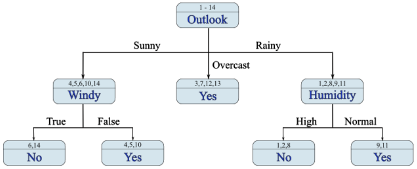
Decision Tree

April 6, 2025

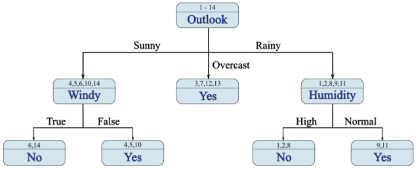
Decision Tree

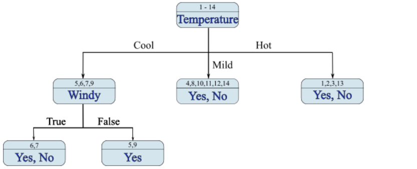
	Outlook	Temperature	Humidity	Windy	Play Golf
1	Rainy	Hot	High	False	**No**
2	Rainy	Hot	High	True	**No**
3	Overcast	Hot	High	False	**Yes**
4	Sunny	Mild	High	False	**Yes**
5	Sunny	Cool	Normal	False	**Yes**
6	Sunny	Cool	Normal	True	**No**
7	Overcast	Cool	Normal	True	**Yes**
8	Rainy	Mild	High	False	**No**
9	Rainy	Cool	Normal	False	**Yes**
10	Sunny	Mild	Normal	False	**Yes**
11	Rainy	Mild	Normal	True	**Yes**
12	Overcast	Mild	High	True	**Yes**
13	Overcast	Hot	Normal	False	**Yes**
14	Sunny	Mild	High	True	**No**



Decision Tree

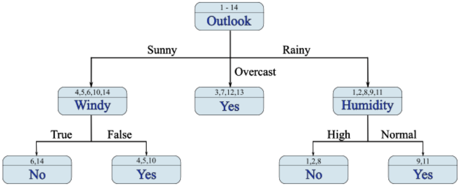
	Outlook	Temperature	Humidity	Windy	Play Golf
1	Rainy	Hot	High	False	**No**
2	Rainy	Hot	High	True	**No**
3	Overcast	Hot	High	False	**Yes**
4	Sunny	Mild	High	False	**Yes**
5	Sunny	Cool	Normal	False	**Yes**
6	Sunny	Cool	Normal	True	**No**
7	Overcast	Cool	Normal	True	**Yes**
8	Rainy	Mild	High	False	**No**
9	Rainy	Cool	Normal	False	**Yes**
10	Sunny	Mild	Normal	False	**Yes**
11	Rainy	Mild	Normal	True	**Yes**
12	Overcast	Mild	High	True	**Yes**
13	Overcast	Hot	Normal	False	**Yes**
14	Sunny	Mild	High	True	**No**





Structure and terminologies

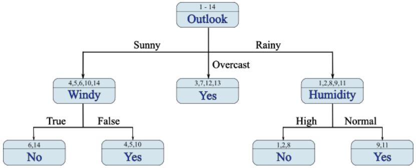
- **Tree structure:**
  - Root node: starts the decision process
  - Internal nodes: test on features
  - Branches: outcomes of tests
  - Leaf nodes: final decisions
- **Key concepts:**
  - Splits data based on features
  - Uses criteria like Gini impurity or information gain
  - Works for both classification and regression



Types of decision tree

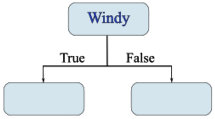
Types of decision tree is determined by the values of leaves.

- Classification decision tree
- Regression decision tree
- Probability decision tree

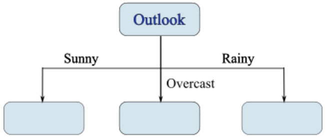


Feature and test

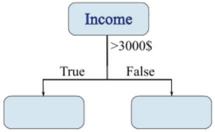
- **Binary feature:**
  Partition data into two subsets
- **Categorical feature:**
  Partition data into categories
- **Numeric feature:**
  Often partition data into two subsets using threshold



Binary Feature



Categorical Feature



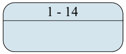
Numeric Feature

Decision tree algorithm

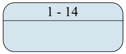
- **Iterative Dichotomiser 3 (ID3)**
  - Based on a greedy search
  - One step ahead algorithm
- **Algorithm**
  1. Start with a single node
  2. Calculate “Information Gain (IG)” of adding a split on each feature
  3. Add the split with the highest IG
  4. Sort data into the leaves of the new tree
  5. Continue recursively with remaining features until some stopping criteria are met

Decision tree algorithm

Decision tree algorithm

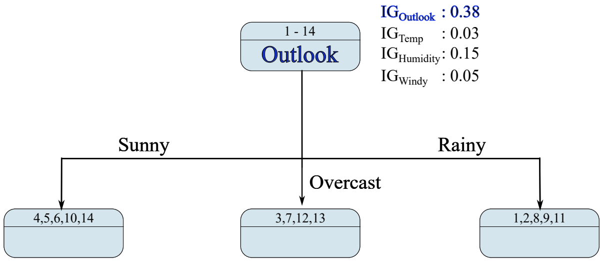


Decision tree algorithm

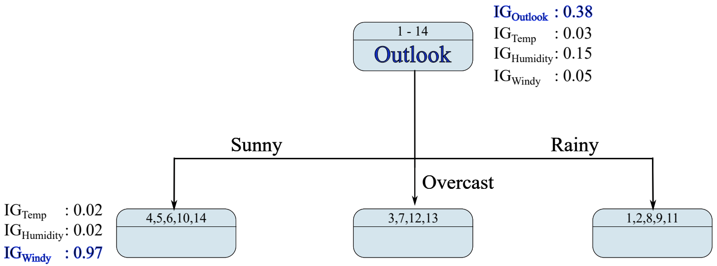


$\mathbf{IG_{Outlook}}$	**: 0.38**
IG_{Temp}	: 0.03
$IG_{Humidity}$	: 0.15
IG_{Windy}	: 0.05

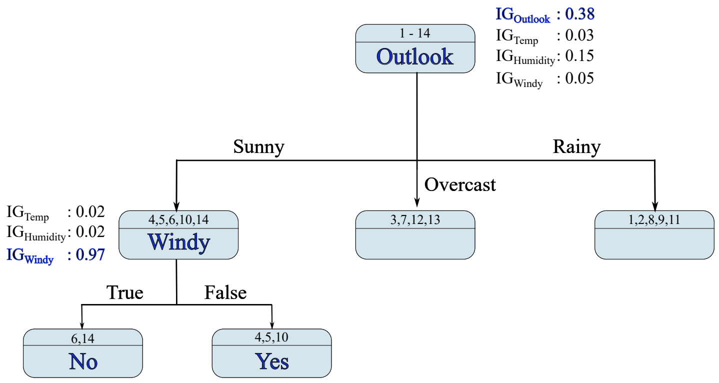
Decision tree algorithm



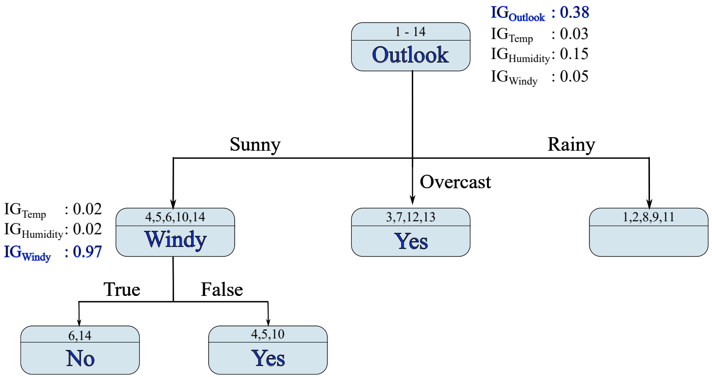
Decision tree algorithm



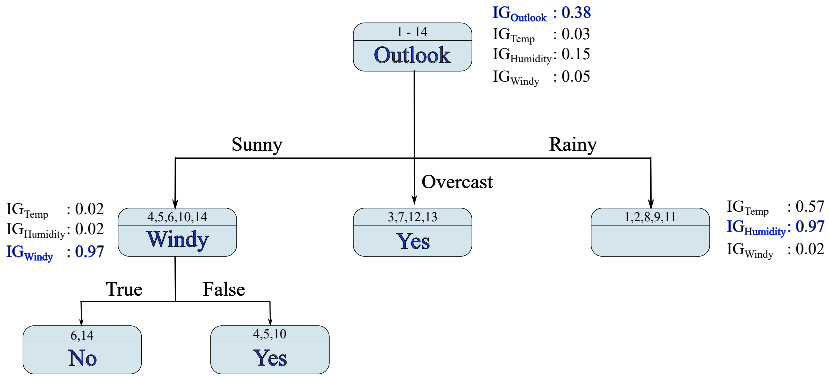
Decision tree algorithm



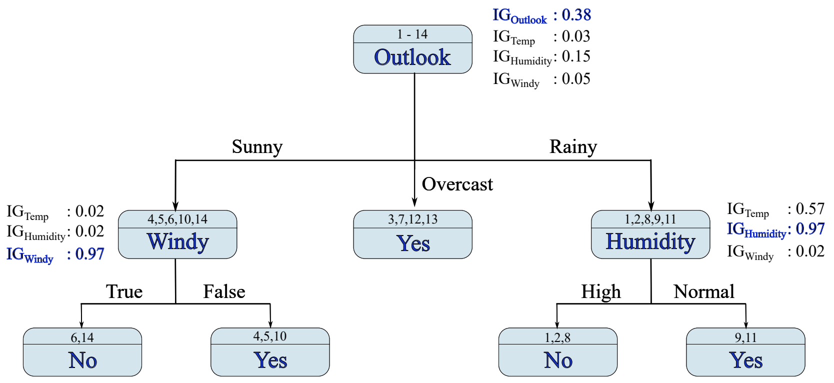
Decision tree algorithm



Decision tree algorithm



Decision tree algorithm

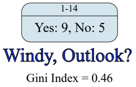


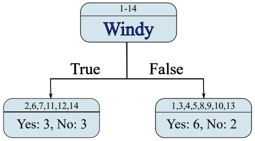
Feature selection for node splitting

- To select the best split to apply for a node, we need to calculate a score for each split corresponding to a feature.
- The split with the highest score will be chosen.
- Two most common measures used to measure the quality of a split are
  - Gini Gain
  - Information Gain

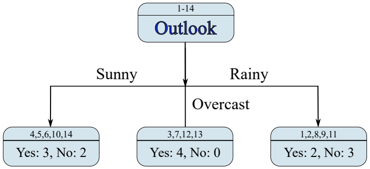
Gini Gain

- **Gini**
  - Probability of randomly picked two points belong to same class
  - $\text{Gini} = \sum_{i=1}^{C} p_i^2$
  - Measure purity/homogeneity
- **Gini Index (GI)**
  - Measure impurity
  - $\text{GI} = 1 - \text{Gini}$
  - $\text{GI}_{\text{split}} = \sum_{sn} \frac{|sn|}{|pn|} \text{GI}_{sn}$ (sn: sub-node, pn: parent node)
- **Gini Gain (GG)**
  - $\text{GG}_{\text{split}} = \text{GI}_{pn} - \text{GI}_{\text{split}}$





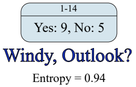
- Gini Index = 6/14*0.5 + 8/14*0.38 = 0.43
- Gini Gain = 0.46 - 0.43 = 0.03

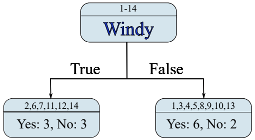


- Gini index = 5/14*0.48 + 4/14*0.0 + 5/14*0.48 = 0.34
- Information Gain = 0.46 - 0.34 = 0.12

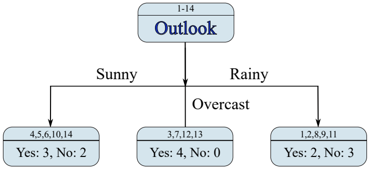
Information Gain

- **Entropy (E)**
  - Measure uncertainty, impurity
  - $E = -\sum_{i=1}^{C} p_i \log_2(p_i)$
  - $E_{split} = \sum_{sn} \frac{|sn|}{|pn|} E_{sn}$ (sn: sub-node, pn: parent node)
- **Information Gain (IG)**
  - Measure the amount of gained information
  - $IG_{split} = E_{pn} - E_{split}$





- Entropy = 6/14*1 + 8/14*0.81 = 0.89
- Information Gain = 0.94 - 0.89 = 0.05



- Entropy = 5/14*0.97 + 4/14*0.0 + 5/14*0.97 = 0.56
- Information Gain = 0.94 - 0.56 = 0.38

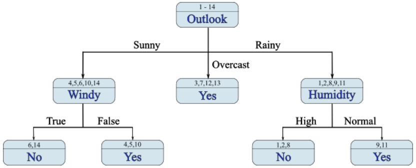
Stopping criteria

As the decision tree algorithm is recursive, we must define stopping criteria. Following are some common stopping criteria.

- Running out of features
- Setting min IG/GG for splitting
- Setting a minimum number of points in the node for splitting
- Leaf is pure
- Setting max to the depth of the tree
- Setting a maximum number of nodes in the tree

Advantages and disadvantages

- **Advantages**
  - Interpretability
  - Less Data Preparation
  - Non-Parametric
  - Versatility: classification and regression, multiple outputs
  - Non-Linearity
- **Disadvantages**
  - Overfitting
  - Bias toward certain features, especially features with many distinct values.
  - Training can be time-consuming, especially for large datasets or deep trees



Decision tree pruning

Pruning aims to avoid overfitting in decision trees.

- **Pre-pruning:** applied before the tree is fully grown. It is done by setting up stopping criteria.
  - Minimum number of examples in a node
  - Maximum depth for a tree
  - Minimum information gain or Gini gain
- **Post-pruning:** applied after the tree is fully grown. Nodes and subtrees are replaced with leaves to reduce complexity
  - **Reduced error pruning:** fusing leaves at their parent node if the fusion does not change the prediction outcome.
  - **Cost-complexity pruning:** removes subtrees based on a cost-complexity function that balances the error rate and complexity of the tree.

Cost complexity pruning (CCP)

- T_t: Subtree rooted at node t
- T_{max}: Full tree
- $|T_t|$: # leaves on the tree
- R(t): Mis-classification error rate of node t

$$R(t) = r(t) \times p(t) = \frac{m}{|t|} \times \frac{|t|}{|N|} = \frac{m}{N}$$

- R(T): Mis-classification error rate of the tree

$$R(T) = \sum_{t \in leaves(T)} R(t)$$

where:

- m, |t|, N: # minority, # samples on node t, # train samples

Cost complexity pruning (CCP)

Objective function:

$$\min_{T \leq T_{max}} R(T) + \alpha|T|$$
$$\alpha = 0 \Longrightarrow T_0 = T_{max}, \ \ldots, \alpha = \infty \Longrightarrow T_\infty = T_{min} \text{ (Root node only)}$$
$$\text{(where: } \alpha\text{: cost complexity parameter, } |T|\text{: complexity)}$$

The trees are nested: $T_\infty \leq \ldots \leq T_0 = T_{max}$

$$\min_{T \leq T_{max}} R(T) + \alpha|T|$$

Cost complexity pruning (CCP)

Pruning rule: Prune the subtree T rooted at node t if the reward is less than the penalty:

$$\frac{R(t)-R(T)}{|T|-1} < \alpha$$

$$R(t) - R(T): \text{ Reward}, \ \alpha(|T|-1): \text{ Penalty}$$

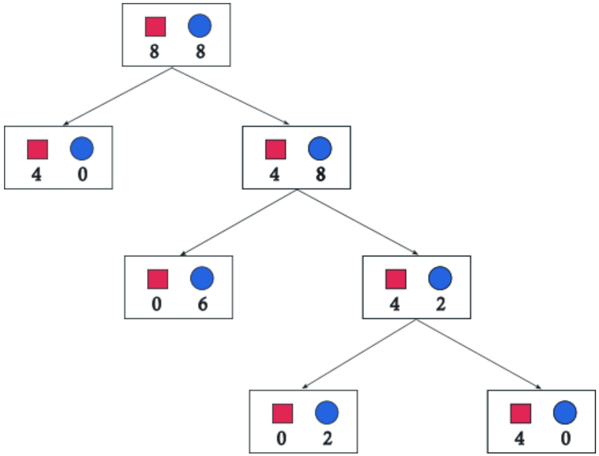
Algorithm:

- For each internal node t, compute: $\frac{R(t)-R(T)}{|T|-1}$
- For a given α, prune the subtree rooted at t if: $\frac{R(t)-R(T)}{|T|-1} < \alpha$

Note that:

- If we increase α, we can start from the previously obtained tree.
- Optimal α will be chosen using cross-validation

Cost complexity pruning (CCP)


8 8
4 0
4 8
0 6
4 2
0 2
4 0