@SSRoy

DECISION TREE CLASSIFICATION

A Simple Complete Tutorial with a Solved Numerical Example

1. What a Decision Tree Is

A decision tree is a model that predicts the **class** of an item by asking a series of simple yes/no questions about its features. It looks like an upside-down tree:

- **Root node:** the top node holding all the training data.
- **Internal node:** a node that asks a question (a split) on one feature and sends the data to its branches.
- **Leaf (terminal) node:** a final node that gives a class label as the answer.

Building the tree means repeatedly **splitting** the data into purer and purer groups, until each group is (almost) all one class. This tutorial covers the **classification** tree only.

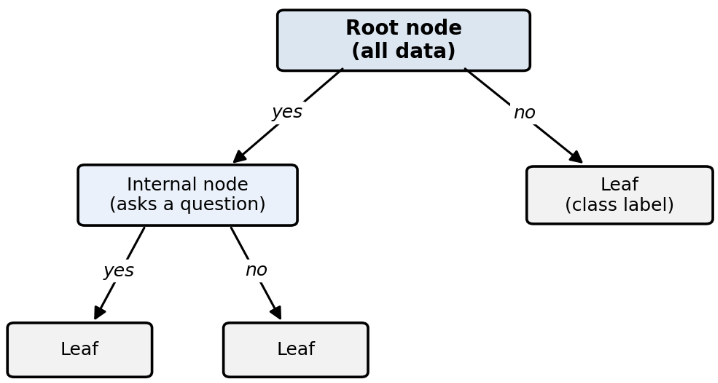

Figure 1. Anatomy of a decision tree: a root node splits into internal nodes and finally into leaves that give class labels.

2. Node Impurity — Measuring "Mixed-ness"

A node is **pure** if all its items belong to one class, and **impure** if the classes are mixed. We need a number to measure this. Let a node t contain items from J classes, and let p(j|t) be the proportion of class j at node t. The impurity is written i(t).

Gini impurity (the CART default)

$$\mathrm{Gini}(t) = 1 - \Sigma\, p(j|t)^2 \quad , \quad \text{summed over } j = 1 \text{ to } J$$

Gini is 0 when the node is pure (one class), and largest when classes are evenly mixed. For two classes it ranges from 0 to 0.5.

Entropy (an alternative measure)

$$\mathrm{Entropy}(t) = -\, \Sigma\, p(j|t) \cdot \log_2 p(j|t)$$

Entropy is 0 for a pure node and largest (= 1 for two equal classes) when fully mixed. Gini and entropy behave very similarly; CART uses Gini by default.

3. Choosing the Best Split

At a node, we try every possible split s (each feature, each threshold) and keep the one that reduces impurity the most. A split sends the parent node t into a left child t_l and a right child t_r. Let p_l and p_r be the fractions of items going left and right. The **goodness of a split** is the drop in impurity:

$$\Delta i(s,t) = i(t) - p_l \cdot i(t_l) - p_r \cdot i(t_r)$$

Here i(t) is the parent impurity and the two subtracted terms are the **weighted** impurities of the children. The best split is the one with the **largest Δi**. (When entropy is used, Δi is called **information gain**.)

4. Growing, Stopping, and Class Assignment

- **Grow:** start at the root and keep splitting each node using the best split, building the tree downward.
- **Stop:** stop splitting a node when it is pure, when it has too few items to split, or when no split improves impurity.
- **Assign a class to a leaf:** label each leaf with the **majority class** of the items in it. A new item is classified by dropping it down the tree to a leaf and taking that leaf's label.

5. Pruning — Avoiding an Over-Grown Tree

A fully grown tree fits the training data perfectly but **overfits** — it memorises noise and predicts poorly on new data. CART fixes this by growing a large tree and then **pruning** it back using **cost-complexity pruning**. For a tree T we define:

$$R_\alpha(T) = R(T) + \alpha \cdot |\tilde{T}|$$

where R(T) is the tree's error (misclassification cost), $|\tilde{T}|$ is the number of leaves (its size/complexity), and $\alpha \geq 0$ is a penalty for each leaf. Small α favours large accurate trees; large α favours small simple trees. Increasing α prunes away the weakest branches, giving a sequence of trees; the best α (and hence the right-sized tree) is chosen by **cross-validation**.

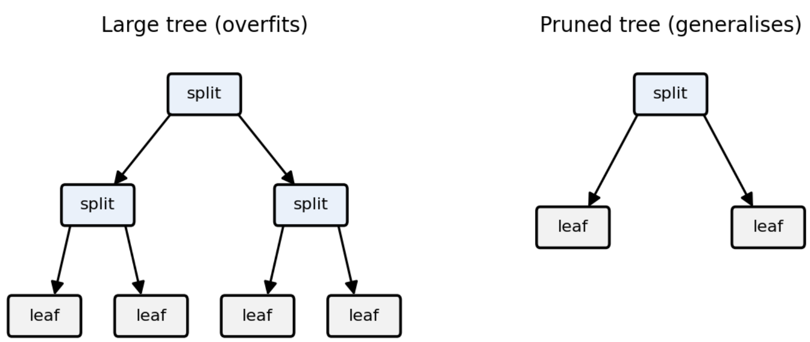

Figure 2. Pruning cuts a large over-fitted tree back to a smaller tree that generalises better to new data.

6. Solved Numerical Example

Ten days are recorded. Target: **PlayOutside** (Yes/No). Two features: Weather (Sunny/Rainy) and Temperature (Hot/Cool). We decide the best first split using Gini.

Day	Weather	Temperature	PlayOutside
1	Sunny	Hot	Yes
2	Sunny	Hot	Yes
3	Sunny	Cool	Yes
4	Sunny	Cool	Yes
5	Sunny	Hot	No
6	Rainy	Cool	No
7	Rainy	Cool	No
8	Rainy	Hot	No

Day	Weather	Temperature	PlayOutside
9	Rainy	Cool	No
10	Rainy	Hot	Yes

Class counts: 5 Yes, 5 No out of 10.

Step 1 — Impurity of the parent (root)

$$\text{Gini(parent)} = 1 - (5/10)^2 - (5/10)^2 = 1 - 0.25 - 0.25 = 0.5$$

Step 2 — Try the split on Weather

Sunny (5 days): 4 Yes, 1 No. Rainy (5 days): 1 Yes, 4 No.

$$\text{Gini(Sunny)} = 1 - (4/5)^2 - (1/5)^2 = 1 - 0.64 - 0.04 = 0.32$$

$$\text{Gini(Rainy)} = 1 - (1/5)^2 - (4/5)^2 = 0.32$$

Weighted child impurity (each child is 5/10 of the data):

$$(5/10)(0.32) + (5/10)(0.32) = 0.32$$

$$\Delta i(\text{Weather}) = 0.5 - 0.32 = 0.18$$

Step 3 — Try the split on Temperature

Hot (5 days): 3 Yes, 2 No. Cool (5 days): 2 Yes, 3 No.

$$\text{Gini(Hot)} = 1 - (3/5)^2 - (2/5)^2 = 0.48\ ,\quad \text{Gini(Cool)} = 0.48$$

$$\text{Weighted} = (5/10)(0.48) + (5/10)(0.48) = 0.48$$

$$\Delta i(\text{Temperature}) = 0.5 - 0.48 = 0.02$$

Step 4 — Choose the best split

Compare the impurity drops:

Split	Δi (impurity drop)	Decision
Weather	0.18	best — chosen
Temperature	0.02	weak

Result: Weather gives the larger Δi (0.18 vs 0.02), so the root splits on Weather. The Sunny branch is mostly “Yes” and the Rainy branch mostly “No”, so each becomes a leaf labelled by its majority class: Sunny → Yes, Rainy → No. A new day is then classified simply by its weather.

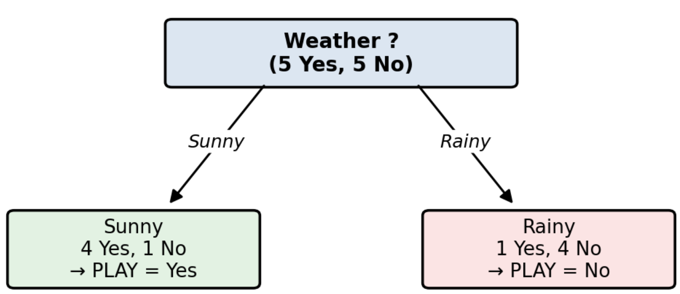

Figure 3. The resulting decision tree for the worked example: split on Weather, then read the class off each leaf.

7. Summary

- A decision tree splits data into purer groups and reads a class off a leaf.
- Impurity is measured by Gini = $1 - \Sigma\, p(j|t)^2$ (or entropy).
- The best split maximises the impurity drop $\Delta i(s,t) = i(t) - p_l\, i(t_l) - p_r\, i(t_r)$.
- Grow the tree, assign each leaf its majority class, then prune with $R_\alpha(T) = R(T) + \alpha|\tilde{T}|$ to avoid overfitting.

Reference: L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees (1984), Chapters 1–4 (classification tree).