

## Outline

- Introduction
- Bayesian Probability Theory
- Sequence Prediction and Data Compression
- **Decision Trees and Ensemble Learning**
  - The ID3 Algorithm
  - The AdaBoost Algorithm
- Bayesian Networks

## Decision Tree Induction

- Widely used, deployed in hundreds of real systems worldwide.
- The best known system C4.5 was developed in Australia (by Prof. Ross Quinlan, formerly of University of Sydney).
- Advantages of using decision trees
  - Decision trees are comprehensible.
  - Simple learning algorithms exists.
  - Decision trees + ensemble learning is one of the best off-the-shelf learning systems out there.

## Learning Decision Trees

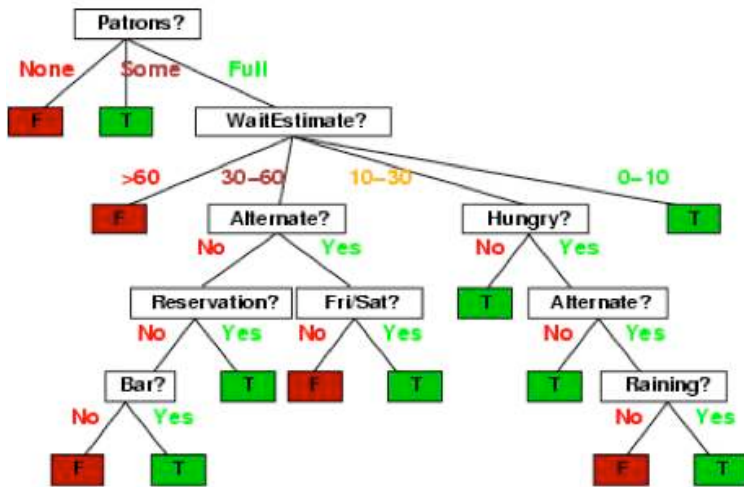
Problem: predict whether a friend will wait for a table at a restaurant, based on the following attributes:

- Alternate: is there an alternative restaurant nearby?
- Bar: is there a comfortable bar area to wait in?
- Fri/Sat: is today Friday or Saturday?
- Hungry: are we hungry?
- Patrons: number of people in the restaurant (None, Some, Full)
- Price: price range (\$, \$\$, \$\$\$)
- Raining: is it raining outside?
- Reservation: have we made a reservation?
- Type: kind of restaurant (French, Italian, Thai, Burger)
- WaitEstimate: estimated waiting time (0-10, 10-30, 30-60, >60)

## Training Data

Example	Attributes										Target
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
X <sub>1</sub>	T	F	F	T	Some	\$\$\$	F	T	French	0-10	T
X <sub>2</sub>	T	F	F	T	Full	\$	F	F	Thai	30-60	F
X <sub>3</sub>	F	T	F	F	Some	\$	F	F	Burger	0-10	T
X <sub>4</sub>	T	F	T	T	Full	\$	F	F	Thai	10-30	T
X <sub>5</sub>	T	F	T	F	Full	\$\$\$	F	T	French	>60	F
X <sub>6</sub>	F	T	F	T	Some	\$\$	T	T	Italian	0-10	T
X <sub>7</sub>	F	T	F	F	None	\$	T	F	Burger	0-10	F
X <sub>8</sub>	F	F	F	T	Some	\$\$	T	T	Thai	0-10	T
X <sub>9</sub>	F	T	T	F	Full	\$	T	F	Burger	>60	F
X <sub>10</sub>	T	T	T	T	Full	\$\$\$	F	T	Italian	10-30	F
X <sub>11</sub>	F	F	F	F	None	\$	F	F	Thai	0-10	F
X <sub>12</sub>	T	T	T	T	Full	\$	F	F	Burger	30-60	T

## True Decision Tree

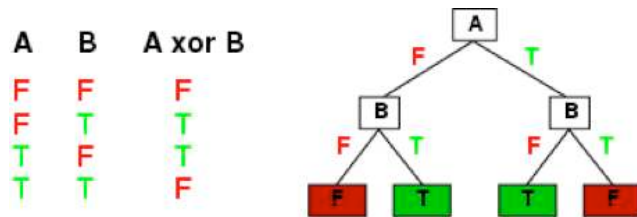


## Decision Trees

- We need to find a good decision tree from the class of all decision trees defined on the given set of attributes.
- Before looking at algorithms, let's look at the expressiveness of decision trees.
- Fact: Decision trees can express any function of the input attributes.

## Expressiveness

- Fact: Decision trees can express any function of the input attributes.
- For example:



- This means there are potentially many trees that fit the data. Occam's razor: prefer small trees.

## Size of Hypothesis Space

- Number of distinct decision trees with  $n$  boolean attributes  
 = number of boolean functions on  $n$  boolean inputs  
 = ?

## Size of Hypothesis Space

- Number of distinct decision trees with  $n$  boolean attributes  
= number of boolean functions on  $n$  boolean inputs  
= number of distinct truth tables with  $2^n$  rows =  $2^{2^n}$ .
- E.g. with 6 boolean attributes, there are 18,446,744,073,709,551,616 distinct functions.
- There are many more distinct decision trees.

## Algorithmic Considerations

- We want to learn from data using the class of decision trees.
- The Bayesian optimal classifier involves a weighted sum over all possible decision trees. (Intractable)
- Finding the smallest decision tree fitting the data (the MAP classifier) also turns out to be intractable.
- Settle for a greedy approximation algorithm.

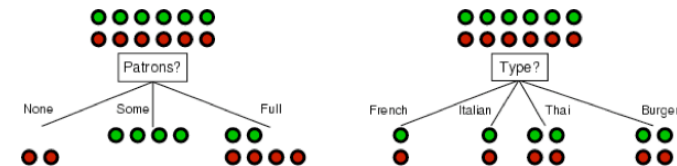
## Decision Tree Algorithm

Idea: Recursively choose “most significant” attribute to split training examples.

```
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
      examplesi ← {elements of examples with best =  $v_i$ }
      subtree ← DTL(examplesi, attributes - best, MODE(examples))
      add a branch to tree with label  $v_i$  and subtree subtree
    return tree
```

## How to Choose an Attribute

Idea: a good attribute splits the examples into subsets that are (ideally) “all positive” or “all negative”.



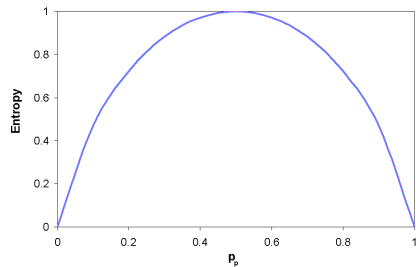
Clearly, the test Patrons? is a better choice.

## The Entropy of a Distribution

- A (discrete) probability distribution  $d = (v_1, p_1), (v_2, p_2), \dots, (v_n, p_n)$  has entropy

$$I(d) = \sum_{i=1}^n -p_i \log_2 p_i.$$

Bernoulli distributions



## Minimising Entropy

- Low entropy is good.
- Let  $\delta(E)$  the empirical distribution of set of examples  $E$ .
- Entropy of a node with examples  $E$  is  $I(\delta(E))$ .
- A chosen attribute  $A$  divides  $E$  into subsets  $E_1, \dots, E_v$  according to their values for  $A$ . The (expected) entropy after the split is

$$I_s(A, E) = \sum_{i=1}^v \frac{|E_i|}{|E|} I(\delta(E_i)).$$

- The quantity *information gain* is defined by

$$IG(A) = I(\delta(E)) - I_s(A, E).$$

- CHOOSE-ATTRIBUTE is defined to pick the attribute with the highest information gain.

## Minimising Entropy

- Sanity check

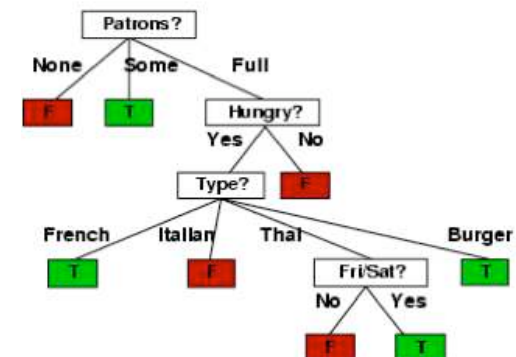
$$IG(\text{Patrons?}) = 0.0541$$

$$IG(\text{Type?}) = 0$$

- Patrons has the highest IG of all attributes and so is chosen by the DTL algorithm as the root

## Example

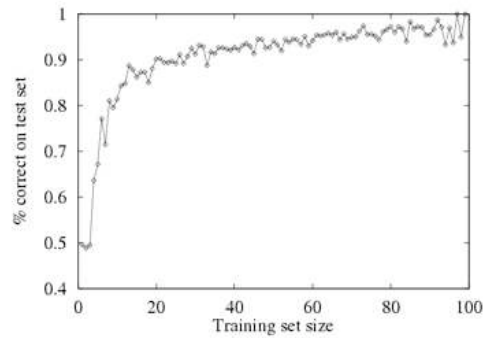
- Decision tree learned from the 12 examples



- Substantially simpler than “true” tree – a more complex hypothesis isn’t justified by small amount of data.

## Performance Measurement

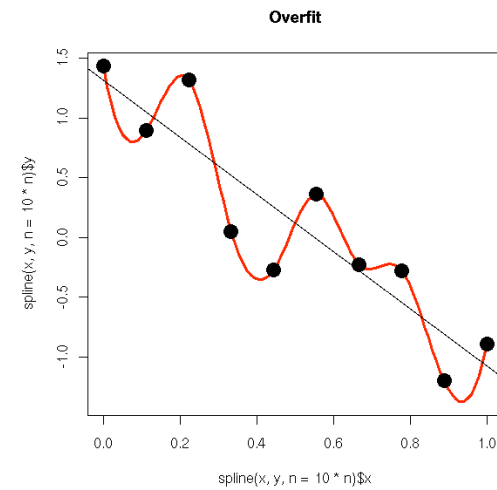
- How do we know our learned function is any good?
- Try it on new test examples.
- Learning curve = % correct on test set as training set size increases



MP3620 (2009)

16

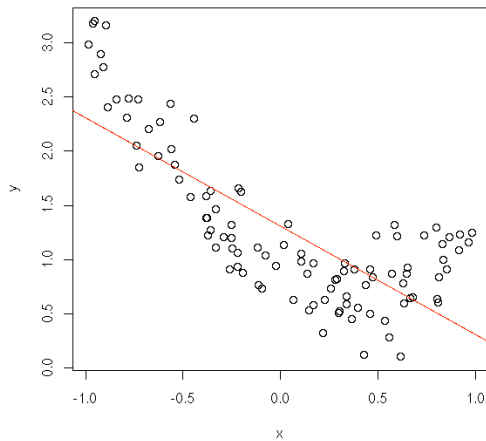
## Problems in Learning: Overfitting



COMP3620 (2009)

17

## Problems in Learning: Underfitting



MP3620 (2009)

18

## N-fold Cross Validation

- Divide training data into  $N$  equal size groups.
- Repeat from  $i = 1$  to  $N$ 
  - Learn on all training data except those in the  $i$ -th group
  - Evaluate learned hypothesis on  $i$ -th group

COMP3620 (2009)

19

## Ensemble Learning

- The idea of ensemble learning methods is to select a collection of hypotheses (instead of one) and combine their predictions.
- A motivating algorithm: construct  $M = 5$  hypotheses from data and then use majority voting to make predictions.
- For the ensemble to misclassify a new example, *at least three hypotheses have to misclassify the example*.
- Under certain conditions, this is much less probable than a misclassification by each single hypothesis.

## AdaBoost

Need a base learning algorithm that

- can handle weighted training examples

$$(x_1, y_1, w_1), \dots, (x_n, y_n, w_n)$$

- is a weak classifier in the sense that it always return a hypothesis predicts slightly better than random guessing.

## AdaBoost

**Algorithm 1:** Adaboost Algorithm (Freund and Schapire)

**Input** : A weak learning algorithm *WeakLearn*, an integer  $T$  specifying number of iterations, and  $N$  labelled training data  $\{(x_1, y_1), \dots, (x_N, y_N)\}$ .

**Output** : A strong classifier  $F$ .

Initialize the weight vector  $w_i^1 = \frac{1}{N}$ , for  $i = 1, \dots, N$ .

**for**  $t \leftarrow 1, 2, \dots, T$  **do**

1.  $\mathbf{p}^t \leftarrow \mathbf{w}^t / \sum_{i=1}^N w_i^t$ .

2. Call *WeakLearn*, providing it with the distribution on  $\mathbf{p}^t$ ; get back a weak learner  $h_t : X \rightarrow \pm 1$ .

3. Calculate the weight error of  $h_t$ :  $\epsilon_t = \sum_{i=1}^N p_i^t \frac{1}{2} |h_t(x_i) - y_i|$ .

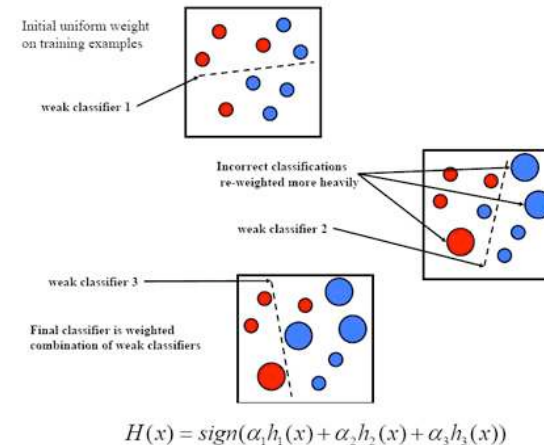
4.  $\alpha_t \leftarrow \log \left( \frac{1-\epsilon_t}{\epsilon_t} \right)$ .

5.  $w_i^{t+1} \leftarrow w_i^t \exp(\alpha_t \frac{1}{2} |h_t(x_i) - y_i|)$ , for  $i = 1, 2, \dots, T$ .

Output the final strong classifier:

$$F(x) = \begin{cases} 1, & \text{if } \sum_{i=1}^T \alpha_i h_i(x) \geq 0, \\ -1, & \text{otherwise.} \end{cases}$$

## AdaBoost Illustration



## AdaBoost Analysis

Theorem: If the base learner is a weak classifier, then AdaBoost will return a hypothesis that classifies the training data perfectly for large enough  $T$ .

Interestingly, in the case of AdaBoost, good performance on training data translates into good performance on test data as well, despite the fact that the complexity of the AdaBoost hypothesis becomes quite high!