




# DECISION TREE LEARNING

CS576 MACHINE LEARNING



Dr. Jin Soung Yoo, Professor  
Department of Computer Science  
Purdue University Fort Wayne

# Reference

- Kelleher et al., Fundamentals of Machine Learning for Predictive Data Analytics, Ch 4
- T. Mitchell, Machine Learning, Ch 3
- E. Alpaydin, Introduction to Machine Learning, Ch 9

# Outline

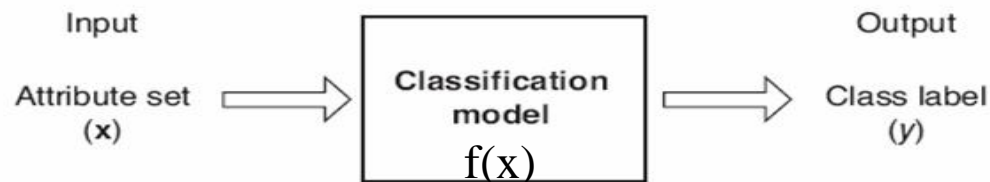
- Classification Task
- Information Theory and Information-based learning
- Decision Tree Learning
- Fundamentals
  - Decision Trees
  - Shannon's Entropy Model
  - Information Gain
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
  - Alternative Feature Selection Metrics
  - Handling Continuous Descriptive Features
  - Predicting Continuous Targets
  - Noisy Data, Overfitting and Tree Pruning

# Classification

- In machine learning and statistics, a **classification problem** revolves around the task of predicting the discrete class labels or categories of given input data.
- Unlike **regression (prediction)**, where the outcome is continuous or numeric, **classification** deals with predicting which among a set of **categories (or classes)** an input belongs to.
- If the number of class labels is 2, the problem is called a **binary classification** problem.
- If the number of class labels is more than 2, it is called a **multiclass classification** problem

# Problem Statement of Classification

- **Given:** A training data set
  - Each instance (example) is a tuple  $(\mathbf{x}, y)$ , where  $\mathbf{x}$  is the set of descriptive feature values that describe the instance and  $y$  is a class label of the instance.
    - The feature set  $\mathbf{x}$  can contain attributes of any type, while the class label  $y$  must be *categorical*.
- **Goal:** Find *a classification model (classifier)* for the class feature as a function of the values of other features.



- The model can be represented in many ways, e.g., as a tree, a probability table, or a vector of real-values parameters.
- The **learning algorithm** that creates the training model is referred to as the *learner*.
- **Objective:** Find the *classifier* (described in terms of *model*) to **determine the class label of new, unseen instances as accurately as possible**.
  - The model is said to classify an instance  $(\mathbf{x}, y)$  correctly if  $\mathbf{f}(\mathbf{x}) = y$

# Applications

## ■ Customer target marketing

- (Class label  $y$ ) The groups (or labels) correspond to the user interest in a particular product.
  - E.g., binary groups - one group may correspond to customers interested in a product, and the other group may contain the remaining customers.
- (Attribute set  $x$ ) The feature variables may correspond to the demographic profiles of the customers.
- (Training data) In many cases, training examples of previous buying behavior are available. These can be used to provide examples of customers who may or may not be interested in a specific product.
- (Learning model) These training examples are used to learn whether or not a customer, with a known demographic profile, but unknown buying behavior, may be interested in a particular product.

# Applications (cont.)

## ■ Document categorization and filtering

- Many applications, such as news services, require real-time classification of documents.
- (Attribute set  $\mathbf{x}$ ) The features correspond to the words in the document.
- (Class label  $\mathbf{y}$ ) The class labels correspond to the various topics, such as politics, sports, current events, and so on.
- (Training data) Previous examples of documents from each topic may be available.
- (Learning model) The models are used to organize the documents in Web portals.

## ■ Categorizing email messages

- (Attribute set  $\mathbf{x}$ ) Features extracted from email message header and content
- (Class label  $\mathbf{y}$ ) spam or non-spam

# Applications (cont.)

## ■ Medical disease management -

### Prediction of treatment outcomes

- (Attribute set  $\mathbf{x}$ ) The features may be extracted from patient medical tests and treatments.
- (Class label  $\mathbf{y}$ ) The class label may correspond to treatment outcomes.
- (Learning model) The model constructed on the features is used to predict treatment outcomes.

### Identification of tumor cells

- (Attribute set  $\mathbf{x}$ ) The features may be extracted from MRI scans.
- (Class label  $\mathbf{y}$ ) The class label is malignant or benign.
- (Learning model) The learning model is desired to identify tumor cells.



# Classification Techniques

- The classification technique consists of a family of related models and a number of algorithms for learning these models.
- The most well-known **classification techniques** are
  - Decision tree classifiers
  - Rule-based classifiers
  - Probabilistic models
  - Instance-based classifiers
  - Support vector machines
  - Neural networks
  - and so on.

# Terminology Note

- The term “**classifier**” and “**model**” are often taken to be synonymous.
- While every model defines a classifier, not every classifier is defined by a single model. E.g.,
  - *k*-nearest neighbor classifiers do not build an explicit model.
  - An ensemble classifier combines the output of a collection of models.
- The term “classifier” is often used in a more general sense to refer to a classification technique.
  - E.g., Decision tree classifier can refer to the decision tree classification technique or a specific classifier (/model) built using that technique.

# Two Phases of Classification

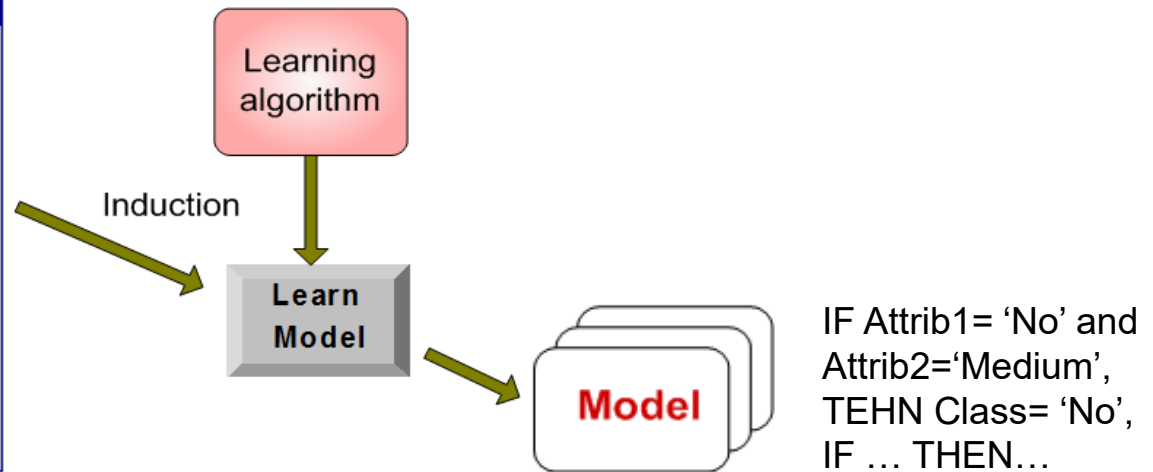
- Most classification algorithms have two phases.
- **Training phase**
  - In this phase, a training model is constructed from the training examples using a learning algorithm. The process to learn(/build) a model is known as **induction**.
- **Testing phase**
  - In this phase, the training model is used to determine the class label (or group identifier) of one or more unseen test instances. This process is also known as **deduction**.

# Phase One: Training

- **Model construction** describing a set of predetermined classes.
  - A set of examples used for model construction is called a **training set**.
  - Each example is assumed to belong to a predefined class, as determined by the class label attribute
  - The model is represented as classification rules, decision trees, or mathematical formulae.

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

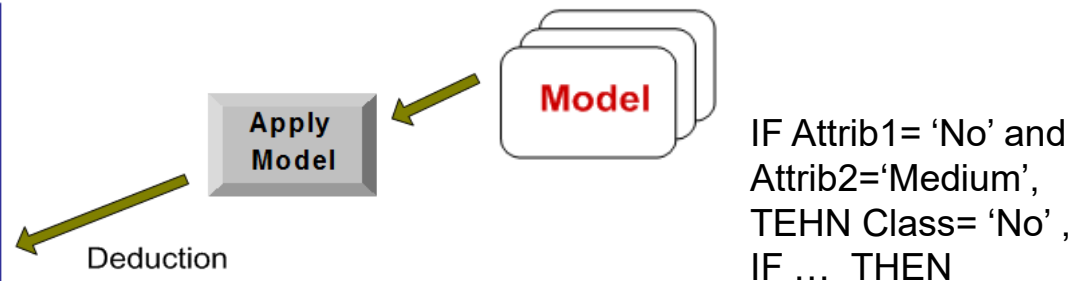


# Phase Two: Test

- **Model test** for classifying future or unknown data points
  - The performance of a model (classifier) can be evaluated by comparing the predicted labels against the true labels of example in **test set** which is independent of training set.

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



- The **accuracy** of the model is the percentage of test samples that are correctly classified by the model
- If the accuracy is acceptable, use the model to classify data instances whose class labels are not known.

# Issues of Classification Model/Algorithm

- **Accuracy**

- Classifier accuracy: predicting class label

- **Speed**

- Time to construct the model (training time)
- Time to use the model (classification/prediction time)

- **Robustness**

- Handling noise and missing values

- **Scalability**

- Efficiency for large datasets

- **Interpretability**

- Understanding and insight provided by the model

- **Other measures**

- e.g., goodness of rules, such as model size or compactness of classification rules

# Outline

- Classification Task
- ☞ **Information Theory and Information-based Learning**
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
- Summary

# Information Theory

- **Information theory** is a branch of applied mathematics and communication theory that deals with the quantification, transmission, and manipulation of information.
- It is the study of how much information is present in the signals or data we receive from our environment.
- Information theory was founded by Claude **Shannon** in the late 1940s
- It has since become a foundational concept in various fields, including computer science, communication, statistics, cryptography, and machine learning.



# Information-based Learning

- **Information-based learning** refers to a framework within machine learning and data analysis that leverages principles from information theory to guide and enhance the learning process.
- When information theory concept is applied to learning algorithms, the **goal** is to select models, features, or data points based on how much information they provide regarding the task at hand.

# Decision Tree Learning

- One of the most prominent examples of **information-based learning** is in the construction of decision trees, particularly the ID3, C4.5, and related algorithms.
- **Decision tree learning** methods build predictive models **using only the most informative features**.
- These algorithms split data based on the feature that provides the most "**information gain**."
  - Information gain is measured by the reduction in **entropy** (a concept from information theory) after the split.

# Decision Tree Learning

- **Decision tree learning** is a method for approximating discrete-valued target functions (i.e., classification), in which the learned function is represented by a *decision tree*
  - Decision Trees can also perform regression tasks (for predicting numeric target values).
- The prediction process is modeled with the use of a set of *hierarchical decisions* on the descriptive feature values, arranged in a tree-like structure.

# Decision Tree Learning

- Decision tree learning is **one of the most widely used and practical methods** for inductive inference
- Decision tree learning **provides comparable classification accuracy** with other classification techniques for many simple data sets
- Decision trees are also the fundamental components of **Random Forests**, which are among the most powerful ML algorithm available today.

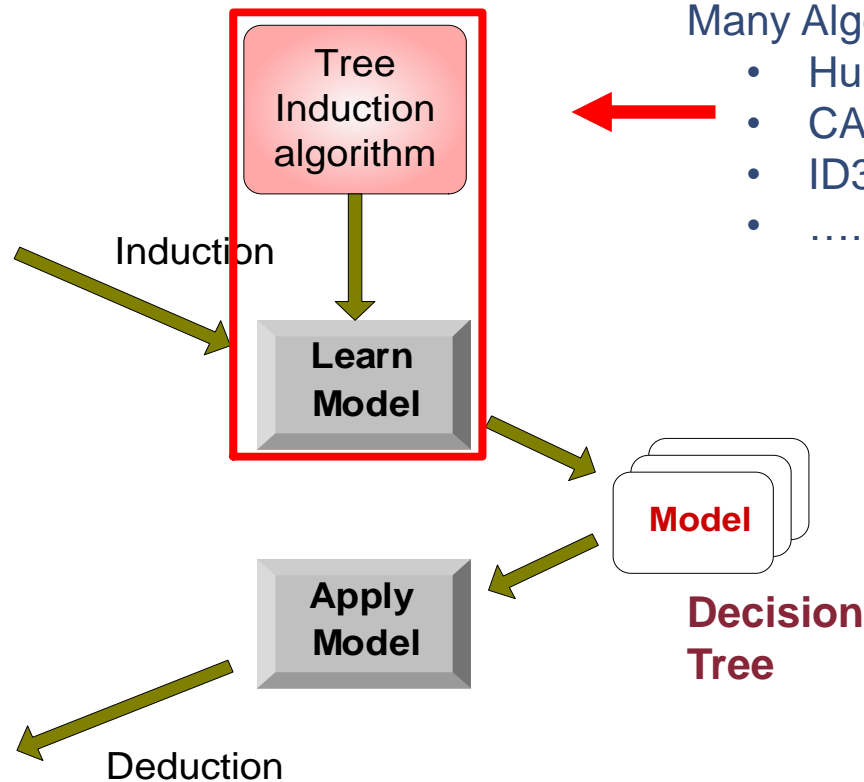
# Classification with Decision Tree

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

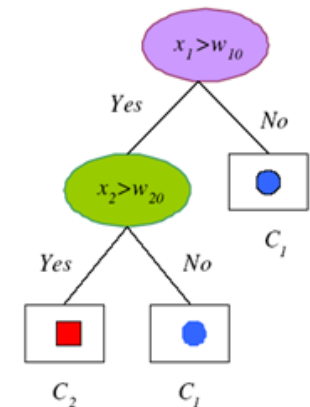
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Many Algorithms:

- Hunt's Algorithm
- CART
- ID3, C4.5
- ....



IF  $x_1 > 10$  AND  $x_2 > 20$  THEN Class= 'C2',  
IF  $x_1 \leq 10$  20 THEN Class= 'C1',  
IF ... THEN...

# Outline

- Classification Task
- Information Theory and Information-based learning
- Decision Tree Learning
- ☞ **Fundamentals**
  - Decision Trees
  - Shannon's Entropy Model
  - Information Gain
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
- Summary

# Example Dataset

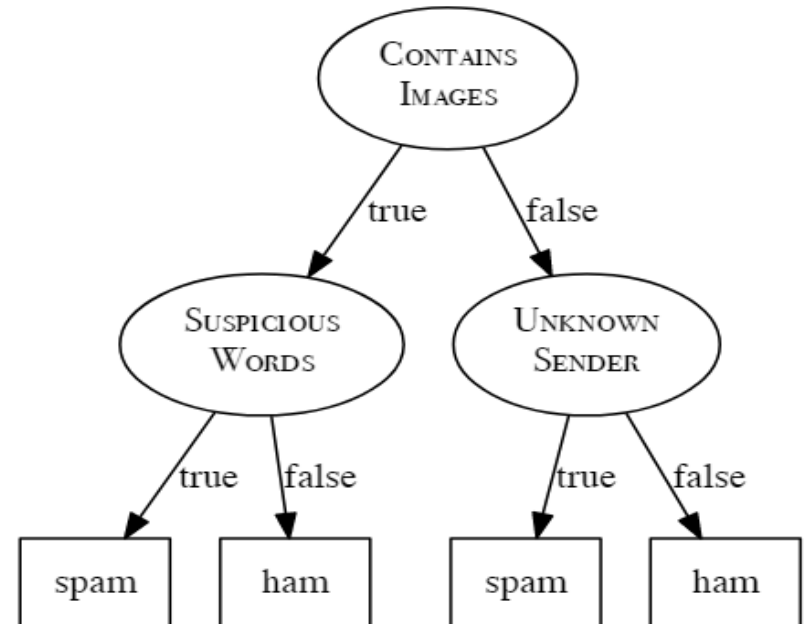
- An email spam prediction dataset
  - **Descriptive features:** Suspicious words, Unknown sender, contains images
  - **Target feature:** Class

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

**Table:** An email spam prediction dataset

# Decision Tree Structure

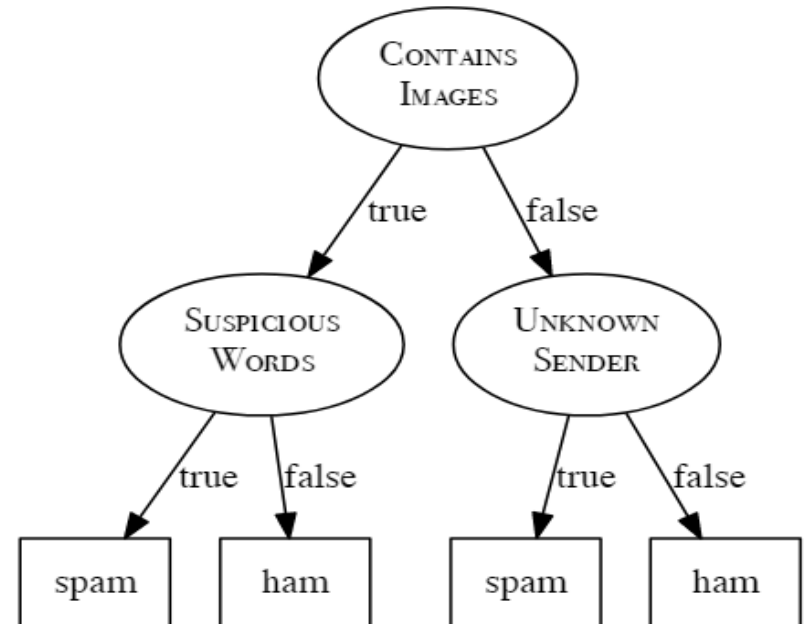
- A **decision tree** is the fundamental structure used in information-based machine learning
- A decision tree consists of:
  - *a root node* (or starting node),
  - *interior nodes*, and
  - *leaf nodes* (or terminating nodes).
- Each of the **non-leaf nodes** (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.
- Each of the **leaf nodes** specifies a predicted classification for the query.





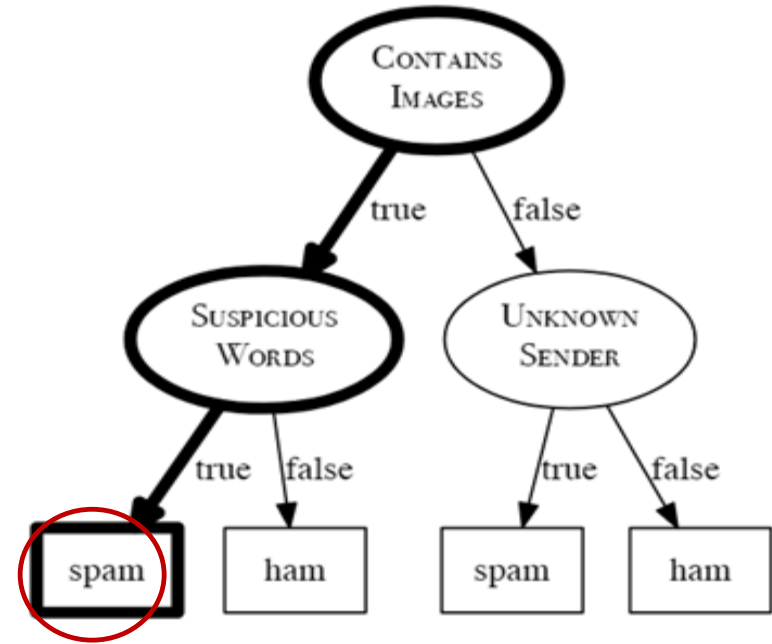
# Decision Tree Model

- A decision tree can be represented with *a disjunction (OR) of conjunctions (AND) of constraints on the feature values*.
  - Each path from the tree root to a leaf corresponds to a conjunction of feature tests. The tree itself is a disjunction of these conjunctions
  - E.g.,  $(\text{Contains Images} = \text{true} \wedge \text{Suspicious words} = \text{true})$   
 $\vee (\text{Contains Images} = \text{true} \wedge \text{Suspicious words} = \text{false})$   
 $\vee \dots$
- The tree model can be also re-represented as sets of *if-then rules*
  - E.g., IF *Contains Images = true* AND *Suspicious words = true* THEN *Class = spam*



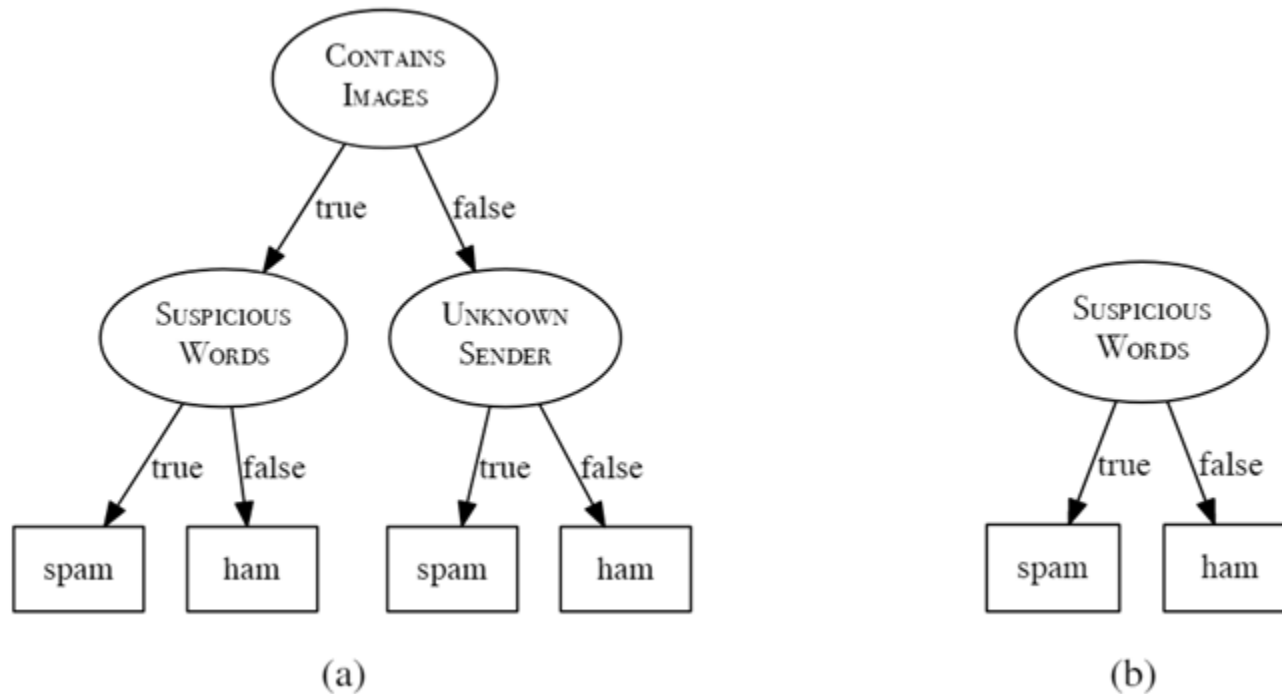
# Using Decision Tree Model

- A given test instance  
< *Suspicious words=true,*  
*Unknown sender=false,*  
*Contains Images =true,*  
*Class =?*>



- The classifier would classify the query instance as *spam* (i.e., the tree predicts that *Class=spam*>

# Different Decision Trees



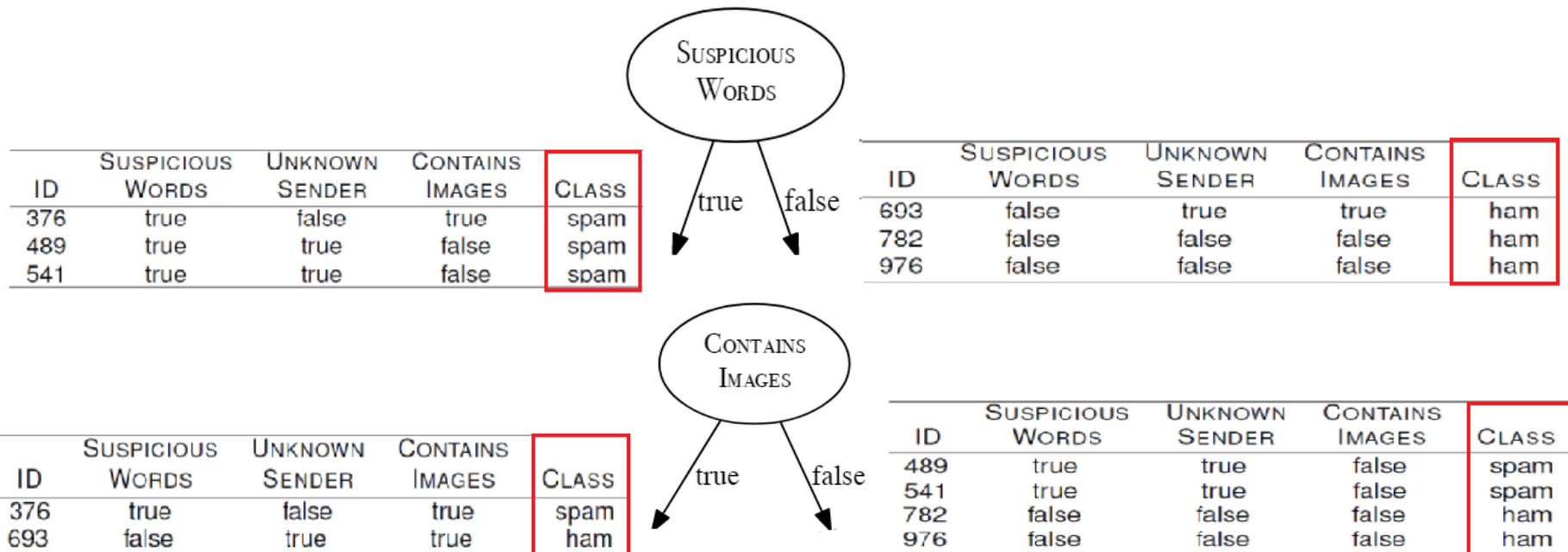
**Figure.** Two decision trees, (a) and (b), that are consistent with the instances in the spam dataset

- There might be a set of different decision trees that are all consistent with a set of training instances. *Which tree should we use?*

# Information-based Learning, Decision Tree

- According to information-based learning, descriptive features that **split the dataset into pure subsets with respect to the class label** provide information about the class label.

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

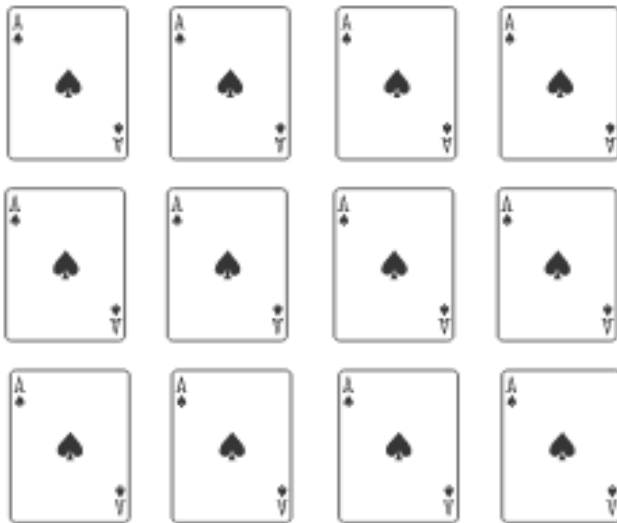


# Information-based Learning, Entropy

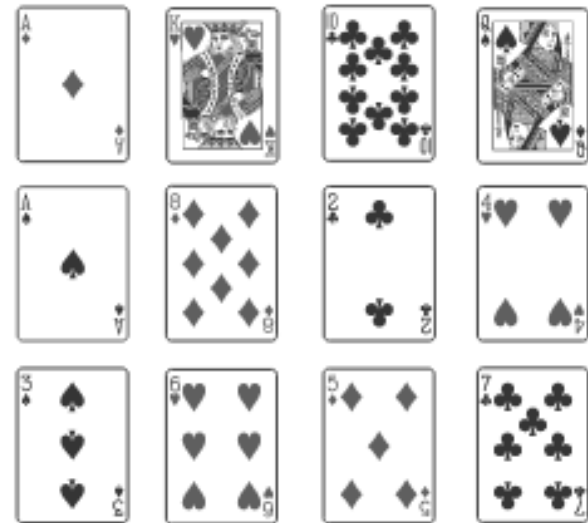
- The **ideal discriminatory feature** will partition the data into *pure* subsets where all the instances in each subset have the same classification.
- One of computational metrics of impurity of a set is **entropy**

# Idea on Entropy

- **Entropy** is related to the probability of a outcome, i.e., the uncertainty associated with guessing the result if you were to make a random selection from the set.
- Example:

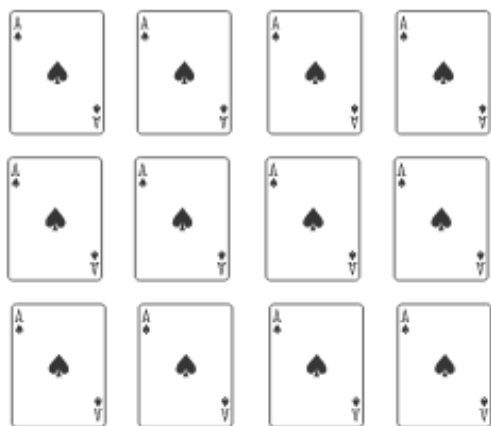


(a) **Entropy(card) = 0**,  
Zero uncertainty with guessing the result

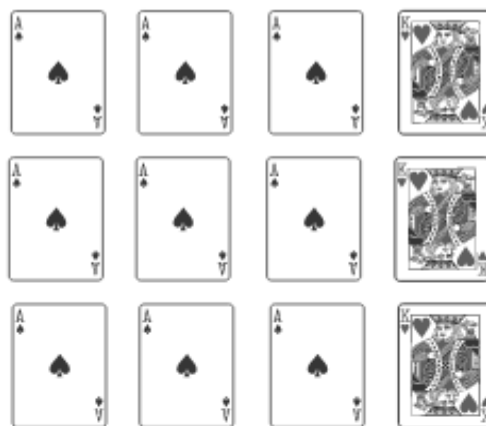


(b) **Entropy(card) = 3.58**,  
12 possible outcomes, Very uncertain for the  
result of the random selection

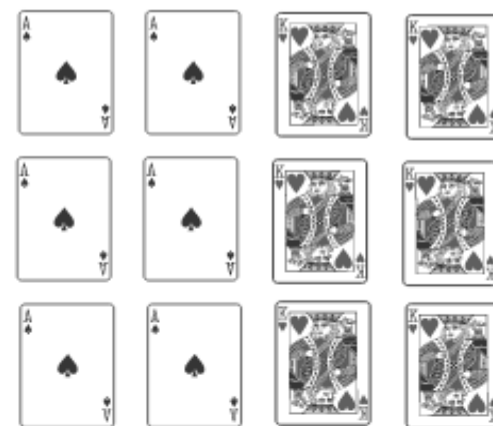
# Example



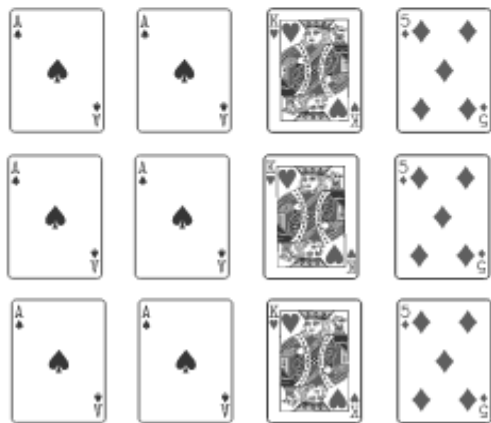
(a)  $H(card) = 0.00$



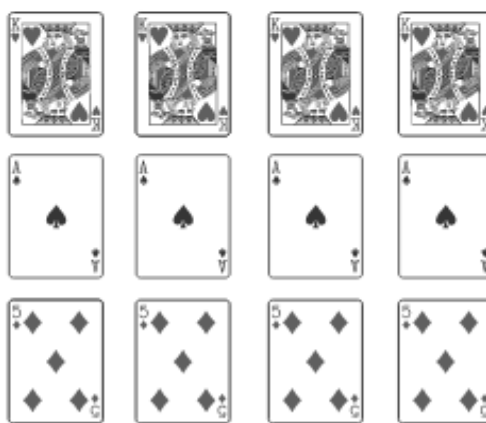
(b)  $H(card) = 0.81$



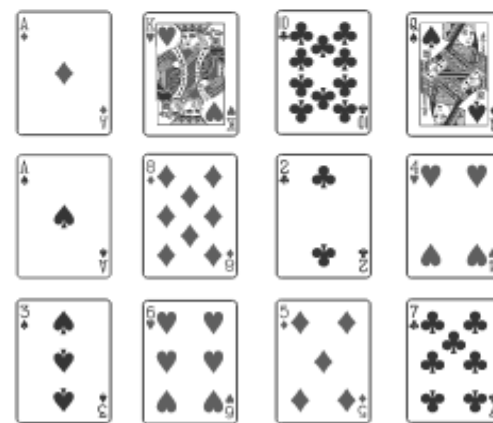
(c)  $H(card) = 1.00$



(d)  $H(card) = 1.50$



(e)  $H(card) = 1.58$



(f)  $H(card) = 3.58$

**Figure:** The entropy of different sets of playing cards measured in bits.

# Shannon's Entropy Model

- The probabilities of the different possible outcomes for the random selection is transformed to **entropy** values.
- Claude Shannon's **entropy model** defines a computational measure of the impurity – **heterogeneity** - of the elements of a set.
- **Entropy** is a weighted sum of the logs of the probabilities of each of the possible outcomes when we make a random selection from a set.

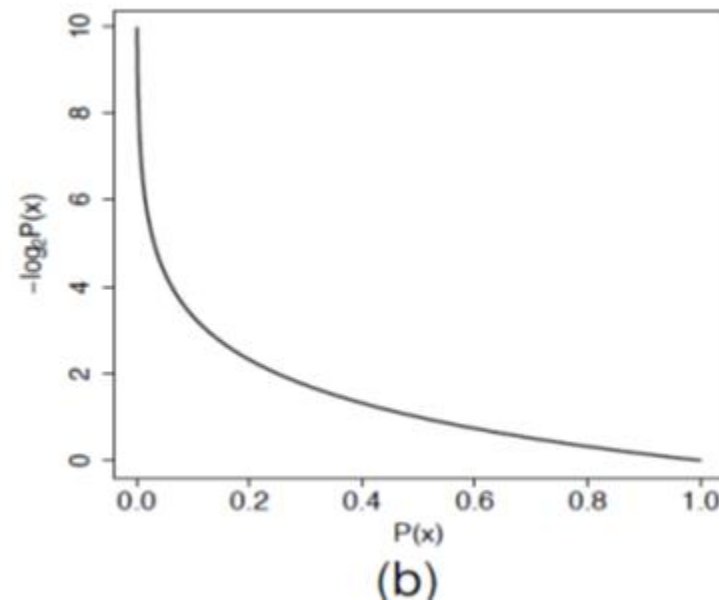
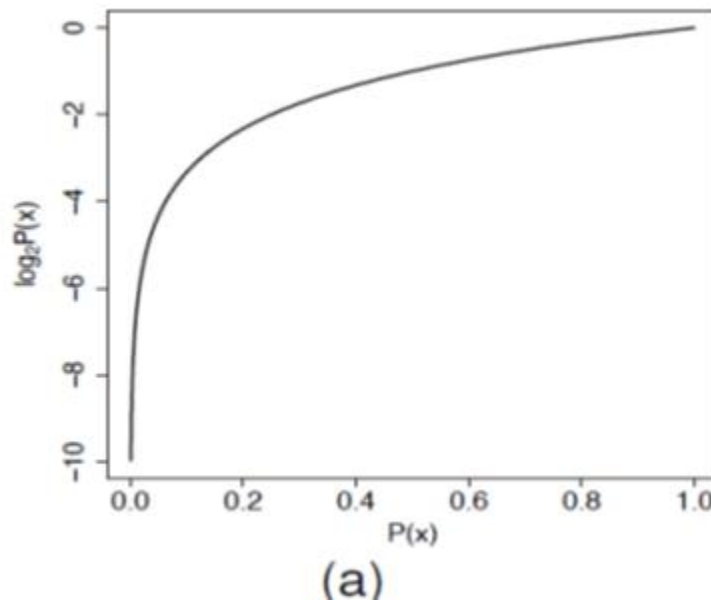
$$H(t) = Entropy(t) = - \sum_{i=1}^l (P(t = i) \times \log_s(P(t = i)))$$

- $l$  is the number of different types (levels) of things in the set
- $P(t = i)$  is the probability that the outcome of randomly selecting an element  $t$  is the type (level)  $i$ .
- $s$  is an arbitrary logarithmic base.



# Computational Metric of Entropy

- $H(t) = \text{Entropy}(t) = -\sum_{i=1}^l (P(t = i) \times \log_2(P(t = i)))$ 
  - High probability in  $P(t = i) \Rightarrow$  Low entropy (Low heterogeneity)
  - Low probability in  $P(t = i) \Rightarrow$  High entropy (High heterogeneity)
- Impact of multiplying by -1



**Figure.** (a) A graph illustrating how the value of a **binary log** (the log to the base 2) of a probability changes across the range of probability values; and (b) the impact of multiplying these values by -1.

# Exercise: Compute Entropy for Each Set

Example 1

C1	<b>0</b>
C2	<b>6</b>

$$P(C1) = 0/6 = 0, \quad P(C2) = 6/6 = 1$$

$$\text{Entropy} = -0 \log 0 - 1 \log 1 = -0 - 0 = \mathbf{0}$$

The entropy is 0 if all members of  $t$  belong to the same class.

Example 2

C1	<b>1</b>
C2	<b>5</b>

$$P(C1) = 1/6, \quad P(C2) = 5/6$$

$$\text{Entropy} = - (1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$$

Example 3

C1	<b>2</b>
C2	<b>4</b>

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Entropy} = - (2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$$

Example 4

C1	<b>3</b>
C2	<b>3</b>

$$P(C1) = 3/6 \quad P(C2) = 3/6$$

$$\text{Entropy} = - (3/6) \log_2 (3/6) - (3/6) \log_2 (3/6) = 1$$

Note that the entropy is 1 if the collection contains an equal number of positive (C1) and negative (C2) examples.

# Decision Tree Learning, Information Gain

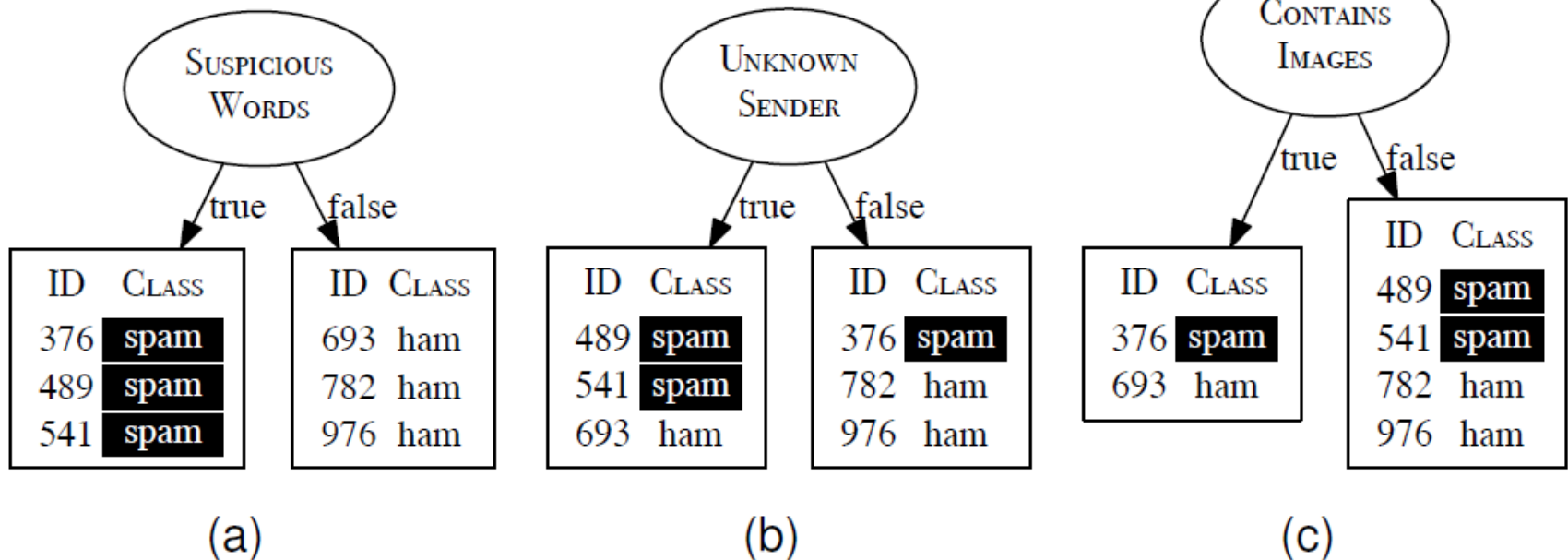
- Decision tree learning **prefers trees that use less tests (shallower trees)**.
- We can make shallow trees by testing the informative features early on the tree.
- For implementing this idea, decision tree learning uses a metric called **information gain**.

# Data Split and Information Gain

- When we partition the input data using each of descriptive features, the instances in the dataset are split differently.

**Table:** An email spam prediction dataset.

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham



# Information Gain

- The reduction in overall entropy of a prediction task when testing a specific feature can be described as **the information gain** of that feature.
- ***Information Gain***( $d, D$ ) of a feature  $d$ , a relative to a collection of examples ( $D$ ) at a node, is the **expected reduction in entropy of target feature  $t$  for data sample  $D$ .**

$$\begin{aligned} & \text{Information Gain}(d, D) \\ &= \text{Entropy}(t, D) - \sum_{l \in \text{levels}(d)} \frac{|D_{d=l}|}{|D|} \times \text{Entropy}(t, D_{d=l}) \end{aligned}$$

- The information gain defines a measure of the effectiveness of a descriptive feature in classifying the training data.

# Computing Information Gain

$$\text{Information Gain}(d, D) = H(t, D) - \text{rem}(d, D)$$

■ **Computing information gain** involves the following three steps:

1. Compute the entropy (heterogeneity, impurity) of the original dataset (or a current partition) with respect to the target feature  $t$

$$H(t, D)$$

2. For each descriptive feature  $d$ , create the sets that result by partitioning the instances using their feature values, and then sum the entropy scores of each of these subsets with their weight

$$\text{rem}(d, D) = \sum_{l \in \text{levels}(d)} \underbrace{\frac{|D_{d=l}|}{|D|}}_{\text{weighting}} \times \underbrace{\text{Entropy}(t, D_{d=l})}_{\text{Entropy of partition } D_{d=l}}$$

3. Subtract the remaining entropy value (step 2) from the original entropy value (step 1) to give the information gain.

$$IG(d, D) = H(t, D) - \text{rem}(d, D)$$

# Example

- Calculate the information gain for each of the descriptive features in the spam email dataset.

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

$$IG(d, D) = H(t, D) - rem(d, D)$$

$$H(t) = \text{Entropy}(t) = -\sum_{i=1}^l (P(t = i) \times \log_s(P(t = i)))$$

- The entropy of the dataset  $H(\text{Class}, D)$

$$\begin{aligned} H(\text{Class}, D) &= -\sum_{i=1}^2 (P(t = i) \times \log_2(P(t = i))) \\ &= -\left(\frac{3}{6} \times \log_2 \frac{3}{6} + \frac{3}{6} \times \log_2 \frac{3}{6}\right) = -(0.5 \times -1 + 0.5 \times -1) = 1 \end{aligned}$$

# Example (cont.)

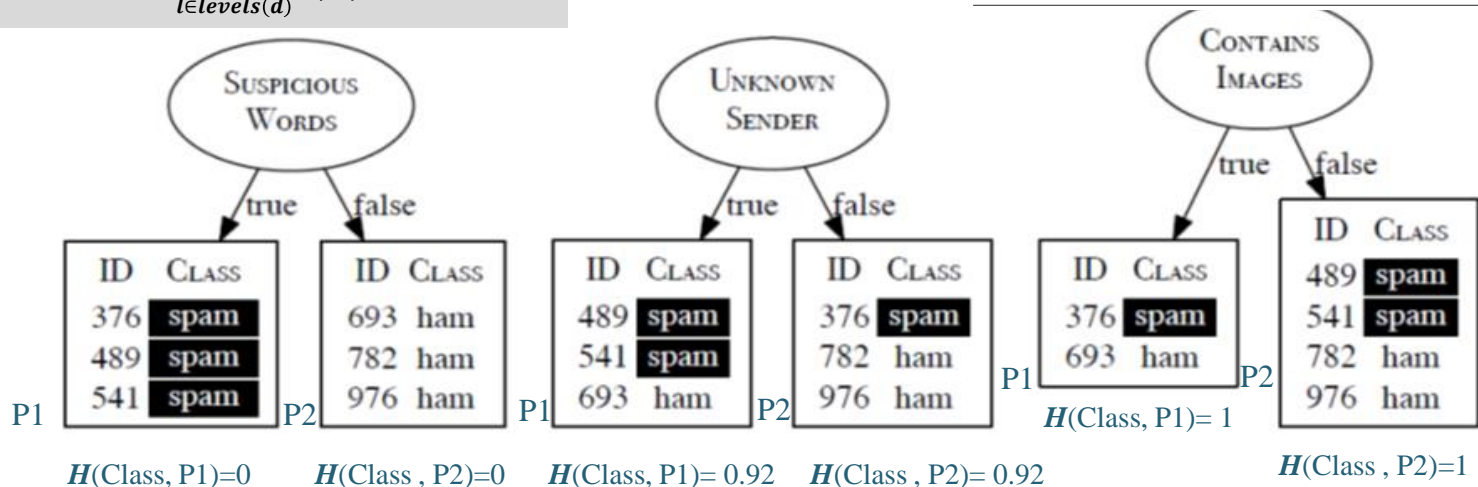
$$IG(d, D) = H(t, D) - \text{rem}(d, D)$$

$$H(t) = \text{Entropy}(t) = -\sum_{i=1}^l (P(t=i) \times \log_s(P(t=i)))$$

$$\text{rem}(d, D) = \sum_{l \in \text{levels}(d)} \frac{|D_{d=l}|}{|D|} \times H(t, D_{d=l})$$

**Table:** An email spam prediction dataset.

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham



- Information Gain (Suspicious Words, D)**

$$= H(\text{Class}, D) - \text{rem}(\text{Suspicious Words}, D)$$

$$= 1 - \left( \frac{3}{6} \times 0 + \frac{3}{6} \times 0 \right) = 1 - 0 = 1$$
- Information Gain (Unknown Sender, D)**

$$= H(\text{Class}, D) - \text{rem}(\text{Unknown Sender}, D)$$

$$= 1 - \left( \frac{3}{6} \times 0.92 + \frac{3}{6} \times 0.92 \right) = 1 - 0.92 = 0.08$$
- Information Gain (Contains Images, D)**

$$= H(\text{Class}, D) - \text{rem}(\text{Contains Images}, D)$$

$$= 1 - \left( \frac{2}{6} \times 1 + \frac{4}{6} \times 1 \right) = 1 - 1 = 0$$



# Outline

- Classification Task
- Information Theory and Information-based learning
- Decision Tree Learning
- Fundamentals
- ☞ **Standard Approach: The ID3 Algorithm**
- Extensions and Variations
- Summary

# Search Problem Setting: Decision Tree Learning

- A set of possible instances  $X$  (*training set*)
  - Each instance  $x$  in  $X$  is a feature vector
    - e.g.,  $\langle \text{Humidity}=\text{low}, \text{Wind}=\text{weak}, \text{Outlook}=\text{rain}, \text{Temp}=\text{hot} \rangle$
- A unknown target function:  $f: X \rightarrow y$ 
  - e.g.,  $y = 1$  if we play tennis on this day, else 0
- A set of function hypotheses  $H = \{h \mid h: X \rightarrow y\}$  (search space)
  - Each hypothesis  $h$  is a decision tree
- Each tree sorts  $x$  to leaf, which assigns  $y$ 
  - Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance.

# Algorithms of Decision Tree Induction

- There are many possible decision trees that can be constructed from a particular data set. Some trees are better than others
- Finding an optional one is computationally expensive due to the exponential size of the search space.
- There are several algorithms which induce a reasonably accurate, although suboptimal, decision tree in a reasonable amount of time.
- Usually, decision tree learning algorithms employ a *greedy strategy* to grow the decision tree in a *top-down fashion* by making a series of locally optimal decisions about which attribute to use when partitioning the training data

# Decision Tree Learning Algorithms

## ■ Basic Algorithms

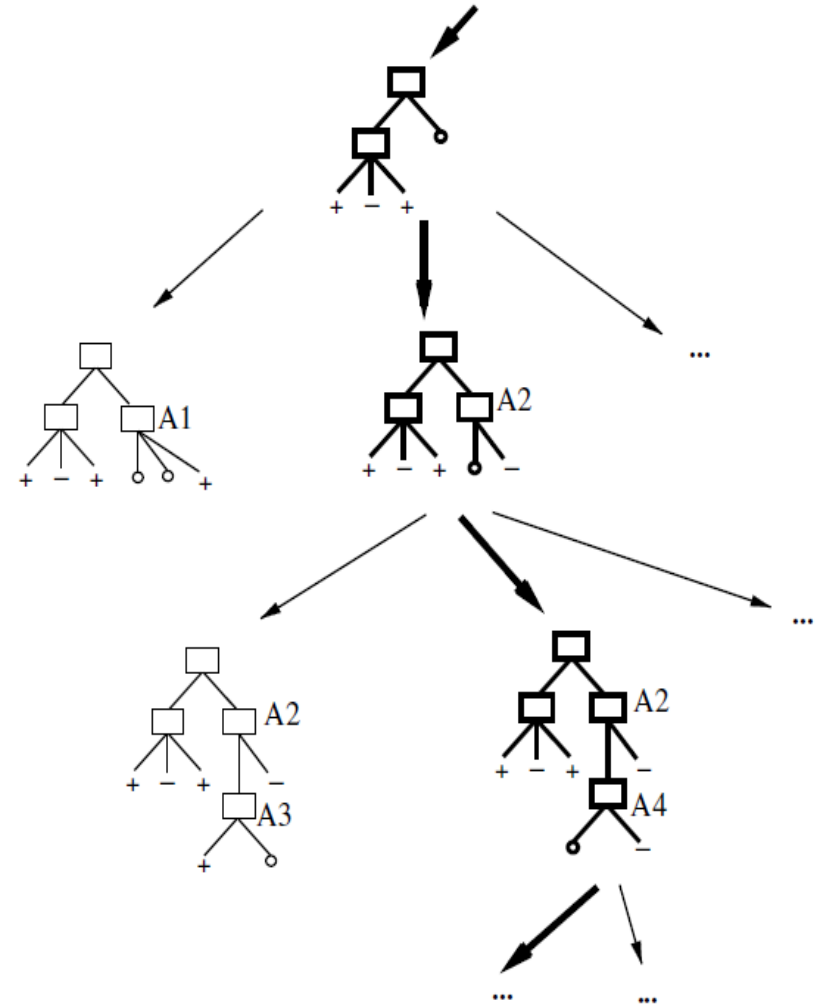
- **ID3** algorithm (Quinlan 1986)
- Its successor **C4.5** algorithm (Quinlan 1993)
- Hunt's algorithm

## ■ Other Algorithms

- RainForest
- BOAT
- SLIQ
- SPRINT
- PUBLIC

# Standard Approach: ID3 Algorithm

- **ID3 Algorithm** (Iterative Dichotomizer 3) attempts to create the shallowest tree that is consistent with the data that it is given.
- The ID3 algorithm builds the tree in a *recursive, depth-first manner*, beginning at the root node and working down to the leaf nodes.



**Figure:** Hypothesis (Model) Space Search by ID3

# ID3 Algorithm

1. The algorithm begins by choosing the best descriptive feature to test (i.e., the best question to ask first) using information gain.
2. A root node is then added to the tree and labelled with the selected test feature.
3. The training dataset is then partitioned using a test with the selected feature.
4. For each partition a branch is grown from the node.
5. The process is then repeated for each of these branches using the relevant partition of the training set in place of the full training set and with the selected test feature excluded from further testing.

# Algorithm Stopping Criterion

## ■ Conditions for stopping the algorithm

- i. All of the instances in the dataset have the same classification. Then return a leaf node with that classification as its label.
- ii. The set of features left to test is empty. Then return a leaf node with the majority class of the dataset as its label.
- iii. The dataset is empty (No instances are left). Return a leaf node with the majority class label of the dataset at the parent node

---

**Algorithm** Pseudocode description of the ID3 algorithm.

---

**Require:** set of descriptive features  $\mathbf{d}$

**Require:** set of training instances  $\mathcal{D}$

- 1: **if** all the instances in  $\mathcal{D}$  have the same target level  $C$  **then return** a decision tree consisting of a leaf node with label  $C$
  - 2: **else if**  $\mathbf{d}$  is empty **then return** a decision tree consisting of a leaf node with the label of the majority target level in  $\mathcal{D}$
  - 3: **else if**  $\mathcal{D}$  is empty **then return** a decision tree consisting of a leaf node with the label of the majority target level of the dataset of the immediate parent node
  - 4: **else**
  - 5:      $\mathbf{d}[best] \leftarrow \arg \max_{d \in \mathbf{d}} IG(d, \mathcal{D})$
  - 6:     make a new node,  $Node_{\mathbf{d}[best]}$ , and label it with  $\mathbf{d}[best]$
  - 7:     partition  $\mathcal{D}$  using  $\mathbf{d}[best]$
  - 8:     remove  $\mathbf{d}[best]$  from  $\mathbf{d}$
  - 9:     **for** each partition  $\mathcal{D}_i$  of  $\mathcal{D}$  **do**
  - 10:         grow a branch from  $Node_{\mathbf{d}[best]}$  to the decision tree created by rerunning *ID3* with  $\mathcal{D} = \mathcal{D}_i$
  - 11:     **end for**
  - 12: **end if**
-



# A Worked Example: ID3 Algorithm

## ■ Dataset: Vegetable Distribution Prediction Data

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	high	chapparal
2	true	moderate	low	riparian
3	true	steep	medium	riparian
4	false	steep	medium	chapparal
5	false	flat	high	conifer
6	true	steep	highest	conifer
7	true	steep	high	chapparal

**Table:** The vegetation classification dataset

$H(\text{VEGETATION}, \mathcal{D})$  \* Here, the heterogeneity is computed using Entropy.

$$\begin{aligned} &= - \sum_{l \in \left\{ \begin{array}{l} \text{chapparal}, \\ \text{riparian}, \\ \text{conifer} \end{array} \right\}} P(\text{VEGETATION} = l) \times \log_2 (P(\text{VEGETATION} = l)) \\ &= - \left( \left( \frac{3}{7} \times \log_2 \left( \frac{3}{7} \right) \right) + \left( \frac{2}{7} \times \log_2 \left( \frac{2}{7} \right) \right) + \left( \frac{2}{7} \times \log_2 \left( \frac{2}{7} \right) \right) \right) \\ &= 1.5567 \end{aligned}$$

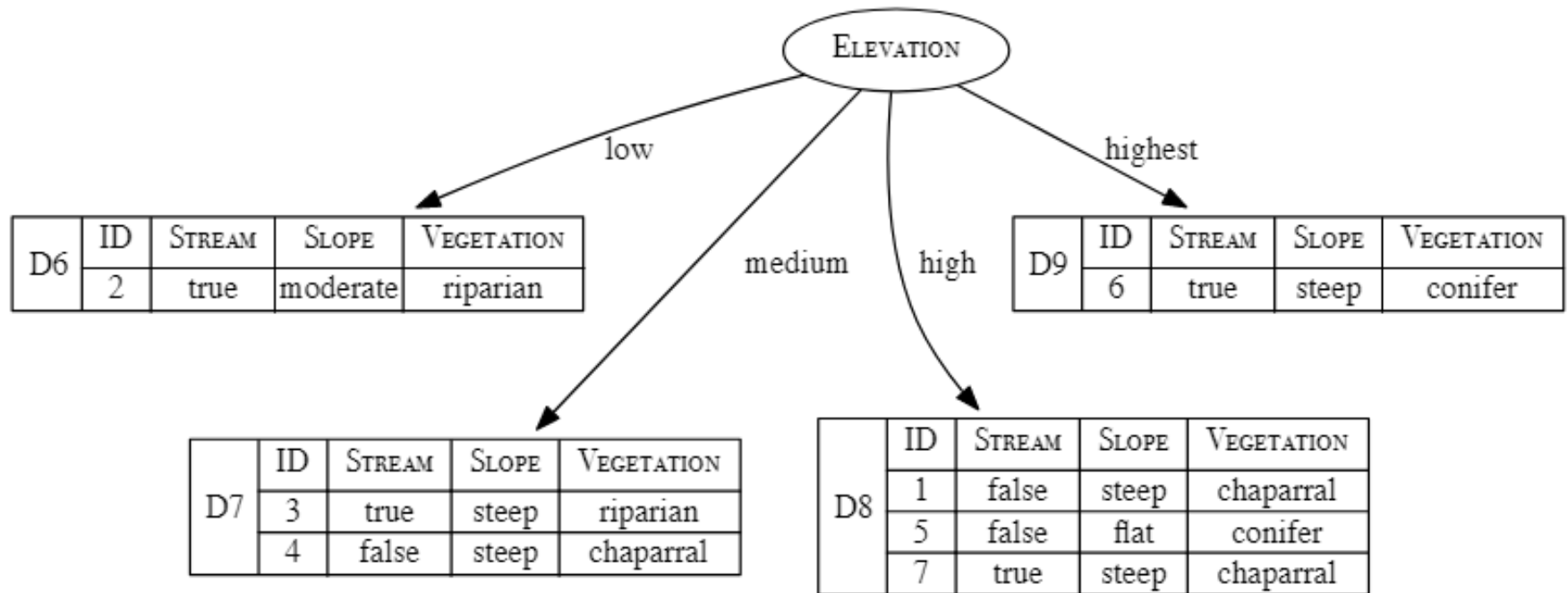
# Example (Cont.)

$$\text{Information Gain}(d, D) = H(t, D) - \sum_{l \in \text{levels}(d)} \frac{|D_{d=l}|}{|D|} \times H(t, D_{d=l})$$

Split by Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
STREAM	<i>true</i>	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	1.5000	1.2507	0.3060
	<i>false</i>	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.9183		
SLOPE	<i>flat</i>	$\mathcal{D}_3$	$\mathbf{d}_5$	0.0	0.9793	0.5774
	<i>moderate</i>	$\mathcal{D}_4$	$\mathbf{d}_2$	0.0		
	<i>steep</i>	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$	1.3710		
ELEVATION	<i>low</i>	$\mathcal{D}_6$	$\mathbf{d}_2$	0.0	0.6793	0.8774
	<i>medium</i>	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$	1.0		
	<i>high</i>	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$	0.9183		
	<i>highest</i>	$\mathcal{D}_9$	$\mathbf{d}_6$	0.0		

**Table:** Partition sets (Part.), entropy, remainder(Rem.) and information gain (Info. Gain) by feature for the dataset

## Example (Cont.)



**Figure:** The decision tree after the data has been split using ELEVATION

# Example (Cont.)

	ID	STREAM	SLOPE	VEGETATION
D7	3	true	steep	riparian
	4	false	steep	chaparral

$$H(\text{VEGETATION}, \mathcal{D}_7)$$

$$= - \sum_{l \in \left\{ \begin{smallmatrix} \text{chaparral,} \\ \text{riparian,} \\ \text{conifer} \end{smallmatrix} \right\}} P(\text{VEGETATION} = l) \times \log_2(P(\text{VEGETATION} = l))$$

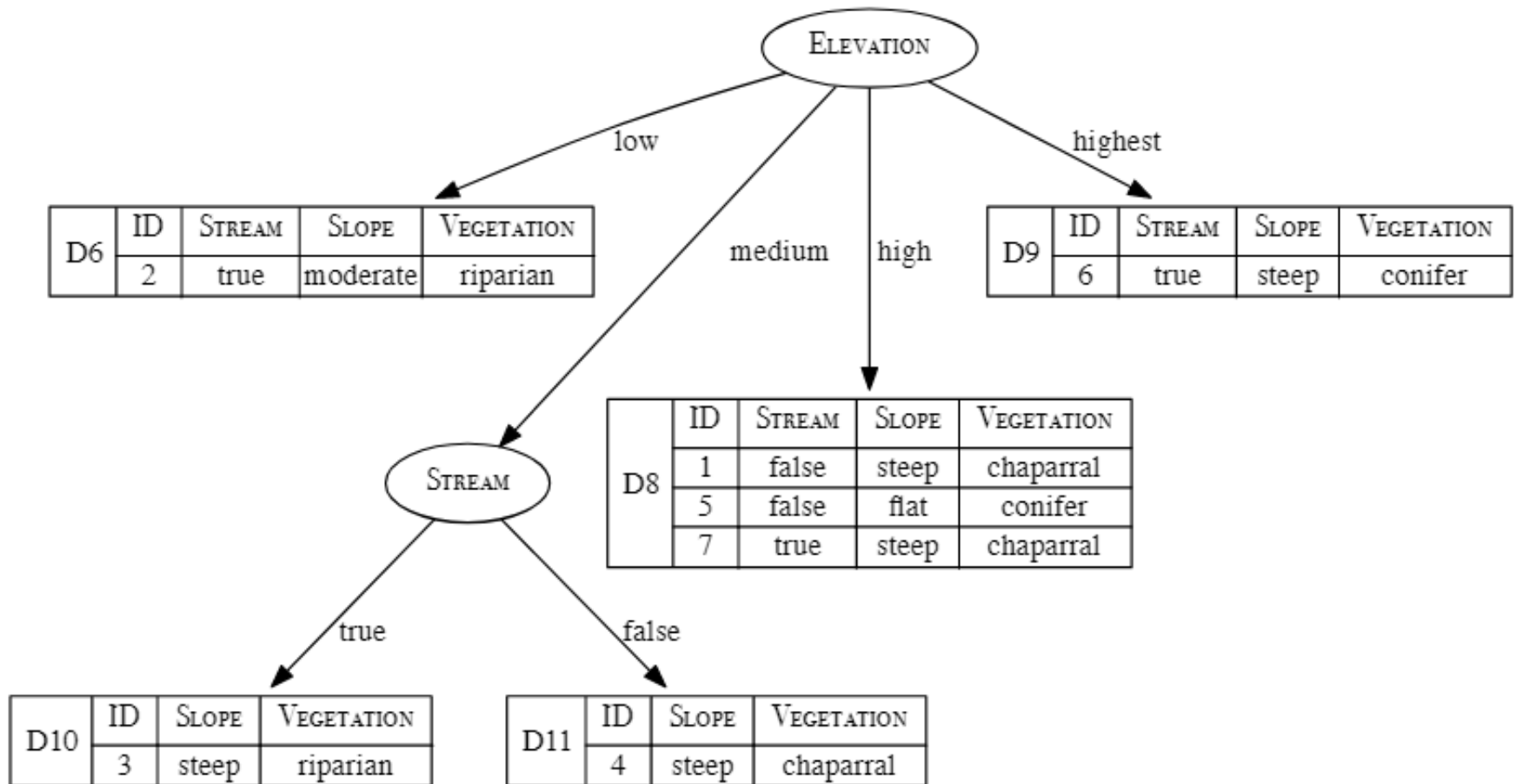
$$= - \left( \left( \frac{1}{2} \times \log_2\left(\frac{1}{2}\right) \right) + \left( \frac{1}{2} \times \log_2\left(\frac{1}{2}\right) \right) + \left( \frac{0}{2} \times \log_2\left(\frac{0}{2}\right) \right) \right)$$

$$= 1.0 \text{ bits}$$

Split by Feature	Level	Part.	Instances	Partition Entropy	Rem.	Info. Gain
STREAM	<i>true</i>	$\mathcal{D}_{10}$	<b>d<sub>3</sub></b>	0.0	0.0	1.0
	<i>false</i>	$\mathcal{D}_{11}$	<b>d<sub>4</sub></b>	0.0		
SLOPE	<i>flat</i>	$\mathcal{D}_{12}$		0.0	1.0	0.0
	<i>moderate</i>	$\mathcal{D}_{13}$		0.0		
	<i>steep</i>	$\mathcal{D}_{14}$	<b>d<sub>3</sub>, d<sub>4</sub></b>	1.0		

**Table:** Partition sets (Part.), entropy, remainder(Rem.) and information gain (Info. Gain) by feature for  $\mathcal{D}_7$  partition

## Example (Cont.)



**Figure:** The state of the decision tree after the  $D_7$  partition has been split using STREAM

# Example (Cont.)

$$H(\text{VEGETATION}, \mathcal{D}_8)$$

$$= - \sum_{l \in \left\{ \begin{matrix} \text{chapparral}, \\ \text{riparian}, \\ \text{conifer} \end{matrix} \right\}} P(\text{VEGETATION} = l) \times \log_2 (P(\text{VEGETATION} = l))$$

$$= - \left( \left( \frac{2}{3} \times \log_2 \left( \frac{2}{3} \right) \right) + \left( \frac{0}{3} \times \log_2 \left( \frac{0}{3} \right) \right) + \left( \frac{1}{3} \times \log_2 \left( \frac{1}{3} \right) \right) \right)$$

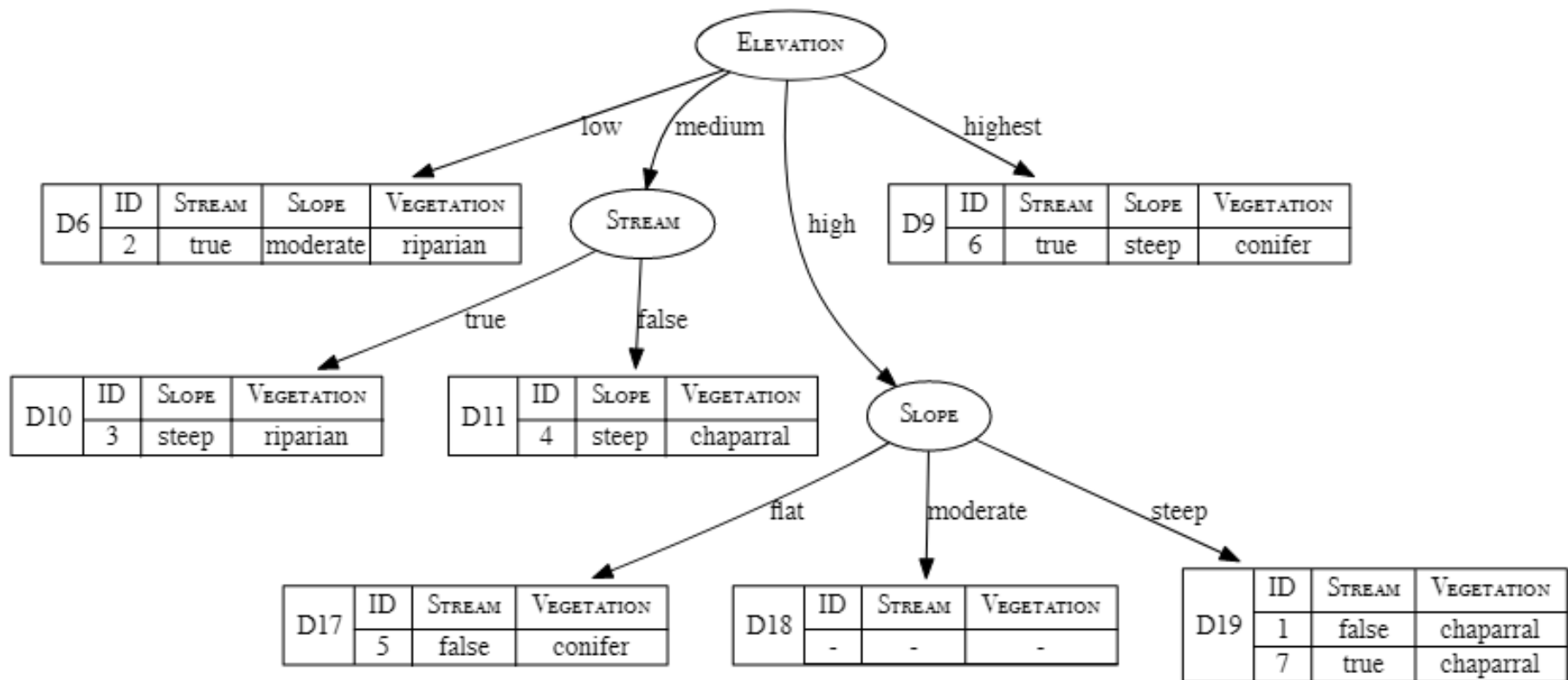
$$= 0.9183 \text{ bits}$$

	ID	STREAM	SLOPE	VEGETATION
D8	1	false	steep	chapparral
	5	false	flat	conifer
	7	true	steep	chapparral

Split by				Partition		Info.
Feature	Level	Part.	Instances	Entropy	Rem.	Gain
STREAM	<i>true</i>	$\mathcal{D}_{15}$	$\mathbf{d}_7$	0.0	0.6666	0.2517
	<i>false</i>	$\mathcal{D}_{16}$	$\mathbf{d}_1, \mathbf{d}_5$	1.0		
SLOPE	<i>flat</i>	$\mathcal{D}_{17}$	$\mathbf{d}_5$	0.0	0.0	0.9183
	<i>moderate</i>	$\mathcal{D}_{18}$		0.0		
	<i>steep</i>	$\mathcal{D}_{19}$	$\mathbf{d}_1, \mathbf{d}_7$	0.0		

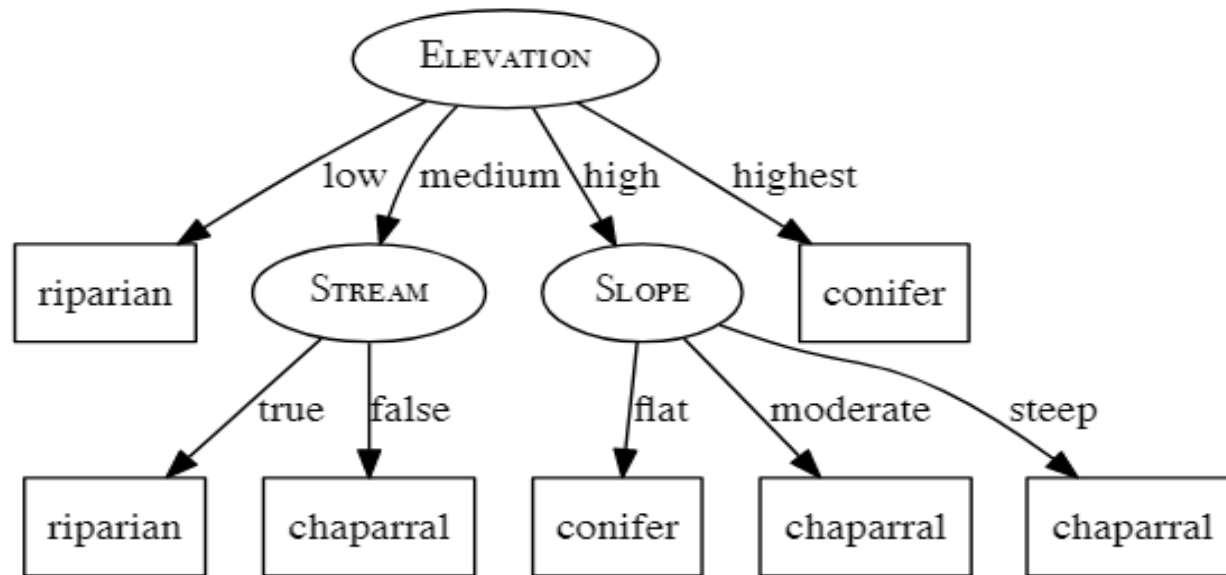
**Table:** Partition sets (Part.), entropy, remainder(Rem.) and information gain (Info. Gain) by feature for  $\mathcal{D}_8$  partition

# Example (Cont.)



**Figure:** The state of the decision tree after the  $D_8$  partition has been split using SLOPE

# Example : Final Decision Tree and The Usage

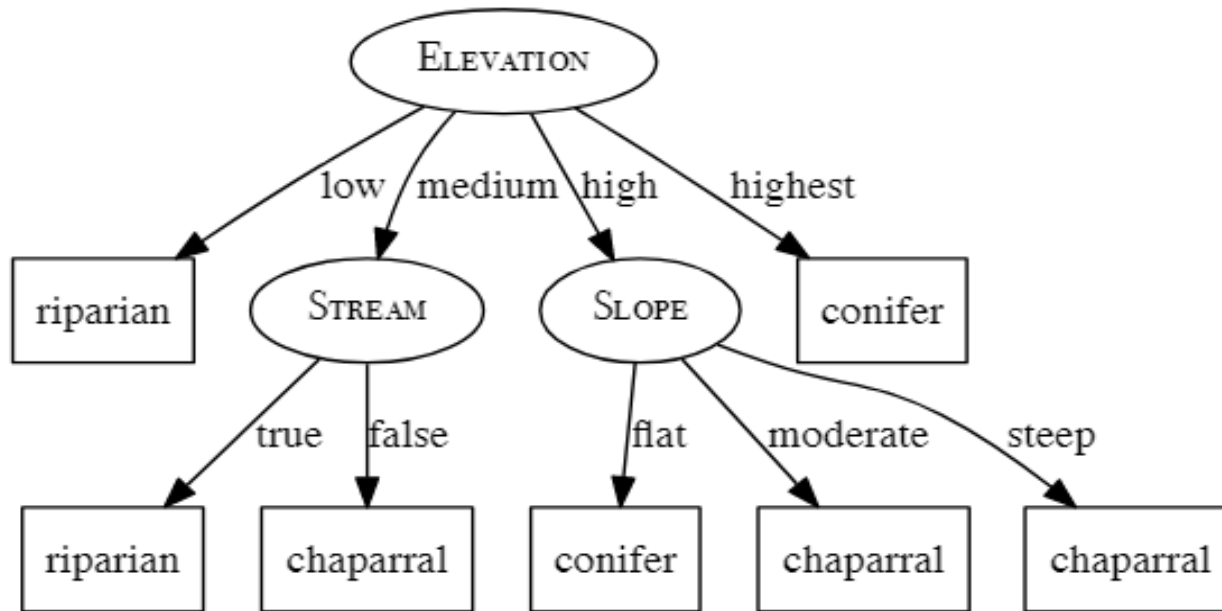


**Figure:** The final vegetation classification decision tree

- Based on the stopping conditions, **each leaf node is labeled as following:**
  - All of the instances in a leaf node have the same class. The leaf node has the classification as its label.
  - If the leaf node has mixed classes. The leaf node is labeled with the majority class of the instances.
  - If the leaf node is empty. The node is labeled with the majority class label of its parent node



# Example : Use the Decision Tree Model



**Figure:** The final vegetation classification decision tree

- What prediction will this decision tree model return for the following query?

< STREAM = 'true', SLOPE='Moderate', ELEVATION='High', VEGETATION=?>

**VEGETATION = 'Chaparral'**

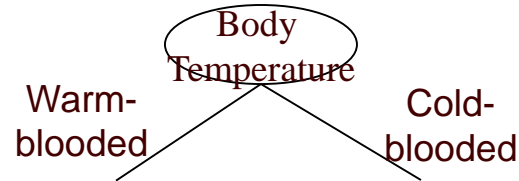
This is an example where the model is attempting to **generalize** beyond the training dataset.

# Methods for Expressing Test Conditions

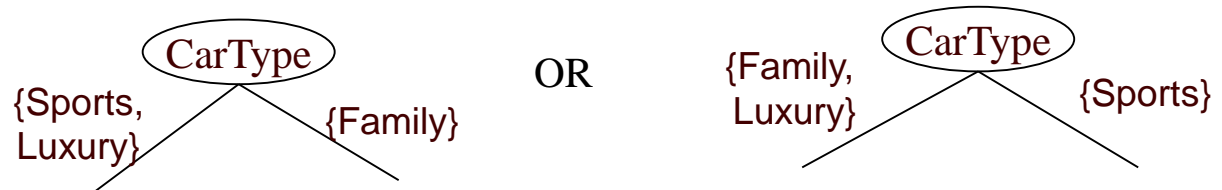
- The **splitting criterion** determines which feature is chosen as the test condition and how the training instances should be partitioned to the child nodes
- Expressing the test condition depends on feature types
  - Binary
  - Nominal
  - Ordinal
  - Continuous
- and depends on number of ways to split
  - 2-way (Binary) split
  - Multi-way split

# Split by Categorical Feature

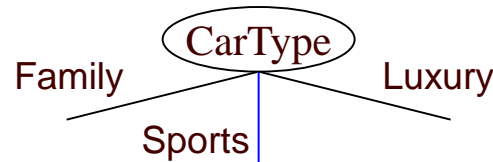
- A way for expressing a feature test condition and its corresponding outcomes for different feature types
- **Binary features:** **binary split** divides values into two subsets.



- **Nominal features:** can produce binary or multiway splits
  - **Binary split** needs to find optional partitioning to divide values into two subsets



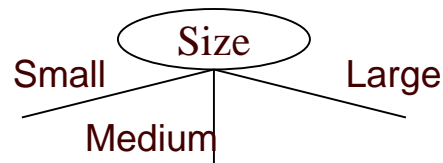
- **Multi-way split** uses as many partitions as distinct values



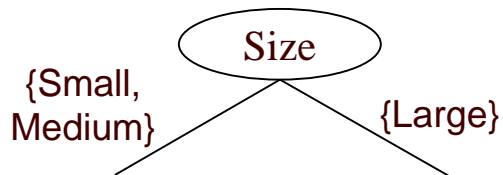
# Split by Categorical Feature (Cont.)

- **Ordinal features** can also produce binary or multiway splits

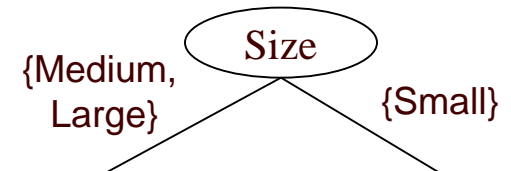
- **Multi-way split**



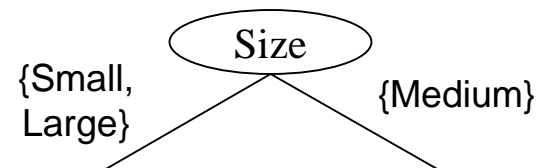
- **Binary split:** Ordinal attribute values can be grouped as long as the grouping does not violate the order property of the attributes



OR

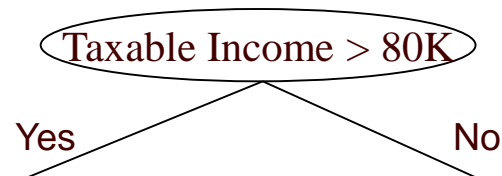


- Q: What about this split?

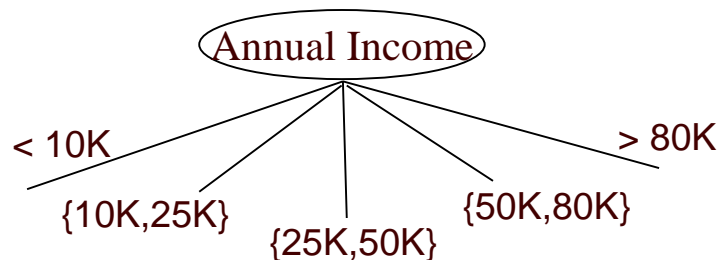


# Split by Contiguous Feature

- **Contiguous features** can produce binary or multiway splits with discretization. **The algorithm considers all possible splits and finds the best cut** – More computation intensive
  - **Binary split:**  $(A < v)$  or  $(A \geq v)$



- **Multi-way split:**  $(v_i \leq A < v_{i+1} \text{ for } i = 1, \dots, k)$



# Insights of ID3

- ID3 maintains only a single current hypothesis (model) as it searches through the space of decision trees.
  - No ability to determine how many alternative decision trees are consistent with the available training data.
- ID3 performs no backtracking in its search
  - The found model is a locally optimal solution that is not globally optimal.
- ID3 uses all the available training instances at each step in the search to make statistically based decisions
  - Robust to noisy data ...

# Inductive Bias in ID3

- **Inductive bias** in machine learning refers to the inherent assumptions or constraints that are incorporated into a learning algorithm (learner or model) to guide its generalization from training data to unseen or new data.
- Inductive bias helps a model to navigate the vast hypothesis space and prioritize certain hypotheses over others, even before it sees any training data.
- What is **ID3's inductive bias**?
  - ID3 prefers shorter (shallower) tree over longer trees. That's a bias towards simpler trees
  - ID3 prefers attributes that offer clear partitions early - Attributes that provide the highest information gain will be placed closer to the root of the tree.

# Outline

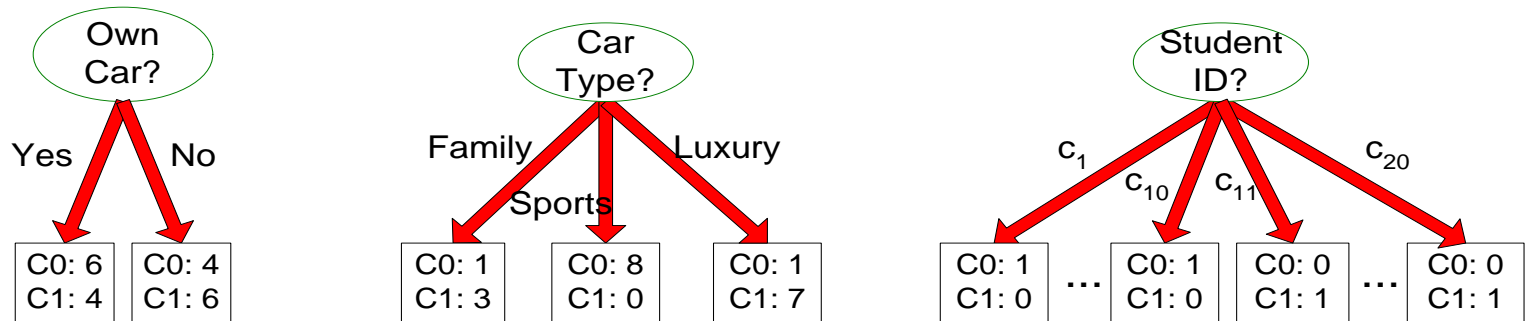
- Classification Task
- Information Theory and Information-based Learning
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- ☞ **Extensions and Variations**
  - Alternative Feature Selection Metrics
  - Handling Continuous Descriptive Features
  - Predicting Continuous Targets
  - Overfitting and Tree Pruning
- Summary



# Attributes with Many Values

## ■ Limitation of Information Gain:

- Among candidate descriptive features, a feature with many distinct values will have the largest information gain. The learning algorithm will select the feature for data partition because the subsets are more likely to be pure.



\* According to IG value, Student ID will be chosen for the test condition.

- The entropy-based information gain is biased towards choosing attributes with a large number of values
- This may result in **overfitting** (selection of an attribute that is non-optimal for prediction) and another problem is **fragmentation**

# Alternative Feature Selection Metric

- One way of addressing the issue of information gain is to use **information gain ratio** which is computed by dividing the information gain of a feature by the amount of information used to determine the value of the feature (i.e., split information) :

$$GR(d, D) = \frac{IG(d, D)}{-\sum_{l \in levels(d)} (P(d = l) \times \log_2(P(d = l)))}$$

, where  $IG(d, D)$  is the information gain of the feature  $d$  for the data set  $D$  and  $P(d = l) = \frac{|D_{d=l}|}{|D|}$

- The large information gain due to small partitions will be penalized with including the number of outcomes
- Information gain ratio is used in C4.5 algorithm.

# Example: Compare IG and GR

## ■ Dataset

**Table:** The vegetation classification dataset.

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	high	chaparral
2	true	moderate	low	riparian
3	true	steep	medium	riparian
4	false	steep	medium	chaparral
5	false	flat	high	conifer
6	true	steep	highest	conifer
7	true	steep	high	chaparral

$$H(\text{STREAM}, \mathcal{D})$$

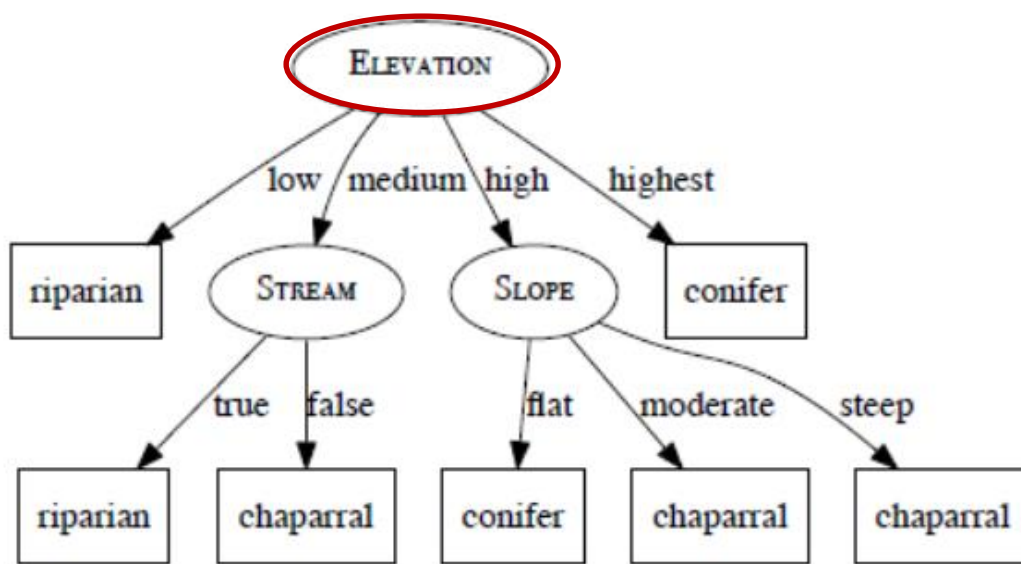
$$\begin{aligned} &= - \sum_{l \in \left\{ \begin{smallmatrix} \text{true}, \\ \text{false} \end{smallmatrix} \right\}} P(\text{STREAM} = l) \times \log_2 (P(\text{STREAM} = l)) \\ &= - \left( \left( \frac{4}{7} \times \log_2 \left( \frac{4}{7} \right) \right) + \left( \frac{3}{7} \times \log_2 \left( \frac{3}{7} \right) \right) \right) \\ &= 0.9852 \end{aligned}$$

$$H(\text{SLOPE}, \mathcal{D})$$

$$\begin{aligned} &= - \sum_{l \in \left\{ \begin{smallmatrix} \text{flat}, \\ \text{moderate}, \\ \text{steep} \end{smallmatrix} \right\}} P(\text{SLOPE} = l) \times \log_2 (P(\text{SLOPE} = l)) \\ &= - \left( \left( \frac{1}{7} \times \log_2 \left( \frac{1}{7} \right) \right) + \left( \frac{1}{7} \times \log_2 \left( \frac{1}{7} \right) \right) \right. \\ &\quad \left. + \left( \frac{5}{7} \times \log_2 \left( \frac{5}{7} \right) \right) \right) \\ &= 1.1488 \end{aligned}$$

$$H(\text{ELEVATION}, \mathcal{D})$$

$$\begin{aligned} &= - \sum_{l \in \left\{ \begin{smallmatrix} \text{low}, \\ \text{medium}, \\ \text{high}, \\ \text{highest} \end{smallmatrix} \right\}} P(\text{ELEVATION} = l) \times \log_2 (P(\text{ELEVATION} = l)) \\ &= - \left( \left( \frac{1}{7} \times \log_2 \left( \frac{1}{7} \right) \right) + \left( \frac{2}{7} \times \log_2 \left( \frac{2}{7} \right) \right) \right. \\ &\quad \left. + \left( \frac{3}{7} \times \log_2 \left( \frac{3}{7} \right) \right) + \left( \frac{1}{7} \times \log_2 \left( \frac{1}{7} \right) \right) \right) \\ &= 1.8424 \end{aligned}$$



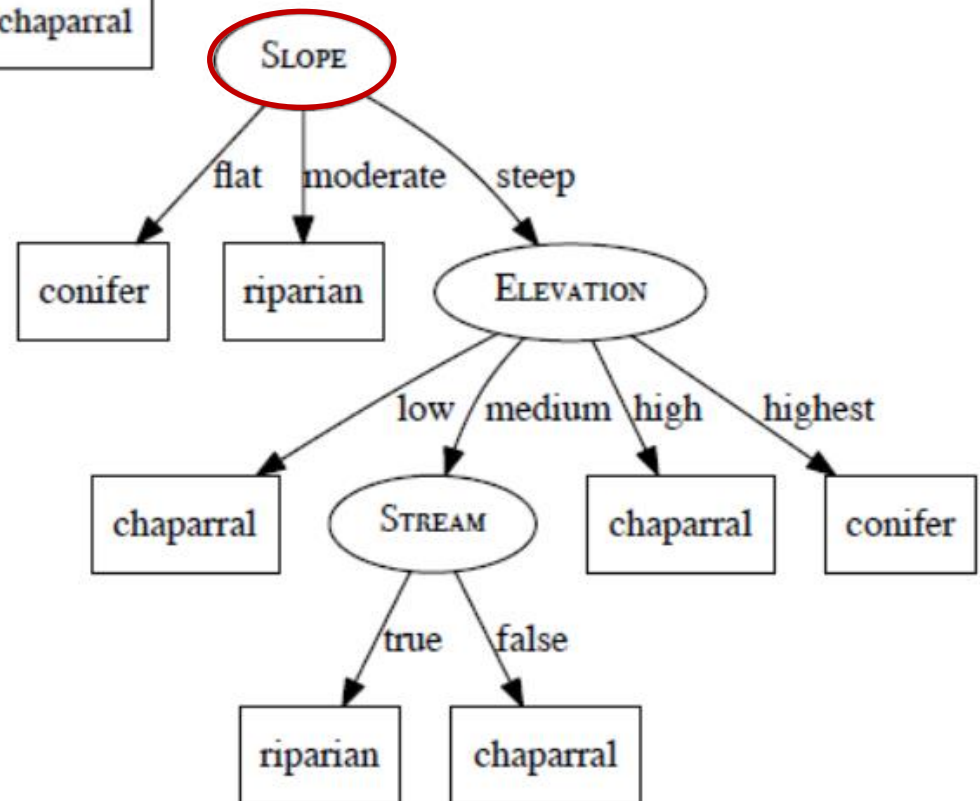
**Figure:** The vegetation classification decision tree generated using **information gain**.

### Information Gain from the root node

$$IG(\text{STREAM}, D) = 0.3060$$

$$IG(\text{SLOPE}, D) = 0.5774$$

$$IG(\text{ELEVATION}, D) = 0.8774$$



**Figure:** The vegetation classification decision tree generated using **information gain ratio**.

### Gain Ratio from the root node

$$GR(\text{STREAM}, D) = \frac{0.3060}{0.9852} = 0.3106$$

$$GR(\text{SLOPE}, D) = \frac{0.5774}{1.1488} = 0.5026$$

$$GR(\text{ELEVATION}, D) = \frac{0.8774}{1.8424} = 0.4762$$

# Alternative Impurity Measures

- There are alternative impurity measures beside Entropy
- **Gini index:**

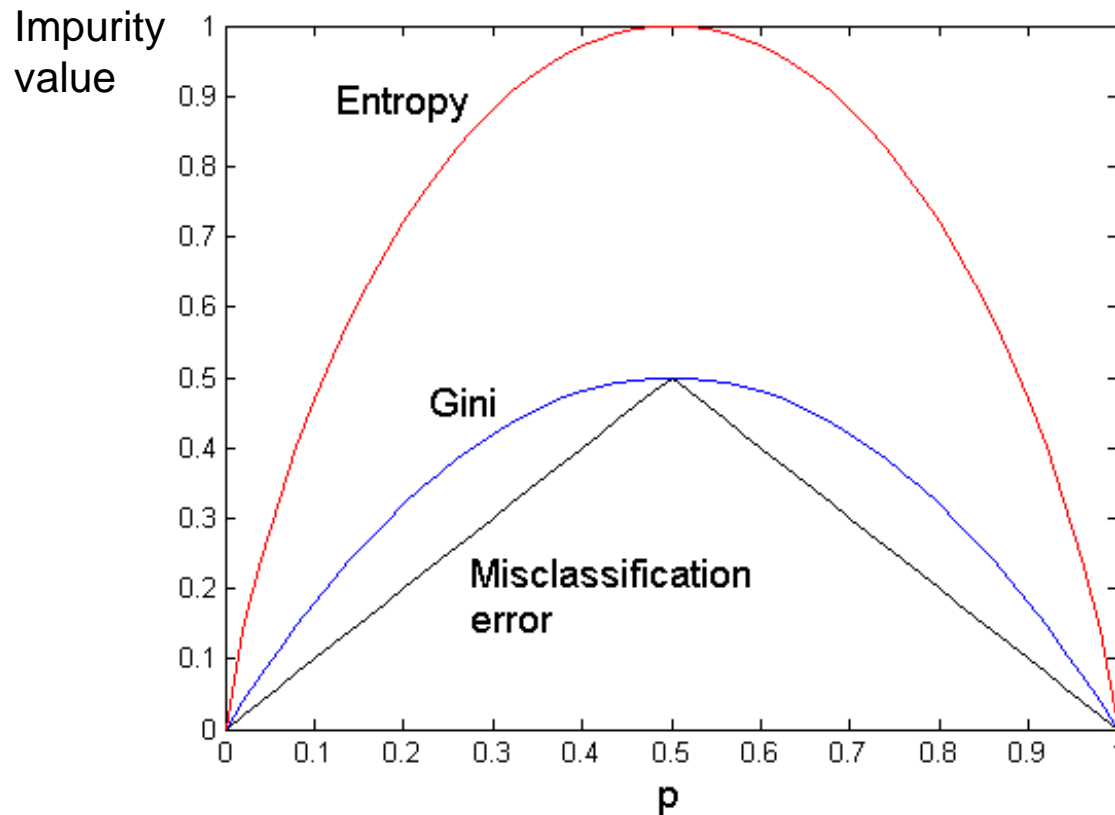
$$Gini(t, D) = 1 - \sum_{l \in levels(t)} P(d = l)^2$$

- The Gini index can be thought of as calculating how often you would misclassify an instance in the dataset if you classified it based on the distribution of classifications in the dataset.
- **Misclassification error:**

$$Missclassi. Error(t, D) = 1 - \max_{l \in levels(t)} P(d = l)$$

# Comparing Impurity Measures

In case of a 2-class problem:



P: fraction of instances that belong to one of the two classes

\*Max value when the class distribution is uniform (i.e., when  $p=0.5$ )

\*Min value when all the records belong to the same class (i.e., when  $p$  equals 0 or 1).

# Gini Index-based Information Gain

- **Gini index based information gain** can be calculated using the Gini index by replacing the entropy measure with the Gini index.

$$IG(d, D) = Gini(t, D) - \sum_{l \in levels(d)} \frac{|D_{d=l}|}{|D|} \times Gini(t, D_{d=l})$$

# Example of Gini Index and Gini Index based IG

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

$Gini(VEGETATION, \mathcal{D})$

$$\begin{aligned}
 &= 1 - \sum_{l \in \left\{ \begin{smallmatrix} chapparal, \\ riparian, \\ conifer \end{smallmatrix} \right\}} P(VEGETATION = l)^2 \\
 &= 1 - \left( (3/7)^2 + (2/7)^2 + (2/7)^2 \right) \\
 &= 0.6531
 \end{aligned}$$

Split by Feature	Level	Part.	Instances	Partition Gini Index	Rem.	Info. Gain
STREAM	<i>true</i>	$\mathcal{D}_1$	$\mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_6, \mathbf{d}_7$	0.6250	0.5476	0.1054
	<i>false</i>	$\mathcal{D}_2$	$\mathbf{d}_1, \mathbf{d}_4, \mathbf{d}_5$	0.4444		
SLOPE	<i>flat</i>	$\mathcal{D}_3$	$\mathbf{d}_5$	0.0	0.4000	0.2531
	<i>moderate</i>	$\mathcal{D}_4$	$\mathbf{d}_2$	0.0		
	<i>steep</i>	$\mathcal{D}_5$	$\mathbf{d}_1, \mathbf{d}_3, \mathbf{d}_4, \mathbf{d}_6, \mathbf{d}_7$	0.5600		
ELEVATION	<i>low</i>	$\mathcal{D}_6$	$\mathbf{d}_2$	0.0	0.3333	0.3198
	<i>medium</i>	$\mathcal{D}_7$	$\mathbf{d}_3, \mathbf{d}_4$	0.5000		
	<i>high</i>	$\mathcal{D}_8$	$\mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_7$	0.4444		
	<i>highest</i>	$\mathcal{D}_9$	$\mathbf{d}_6$	0.0		



# Decision Tree (C4.5) with Gini Index

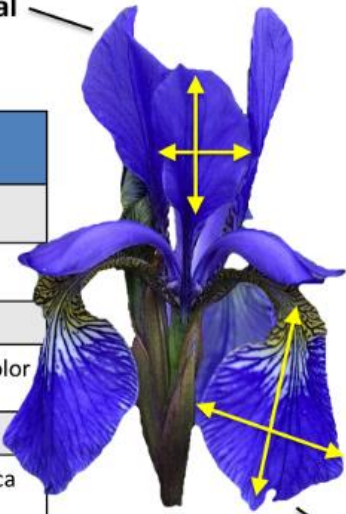
## ■ Iris Dataset (Iris data set)

- 150 instances
- 3 descriptive features
- 3 class labels: setosa, versicolor, virginica

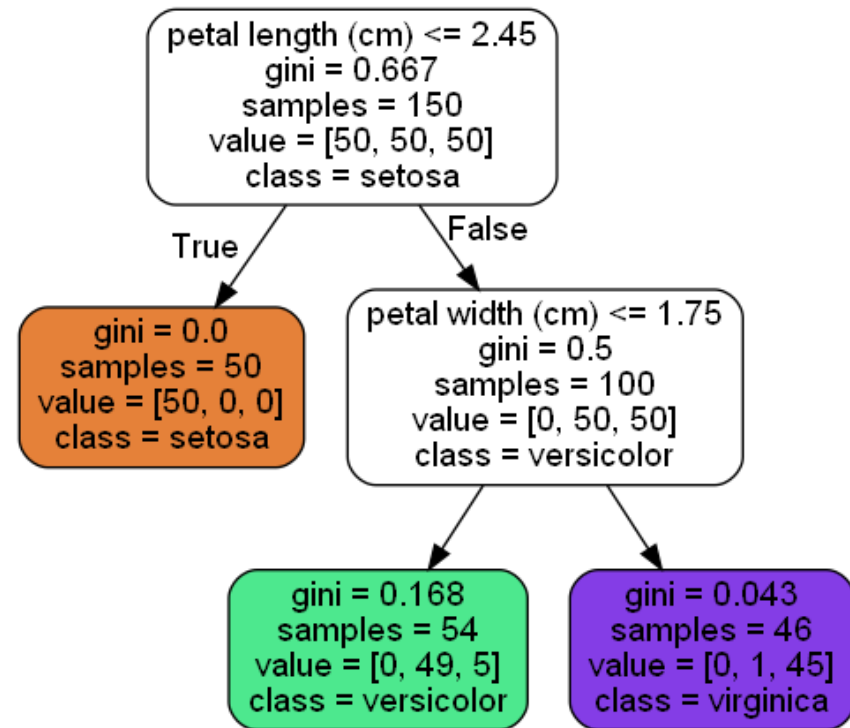
	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

**Features**  
(attributes, measurements, dimensions)

**Class labels**  
(targets)



## ■ A decision tree



# Outline

- Classification Task
- Information Theory and Information-based Learning
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
  - Alternative Feature Selection Metrics
  - ☞ **Handling Continuous Descriptive Features**
  - Predicting Continuous Targets
  - Overfitting and Tree Pruning
- Summary

# Continuous Descriptive Features

- How to determine the test condition of a continuous descriptive feature?
  - The easiest way to handle continuous valued features is to turn them into Boolean features by defining a threshold and using this threshold to partition the instances based their value of the continuous feature.
- How to select the best value for the threshold ?
  - Find a boundary for splitting which produce the greatest information gain

# Example

Sorted Values  
Split Positions

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No			
→  →	Annual Income																					
	60		70		75		85		90		95		100		120		125		220			
	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>		
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0		
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		0.300		0.343		0.375		0.400		0.420	

# Continuous Descriptive Features (Cont.)

- Once a threshold has been set, the dynamically created new binary feature can compete with the other categorical features for selection as the splitting feature at that node.
  - E.g., *Playtennis* = *true* if *Temperature* > *c*
    - e.g., *Temperature* > 72.5

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

- This process can be repeated at each node as the tree grows.

# Example

- Dataset for predicting the vegetation in an area

ID	STREAM	SLOPE	ELEVATION	VEGETATION
1	false	steep	3,900	chapparal
2	true	moderate	300	riparian
3	true	steep	1,500	riparian
4	false	steep	1,200	chapparal
5	false	flat	4,450	conifer
6	true	steep	5,000	conifer
7	true	steep	3,000	chapparal

- Training instances sorted by a continuous ELEVATION feature.

ID	STREAM	SLOPE	ELEVATION	VEGETATION
2	true	moderate	300	riparian
4	false	steep	1,200	chapparal
3	true	steep	1,500	riparian
7	true	steep	3,000	chapparal
1	false	steep	3,900	chapparal
5	false	flat	4,450	conifer
6	true	steep	5,000	conifer

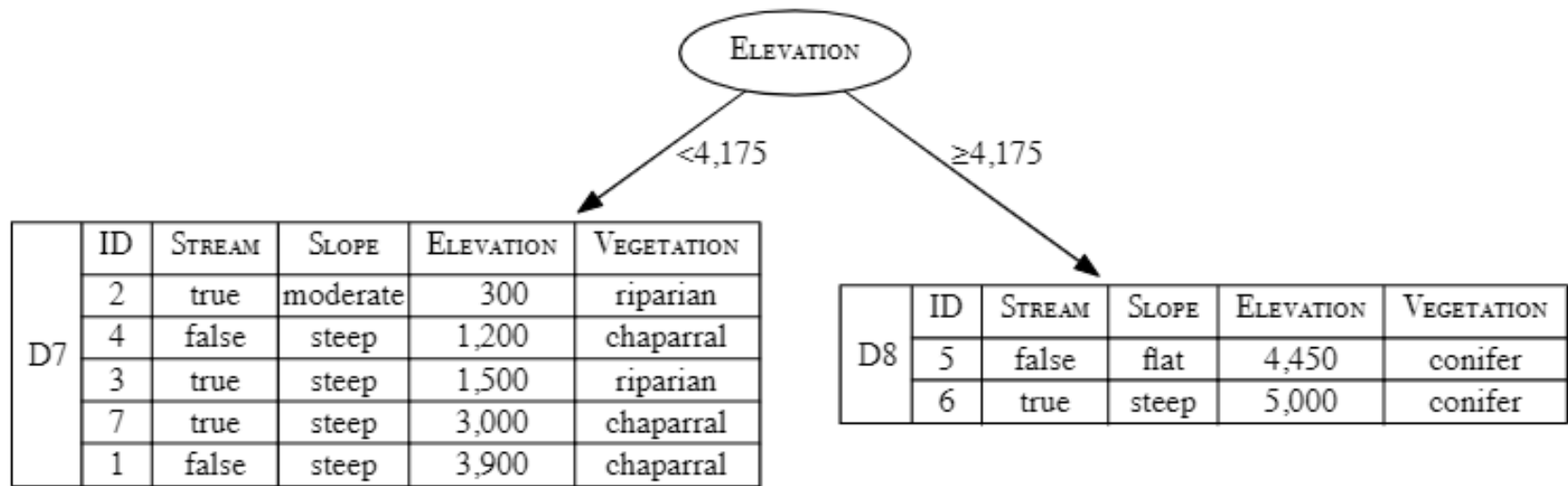
## Example (Cont.)

Split by Threshold	Part.	Instances	Partition Entropy	Rem.	Info. Gain
$\geq 750$	$\mathcal{D}_1$	$\mathbf{d}_2$	0.0	1.2507	0.3060
	$\mathcal{D}_2$	$\mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.4591		
$\geq 1,350$	$\mathcal{D}_3$	$\mathbf{d}_2, \mathbf{d}_4$	1.0	1.3728	0.1839
	$\mathcal{D}_4$	$\mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.5219		
$\geq 2,250$	$\mathcal{D}_5$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3$	0.9183	0.9650	0.5917
	$\mathcal{D}_6$	$\mathbf{d}_7, \mathbf{d}_1, \mathbf{d}_5, \mathbf{d}_6$	1.0		
$\geq 4,175$	$\mathcal{D}_7$	$\mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_3, \mathbf{d}_7, \mathbf{d}_1$	0.9710	0.6935	0.8631
	$\mathcal{D}_8$	$\mathbf{d}_5, \mathbf{d}_6$	0.0		

**Table:** Partition sets (Part.), entropy, remainder (Rem.), and information gain (Info. Gain) for the candidate ELEVATION thresholds:  $\geq 750$ ,  $\geq 1350$ ,  $\geq 2250$ , and  $\geq 4175$

- The boundary between  $\mathbf{d}_2$  and  $\mathbf{d}_4$  is  $\frac{300 + 1,200}{2} = 750$
- The boundary between  $\mathbf{d}_4$  and  $\mathbf{d}_3$  is  $\frac{1,200 + 1,500}{2} = 1,350$
- The boundary between  $\mathbf{d}_3$  and  $\mathbf{d}_7$  is  $\frac{1,500 + 3,000}{2} = 2,250$
- The boundary between  $\mathbf{d}_1$  and  $\mathbf{d}_5$  is  $\frac{3,900 + 4,450}{2} = 4,175$

## Example (Cont.)



**Figure:** The vegetation classification decision tree after the dataset has been split using  $\text{ELEVATION} \geq 4175$



# Outline

- Classification Task
- Information Theory and Information-based Learning
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
  - Alternative Feature Selection Metrics
  - Handling Continuous Descriptive Features
  - 👉 **Predicting Continuous Targets**
  - Overfitting and Tree Pruning
- Summary

# Predicting Continuous Targets

- Decision Trees can also perform **regression tasks** (for predicting numeric target values)
- A *regression tree* is constructed in almost the same manner as a classification tree, except the impurity measure is replaced by a measure appropriate for regression
- Regression trees are constructed so as to reduce the *variance* (or *mean square error*) in the set of training examples at each of the leaf nodes in the tree
  - We can do this by adapting the ID3 algorithm to use a measure of variance rather than a measure of classification impurity (entropy) when selecting the best feature

# Predicting Continuous Targets

- The impurity (**variance**) at a node can be calculated using:

$$var(t, D) = \frac{\sum_{i=1}^n (t_i - \bar{t})^2}{n-1}$$

where  $D$  is the dataset that has reached the node;  $n$  is the number of instances in  $D$ ;  $\bar{t}$  is the mean of the target feature for the dataset  $D$ ;  $t_i$  is the target value of each instance in  $D$ .

- We select the feature to split on at a node by selecting the feature that minimizes the weighted variance across the resulting partitions:

$$d[best] = \arg \min_{d \in \mathbf{d}} \sum_{l \in levels(d)} \frac{|D_{d=l}|}{|D|} \times var(t, D_{d=l})$$

# Example

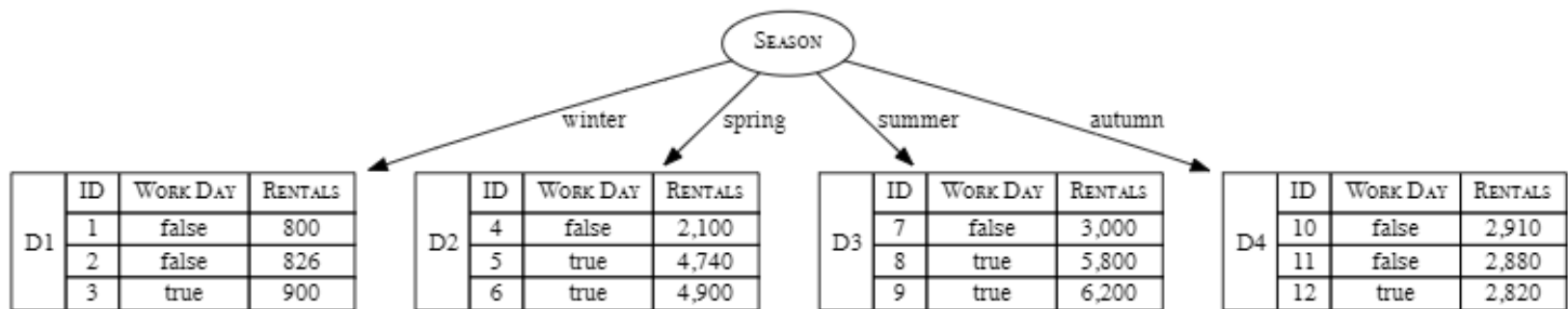
ID	SEASON	WORK	
		DAY	RENTALS
1	winter	false	800
2	winter	false	826
3	winter	true	900
4	spring	false	2,100
5	spring	true	4,740
6	spring	true	4,900
7	summer	false	3,000
8	summer	true	5,800
9	summer	true	6,200
10	autumn	false	2,910
11	autumn	false	2,880
12	autumn	true	2,820

**Table:** A dataset listing the number of bikes rental per pay.

# Example (Cont.)

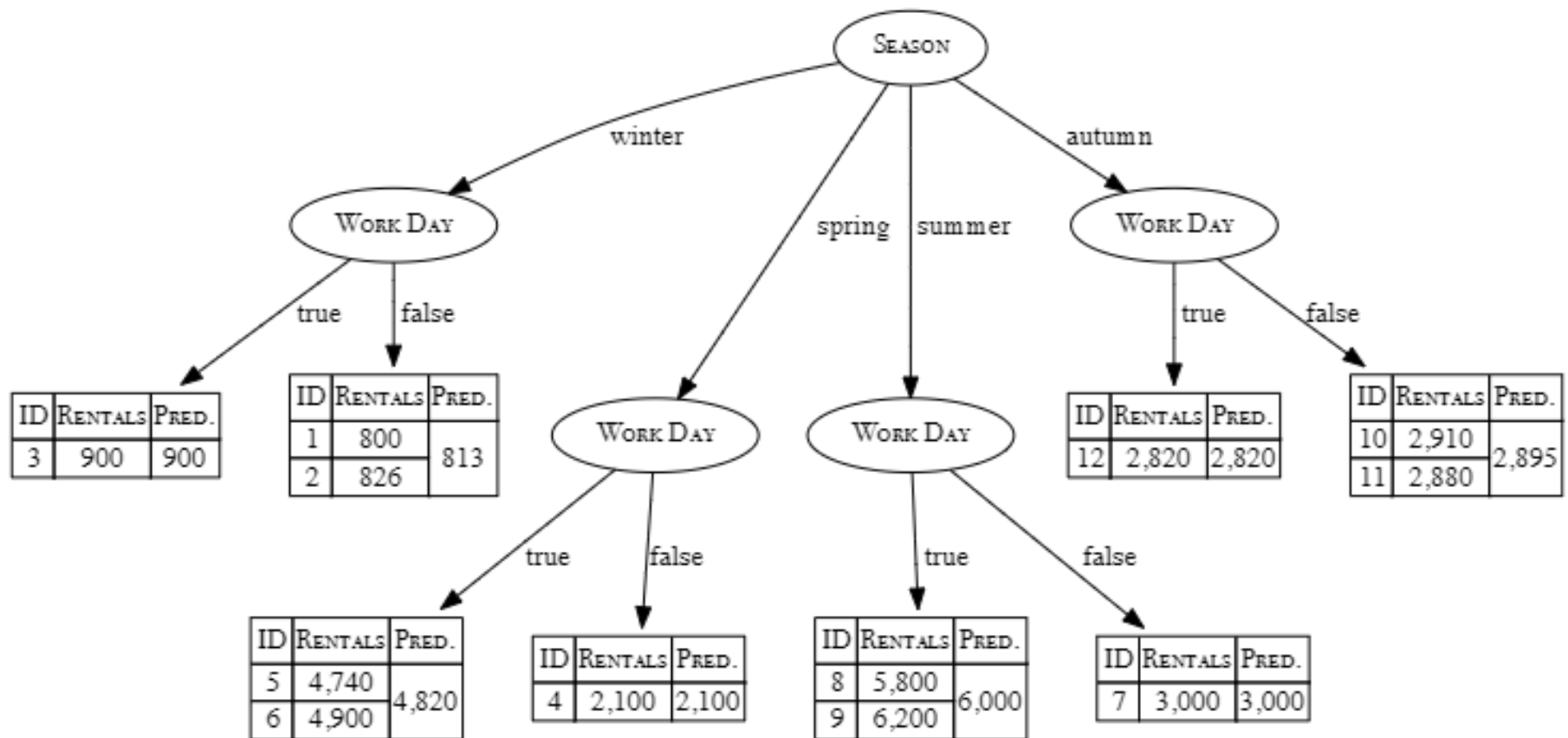
**Table:** The partitioning of the dataset based on SEASON and WORK DAY features and the computation of the weighted variance for each partition

Split by Feature	Level	Part.	Instances	$\frac{ \mathcal{D}_{d=l} }{ \mathcal{D} }$	$var(t, \mathcal{D})$	Weighted Variance
SEASON	winter	$\mathcal{D}_1$	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3$	0.25	2,692	1,379,331
	spring	$\mathcal{D}_2$	$\mathbf{d}_4, \mathbf{d}_5, \mathbf{d}_6$	0.25	2,472,533	
	summer	$\mathcal{D}_3$	$\mathbf{d}_7, \mathbf{d}_8, \mathbf{d}_9$	0.25	3,040,000	
	autumn	$\mathcal{D}_4$	$\mathbf{d}_{10}, \mathbf{d}_{11}, \mathbf{d}_{12}$	0.25	2,100	
WORK DAY	true	$\mathcal{D}_5$	$\mathbf{d}_3, \mathbf{d}_5, \mathbf{d}_6, \mathbf{d}_8, \mathbf{d}_9, \mathbf{d}_{12}$	0.50	4,026,346	2,551,813
	false	$\mathcal{D}_6$	$\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_4, \mathbf{d}_7, \mathbf{d}_{10}, \mathbf{d}_{11}$	0.50	1,077,280	



**Figure:** The decision tree resulting from splitting the data using the feature SEASON

# Example (Cont.)



**Figure:** The final **regression tree** induced from the dataset. To illustrate how the tree generates predictions, this tree lists the instances that ended up at each leaf node and the prediction (PRED. – **mean of target feature values**) made by each leaf node

# Outline

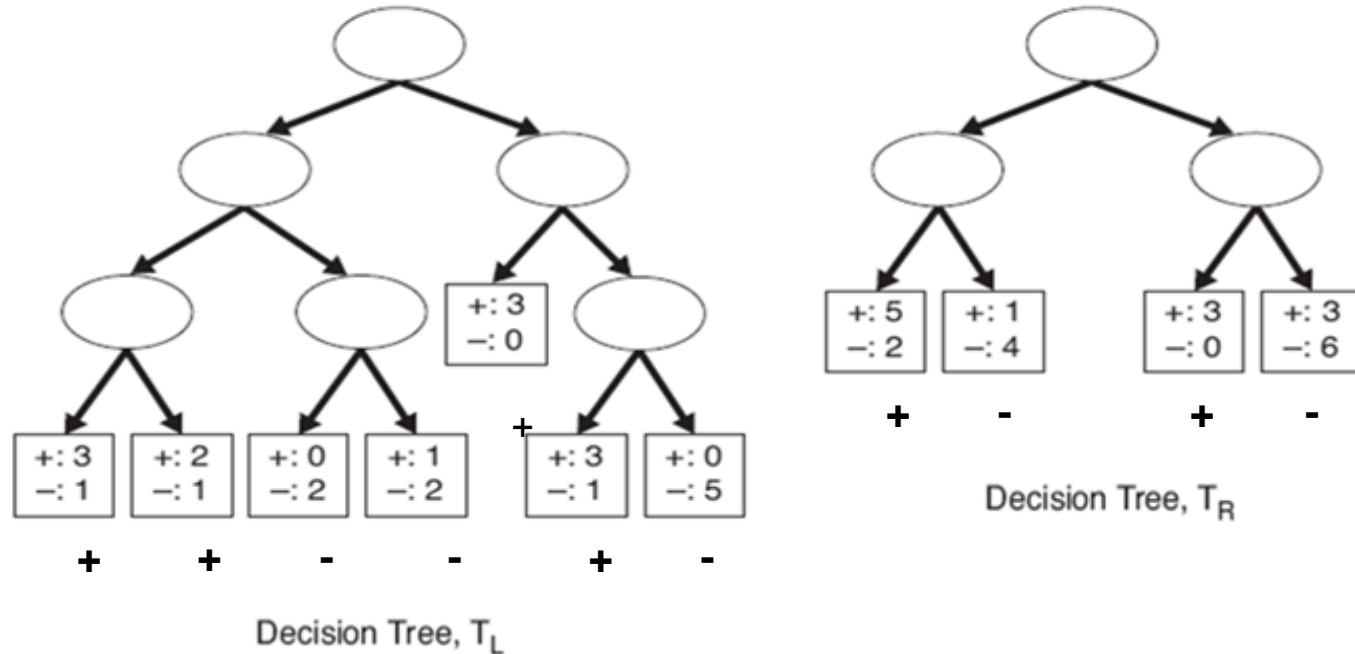
- Classification Task
- Information Theory and Information-based Learning
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
  - Alternative Feature Selection Metrics
  - Handling Continuous Descriptive Features
  - Predicting Continuous Targets
  - ☞ **Overfitting and Tree Pruning**
- Summary

# Type of Errors

- ***Training errors*** (*apparent / resubstitution error*)
  - Errors committed on the training set
- ***Test errors***
  - Errors committed on the test set
- ***Generalization errors***
  - Expected error of a model over random selection of records from same distribution



# Example: Training Error (Resubstitution Error)

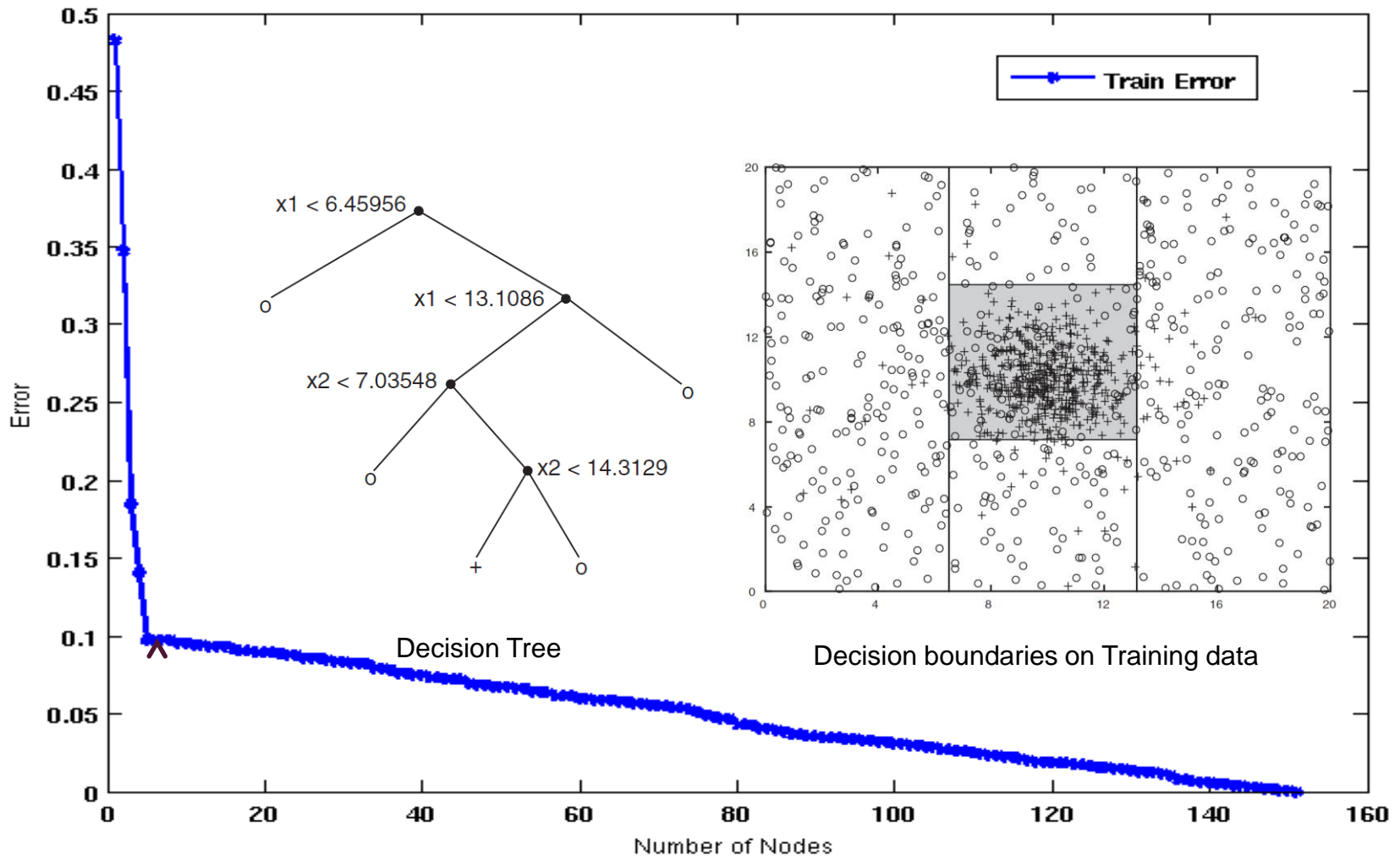


( \* Classification decision at each leaf node according to the majority class )

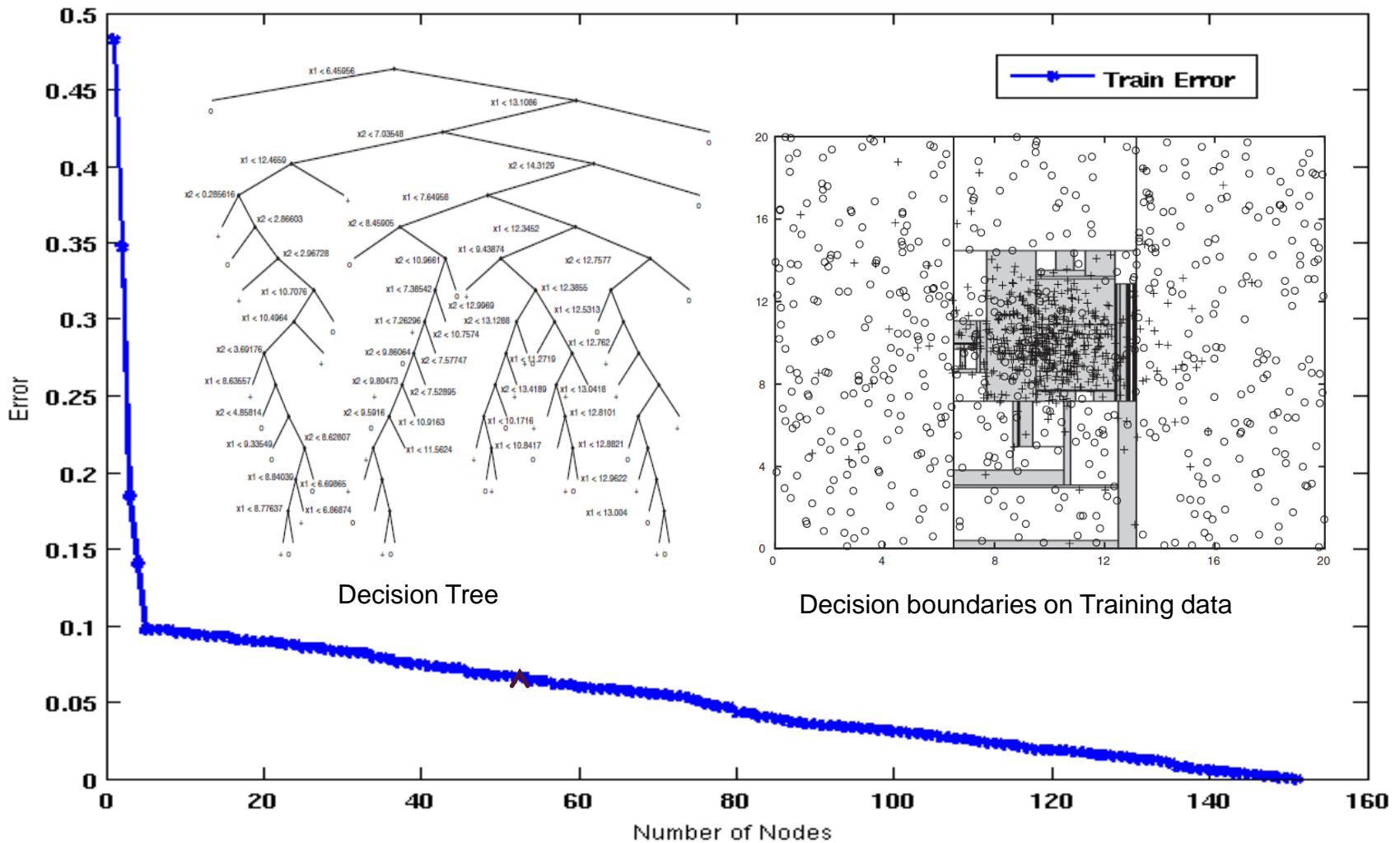
$$\begin{aligned} error(T_L) &= 4 \text{ errors} / 24 \text{ examples} \\ &= 0.168 \end{aligned}$$

$$\begin{aligned} error(T_R) &= 6 \text{ errors} / 24 \text{ examples} \\ &= 0.25 \end{aligned}$$

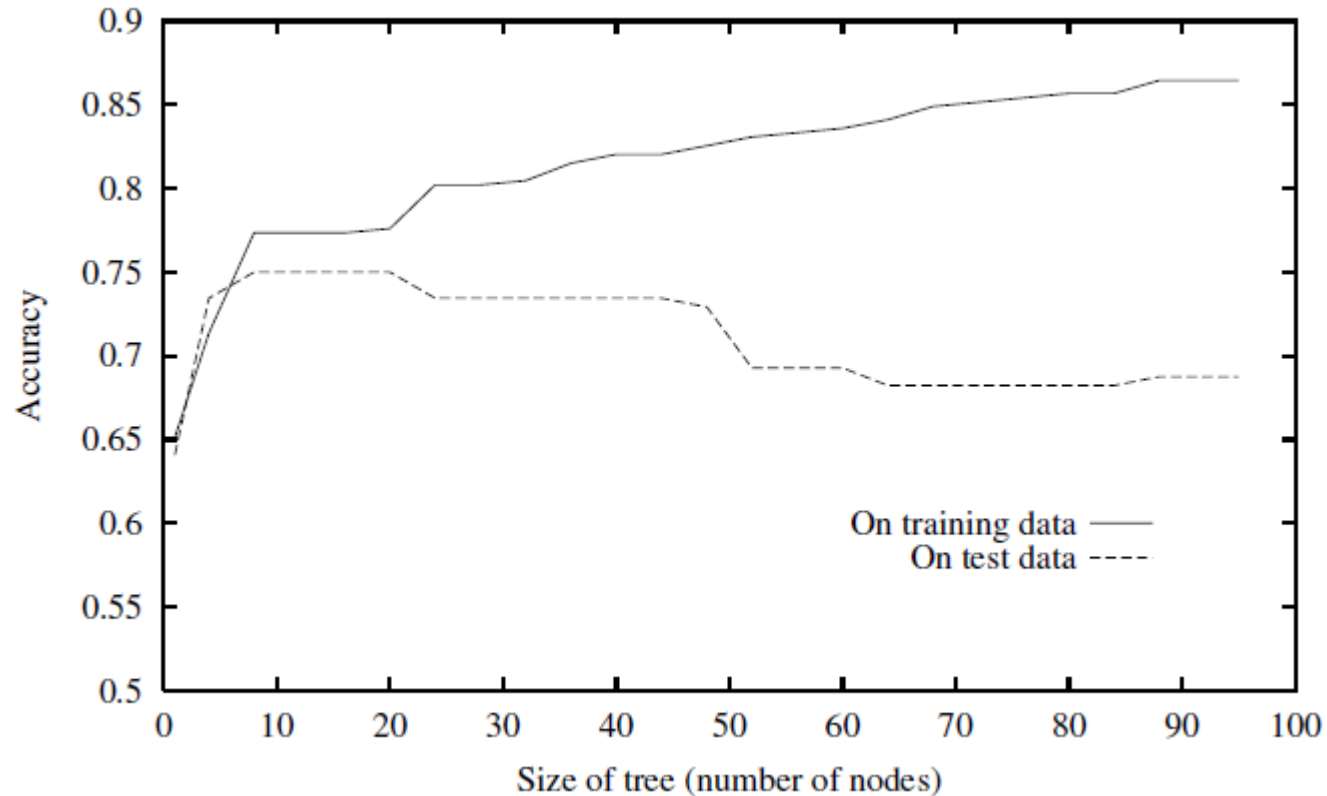
# Decision Tree with 5 nodes and the Training Error



# Decision Tree with 50 nodes and the Training Error



# Training Error and Test Error



Predictably, the accuracy of the tree over the training examples increases monotonically as the tree is grown.

However, the accuracy measured over the independent test examples first increases, then decreases

# When Overfitting Happens

- One of practical issues in learning decision trees is determining how deeply to grow the decision tree to perfectly classify the training examples
- But it is difficult when
  - there is noise or random errors in data
  - the number of training examples is too small to produce a representative sample of the true target function
- This case can produce trees that *overfit* the training examples.

# Avoiding Overfitting

- An experimental study of ID3 (Mingers 1989) shows that overfitting was found to decrease the accuracy of learned decision tree by 10 - 25% on most problem
- How can we avoid overfitting?
  - Stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data.
    - when data split not statistically significant
  - Grow full tree, then post-prune (more successful in practice)

# Pre-Pruning

- ***Pre-Pruning*** (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
    - We can bias our tree or place limits on them to prevent overfitting
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - **More restrictive conditions for early stopping:**
    - Stop if number of instances is less than some user-specified threshold
    - Stop if expanding the current node does not improve impurity measures
    - Stop if estimated generalization error falls below certain threshold

# Post-Pruning

## ■ *Post-Pruning*

- First, grow decision tree to its entirety
- Then the tree trimming can be done by replacing a subtree with
  - (1) a new leaf node whose class label is determined from the majority class of instances affiliated with the subtree (known as **subtree replacement**), or
  - (2) the most frequently used branch of the subtree (known as **subtree raising**)
- The tree-pruning step terminates no further improvement in the generalization error estimate is observed beyond a certain threshold.



# Examples of Post-pruning

## Decision Tree:

```
depth = 1 :  
| breadth > 7 : class 1  
| breadth <= 7 :  
| | breadth <= 3 :  
| | | ImagePages > 0.375 : class 0  
| | | ImagePages <= 0.375 :  
| | | | totalPages <= 6 : class 1  
| | | | totalPages > 6 :  
| | | | | breadth <= 1 : class 1  
| | | | | breadth > 1 : class 0  
| | width > 3 :  
| | | MultiIP = 0:  
| | | | ImagePages <= 0.1333 : class 1  
| | | | ImagePages > 0.1333 :  
| | | | | breadth <= 6 : class 0  
| | | | | breadth > 6 : class 1  
| | | MultiIP = 1:  
| | | | TotalTime <= 361 : class 0  
| | | | TotalTime > 361 : class 1  
| depth > 1 :  
| | MultiAgent = 0:  
| | | depth > 2 : class 0  
| | | depth <= 2 :  
| | | | MultiIP = 1 : class 0  
| | | | MultiIP = 0:  
| | | | | breadth <= 6 : class 0  
| | | | | breadth > 6 :  
| | | | | | RepeatedAccess <= 0.0322 : class 0  
| | | | | | RepeatedAccess > 0.0322 : class 1  
| | MultiAgent = 1:  
| | | totalPages <= 81 : class 0  
| | | totalPages > 81 : class 1
```

The subtree rooted at depth=1 has been replaced by one of its branches corresponding to  $\text{breadth} \leq 7$ ,  $\text{width} > 3$  and  $\text{MultiIP}=1$

## Simplified Decision Tree:

```
depth = 1 :  
| ImagePages <= 0.1333 : class 1  
| ImagePages > 0.1333 :  
| | breadth <= 6 : class 0  
| | breadth > 6 : class 1  
depth > 1 :  
| MultiAgent = 0: class 0  
| MultiAgent = 1:  
| | totalPages <= 81 : class 0  
| | totalPages > 81 : class 1
```

Subtree  
Raising

Subtree  
Replacement

The subtree corresponding to  $\text{depth} > 1$  and  $\text{MultiAgent} = 0$  has been replaced by a leaf node assigned to class 0, using subtree replacement.

# Outline

- Classification Task
- Information Theory and Information-based Learning
- Decision Tree Learning
- Fundamentals
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
- ➡ **Summary**

# Summary

- Classification Task
- Information Theory and Information-based Learning
- Fundamentals
  - Decision Trees
  - Shannon's Entropy Model
  - Information Gain
- Standard Approach: The ID3 Algorithm
- Extensions and Variations
  - Alternative Feature Selection Metrics
  - Handling Continuous Descriptive Features
  - Predicting Continuous Targets
  - Overfitting and Tree Pruning