da
4	Ngăm	Thấp	Trung bình	Không	Cháy nắng
1	Sẫm	Trung bình	Nhẹ	Không	Cháy nắng
6	Sáng	Trung bình	Nặng	Có	Không
2	Sẫm	Cao	Trung bình	Có	Không
4	Ngăm	Thấp	Trung bình	Không	Cháy nắng
2	Sẫm	Cao	Trung bình	Có	Không

Bước 1: Tạo các mẫu Bootstrap

- Mẫu D2 (giả định): Lấy ngẫu nhiên 6 mẫu khác, ta được tập dữ liệu

STT	Màu da	Tầm vóc	Thể trạng	Dùng kem chống nắng	Tình trạng da
1	Sẫm	Trung bình	Nhẹ	Không	Cháy nắng
5	Ngăm	Trung bình	Nặng	Không	Cháy nắng
2	Sẫm	Cao	Trung bình	Có	Không
6	Sáng	Trung bình	Nặng	Có	Không
5	Ngăm	Trung bình	Nặng	Không	Cháy nắng
3	Sáng	Thấp	Trung bình	Có	Không

Bước 1: Tạo các mẫu Bootstrap

- Mẫu D3 (giả định): Lấy ngẫu nhiên 6 mẫu khác, ta được tập dữ liệu

STT	Màu da	Tầm vóc	Thể trạng	Dùng kem chống nắng	Tình trạng da
3	Sáng	Thấp	Trung bình	Có	Không
4	Ngăm	Thấp	Trung bình	Không	Cháy nắng
1	Sẫm	Trung bình	Nhẹ	Không	Cháy nắng
4	Ngăm	Thấp	Trung bình	Không	Cháy nắng
5	Ngăm	Trung bình	Nặng	Không	Cháy nắng
3	Sáng	Thấp	Trung bình	Có	Không

Bước 2: Xây dựng các Cây Quyết định

- Xây dựng Cây 1 (T1) trên mẫu D1
- Nút gốc: Mẫu D1 có 3 “Cháy nắng” và 3 “Không”, do đó Ggốc = 0.5.
- Phân chia:
  - Chọn ngẫu nhiên 2 đặc trưng: {Màu da, Dùng kem}.
  - So sánh Gini Gain:
    - Chia theo “Dùng kem” tạo 2 nhánh thuần khiết. Gini sau chia = 0 → Gini Gain = 0.5.
    - Chia theo “Màu da” tạo 3 nhánh. Gini sau chia ≈ 0.167 → Gini Gain = 0.333.
  - Kết luận: Chọn chia theo “Dùng kem”. Vì các nút con đã thuần khiết, cây dừng lại.

Bước 2: Xây dựng các Cây Quyết định

- Xây dựng Cây 2 (T2) trên mẫu D2
- Nút gốc: Mẫu D2 có 3 “Cháy nắng” và 3 “Không”, Ggốc = 0.5.
- Phân chia (lần 1):
  - Chọn ngẫu nhiên 2 đặc trưng: {Tầm vóc, Thể trạng}.
  - So sánh Gini Gain:
    - Chia theo “Tầm vóc” tạo 3 nhánh. Gini sau chia ≈ 0.167 → Gini Gain = 0.333.
    - Chia theo “Thể trạng” tạo 3 nhánh. Gini sau chia ≈ 0.333 → Gini Gain = 0.167.
  - Kết luận: Ta chọn chia theo “Tầm vóc” vì có Gini Gain cao hơn.

Bước 2: Xây dựng các Cây Quyết định

- Xây dựng Cây 2 (T2) trên mẫu D2
- Nút gốc: Mẫu D2 có 3 “Cháy nắng” và 3 “Không”, Ggốc = 0.5.
- Phân chia (lần 2) cho nút “Trung bình”:
  - Nhánh “Trung bình” (gồm hai mẫu {1, 6}) chưa thuần khiết, có 1 “Cháy nắng” và 1 “Không” (G = 0.5).
  - Chọn ngẫu nhiên 2 đặc trưng mới: {Màu da, Dùng kem}.
  - Chia theo “Dùng kem” sẽ tách {1} (nhánh “Không”) và {6} (nhánh “Có”). Cách chia này tạo ra 2 nút con thuần khiết, do đó Gini của chúng đều bằng 0 và Gini Gain được tính bằng 0.5. Đây là mức giảm impurity tối đa, nên ta chọn cách chia này.

Bước 2: Xây dựng các Cây Quyết định

- Xây dựng Cây 3 (T3) trên mẫu D3
- Nút gốc: Mẫu D3 có 4 “Cháy nắng” và 2 “Không”, do đó Ggốc ≈ 0.444.
- Phân chia:
  - Chọn ngẫu nhiên 2 đặc trưng {Thể trạng, Dùng kem}.
  - Chia theo “Dùng kem” cho Gini Gain cao nhất (0.444).
  - Kết luận: Cây dừng lại vì các nút con đã thuần khiết.

Bước 3: Dự đoán và Tổng hợp kết quả

- Bây giờ, ta đưa mẫu kiểm tra ID=7 (Nâu, Trung bình, Nhẹ, Có) vào cả ba cây. Dự đoán của từng cây:
  - Cây T1: Mẫu 7 có “Dùng kem = Có” → Dự đoán là Không.
  - Cây T2: Mẫu 7 có “Tầm vóc= Trung bình” và “Dùng kem = Có” → Dự đoán là Không.
  - Cây T3: Mẫu 7 có “Dùng kem = Có” → Dự đoán là Không.
- Tổng hợp kết quả (Majority Voting)
  - Phiếu từ T1: Không.
  - Phiếu từ T2: Không.
  - Phiếu từ T3: Không.
- Kết luận: Mô hình Random Forest dự đoán Kết quả cho mẫu ID=7 là Không cháy nắng.

	A	B	C	D	E
1	Radius_mean	Texture_mean	Perimeter_mean	Area_mean	Diagnosis
2	17.99	10.38	122.8	1001	1
3	20.57	17.77	132.9	1326	1
4	19.69	21.25	130	1203	1
5	11.42	20.38	77.58	386.1	1
6	20.29	14.34	135.1	1297	1
7	12.45	15.7	82.57	477.1	1
8	18.25	19.98	119.6	1040	1
9	13.71	20.83	90.2	577.9	1
10	13	21.82	87.5	519.8	1
11	12.46	24.04	83.97	475.9	1
12	16.02	23.24	102.7	797.8	1
13	15.78	17.89	103.6	781	1
14	14.61	15.69	92.68	664.9	0
15	12.76	13.37	82.29	504.1	0

PlayTennis: training examples

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No