Chapter ML:VI

VI. Decision Trees

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Classification Problems with Nominal Features

Setting:

- X is a multiset of feature vectors.
- C is a set of classes.
- $D = \{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_n, c_n)\} \subseteq X \times C$ is a multiset of examples.

Learning task:

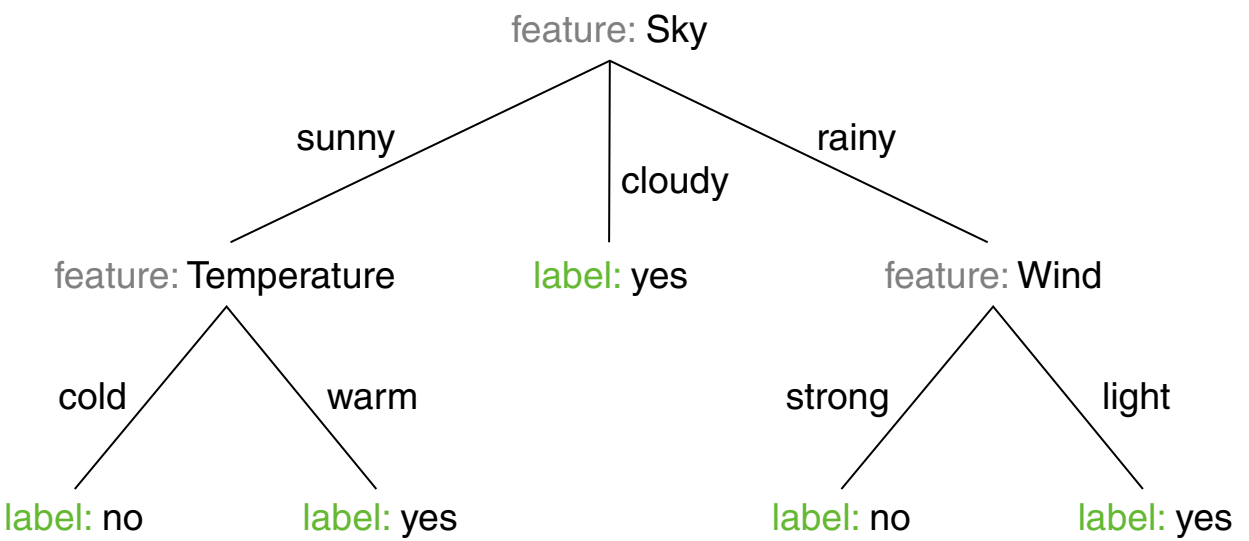
- Fit D using a decision tree T.

©STEIN/LETTMANN 2024

Decision Trees Basics

Decision Tree for the Concept “EnjoySurfing” [concept learning]

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

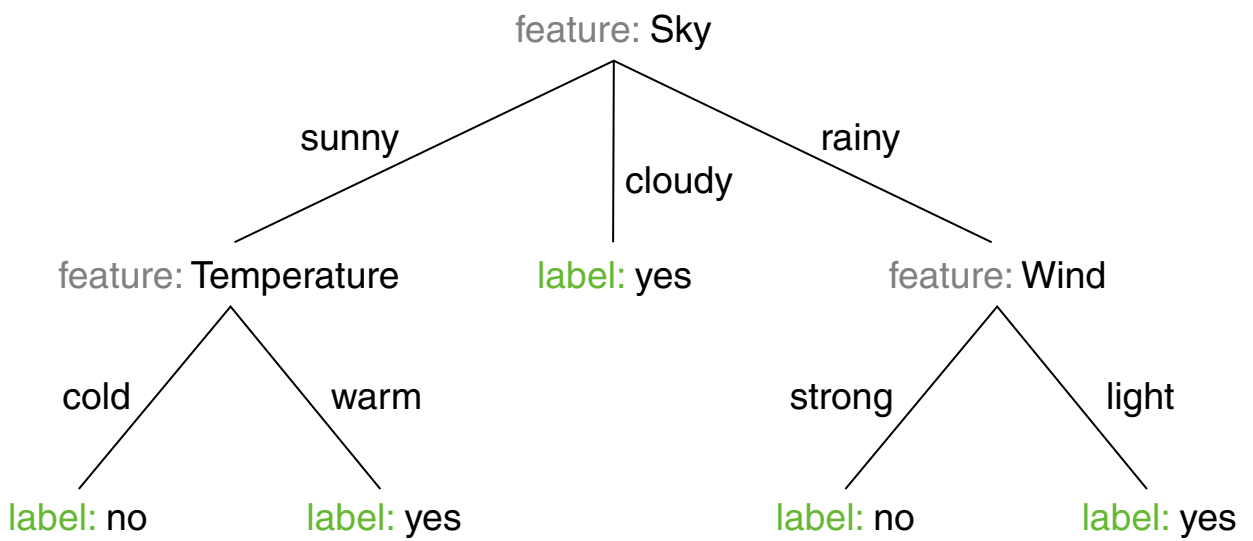


©STEIN/LETTMANN 2024

Decision Trees Basics

Decision Tree for the Concept “EnjoySurfing” [concept learning]

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



Splitting of X at the root node:

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.

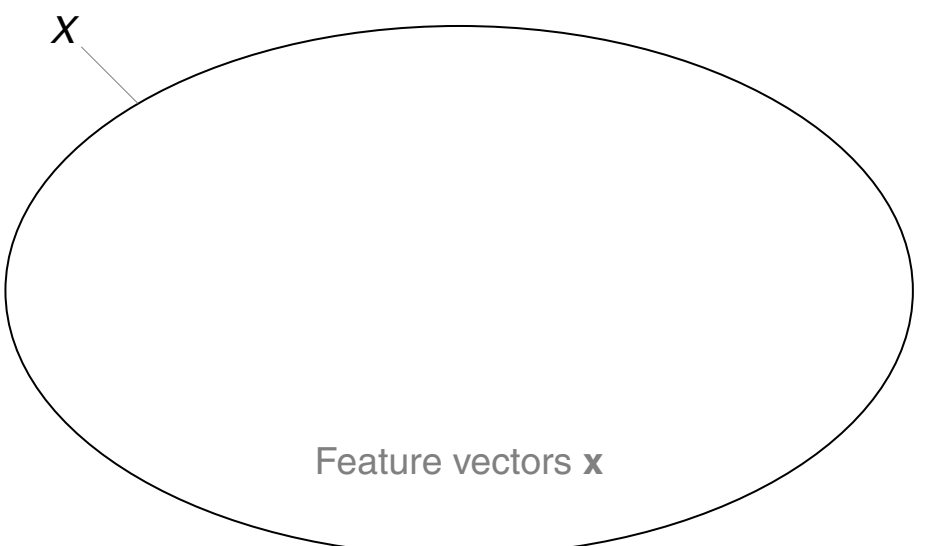
©STEIN/LETTMANN 2024

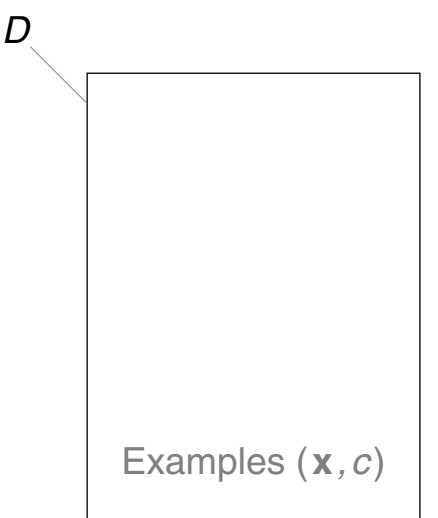
Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.





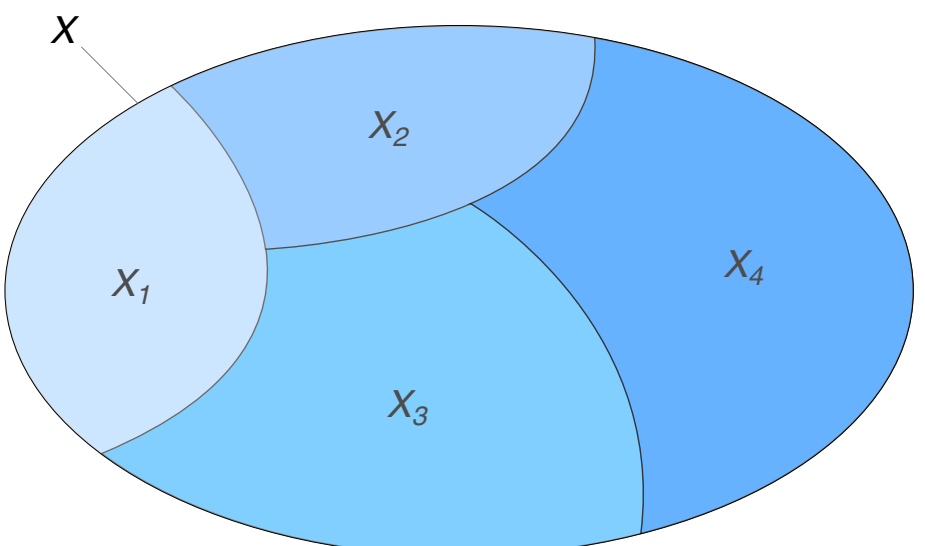
©STEIN/LETTMANN 2024

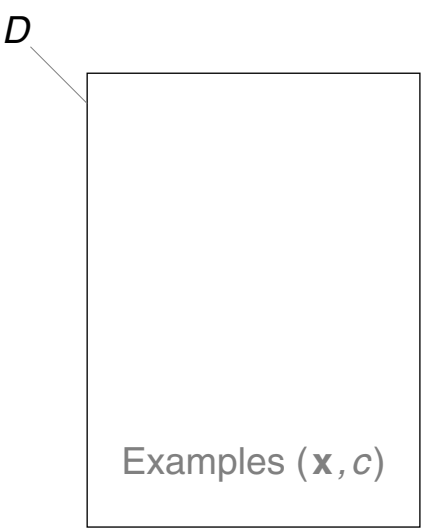
Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.





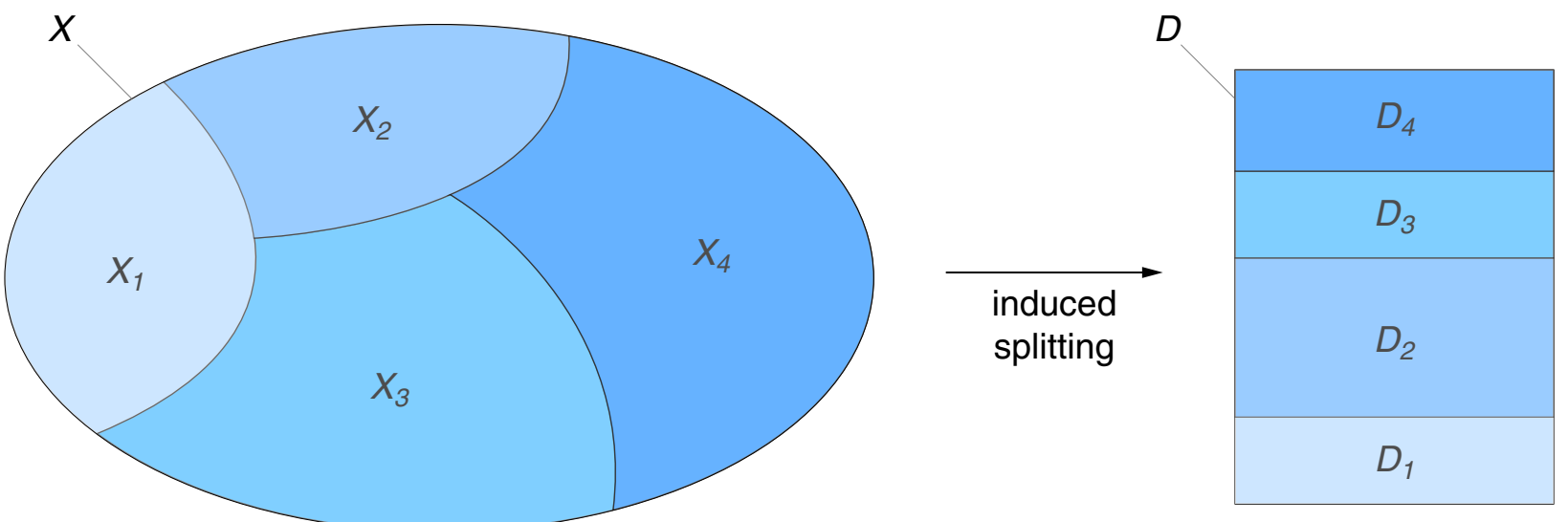
©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.



©STEIN/LETTMANN 2024

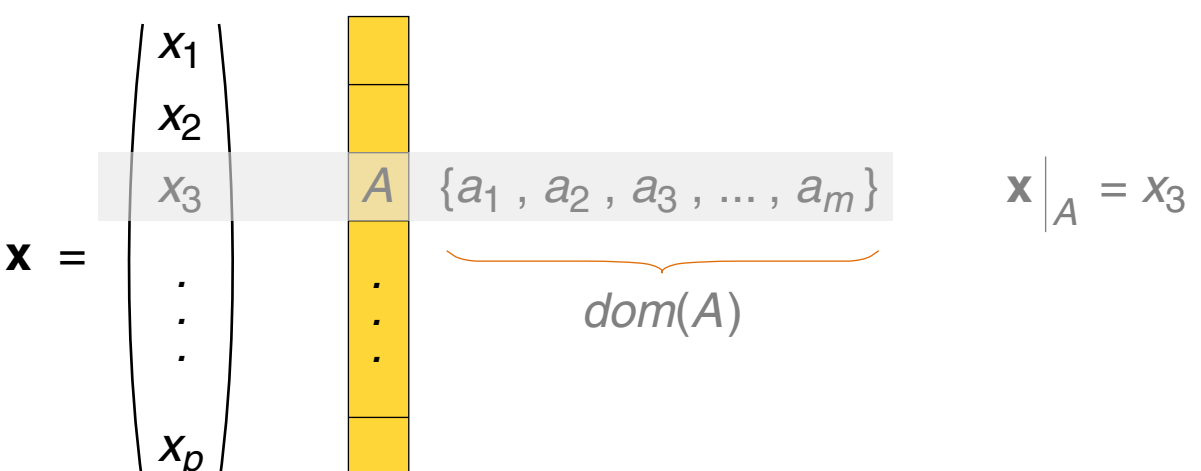
Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.

A splitting of X depends on the measurement scale of a feature A:



©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.

A splitting of X depends on the measurement scale of a feature A:

1. m-ary splitting induced by a (nominal) feature A with finite domain:
   $dom(A) = \{a_1, \ldots, a_m\}: \quad X = \{\mathbf{x} \in X : \mathbf{x}|_A = a_1\} \cup \ldots \cup \{\mathbf{x} \in X : \mathbf{x}|_A = a_m\}$

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.

A splitting of X depends on the measurement scale of a feature A:

1. m-ary splitting induced by a (nominal) feature A with finite domain:
   $dom(A) = \{a_1, \ldots, a_m\}: \quad X = \{\mathbf{x} \in X : \mathbf{x}|_A = a_1\} \cup \ldots \cup \{\mathbf{x} \in X : \mathbf{x}|_A = a_m\}$
2. Binary splitting induced by a (nominal) feature A:
   $V \subset dom(A): \qquad X = \{\mathbf{x} \in X : \mathbf{x}|_A \in V\} \cup \{\mathbf{x} \in X : \mathbf{x}|_A \notin V\}$

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 1 (Splitting, Induced Splitting)

Let X be a multiset of feature vectors and D a multiset of examples. A splitting of X is a decomposition of X into mutually exclusive subsets $X_1, \ldots, X_m$.
I.e., $X = X_1 \cup \ldots \cup X_m$ with $X_l \neq \emptyset$ and $X_l \cap X_{l'} = \emptyset$, where $l, l' \in \{1, \ldots, m\}, l \neq l'$.

A splitting $X_1, \ldots, X_m$ of feature vectors X induces a splitting $D_1, \ldots, D_m$ of examples D, where D_l, $l = 1, \ldots, m$, is defined as $\{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X_l\}$.

A splitting of X depends on the measurement scale of a feature A:

1. m-ary splitting induced by a (nominal) feature A with finite domain:
   $dom(A) = \{a_1, \ldots, a_m\}: \quad X = \{\mathbf{x} \in X : \mathbf{x}|_A = a_1\} \cup \ldots \cup \{\mathbf{x} \in X : \mathbf{x}|_A = a_m\}$
2. Binary splitting induced by a (nominal) feature A:
   $V \subset dom(A): \qquad X = \{\mathbf{x} \in X : \mathbf{x}|_A \in V\} \cup \{\mathbf{x} \in X : \mathbf{x}|_A \notin V\}$
3. Binary splitting induced by an ordinal feature A:
   $v \in dom(A): \qquad X = \{\mathbf{x} \in X : \mathbf{x}|_A \succeq v\} \cup \{\mathbf{x} \in X : \mathbf{x}|_A \prec v\}$

©STEIN/LETTMANN 2024

Remarks:

- $\mathbf{x}|_A$ denotes the projection operator, which returns that vector component (dimension) of $\mathbf{x}$, $\mathbf{x} = (x_1, \ldots, x_p)$, 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. Such kinds of splittings are called “monothetic splittings”.

  By contrast, a polythetic splitting considers several features at the same time.

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 2 (Decision Tree)

Let X be a set of features and C 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.

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 2 (Decision Tree)

Let X be a set of features and C 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.

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

How to *classify* some $\mathbf{x} \in X$ given a decision tree T:

1. Find the root node t of T.
2. If t is a non-leaf node, find among its successors that node t' whose subset of the splitting of $X(t)$ contains $\mathbf{x}$. Repeat Step 2 with $t = t'$.
3. If t is a leaf node, label $\mathbf{x}$ with the associated class.

©STEIN/LETTMANN 2024

Decision Trees Basics

Definition 2 (Decision Tree)

Let X be a set of features and C 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.

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

How to *classify* some $\mathbf{x} \in X$ given a decision tree T:

1. Find the root node t of T.
2. If t is a non-leaf node, find among its successors that node t' whose subset of the splitting of $X(t)$ contains $\mathbf{x}$. Repeat Step 2 with $t = t'$.
3. If t is a leaf node, label $\mathbf{x}$ with the associated class.

The set of possible decision trees over D forms the hypothesis space H.

[hypothesis space: LMS, Find-S, log. regression]

©STEIN/LETTMANN 2024

Remarks:

- The classification of an $\mathbf{x} \in X$ 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 $\mathbf{x}$ 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=light THEN EnjoySurfing=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.
- Since at all non-leaf nodes of T 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 Ross Quinlan.
- Recap. The string “Iff” or “iff” is an abbreviation for “If and only if”.

©STEIN/LETTMANN 2024

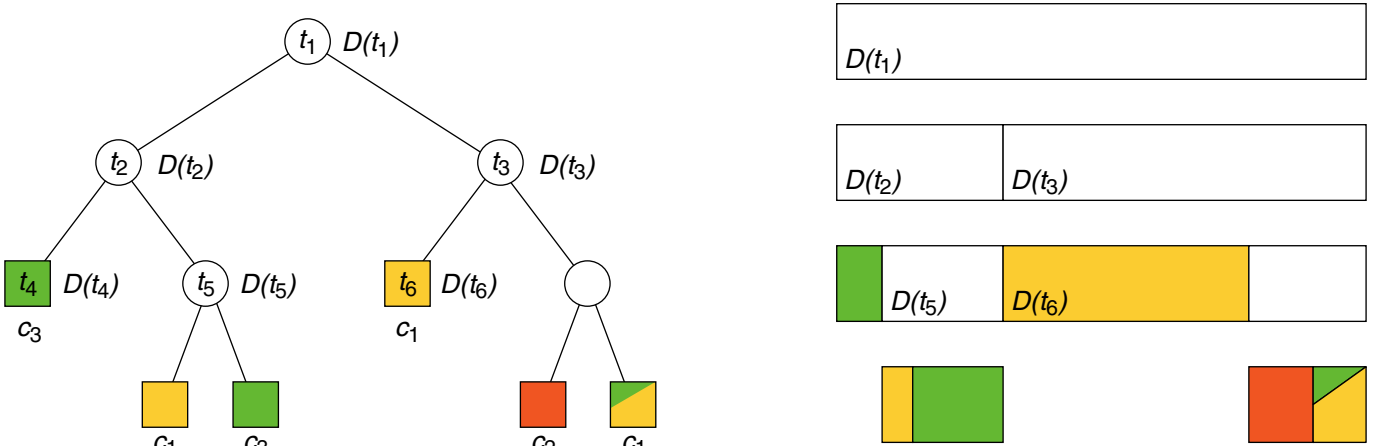
Decision Trees Basics

Notation

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

- $X(t)$ denotes the subset of X that is represented by t. [decision tree definition (1)]
- $D(t)$ denotes the subset of the example set D that is represented by t, i.e., $D(t) = \{(\mathbf{x}, c) \in D \mid \mathbf{x} \in X(t)\}$. [splitting definition]

Illustration (colors correspond to classes) :



©STEIN/LETTMANN 2024

Remarks:

- The set $X(t)$ consists 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.
- Each 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:

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Algorithm Template: Construction [illustration]

Algorithm:	*DT-construct*	Decision Tree Construction
Input:	D	Multiset of examples.
Output:	t	Root node of a decision tree.

DT-construct(D)

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Algorithm Template: Construction [illustration]

Algorithm:	*DT-construct*	Decision Tree Construction
Input:	D	Multiset of examples.
Output:	t	Root node of a decision tree.

DT-construct(D)

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Algorithm Template: Construction [illustration]

Algorithm:	*DT-construct*	Decision Tree Construction
Input:	D	Multiset of examples.
Output:	t	Root node of a decision tree.

DT-construct(D)

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

©STEIN/LETTMANN 2024

Remarks:

- Functions of *DT-construct*() :
  - *createNode*()
    Returns the data strucure for a node according to the definition.
  - *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)
    Assesses 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.

- 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Algorithm Template: Classification

Algorithm:	*DT-classify*	Decision Tree Classification
Input:	$\mathbf{x}$	Feature vector.
	t	Root node of a decision tree.
Output:	$y(\mathbf{x})$	Class of feature vector $\mathbf{x}$ in the decision tree below t.

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

```
1. IF isLeafNode(t)
   THEN return(label(t))
   ELSE return(DT-classify(x, splitSuccessor(t, x))
```


©STEIN/LETTMANN 2024

Remarks:

- 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

When to Use Decision Trees

Problem characteristics that suggest a decision tree classifier:

- the objects can be described by feature-value combinations
- the domain and range of the target function are discrete
- classification decisions should be comprehensible and explainable
- hypotheses can be represented in disjunctive normal form. Example:

  $\alpha =$ (Sky=sunny $\wedge$ Temperature=warm) $\vee$ Sky=cloudy $\vee$ (Sky=rainy $\wedge$ Wind=light)

©STEIN/LETTMANN 2024

Decision Trees Basics

When to Use Decision Trees

Problem characteristics that suggest a decision tree classifier:

- the objects can be described by feature-value combinations
- the domain and range of the target function are discrete
- classification decisions should be comprehensible and explainable
- hypotheses can be represented in disjunctive normal form. Example:

  $\alpha =$ (Sky=sunny $\wedge$ Temperature=warm) $\vee$ Sky=cloudy $\vee$ (Sky=rainy $\wedge$ Wind=light)

Exemplary fields of application:

- medical diagnosis
- fault detection in technical systems
- risk analysis in financial applications
- scheduling tasks such as calendar management
- classification of design flaws and patterns in software engineering

©STEIN/LETTMANN 2024

Decision Trees Basics

On the Construction of Decision Trees [illustration]

- 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 assess decision tree performance?

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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 $\mathbf{x}$ 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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 $\mathbf{x}$ 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment of Decision Trees: Size

- Leaf node number

  The leaf node number corresponds to the 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment of Decision Trees: Size

- Leaf node number

  The leaf node number corresponds to the 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment of Decision Trees: Size (continued)

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

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

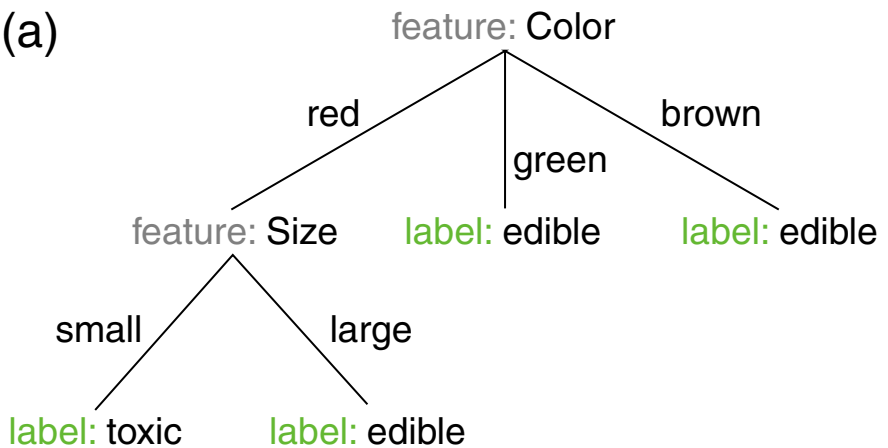
©STEIN/LETTMANN 2024

Decision Trees Basics

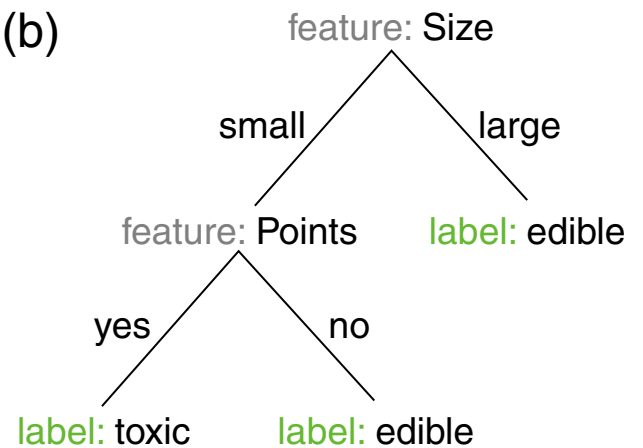
Assessment of Decision Trees: Size (continued)

The following trees correctly classify all examples in D:

(a)



(b)



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

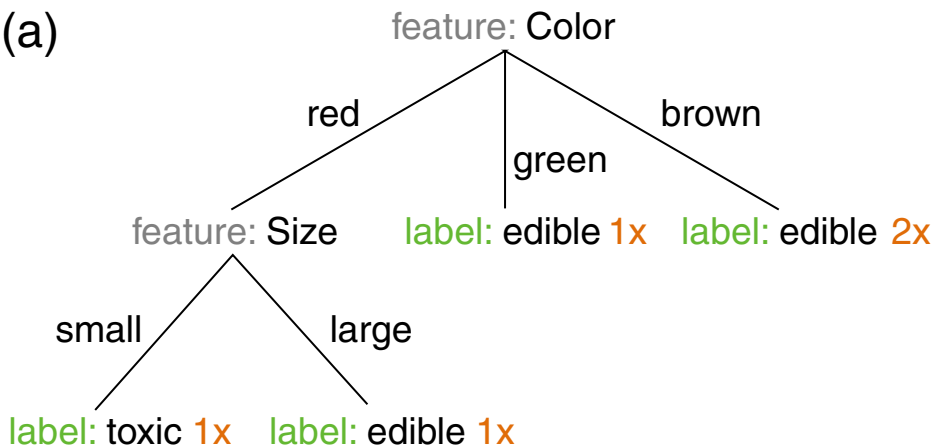
©STEIN/LETTMANN 2024

Decision Trees Basics

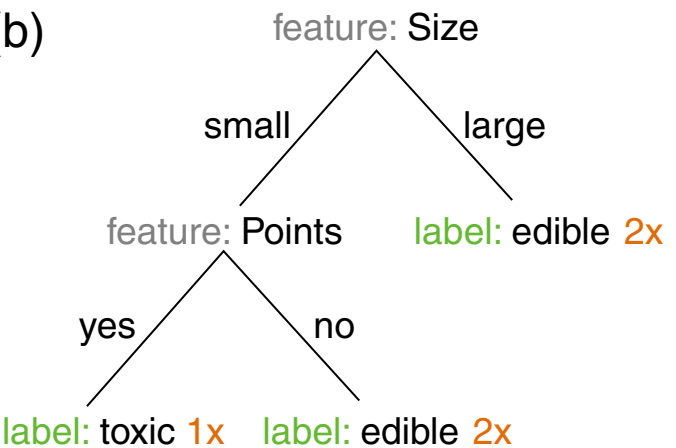
Assessment of Decision Trees: Size (continued)

The following trees correctly classify all examples in D:

(a)



(b)



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

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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.

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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:

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

[illustration]

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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:

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

[illustration]

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

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

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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:

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

[illustration]

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

$$\textit{Err}(t, D(t)) = \frac{|\{(\mathbf{x}, c) \in D(t) : \textit{label}(t) \neq c\}|}{|D(t)|} = 1 - \overbrace{\max_{c \in C} \frac{|\{(\mathbf{x}, c) \in D(t)\}|}{|D(t)|}}^{\text{max. accuracy at } t}$$

©STEIN/LETTMANN 2024

Decision Trees Basics

Assessment 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:

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

[illustration]

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

$$\textit{Err}(t, D(t)) = \frac{|\{(\mathbf{x}, c) \in D(t) : \textit{label}(t) \neq c\}|}{|D(t)|} = \overbrace{1 - \max_{c \in C} \frac{|\{(\mathbf{x}, c) \in D(t)\}|}{|D(t)|}}^{\text{max. accuracy at } t}$$

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

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

©STEIN/LETTMANN 2024

Remarks:

- 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 test set and $\mathit{Err}(T, D)$ denotes the holdout error.

- If D has been used as training set, a reliable interpretation of the (training) error $\mathit{Err}(T, D)$ in terms of $\mathit{Err}^*(T)$ requires the Inductive Learning Hypothesis to hold.

- Observe the difference between $\max f()$ and 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 true misclassification rate $\mathit{Err}^*(T)$ is based on a probability measure P (and not on relative frequencies). For a node t of T this probability becomes minimum iff:

$$\mathit{label}(t) = \operatorname*{argmax}_{c \in C} P(\mathit{C}{=}c \mid \mathbf{D}{=}X(t)),$$

where C denotes a random variable with range C, the set of classes. $\mathbf{D}{=}X(t)$ is a data event where **D** denotes a set of random vectors with realization $X(t)$.

©STEIN/LETTMANN 2024

Remarks (misclassification costs) :

- The assessment of decision trees can also be based on misclassification costs:

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

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

$$\textit{Err}_{\textit{cost}}(T, D) \;=\; \sum_{t \in \textit{leaves}(T)} \frac{|D(t)|}{|D|} \cdot \textit{Err}_{\textit{cost}}(t, D(t))$$

- As before, observe the difference between $\min f()$ and 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.

©STEIN/LETTMANN 2024