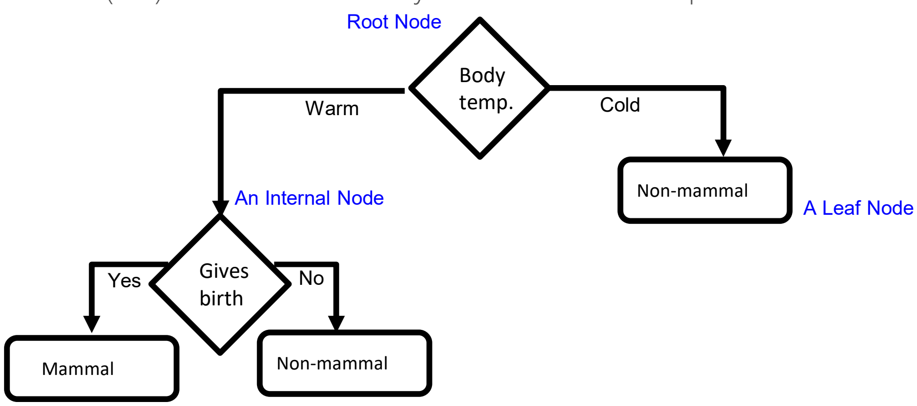
Decision Trees

- A Decision Tree (DT) defines a hierarchy of rules to make a prediction



- Root and internal nodes test rules. Leaf nodes make predictions
- Decision Tree (DT) learning is about learning such a tree from labeled data

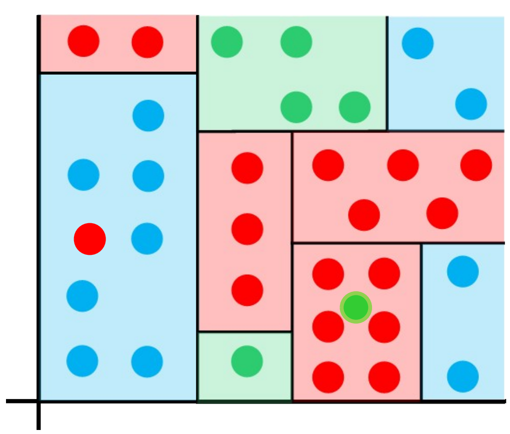
1

Learning Decision Trees with Supervision

- The basic idea is very simple
- Recursively partition the training data into homogeneous regions

What do you mean by "homogeneous" regions?

A homogeneous region will have all (or a majority of) training inputs with the same/similar outputs



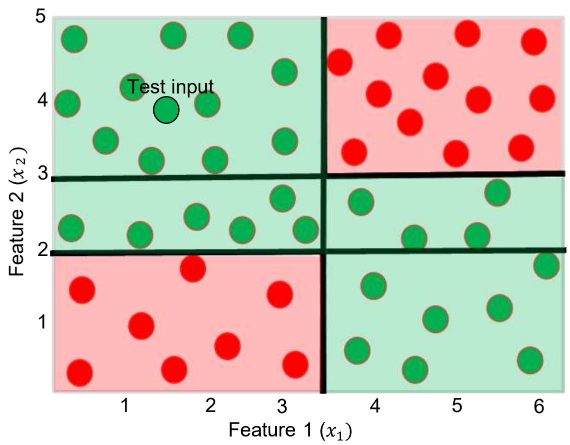
Even though the rule within each group is simple, we are able to learn a fairly sophisticated model overall (note in this example, each rule is a simple horizontal/vertical classifier but the overall decision boundary is rather sophisticated)

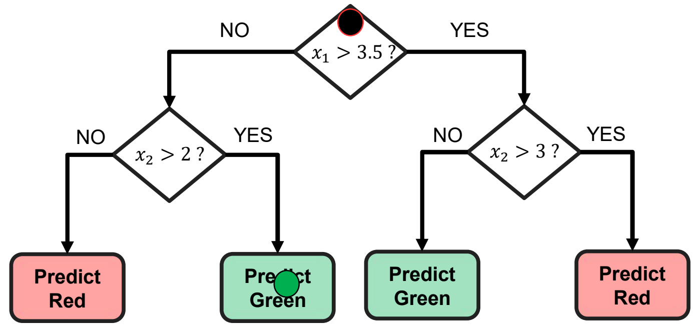
- Within each group, fit a simple supervised learner (e.g., predict the majority label)

2

3

Decision Trees for Classification





DT is very efficient at test time: To predict the label of a test point, nearest neighbors will require computing distances from 48 training inputs. DT predicts the label by doing just 2 feature-value comparisons! Way faster!

Remember: Root node contains all training inputs
Each leaf node receives a subset of training inputs

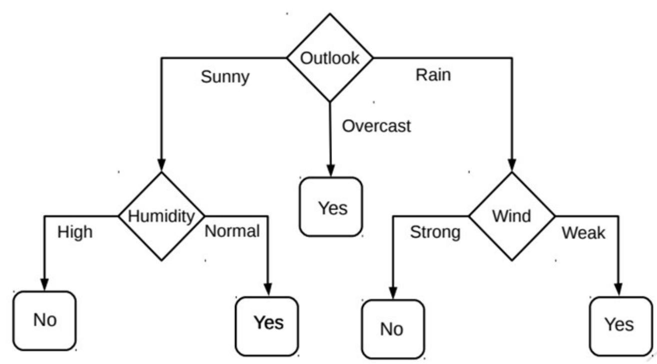
3

4

Decision Trees for Classification: Another Example

- Deciding whether to play or not to play Tennis on a Saturday
- Each input (Saturday) has 4 categorical features: Outlook, Temp., Humidity, Wind
- A binary classification problem (play vs no-play)
- Below Left: Training data, Below Right: A decision tree constructed using this data

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



Example credit: Tom Mitchell

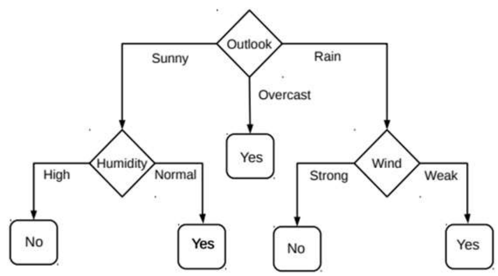
4

Decision Tree Construction: An Example

5

- Let's consider the playing Tennis example
- Assume each internal node will test the value of one of the features

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Question: Why does it make more sense to test the feature "outlook" first?
- Answer: Of all the 4 features, it's the most informative
  - It has the highest information gain as the root node

5

Entropy and Information Gain

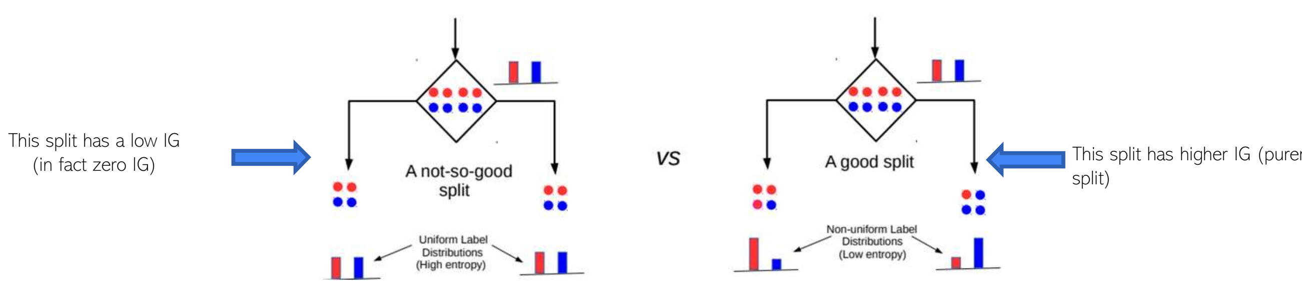
6

- Assume a set of labelled inputs $\boldsymbol{S}$ from C classes, p_c as fraction of class c inputs
- Entropy of the set $\boldsymbol{S}$ is defined as $\mathrm{H}(\boldsymbol{S}) = -\sum_{c \in C} p_c \log p_c$
- Suppose a rule splits $\boldsymbol{S}$ into two smaller disjoint sets $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$
- Reduction in entropy after the split is called information gain

Uniform sets have high entropy (all classes roughly equally present)

Skewed sets low entropy

$$IG = H(S) - \frac{|S_1|}{|S|} H(S_1) - \frac{|S_2|}{|S|} H(S_2)$$



6

Entropy and Information Gain

7

- Let's use IG based criterion to construct a DT for the Tennis example
- At root node, let's compute IG of each of the 4 features
- Consider feature "wind". Root contains all examples $S = [9+,5-]$

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

$$H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$$

$$S_{\text{weak}} = [6+, 2-] \Rightarrow \mathrm{H}(S_{\text{weak}}) = 0.811$$

$$S_{\text{strong}} = [3+, 3-] \Rightarrow \mathrm{H}(S_{\text{strong}}) = 1$$

$$IG(S, wind) = H(S) - \frac{|S_{\text{weak}}|}{|S|}H(S_{\text{weak}}) - \frac{|S_{\text{strong}}|}{|S|}H(S_{\text{strong}}) = 0.94 - 8/14 * 0.811 - 6/14 * 1 = 0.048$$

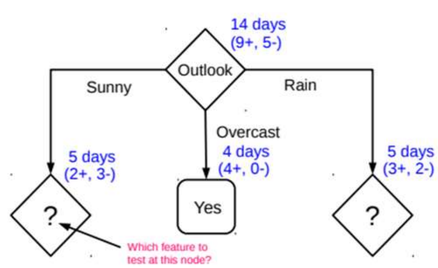
- Likewise, at root: IG(S, outlook) = 0.246, IG(S, humidity) = 0.151, IG(S,temp) = 0.029
- Thus we choose "outlook" feature to be tested at the root node
- Now how to grow the DT, i.e., what to do at the next level? Which feature to test next?
- Rule: Iterate - for each child node, select the feature with the highest IG

7

Growing the tree

8

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no



- Proceeding as before, for level 2, left node, we can verify that
  - IG(S,temp) = 0.570, IG(S, humidity) = 0.970, IG(S, wind) = 0.019
- Thus **humidity** chosen as the feature to be tested at level 2, **left** node
- No need to expand the middle node (already "pure" - all "yes" training examples)
- Can also verify that **wind** has the largest IG for the **right** node
- Note: If a feature has already been tested along a path earlier, we don't consider it again

8

9

Avoiding Overfitting in DTs

- Desired: a DT that is not too big in size, yet fits the training data reasonably
- Note: An example of a very simple DT is "decision-stump"
  - A decision-stump only tests the value of a single feature (or a simple rule)
  - Not very powerful in itself but often used in large ensembles of decision stumps
- Mainly two approaches to prune a complex DT
  - Prune while building the tree (stopping early)
  - Prune after building the tree (post-pruning)

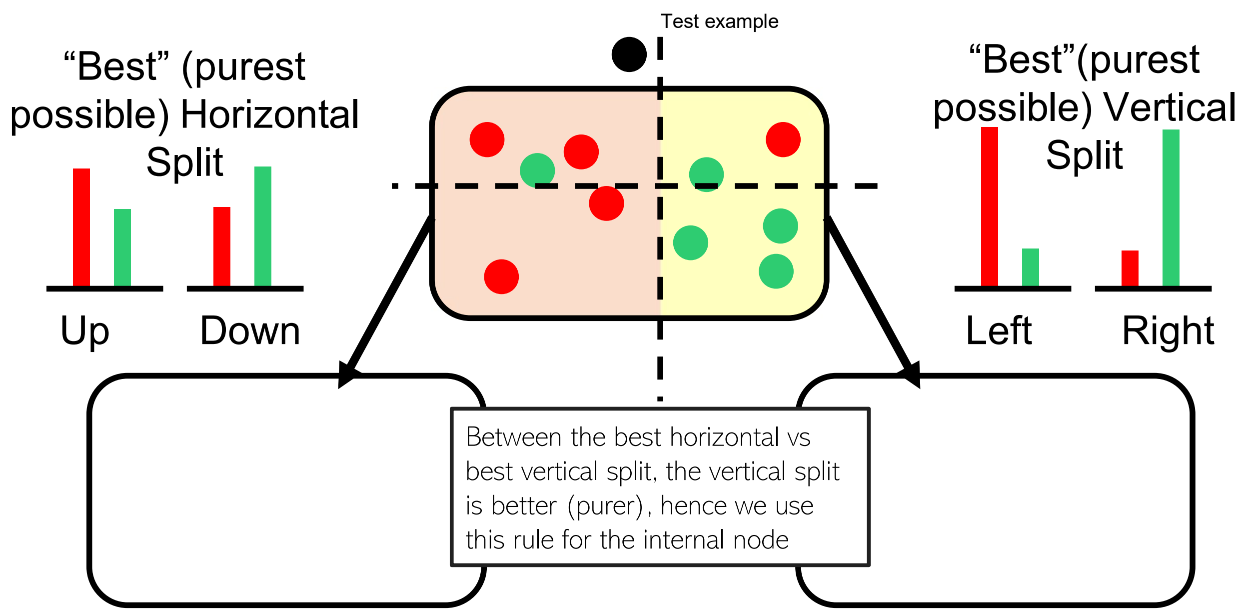
Either can be done using a validation set

- Gini-index defined as $\sum_{c=1}^{C} 1 - p_c{}^{\wedge}2$ can be an alternative to IG
- For DT regression, variance in the outputs can be used to assess purity

9

10

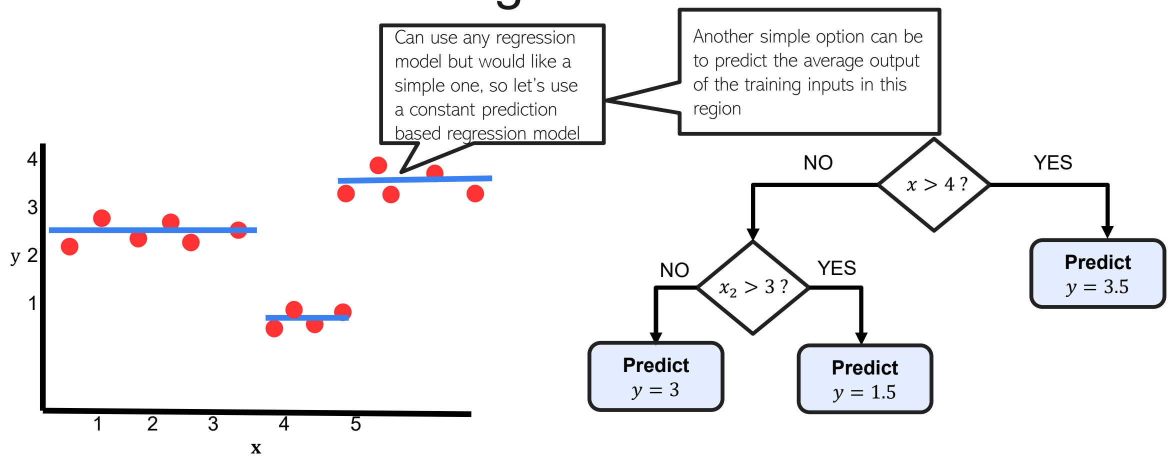
An Illustration: DT with Real-Valued Features



Between the best horizontal vs best vertical split, the vertical split is better (purer), hence we use this rule for the internal node

10

Decision Trees for Regression



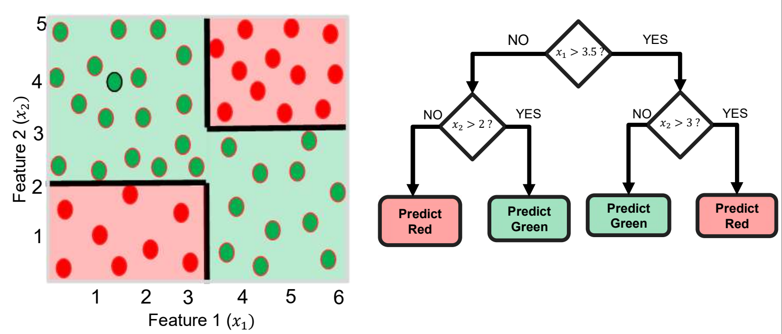
To predict the output for a test point, nearest neighbors will require computing distances from 15 training inputs. DT predicts the label by doing just at most feature-value comparisons! Way faster!

Decision Trees: A Summary

.. thus helping us learn complex rule as a combination of several simpler rules

Some key strengths:

- Simple and easy to interpret
- Nice example of "divide and conquer" paradigm in machine learning
- Easily handle different types of features (real, categorical, etc.)
- Very fast at test time
- Multiple DTs can be combined via ensemble methods: more powerful (e.g., Decision Forests; will see later)



- Used in several real-world ML applications, e.g., recommender systems, gaming (Kinect)

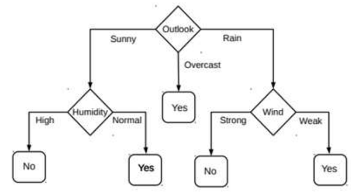
Some key weaknesses:

- Learning optimal DT is (NP-hard) intractable. Existing algos mostly greedy heuristics
- Can sometimes become very complex unless some pruning is applied

When to stop growing the tree?

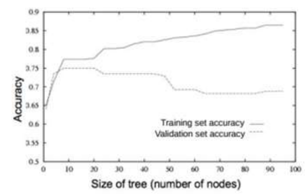
13

day	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

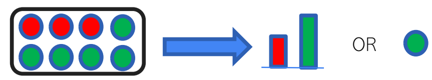


- Stop expanding a node further (i.e., make it a leaf node) when
  - It consist of all training examples having the same label (the node becomes "pure")
  - We run out of features to test along the path to that node
  - The DT starts to overfit (can be checked by monitoring the validation set accuracy)

To help prevent the tree from growing too much!



- Important: No need to obsess too much for purity
  - It is okay to have a leaf node that is not fully pure, e.g., this
  - At test inputs that reach an impure leaf, can predict probability of belonging to each class (in above example, p(red) = 3/8, p(green) = 5/8), or simply predict the majority label



13

- **Information Gain (IG)**
- **Definition:** Information Gain measures the reduction in entropy (uncertainty) after a dataset is split on an attribute.
- **Formula:**

$$IG(T, A) = Entropy(T) - \sum_{v \in \text{Values}(A)} \frac{|T_v|}{|T|} \cdot Entropy(T_v)$$

- **Used in:** ID3 and C4.5 decision tree algorithms.
- **Best for:** When you want to choose the attribute that gives the **most pure subgroups**.
- **Entropy:**

$$Entropy(S) = -\sum p_i \log_2(p_i)$$

- **In Big Data:** Can be computationally intensive on large datasets, especially with continuous attributes.

14

Gini Index

- **Definition**: Gini Index measures the impurity of a dataset; the lower the Gini, the purer the node.
- **Formula**:

$$Gini(S) = 1 - \sum_{i=1}^{n} p_i^2$$

- **Used in**: CART (Classification and Regression Trees).
- **Best for**: Faster computation than Information Gain; works well in large-scale datasets.
- **In Big Data**: Efficient and scalable, making it a preferred choice in systems like Spark MLlib.

Feature	Information Gain	Gini Index
Basis	Entropy	Probability of misclassification
Algorithm	ID3, C4.5	CART
Computation	More intensive	Less intensive
Bias	Toward attributes with more levels	Slightly less biased

15

Solved Example

Instance	Classification	a1	a2
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

Attribute: a1

$Values\,(a1) = T, F$

$S = [3+,\ 3-] \qquad Entropy(S) = 1.0$

$S_T = [2+,\ 1-] \qquad Entropy(S_T) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$S_F \leftarrow [1+,\ 2-] \qquad Entropy(S_F) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$

$$Gain\,(S, a1) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a1) = Entropy(S) - \frac{3}{6}Entropy(S_T) - \frac{3}{6}Entropy(S_F)$$

$$Gain(S, a1) = 1.0 - \frac{3}{6} * 0.9183 - \frac{3}{6} * 0.9183 = 0.0817$$

Attribute: a2

$Values\,(a2) = T, F$

$S = [3+,\ 3-] \qquad Entropy(S) = 1.0$

$S_T = [2+,\ 2-] \qquad Entropy(S_T) = 1.0$

$S_F \leftarrow [1+,\ 1-] \qquad Entropy(S_F) = 1.0$

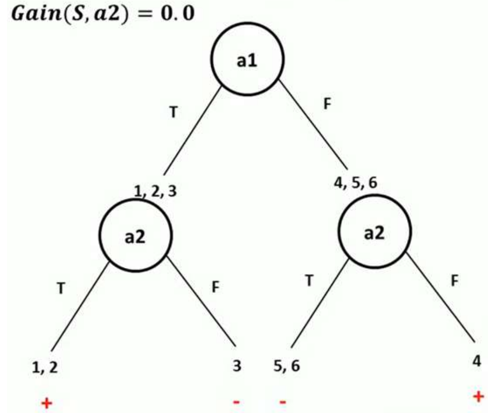
$$Gain\,(S, a2) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{4}{6}Entropy(S_T) - \frac{2}{6}Entropy(S_F)$$

$$Gain(S, a2) = 1.0 - \frac{4}{6} * 1.0 - \frac{2}{6} * 1.0 = 0.0$$

$Gain(S, a1) = 0.0817 \;\; - \; Maximum\ Gain$

$Gain(S, a2) = 0.0$



16

Solved Example

ID	Leaf Color	Size	Spots	Type
1	Green	Tall	No	Edible
2	Brown	Short	Yes	Poisonous
3	Green	Tall	Yes	Edible
4	Green	Short	No	Edible
5	Brown	Tall	No	Poisonous
6	Brown	Short	No	Poisonous
7	Green	Short	Yes	Poisonous
8	Brown	Tall	Yes	Poisonous
9	Green	Tall	No	Edible
10	Brown	Short	No	Poisonous

Target Counts:

- Edible = 4 (IDs 1, 3, 4, 9)
- Poisonous = 6 (IDs 2, 5, 6, 7, 8, 10)

$$Gini_{parent} = 1 - (0.4)^2 - (0.6)^2 = 1 - 0.16 - 0.36 = 0.48$$

🔀 Try splitting by **Spots**

Spots = Yes (IDs 2, 3, 7, 8):

- Poisonous: 3 (2, 7, 8)
- Edible: 1 (3)
- Gini =

$$1 - (3/4)^2 - (1/4)^2 = 1 - 0.5625 - 0.0625 = 0.375$$

Spots = No (IDs 1, 4, 5, 6, 9, 10):

- Poisonous: 3 (5, 6, 10)
- Edible: 3 (1, 4, 9)
- Gini =

$$1 - (0.5)^2 - (0.5)^2 = 0.5$$

🎯 **Weighted Gini:**

$$Gini_{split} = \frac{4}{10} \cdot 0.375 + \frac{6}{10} \cdot 0.5 = 0.15 + 0.3 = 0.45$$

➡️ **Improvement from 0.48 to 0.45.**

🔀 Try splitting by **Size**

Tall Plants (IDs 1, 3, 5, 8, 9):

- Types:
  - Edible: 1, 3, 9 → 3
  - Poisonous: 5, 8 → 2
- Gini =

$$1 - (3/5)^2 - (2/5)^2 = 1 - 0.36 - 0.16 = 0.48$$

Short Plants (IDs 2, 4, 6, 7, 10):

- Types:
  - Edible: 4 → 1
  - Poisonous: 2, 6, 7, 10 → 4
- Gini =

$$1 - (4/5)^2 - (1/5)^2 = 1 - 0.64 - 0.04 = 0.32$$

🎯 **Weighted Gini for Size split:**

$$Gini_{split} = \frac{5}{10} \cdot 0.48 + \frac{5}{10} \cdot 0.32 = 0.24 + 0.16 = 0.40$$

🔀 Try splitting by **Leaf Color**

Green (IDs 1, 3, 4, 7, 9):

- Edible: 4 (1, 3, 4, 9)
- Poisonous: 1 (7)
- Gini =

$$1 - (4/5)^2 - (1/5)^2 = 1 - 0.64 - 0.04 = 0.32$$

Brown (IDs 2, 5, 6, 8, 10):

- All 5 are Poisonous
- Gini =

$$1 - (1)^2 = 0$$

🎯 **Weighted Gini:**

$$Gini_{split} = \frac{5}{10} \cdot 0.32 + \frac{5}{10} \cdot 0 = 0.16$$

✅ **Huge improvement! Gini goes from 0.48 → 0.16**

Attribute	Gini After Split
Leaf Color	0.16 ✅ Best
Size	0.4
Spots	0.45
No Split (Parent)	0.48

Learning Outcomes

Category	Keywords / Phrases
Structure Elements	Root node, Decision node, Leaf node, Terminal node, Branch, Path, Subtree, Tree depth
Algorithms	ID3 (Iterative Dichotomiser 3), CART (Classification and Regression Trees)
Splitting Criteria	Entropy, Information Gain, Gini Impurity
Processes	Splitting, Recursive partitioning, Pruning, Pre-pruning, Post-pruning
Types of Trees	Classification Tree, Regression Tree, Binary Tree, Multi-way Tree
Performance Metrics	Accuracy, Precision, Recall, F1 Score, Confusion Matrix, ROC Curve
Challenges	Overfitting, Underfitting, Bias-variance tradeoff
Applications	Classification, Regression, Feature selection, Predictive modeling, Decision support