CLASSIFICATION

— DECISION TREE

DR. XIAOHUI YUAN

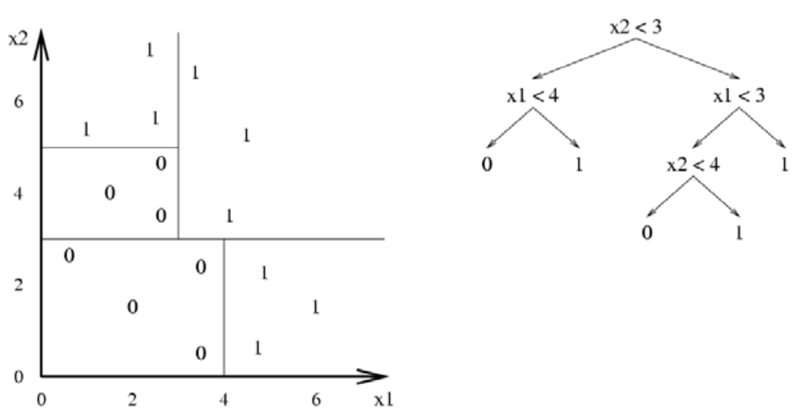
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIVERSITY OF NORTH TEXAS

Decision Trees

- Decision tree
  - **Internal node** denotes a test on a feature
  - **Branch** represents the path of a test outcome
  - **Leaf nodes** represent the class labels or class distribution
  - At each node, one attribute is chosen to split the training examples into *distinct classes as much as possible*
  - A new case is classified by following a matching path to a leaf node.
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers

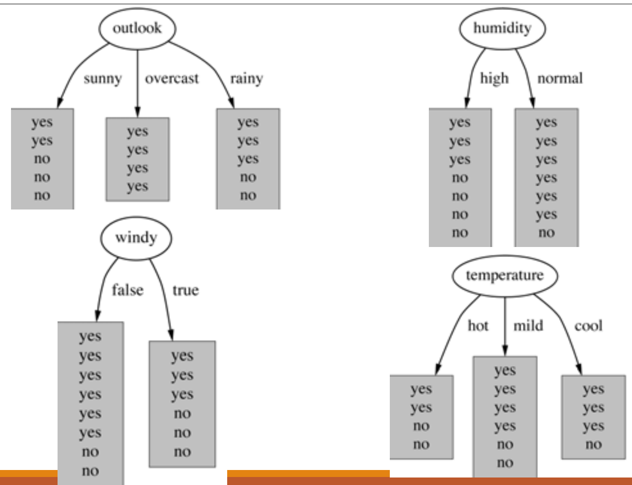
Graphical Representation



How to Construct a Tree?

- Greedy algorithm
  - Tree is constructed in a top-down, recursive manner
  - Examples are partitioned recursively based on the selected attributes.
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning
- Majority voting is employed for classifying in leaves

Which Attribute to Select?



Choosing the Splitting Attributes

- **The goal is to have the resulted decision tree as small as possible** (to avoid overfitting)
- The main decision in the algorithm is the selection of the next attribute to make condition on.
- At each node, the available attributes are evaluated on the basis of separating the training examples.
- The objective function (a.k.a. Goodness function) include
  - Information gain (ID3/C4.5)
  - Information gain ratio
  - Gini index (CART)

Information Gain

- Select the attribute with the **greatest information gain** (information gain is the expected reduction in entropy).
- Assume that there are two classes, *P* and *N*
  - Let the set of examples *S* contain *p* elements of class *P* and *n* elements of class *N*. The amount of information needed to decide if an arbitrary example in *S* belongs to *P* or *N* is defined as

$$E(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$ and $0 \log(0) \equiv 0$

- With an attribute A, data are partitioned into subsets. The information gained by branching on *A* is as follows

$$Gain(A) = E(Current\ set) - \sum P(childset)E(childset)$$

Person	Hair Length	Weight	Age	Class
Homer	0”	250	36	M
Marge	10”	150	34	F
Bart	2”	90	10	M
Lisa	6”	78	8	F
Maggie	4”	20	1	F
Abe	1”	170	70	M
Selma	8”	160	41	F
Otto	10”	180	38	M
Krusty	6”	200	45	M

Learn from

Predict for

Comic	8”	290	38	?

$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

Entropy(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= 0.9911

yes — Hair Length <= 5? — no

Let us try splitting on *Hair length*

Entropy(1**F**,3**M**) = -(1/4)$\log_2$(1/4) - (3/4)$\log_2$(3/4)
= **0.8113**

Entropy(3**F**,2**M**) = -(3/5)$\log_2$(3/5) - (2/5)$\log_2$(2/5)
= **0.9710**

$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

Gain(Hair Length <= 5) = 0.9911 – (4/9 * **0.8113** + 5/9 * **0.9710**) = **0.0911**

$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

Entropy(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= 0.9911

yes — Weight <= 160? — no

Let us try splitting on *Weight*

Entropy(4**F**,1**M**) = -(4/5)$\log_2$(4/5) - (1/5)$\log_2$(1/5)
= **0.7219**

Entropy(0**F**,4**M**) = -(0/4)$\log_2$(0/4) - (4/4)$\log_2$(4/4)
= ***0***

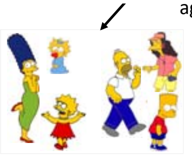
$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

Gain(Weight <= 160) = 0.9911 – (5/9 * **0.7219** + 4/9 * **0**) = **0.5900**

$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

Entropy(4**F**,5**M**) = -(4/9)$\log_2$(4/9) - (5/9)$\log_2$(5/9)
= 0.9911

yes — age <= 40? — no



Let us try splitting on *Age*

Entropy(3**F**,3**M**) = -(3/6)$\log_2$(3/6) - (3/6)$\log_2$(3/6)
= **1**

Entropy(1**F**,2**M**) = -(1/3)$\log_2$(1/3) - (2/3)$\log_2$(2/3)
= **0.9183**

$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

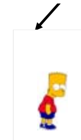
Gain(Age <= 40) = 0.9911 – (6/9 * **1** + 3/9 * **0.9183**) = **0.0183**

Of the 3 features we had, *Weight* was the best. But while people who weigh over 160 are perfectly classified (as males), the under 160 people are not classified correctly …

This time we find that we can split on *Hair length,* and we are done!

yes — Weight <= 160? — no

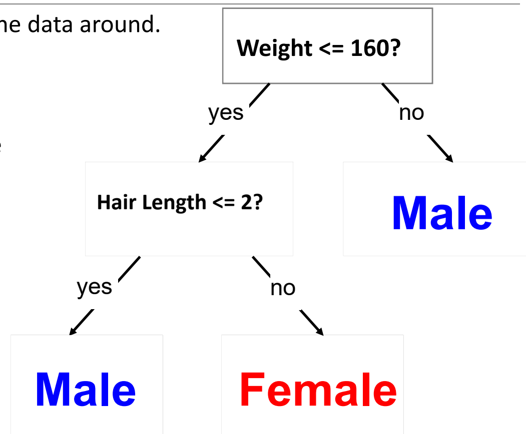
yes — Hair Length <= 2? — no



Applying Decision Trees

We don't need to keep the data around.
Just the test conditions.

How would these people be classified?



Gini Index (used in CART)

- If a data set T contains examples from n classes, Gini index, gini(T), is defined as

$$gini(T) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in T.

- gini(T) is minimized if the classes in T are skewed.
- After splitting T into two subsets T_1 and T_2 with sizes N_1 and N_2, the Gini index of the subsets is

$$gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

- The attribute providing smallest $gini_{split}(T)$ is chosen to split the node.

Decision Tree Learning

Aim: find a small tree consistent with the training examples
Idea: (**recursively**) choose "the most significant" attribute as the **root** of the (**sub**)tree

```
function DTL(examples, attributes, default) returns a decision tree
   if examples is empty then return default
   else if all examples have the same classification then return the classification
   else if attributes is empty then return MODE(examples)
   else
       best ← CHOOSE-ATTRIBUTE(attributes, examples)
       tree ← a new decision tree with root test best
       for each value v_i of best do
           examples_i ← {elements of examples with best = v_i}
           subtree ← DTL(examples_i, attributes − best, MODE(examples))
           add a branch to tree with label v_i and subtree subtree
       return tree
```