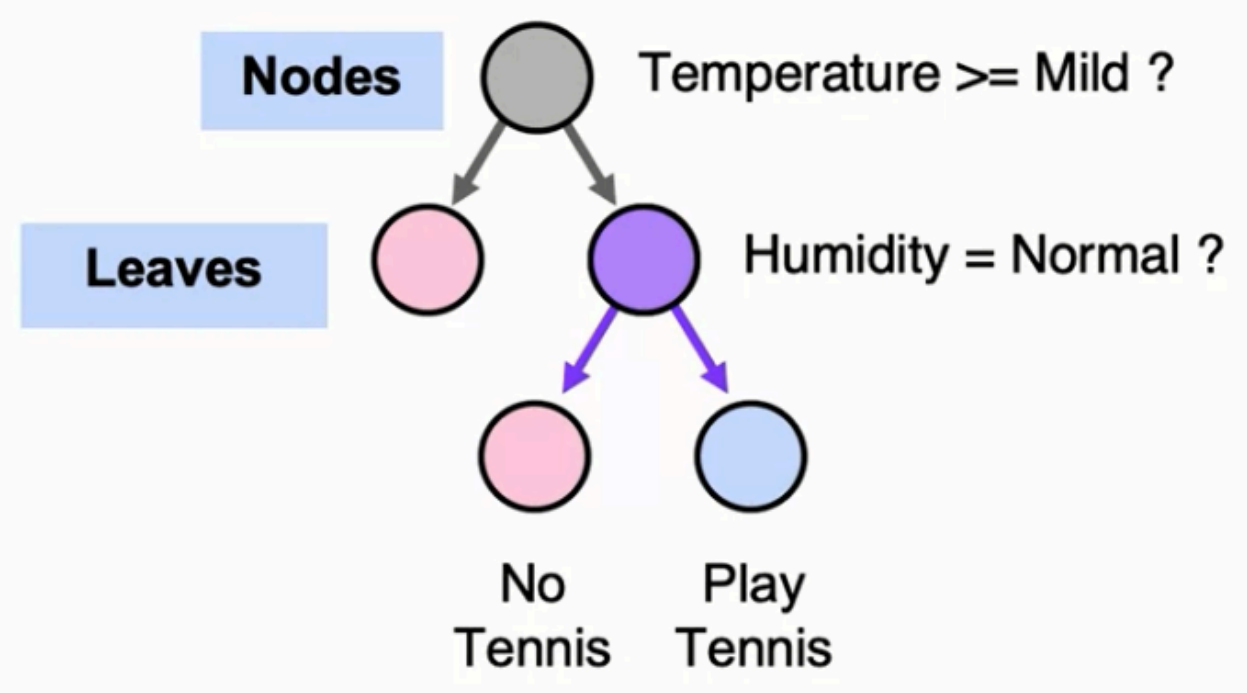
Course3_Module4

Decision Trees

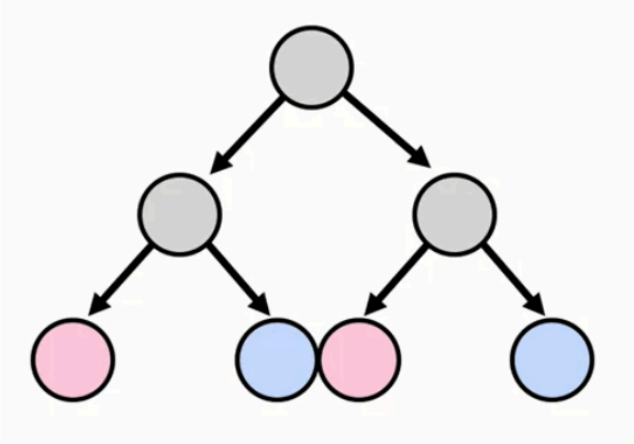


1) Bài toán & trực giác nhanh

- Mục tiêu: dự đoán nhãn (ví dụ: “Play Tennis” = Yes/No) dựa trên đặc trưng như nhiệt độ, độ ẩm, gió, outlook… Cây sẽ “chẻ” dữ liệu theo các câu hỏi yes/no để đi đến lá (kết luận).
- Cây cho **phân loại** (nhãn rời rạc) và cũng có **hồi quy** (giá trị liên tục: giá trị ở lá là trung bình các điểm rơi vào lá).

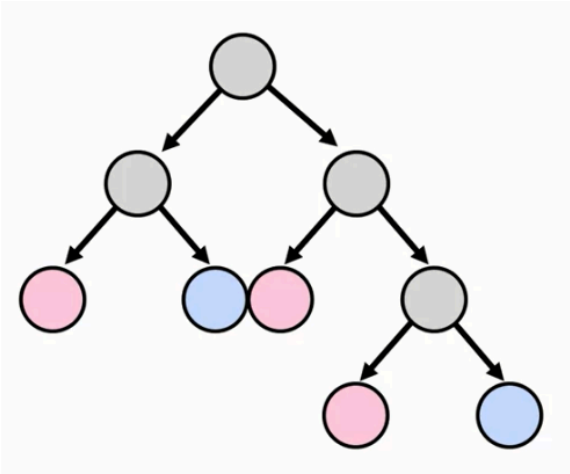
2) Xây cây kiểu “tham lam” (greedy)

- Ở mỗi nút, chọn **chia tách tốt nhất** theo một thước đo “độ không thuần khiết” (impurity).



- Select a feature and split data into binary tree.
- Continue splitting with available features.

- Lặp lại trên các nút con cho đến khi đạt tiêu chí dừng (lá thuần, đạt max_depth, v.v.).



Until:
- Leaf node(s) are **pure** (only **one class** remains).
- A **maximum depth** is reached.
- A **performance metric** is achieved.

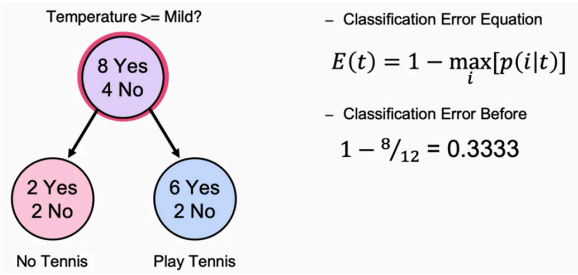
3) Các thước đo "impurity"

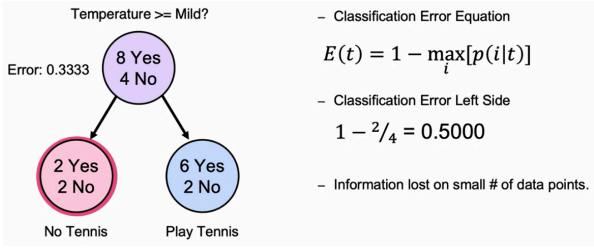
Cho bài toán phân loại 2 lớp (mở rộng được cho nhiều lớp), ký hiệu p_k là tỉ lệ mẫu lớp k trong một nút.

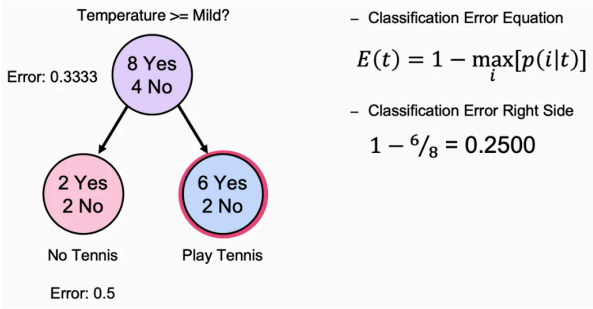
3.1) Classification error (tối giản)

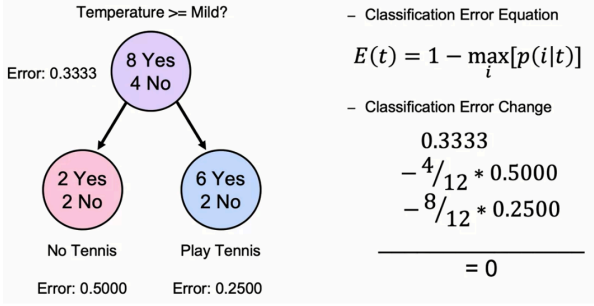
D

- Dễ hiểu nhưng **"phẳng"**: ít nhạy khi xác suất thay đổi gần 50/50 ⇒ thường **không tốt** để chọn split so với Entropy/Gini.





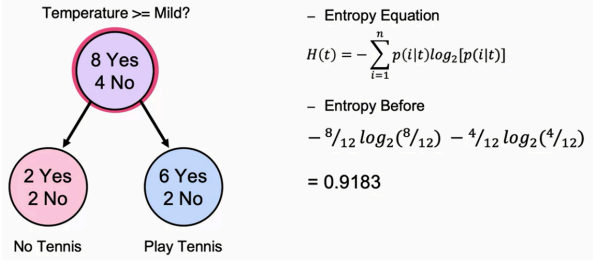


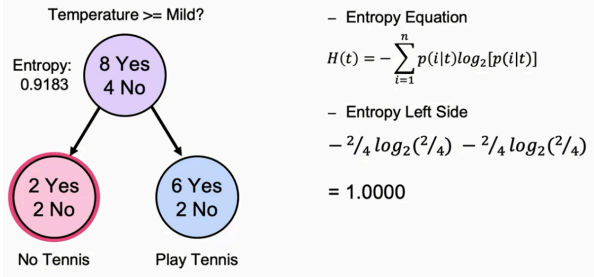


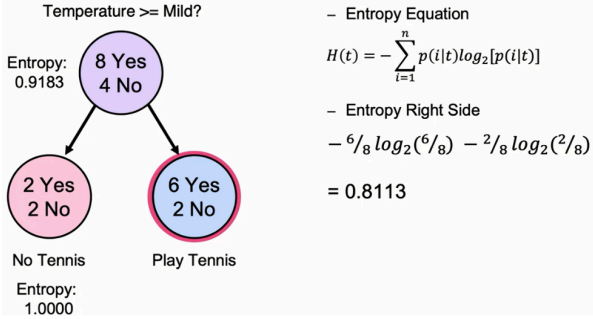
3.2) Entropy (ID3/Information Gain)

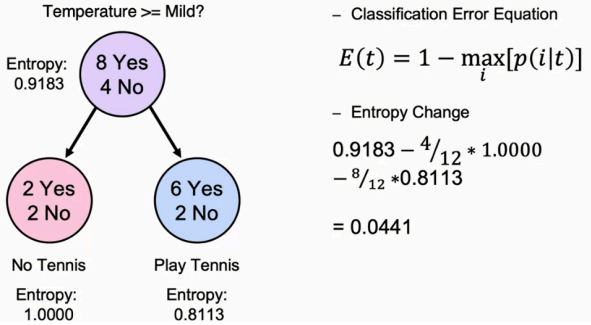
D

- Đo “mức hỗn loạn/không chắc chắn”. H = 0 khi nút thuần; lớn nhất khi phân bố đều.







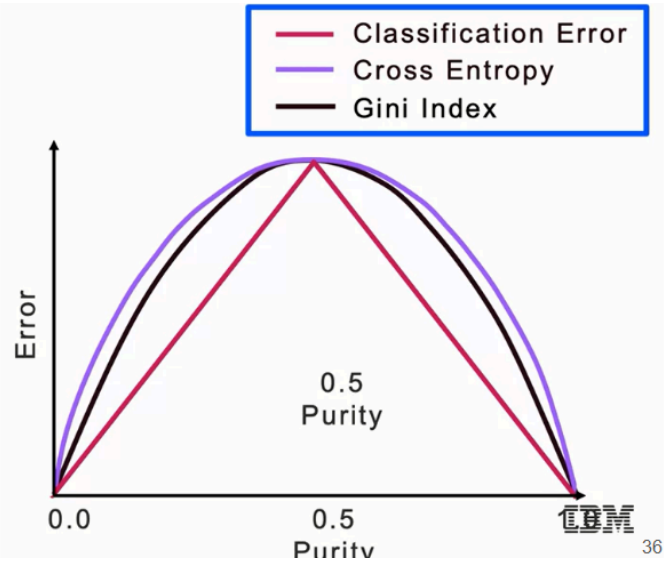


3.3) Gini (CART — mặc định trong scikit-learn)

- Hành vi tương tự Entropy (cũng có "độ cong"), **không có log** nên tính nhanh; là mặc định của `DecisionTreeClassifier` trong scikit-learn.

- In practice, **Gini index** often used for splitting.
- Function is similar to entropy - has bulge.
- Does not contain logarithm.

$$G(t) = 1 - \sum_{i=1}^{n} p(i|t)^2$$



4) Information Gain (IG) — vì sao Entropy/Gini hay hơn Classification error?

- Ý tưởng: một split tốt phải **giảm bất định** nhiều nhất. Với Entropy:

- Classification error “phẳng” ⇒ dễ rơi vào tình huống IG ≈ 0 dù split hợp lý; Entropy/Gini có “độ cong” ⇒ tiếp tục khuyến khích tách đến khi lá thuần hơn.

Ví dụ tính IG — **step by step (nhị phân, Entropy)**

Giả sử một nút cha có 10 mẫu: 6 Pos, 4 Neg.

1. Entropy nút cha:

$$$$

2. Sau khi split theo thuộc tính A, ta có hai nút con:

- Con L: 4 Pos, 0 Neg ⇒ $H_L = 0$ (thuần)
- Con R: 2 Pos, 4 Neg ⇒ $H_R = -\left(\frac{2}{6}\log_2\frac{2}{6} + \frac{4}{6}\log_2\frac{4}{6}\right) \approx 0.918$

3. Entropy trung bình sau split:

$$$$

4. **IG**:

$$$$

5) Tiêu chí dừng (khi nào ngừng tách?)

- **Lá thuần** (impurity ~ 0) hoặc **không còn split nào làm tăng IG/giảm impurity**.
- Giới hạn cấu trúc: `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_leaf_nodes`, hoặc ngưỡng `min_impurity_decrease`.

6) Overfitting & Pruning

- Cây dễ **overfit** (độ biến thiên cao: thay dữ liệu một chút có thể đổi cấu trúc nhiều). Cách khắc phục: giới hạn độ sâu/lá, **pruning.**
- **Cost-Complexity Pruning (CART)**: tối ưu $R_\alpha(T) = R(T) + \alpha\,|T|$ (độ lỗi + “phạt” theo số nút). Trong scikit-learn, tham số `ccp_alpha` càng lớn ⇒ cắt tỉa càng mạnh; có hàm `cost_complexity_pruning_path` để dò bộ α ứng viên.

7) Ưu / nhược điểm

- **Ưu**: dễ diễn giải (if–then–else), xử lý cả nhị phân/ordinal/liên tục, ít cần chuẩn hoá đặc trưng.
- **Nhược**: dễ overfit, nhạy với nhiễu/biến động dữ liệu; cần cross-validation, giới hạn độ sâu hoặc pruning.

8) Ghi nhớ nhanh (cheatsheet)

- **Thứ tự ưu tiên split** (thực dụng): dùng **Gini** (nhanh, mặc định sklearn) hoặc **Entropy + IG** (lý thuyết ID3). Tránh dùng **Classification error** để chọn split vì quá "phẳng".
- **Dừng + Regularize**: `max_depth`, `min_samples_leaf`, `min_samples_split`, `max_leaf_nodes`, `min_impurity_decrease`.
- **Pruning (hậu tia)**: dò `ccp_alpha` với `cost_complexity_pruning_path`, chọn α cân bằng bias–variance.

Phụ lục A — So sánh ba thước đo impurity (2 lớp)

Đo lường	Công thức	Ý nghĩa nhanh	Phạm vi
Classification error	$1 - \max(p, 1-p)$	Tỉ lệ sai nếu chọn lớp đa số	$[0, 0.5]$
Gini	$1 - (p^2 + (1-p)^2)$	Xác suất chọn ngẫu nhiên & gán nhầm	$[0, 0.5]$
Entropy	$-[p \log_2 p + (1-p)\log_2(1-p)]$	"Hỗn loạn"/bất định thông tin	$[0, 1]$

> Trong thực hành chọn split, Gini/Entropy tốt hơn Classification error vì nhạy với thay đổi gần 0.5.

Phụ lục B — sklearn

```
from sklearn.tree import DecisionTreeClassifier
```

```
clf = DecisionTreeClassifier(
    criterion="gini",       # hoặc "entropy"
    max_depth=None,         # đặt số nguyên để chống overfit
    min_samples_leaf=1,
    min_samples_split=2,
    ccp_alpha=0.0           # >0 để bật cost-complexity pruning
)
clf.fit(X_train, y_train)
```

- Gợi ý: thử nhiều `ccp_alpha` từ `cost_complexity_pruning_path` và chọn theo cross-val.