

# Lecture 12

## Decision Trees

Used for classification and regression tasks

It consists of:

- Nodes: represent tests on features (attributes).
- Branches: outcomes of the tests.
- Leaves (terminal nodes): represent the class labels (for classification) or predicted values (for regression).

Working:

1. Starts at the root node with the entire dataset.
2. Recursively splits data based on the most informative feature at each node.
3. Splitting criteria aim to create groups that are as homogeneous as possible (all one class in classification).
4. Splitting continues until a stopping condition is met:
  - a. All samples in a node belong to one class,
  - b. No features remain to split,
  - c. Or a maximum tree depth is reached.

### Hunt's Algorithm:

1. Start at the root node with all training samples.
2. If all samples belong to the same class:
  - Create a leaf node with that class label.
  - Stop splitting here.
3. If samples belong to multiple classes:
  - Select the best attribute (feature) to split the data, based on a splitting criterion (e.g., information gain, Gini impurity).
  - Partition the dataset into subsets based on this attribute's values.
4. For each subset:
  - Create a child node.
  - Recursively apply Hunt's algorithm on the subset:
  - Repeat steps 2 and 3.
5. Continue until you reach pure subsets or no attributes remain.

### Ways to split a categorical attribute:

#### 1. Binary Split

- a. The attribute's values are divided into two groups (two branches).
- b. Common for both continuous and categorical attributes.
- c. Example for a numeric attribute (e.g., Age):
  - i. Split into  $\text{Age} < 30$  and  $\text{Age} \geq 30$ .
- d. Example for categorical attribute:
  - i. Group categories {Red, Blue} vs {Green}.
- e. Pros: Simpler tree structure, easier to interpret and prune

#### 2. Multi – Way Split

- a. The attribute's distinct values create more than two branches.
- b. Usually used with categorical attributes that have many categories.
- c. Example: For a city attribute with values {London, Paris, NY}, make 3 branches, one for each city.
- d. Pros: Can be intuitive and directly splits based on each category but can lead to more complex trees.

Now the attributes (features) can be **categorical** or **continuous** (numeric), and that the way we split these attributes in decision trees depends on their type.

- **Categorical attributes:** splits can be binary or multi-way based on distinct categories.
- **Continuous attributes:** techniques to find effective splits are such as binning or thresholding.

### Ways to handle a continuous attribute:

#### 1. Binning (Discretization) Before Tree Building

- a. Divide continuous values into discrete bins or groups before using them in the tree.
- b. Binning can be done using methods like:
  - i. Equal-width or equal-frequency binning
  - ii. Clustering-based binning (group similar values together)
- c. This converts continuous variables into categorical ones, simplifying splitting.

#### 2. Dynamic Threshold Computation While Building the Tree

- a. Instead of pre-binning, determine the best splitting threshold  $t$  for the attribute during tree construction.
- b. The algorithm tries splits like  $A > t$  vs  $A \leq t$
- c. It searches for  $t$  that best separates the classes based on impurity measures (Gini, entropy).
- d. This is more flexible and often results in better trees.

### Example Binning:

Age: [22, 25, 47, 52, 46, 56, 55, 60]

Method: Equal-width binning into N bins

N = 3

Range = Max – Min = 60 – 22 = 38

$$\text{Bin width} = \frac{\text{Range}}{N} = \frac{38}{3} = 12.67$$

Define bins:

- Bin 1:  $22 + 12.67 = 34.67$ , there Bin 1 is  $22 \leq \text{Age} \leq 34.67$
- Bin 2:  $34.67 + 12.67 = 47.34$ , there Bin 2 is  $34.67 \leq \text{Age} \leq 47.34$
- Bin 3:  $47.34 + 12.67 = 60$ , there Bin 3 is  $47.34 \leq \text{Age} \leq 60$

Age	Bins
22	1
25	2
47	2
52	3

Age	Bins
46	2
56	3
55	3
60	3

### Example Thresholding:

Age	Class (Given)
22	Low
25	Low
47	High
52	High
46	High
56	High
55	High
60	High

Candidate thresholds are between the sorted age values:

$$\frac{(22 + 25)}{2} = 23.5; \frac{25 + 46}{2} = 35.5; \frac{46 + 47}{2} = 46.5; \frac{47 + 52}{2} = 49.5; \dots$$

For each candidate threshold  $t$ , split data into:

*Left group:*  $\text{Age} \leq t$  and *Right group:*  $\text{Age} \geq t$

Example split at  $t = 46.5$ :

*Left Group* = 22 (Low), 25 (Low), 46 (High)

*Right Group* = 47 (High), 52 (High), 55 (High), 56 (High), 60 (High)

## GINI Index

- The Gini Index measures the impurity or disorder of a dataset with respect to the class distribution.
- It quantifies how often a randomly chosen element from the set would be incorrectly labelled if it was labelled randomly according to the class distribution in that set.
- Aim: split data into subsets with the lowest Gini impurity (more “pure” subsets).

$$Gini = 1 - \sum_{i=1}^J p_i^2,$$

where  $p_i = \frac{\text{no. of samples of class } i}{\text{total no. of samples}}$  and  $J = \text{total no. of classes}$

- **Gini = 0**: All instances belong to one class (perfect purity).
- **Gini close to 0.5 (for binary classes)**: Maximum impurity (equal number of classes).
- Lower Gini means better splits.
- For **Gini = 1**, the sum of the squared class probabilities must be zero, i.e.

$$\sum_{i=1}^J p_i^2 = 0$$

Not possible since  $p_i > 0$  and  $\sum p_i = 1$ , this cannot happen.

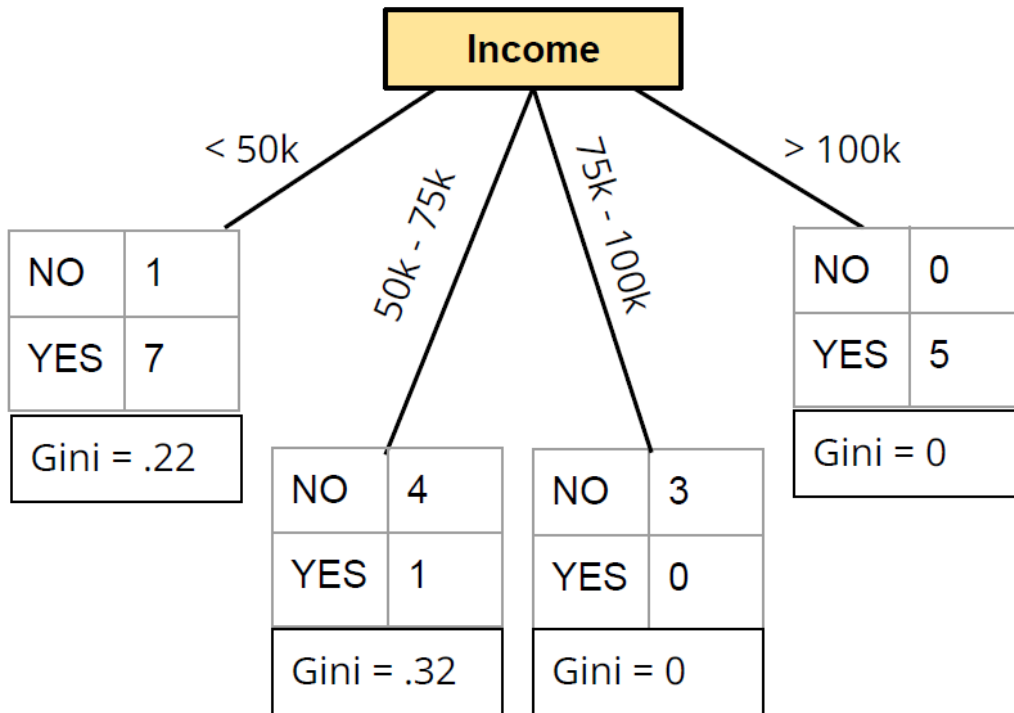
At minimum, if classes are evenly split among many classes,  $\sum p_i^2$  is minimized, never zero.

For Binary Class: Max possible value for Gini is 0.5 (distributions are equal)

For Multi Class: maximum Gini is when the classes are uniformly distributed:

$$p_i = \frac{1}{J} \rightarrow Gini = 1 - J \left( \frac{1}{J} \right)^2 = 1 - \frac{1}{J}$$

As  $J$  (number of classes) increases, the maximum Gini approaches 1 but never equals 1 exactly.



**Left Node: Income < 50k**

Calculate proportions:

$$p_{NO} = \frac{1}{8} = 0.125, \quad p_{YES} = \frac{7}{8} = 0.875$$

Compute Gini:

$$\begin{aligned}
 Gini &= 1 - (p_{NO}^2 + p_{YES}^2) = 1 - (0.125^2 + 0.875^2) \\
 &= 1 - (0.015625 + 0.765625) \\
 &= 1 - 0.78125 = 0.21875 \approx 0.22
 \end{aligned}$$

Similarly Do all.

**Weighted GINI Split:**

Income range	No Count	Yes Count	Total Count	Gini Impurity
< 50k	1	7	8	0.22
50k – 75k	4	1	5	0.32
75k – 100k	3	0	3	0
> 100k	0	5	5	0
Total	8	13	21	-

$$Gini_{split} = \sum_{i=1}^{Splits} \frac{N_i}{N} (Gini_i)$$

$$Gini_{split} = \left(\frac{8}{21}\right)(0.22) * \left(\frac{5}{21}\right)(0.32) * \left(\frac{3}{21}\right)(0) * \left(\frac{5}{21}\right)(0)$$

$$Gini_{split} = 0.0838 + 0.0762 + 0 + 0 = \mathbf{0.16}$$

Impurity of parent node (before split)

$$Gini_{parent} = 1 - \left( \left(\frac{8}{21}\right)^2 + \left(\frac{13}{21}\right)^2 \right)$$

$$= 1 - (0.381^2 + 0.619^2)$$

$$= 1 - 0.145 - 0.388 = \mathbf{0.472}$$

**Gini Gain** (Impurity reduction):

$$Gini\ Gain = Gini_{parent} - Gini_{split}$$

$$= 0.472 - 0.16 = \mathbf{0.312}$$

We Prefer Higher GINI Gain Between Two Splits

## Limitations

1. Overfitting (Too Complex Trees)
  - a. Decision trees can easily become very complex by creating many branches, perfectly fitting the training data but failing to generalize to new data.
  - b. Overfitting leads to poor predictive performance on unseen data.
2. Instability
  - a. Small changes in the training data can cause large changes in the tree structure.
  - b. Trees are sensitive to noise and outliers.
3. Bias Toward Features with More Levels
  - a. Features with many categories can dominate splits even if they aren't more informative.
4. Difficulty Modelling Some Relations
  - a. Trees split data axis-aligned, which may not capture relationships well if the true decision boundary is diagonal or nonlinear in complex ways.

## Solutions

1. Early Stopping (Pre-Pruning)
  - a. Stop the tree growth early, before perfectly fitting the data. Possible criteria:
  - b. Limit maximum tree depth.
  - c. Require minimum number of samples in leaf nodes.
  - d. Stop splitting if impurity improvement is below a threshold.
2. Pruning (post-pruning)
  - a. Grow a full tree first, then prune back branches that do not improve performance on validation data.
  - b. Pruning helps reduce complexity and overfitting.

Limitation	Brief Explanation	Solutions
Overfitting	Tree too complex fits noise	Early stopping, Pruning
Instability	Small data changes affect tree structure	Ensembles (Random Forest)
Biased splits	Preference for attributes with many categories	Feature selection
Limited boundary shaping	Axis-aligned splits may poorly capture decision boundary	Use other models or ensembles