

# Chapter ML:III

## III. Decision Trees

- ❑ Decision Trees Basics
- ❑ Impurity Functions
- ❑ Decision Tree Algorithms
- ❑ Decision Tree Pruning

# Decision Trees Basics

## Specification of Classification Problems [[ML Introduction](#)]

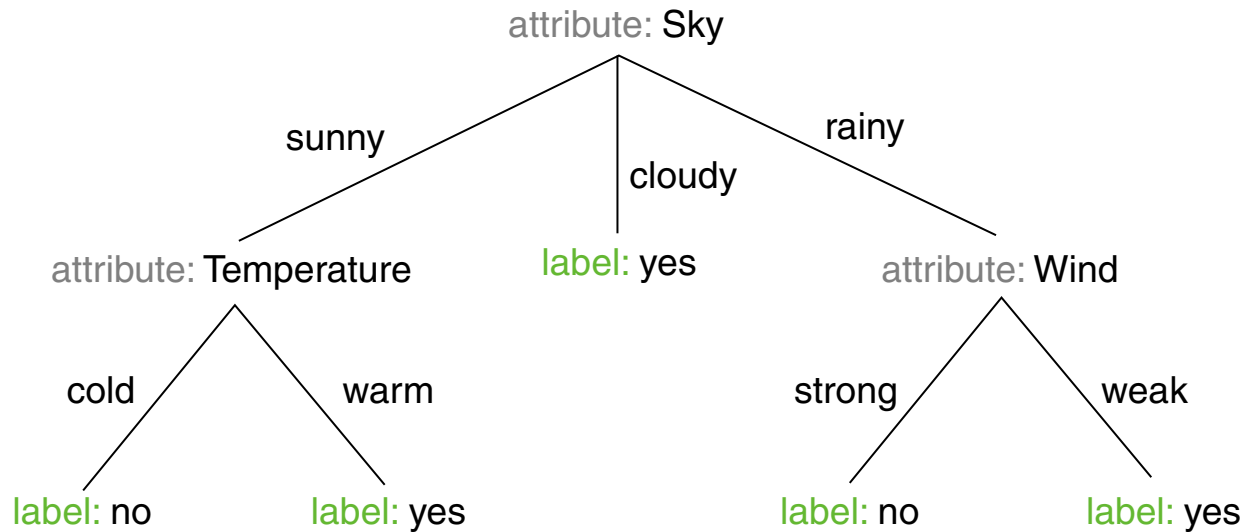
Characterization of the model (model world):

- $X$  is a set of feature vectors, also called feature space.
- $C$  is a set of classes.
- $c : X \rightarrow C$  is the ideal classifier for  $X$ .
- $D = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \dots, (\mathbf{x}_n, c(\mathbf{x}_n))\} \subseteq X \times C$  is a set of examples.

# Decision Trees Basics

## Decision Tree for the Concept “EnjoySport”

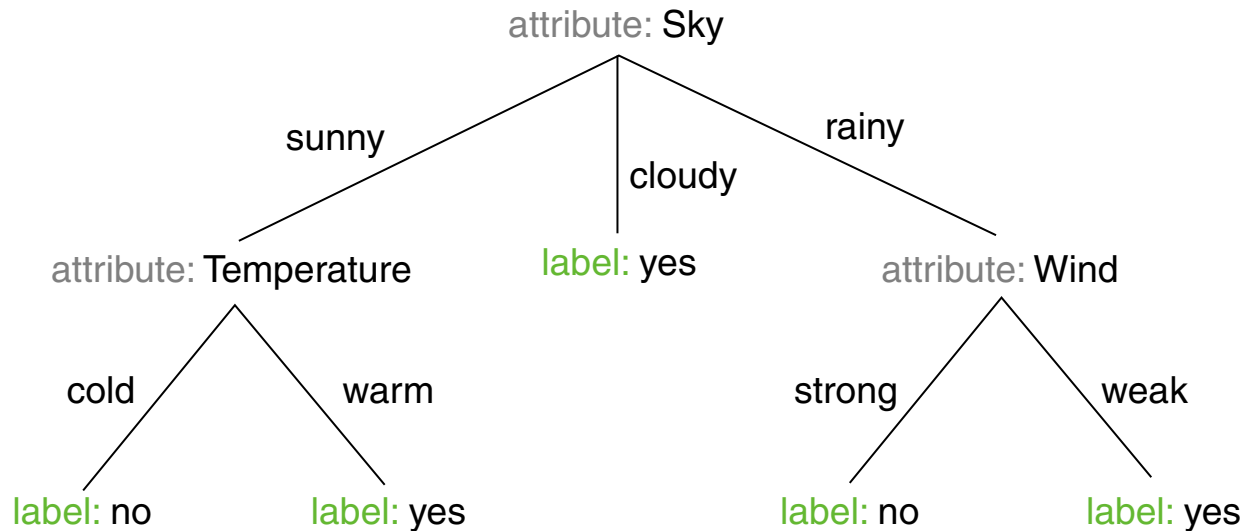
Example	Sky	Temperature	Humidity	Wind	Water	Forecast	EnjoySport
1	sunny	warm	normal	strong	warm	same	yes
2	sunny	warm	high	strong	warm	same	yes
3	rainy	cold	high	strong	warm	change	no
...							



# Decision Trees Basics

## Decision Tree for the Concept “EnjoySport”

Example	Sky	Temperature	Humidity	Wind	Water	Forecast	EnjoySport
1	sunny	warm	normal	strong	warm	same	yes
2	sunny	warm	high	strong	warm	same	yes
3	rainy	cold	high	strong	warm	change	no
...							



Partitioning of  $X$  at the root node:

$$X = \{\mathbf{x} \in X : \mathbf{x}|_{\text{Sky}} = \text{sunny}\} \cup \{\mathbf{x} \in X : \mathbf{x}|_{\text{Sky}} = \text{cloudy}\} \cup \{\mathbf{x} \in X : \mathbf{x}|_{\text{Sky}} = \text{rainy}\}$$

# Decision Trees Basics

## Definition 1 (Splitting)

Let  $X$  be a feature space and let  $D$  be a set of examples. A splitting of  $X$  is a partitioning of  $X$  into mutually exclusive subsets  $X_1, \dots, X_s$ . I.e.,  $X = X_1 \cup \dots \cup X_s$  with  $X_j \neq \emptyset$  and  $X_j \cap X_{j'} = \emptyset$ , where  $j, j' \in \{1, \dots, s\}, j \neq j'$ .

A splitting  $X_1, \dots, X_s$  of  $X$  induces a splitting  $D_1, \dots, D_s$  of  $D$ , where  $D_j$ ,  $j = 1, \dots, s$ , is defined as  $\{(\mathbf{x}, c(\mathbf{x})) \in D \mid \mathbf{x} \in X_j\}$ .

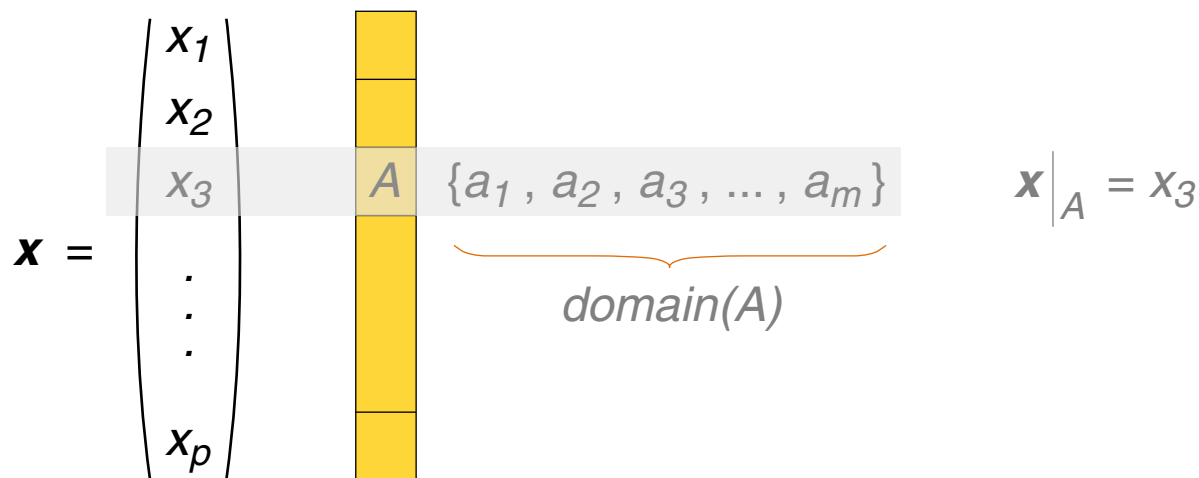
# Decision Trees Basics

## Definition 1 (Splitting)

Let  $X$  be a feature space and let  $D$  be a set of examples. A splitting of  $X$  is a partitioning of  $X$  into mutually exclusive subsets  $X_1, \dots, X_s$ . I.e.,  $X = X_1 \cup \dots \cup X_s$  with  $X_j \neq \emptyset$  and  $X_j \cap X_{j'} = \emptyset$ , where  $j, j' \in \{1, \dots, s\}, j \neq j'$ .

A splitting  $X_1, \dots, X_s$  of  $X$  induces a splitting  $D_1, \dots, D_s$  of  $D$ , where  $D_j$ ,  $j = 1, \dots, s$ , is defined as  $\{(\mathbf{x}, c(\mathbf{x})) \in D \mid \mathbf{x} \in X_j\}$ .

A splitting depends on the measurement scale of a feature:



# Decision Trees Basics

## Definition 1 (Splitting)

Let  $X$  be a feature space and let  $D$  be a set of examples. A splitting of  $X$  is a partitioning of  $X$  into mutually exclusive subsets  $X_1, \dots, X_s$ . I.e.,  $X = X_1 \cup \dots \cup X_s$  with  $X_j \neq \emptyset$  and  $X_j \cap X_{j'} = \emptyset$ , where  $j, j' \in \{1, \dots, s\}, j \neq j'$ .

A splitting  $X_1, \dots, X_s$  of  $X$  induces a splitting  $D_1, \dots, D_s$  of  $D$ , where  $D_j$ ,  $j = 1, \dots, s$ , is defined as  $\{(\mathbf{x}, c(\mathbf{x})) \in D \mid \mathbf{x} \in X_j\}$ .

A splitting depends on the measurement scale of a feature:

1.  $m$ -ary splitting induced by a (nominal) feature  $A$  with finite domain:

$$A = \{a_1, \dots, a_m\} : X = \{\mathbf{x} \in X : \mathbf{x}|_A = a_1\} \cup \dots \cup \{\mathbf{x} \in X : \mathbf{x}|_A = a_m\}$$

# Decision Trees Basics

## Definition 1 (Splitting)

Let  $X$  be a feature space and let  $D$  be a set of examples. A splitting of  $X$  is a partitioning of  $X$  into mutually exclusive subsets  $X_1, \dots, X_s$ . I.e.,  $X = X_1 \cup \dots \cup X_s$  with  $X_j \neq \emptyset$  and  $X_j \cap X_{j'} = \emptyset$ , where  $j, j' \in \{1, \dots, s\}, j \neq j'$ .

A splitting  $X_1, \dots, X_s$  of  $X$  induces a splitting  $D_1, \dots, D_s$  of  $D$ , where  $D_j$ ,  $j = 1, \dots, s$ , is defined as  $\{(\mathbf{x}, c(\mathbf{x})) \in D \mid \mathbf{x} \in X_j\}$ .

A splitting depends on the measurement scale of a feature:

1.  $m$ -ary splitting induced by a (nominal) feature  $A$  with finite domain:

$$A = \{a_1, \dots, a_m\} : X = \{\mathbf{x} \in X : \mathbf{x}|_A = a_1\} \cup \dots \cup \{\mathbf{x} \in X : \mathbf{x}|_A = a_m\}$$

2. Binary splitting induced by a (nominal) feature  $A$ :

$$A' \subset A : X = \{\mathbf{x} \in X : \mathbf{x}|_A \in A'\} \cup \{\mathbf{x} \in X : \mathbf{x}|_A \notin A'\}$$

3. Binary splitting induced by an ordinal feature  $A$ :

$$v \in \text{dom}(A) : X = \{\mathbf{x} \in X : \mathbf{x}|_A \succeq v\} \cup \{\mathbf{x} \in X : \mathbf{x}|_A \prec v\}$$



## Remarks:

- ❑ The syntax  $\mathbf{x}|_A$  denotes the projection operator, which returns that vector component (dimension) of  $\mathbf{x} = (x_1, \dots, x_p)$  that is associated with the feature  $A$ . Without loss of generality this projection can be presumed being unique.
- ❑ A splitting of  $X$  into two disjoint, non-empty subsets is called a binary splitting.
- ❑ We consider only splittings of  $X$  that are induced by a splitting of a single feature  $A$  of  $X$ .  
Keyword: monothetic splitting.  
By contrast, a polythetic splitting considers several features at the same time.

# Decision Trees Basics

## Definition 2 (Decision Tree)

Let  $X$  be a feature space and let  $C$  be a set of classes. A decision tree  $T$  for  $X$  and  $C$  is a finite tree with a distinguished root node. A non-leaf node  $t$  of  $T$  has assigned (1) a set  $X(t) \subseteq X$ , (2) a splitting of  $X(t)$ , and (3) a one-to-one mapping of the subsets of the splitting to its successors.

$X(t) = X$  iff  $t$  is root node. A leaf node of  $T$  has assigned a class from  $C$ .

# Decision Trees Basics

## Definition 2 (Decision Tree)

Let  $X$  be a feature space and let  $C$  be a set of classes. A decision tree  $T$  for  $X$  and  $C$  is a finite tree with a distinguished root node. A non-leaf node  $t$  of  $T$  has assigned (1) a set  $X(t) \subseteq X$ , (2) a splitting of  $X(t)$ , and (3) a one-to-one mapping of the subsets of the splitting to its successors.

$X(t) = X$  iff  $t$  is root node. A leaf node of  $T$  has assigned a class from  $C$ .

Classification of some  $\mathbf{x} \in X$  given a decision tree  $T$ :

1. Find the root node of  $T$ .
2. If  $t$  is a non-leaf node, find among its successors that node whose subset of the splitting of  $X(t)$  contains  $\mathbf{x}$ . Repeat this step.
3. If  $t$  is a leaf node, label  $\mathbf{x}$  with the respective class.

→ The set of possible decision trees forms the hypothesis space  $H$ .

## Remarks:

- ❑ The classification of an  $x \in X$  determines a unique path from the root node of  $T$  to some leaf node of  $T$ .
- ❑ At each non-leaf node a particular feature of  $x$  is evaluated in order to find the next node along with a possible next feature to be analyzed.
- ❑ Each path from the root node to some leaf node corresponds to a conjunction of feature values, which are successively tested. This test can be formulated as a decision rule.

### Example:

IF Sky=rainy AND Wind=weak THEN EnjoySport=yes

If all tests in  $T$  are of the kind shown in the example, namely, an equality test regarding a feature value, all feature domains must be finite.

- ❑ If in all non-leaf nodes of  $T$  only one feature is evaluated at a time,  $T$  is called a monothetic decision tree. Examples for *polythetic* decision trees are the so-called oblique decision trees.
- ❑ Decision trees became popular in 1986, with the introduction of the ID3 Algorithm by J. R. Quinlan.

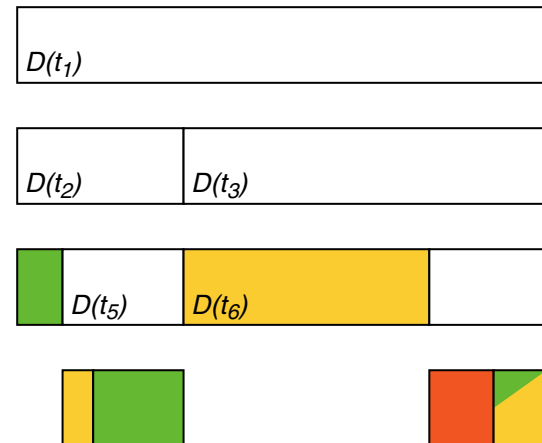
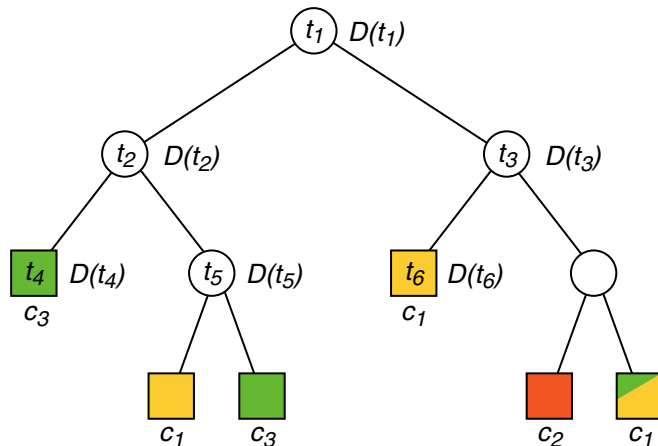
# Decision Trees Basics

## Notation

Let  $T$  be decision tree for  $X$  and  $C$ , let  $D$  be a set of examples [model], and let  $t$  be a node of  $T$ . Then we agree on the following notation:

- $X(t)$  denotes the subset of the feature space  $X$  that is represented by  $t$ .  
(as used in the [decision tree definition](#))
- $D(t)$  denotes the subset of the example set  $D$  that is represented by  $t$ , where  $D(t) = \{(\mathbf{x}, c(\mathbf{x})) \in D \mid \mathbf{x} \in X(t)\}$ . (see the [splitting definition](#))

Illustration:



## Remarks:

- ❑ The set  $X(t)$  is comprised of those members  $\mathbf{x}$  of  $X$  that are filtered by a path from the root node of  $T$  to the node  $t$ .
- ❑  $leaves(T)$  denotes the set of all leaf nodes of  $T$ .
- ❑ A single node  $t$  of a decision tree  $T$ , and hence  $T$  itself, encode a piecewise constant function. This way,  $t$  as well as  $T$  can form complex, non-linear classifiers. The functions encoded by  $t$  and  $T$  differ in the number of evaluated features of  $\mathbf{x}$ , which is one for  $t$  and the tree height for  $T$ .
- ❑ In the following we will use the symbols “ $t$ ” and “ $T$ ” to denote also the classifiers that are encoded by a node  $t$  and a tree  $T$  respectively:

$$t, T : X \rightarrow C \quad (\text{instead of } y_t, y_T : X \rightarrow C)$$

# Decision Trees Basics

## Algorithm Template: Construction

Algorithm: *DT-construct* Decision Tree Construction

Input:  $D$  (Sub)set of examples.

Output:  $t$  Root node of a decision (sub)tree.

*DT-construct*( $D$ )

1.  $t = \text{newNode}()$   
    $\text{label}(t) = \text{representativeClass}(D)$
2. **IF**  $\text{impure}(D)$   
   **THEN**  $\text{criterion} = \text{splitCriterion}(D)$   
   **ELSE**  $\text{return}(t)$
3.  $\{D_1, \dots, D_s\} = \text{decompose}(D, \text{criterion})$
4. **FOREACH**  $D'$  **IN**  $\{D_1, \dots, D_s\}$  **DO**  
    $\text{addSuccessor}(t, \text{DT-construct}(D'))$   
**ENDDO**
5.  $\text{return}(t)$

[\[Illustration\]](#)

# Decision Trees Basics

## Algorithm Template: Classification

Algorithm: *DT-classify* Decision Tree Classification

Input:  $\mathbf{x}$  Feature vector.  
 $t$  Root node of a decision (sub)tree.

Output:  $y(\mathbf{x})$  Class of feature vector  $\mathbf{x}$  in the decision (sub)tree below  $t$ .

*DT-classify*( $\mathbf{x}, t$ )

```
1. IF isLeafNode( $t$ )  
   THEN return(label( $t$ ))  
   ELSE return(DT-classify( $\mathbf{x}, \textit{splitSuccessor}(t, \mathbf{x})$ )
```



## Remarks:

- ❑ Since *DT-construct* assigns to each node of a decision tree  $T$  a class, each subtree of  $T$  (as well as each pruned version of a subtree of  $T$ ) represents a valid decision tree on its own.
- ❑ Functions of *DT-construct*:
  - *representativeClass*( $D$ )  
Returns a representative class for the example set  $D$ . Note that, due to pruning, each node may become a leaf node.
  - *impure*( $D$ )  
Evaluates the (im)purity of a set  $D$  of examples.
  - *splitCriterion*( $D$ )  
Returns a split criterion for  $X(t)$  based on the examples in  $D(t)$ .
  - *decompose*( $D$ , *criterion*)  
Returns a splitting of  $D$  according to *criterion*.
  - *addSuccessor*( $t$ ,  $t'$ )  
Inserts the successor  $t'$  for node  $t$ .
- ❑ Functions of *DT-classify*:
  - *isLeafNode*( $t$ )  
Tests whether  $t$  is a leaf node.
  - *splitSuccessor*( $t$ ,  $\mathbf{x}$ )  
Returns the (unique) successor  $t'$  of  $t$  for which  $\mathbf{x} \in X(t')$  holds.

# Decision Trees Basics

## When to Use Decision Trees

Problem characteristics that may suggest a decision tree classifier:

- ❑ the objects can be described by feature-value combinations
- ❑ the domain and range of the target function are discrete
- ❑ hypotheses can be represented in disjunctive normal form
- ❑ the training set contains noise

Selected application areas:

- ❑ medical diagnosis
- ❑ fault detection in technical systems
- ❑ risk analysis for credit approval
- ❑ basic scheduling tasks such as calendar management
- ❑ classification of design flaws in software engineering

# Decision Trees Basics

## On the Construction of Decision Trees

- ❑ How to exploit an example set both efficiently and effectively?
- ❑ According to what rationale should a node become a leaf node?
- ❑ How to assign a class for nodes of impure example sets?
- ❑ How to evaluate decision tree performance?

# Decision Trees Basics

## Evaluation of Decision Trees

1. Size

2. Classification error

# Decision Trees Basics

## Evaluation of Decision Trees

### 1. Size

Among those theories that can explain an observation, the most simple one is to be preferred (*Ockham's Razor*):

*Entia non sunt multiplicanda sine necessitate.*

[Johannes Clauberg 1622-1665]

Here: among all decision trees of minimum classification error we choose the one of smallest size.

### 2. Classification error

Quantifies the rigor according to which a class label is assigned to  $x$  in a leaf node of  $T$ , based on the examples in  $D$ . [\[illustration\]](#)

If all leaf nodes of a decision tree  $T$  represent a single example of  $D$ , the classification error of  $T$  with respect to  $D$  is zero.

# Decision Trees Basics

## Evaluation of Decision Trees: Size

- ❑ Leaf node number
- ❑ Tree height
- ❑ External path length
- ❑ Weighted external path length

# Decision Trees Basics

## Evaluation of Decision Trees: Size

- ❑ Leaf node number

The leaf node number corresponds to number of rules that are encoded in a decision tree.

- ❑ Tree height

The tree height corresponds to the maximum rule length and bounds the number of premises to be evaluated to reach a class decision.

- ❑ External path length

The external path length totals the lengths of all paths from the root of a tree to its leaf nodes. It corresponds to the space to store all rules that are encoded in a decision tree.

- ❑ Weighted external path length

The weighted external path length is defined as the external path length with each length value weighted by the number of examples in  $D$  that are classified by this path.

# Decision Trees Basics

## Evaluation of Decision Trees: Size (continued)

Example set  $D$  for mushrooms, implicitly defining a feature space  $X$  over the three dimensions color, size, and points:

	Color	Size	Points	Eatability
1	red	small	yes	toxic
2	brown	small	no	eatable
3	brown	large	yes	eatable
4	green	small	no	eatable
5	red	large	no	eatable

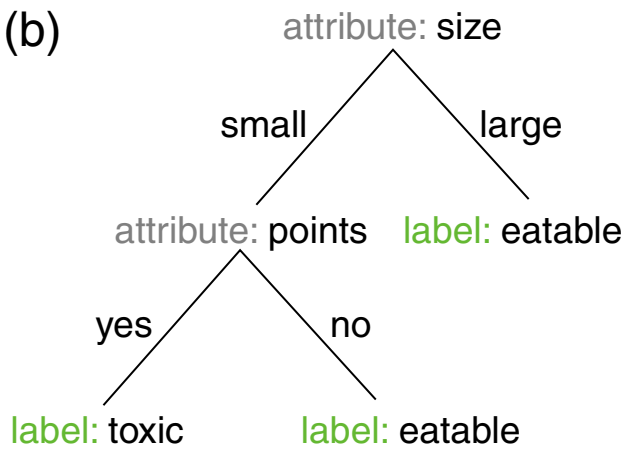
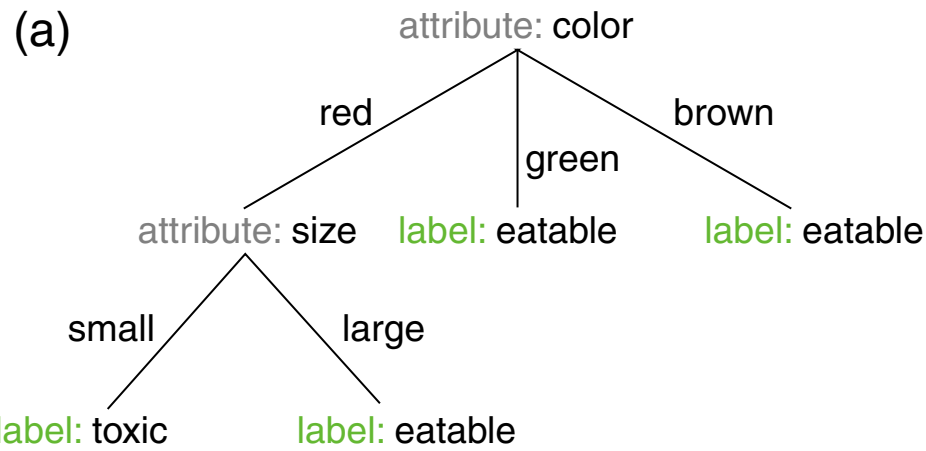




# Decision Trees Basics

## Evaluation of Decision Trees: Size (continued)

The following trees correctly classify all examples in  $D$  :

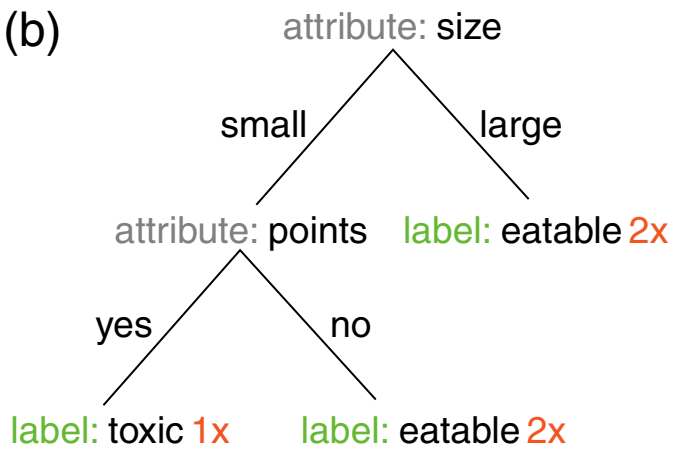
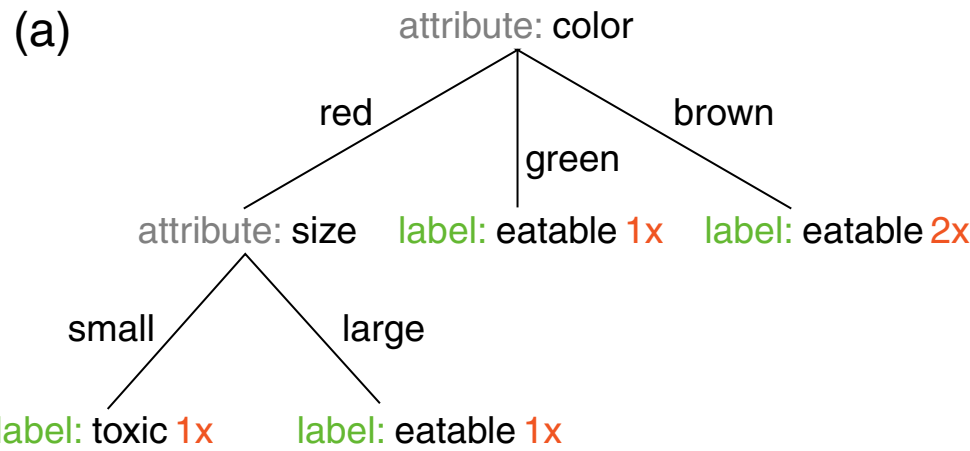


Criterion	(a)	(b)
Leaf node number	4	3
Tree height	2	2
External path length	6	5

# Decision Trees Basics

## Evaluation of Decision Trees: Size (continued)

The following trees correctly classify all examples in  $D$  :



Criterion	(a)	(b)
Leaf node number	4	3
Tree height	2	2
External path length	6	5
Weighted external path length	7	8

# Decision Trees Basics

## Evaluation of Decision Trees: Size (continued)

### Theorem 3 (External Path Length Bound)

The problem to decide for a set of examples  $D$  whether or not a decision tree exists whose external path length is bounded by  $b$ , is NP-complete.

# Decision Trees Basics

## Evaluation of Decision Trees: Classification Error

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows [\[Illustration\]](#):

$$label(t) = \operatorname{argmax}_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|}$$

# Decision Trees Basics

## Evaluation of Decision Trees: Classification Error

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows [\[Illustration\]](#):

$$label(t) = \operatorname{argmax}_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|}$$

Misclassification rate of node classifier  $t$  wrt.  $D(t)$ :

$$Err(t, D(t)) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) \neq label(t)\}|}{|D(t)|} = 1 - \max_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|}$$

# Decision Trees Basics

## Evaluation of Decision Trees: Classification Error

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows [\[Illustration\]](#):

$$label(t) = \operatorname{argmax}_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|}$$

Misclassification rate of node classifier  $t$  wrt.  $D(t)$ :

$$Err(t, D(t)) = \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) \neq label(t)\}|}{|D(t)|} = 1 - \max_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|}$$

Misclassification rate of decision tree classifier  $T$  wrt.  $D$ :

$$Err(T, D) = \sum_{t \in \text{leaves}(T)} \frac{|D(t)|}{|D|} \cdot Err(t, D(t))$$

## Remarks:

- ❑ Observe the difference between  $\max(f)$  and  $\operatorname{argmax}(f)$ . Both expressions maximize  $f$ , but the former returns the maximum  $f$ -value (the image) while the latter returns the argument (the preimage) for which  $f$  becomes maximum:
  - $\max_{c \in C}(f(c)) = \max\{f(c) \mid c \in C\}$
  - $\operatorname{argmax}_{c \in C}(f(c)) = c^* \Rightarrow f(c^*) = \max_{c \in C}(f(c))$
- ❑ The classifiers  $t$  and  $T$  may not have been constructed using  $D(t)$  as training data. I.e., the example set  $D(t)$  is in the role of a holdout test set.
- ❑ The true misclassification rate  $Err^*(T)$  is based on a probability measure  $P$  on  $X \times C$  (and not on relative frequencies). For a node  $t$  of  $T$  this probability becomes minimum iff:

$$label(t) = \operatorname{argmax}_{c \in C} P(c \mid X(t))$$

- ❑ If  $D$  has been used as training set, a reliable interpretation of the (training) error  $Err(T, D)$  in terms of  $Err^*(T)$  requires the Inductive Learning Hypothesis to hold. This implies, among others, that the distribution of  $C$  over the feature space  $X$  corresponds to the distribution of  $C$  over the training set  $D$ .

# Decision Trees Basics

## Evaluation of Decision Trees: Misclassification Costs

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . In addition, there is a cost measure for misclassification. Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows:

$$label(t) = \operatorname{argmin}_{c' \in C} \sum_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|} \cdot cost(c' \mid c)$$



# Decision Trees Basics

## Evaluation of Decision Trees: Misclassification Costs

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . In addition, there is a cost measure for misclassification. Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows:

$$label(t) = \operatorname{argmin}_{c' \in C} \sum_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|} \cdot cost(c' | c)$$

Misclassification costs of node classifier  $t$  wrt.  $D(t)$ :

$$Err_{cost}(t, D(t)) = \frac{1}{|D_t|} \cdot \sum_{(\mathbf{x}, c(\mathbf{x})) \in D(t)} cost(label(t) | c(\mathbf{x})) = \min_{c' \in C} \sum_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|} \cdot cost(c' | c)$$

# Decision Trees Basics

## Evaluation of Decision Trees: Misclassification Costs

Given a decision tree  $T$ , a set of examples  $D$ , and a node  $t$  of  $T$  that represents the example subset  $D(t) \subseteq D$ . In addition, there is a cost measure for misclassification. Then, the class that is assigned to  $t$ ,  $label(t)$ , is defined as follows:

$$label(t) = \operatorname{argmin}_{c' \in C} \sum_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|} \cdot cost(c' | c)$$

Misclassification costs of node classifier  $t$  wrt.  $D(t)$  :

$$Err_{cost}(t, D(t)) = \frac{1}{|D(t)|} \cdot \sum_{(\mathbf{x}, c(\mathbf{x})) \in D(t)} cost(label(t) | c(\mathbf{x})) = \min_{c' \in C} \sum_{c \in C} \frac{|\{(\mathbf{x}, c(\mathbf{x})) \in D(t) : c(\mathbf{x}) = c\}|}{|D(t)|} \cdot cost(c' | c)$$

Misclassification costs of decision tree classifier  $T$  wrt.  $D$  :

$$Err_{cost}(T, D) = \sum_{t \in leaves(T)} \frac{|D(t)|}{|D|} \cdot Err_{cost}(t, D(t))$$

## Remarks:

- Again, observe the difference between  $\min(f)$  and  $\operatorname{argmin}(f)$ . Both expressions minimize  $f$ , but the former returns the minimum  $f$ -value (the image) while the latter returns the argument (the preimage) for which  $f$  becomes minimum.