

Content

- Introduction to learning
- Learning agents
- Inductive learning
- Typologies of learning
- Learning decision trees



Learning

Learning is essential for unknown environments,
i.e., when designer lacks omniscience

Learning is useful as a system construction method,
i.e., expose the agent to reality rather than trying to write it down

Learning modifies the agent's decision mechanisms to improve performance



Machine Learning

Improving performance after making observation on the world.
Why machine learning ? Three main applications (in AI):

Data mining

- Decisions using historical data to improve decisions: e.g. medical records -> medical knowledge
- Current challenge dimension of data (big) from Web, sensors (IoT), Biology ...

Software applications we can't program by hand

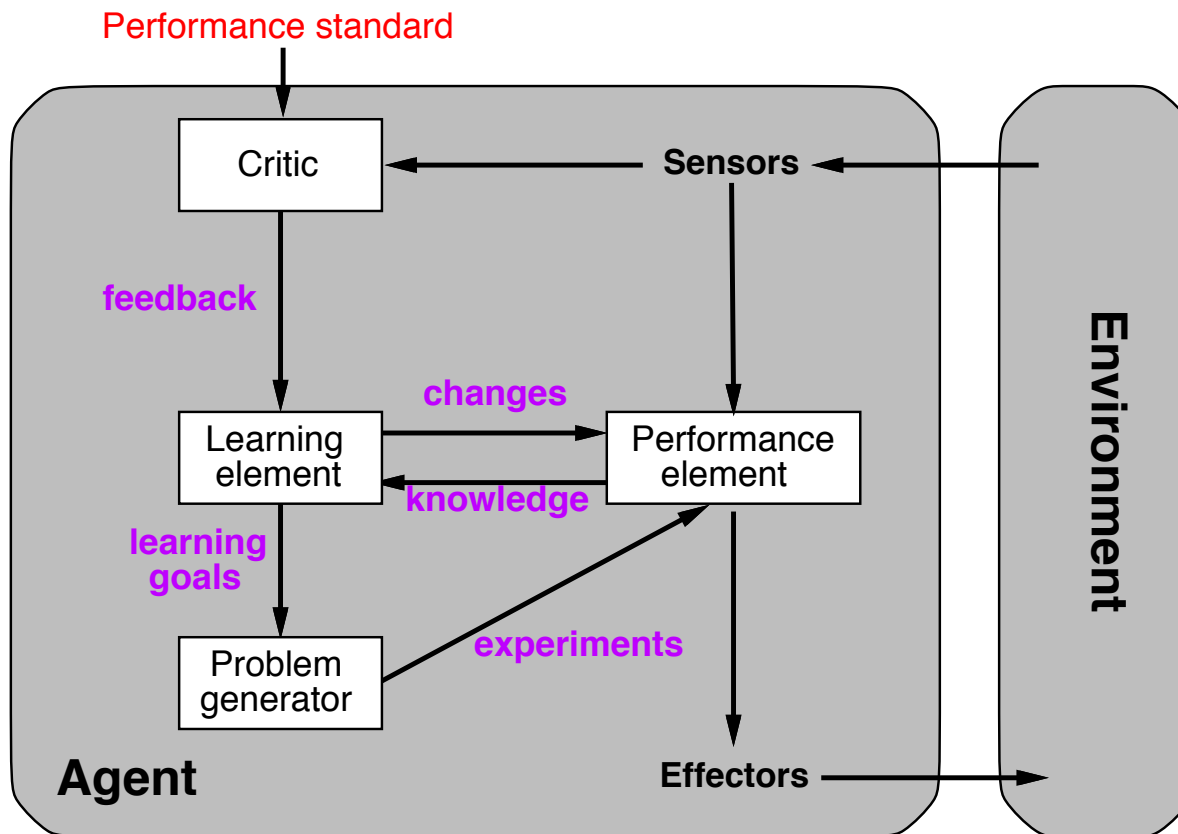
- Autonomous driving, speech recognition, NLP, Artificial vision, FinTech applications ...

Self customizing programs

- Learn user behaviours to provide better services: Newsreader Netflix, Amazon ...



Learning agents



Learning element

Design of learning element is dictated by

- ◇ what type of performance element is used
- ◇ which functional component is to be learned
- ◇ how that functional component is represented
- ◇ what kind of feedback is available

Example scenarios:

Performance element	Component	Representation	Feedback
Alpha–beta search	Eval. fn.	Weighted linear function	Win/loss
Logical agent	Transition model	Successor–state axioms	Outcome
Utility–based agent	Transition model	Dynamic Bayes net	Outcome
Simple reflex agent	Percept–action fn	Neural net	Correct action

Supervised learning: correct answers for each instance

Reinforcement learning: occasional rewards



Inductive learning (a.k.a. Science)

Simplest form: learn a function from examples (**tabula rasa**)

f is the target function

An example is a pair $x, f(x)$, e.g.,

O	O	X
	X	
X		

, $+1$

Problem: find a(n) hypothesis h
such that $h \approx f$
given a training set of examples

(This is a highly simplified model of real learning:

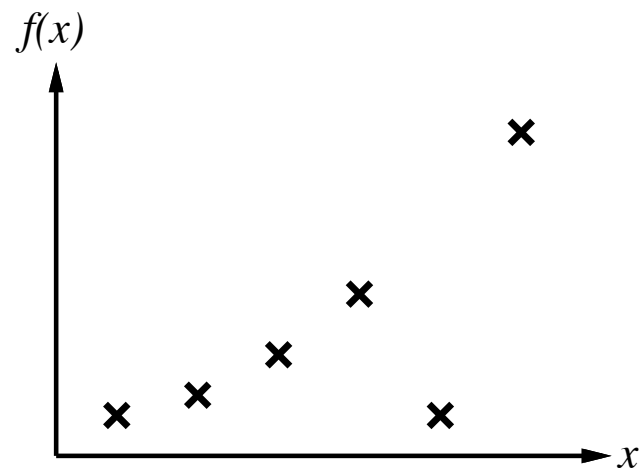
- Ignores prior knowledge
- Assumes a deterministic, observable “environment”
- Assumes examples are given
- Assumes that the agent wants to learn f —why?)



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

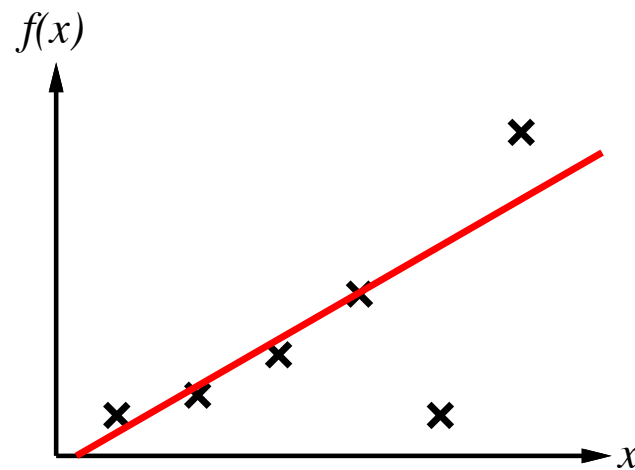
E.g., curve fitting:



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

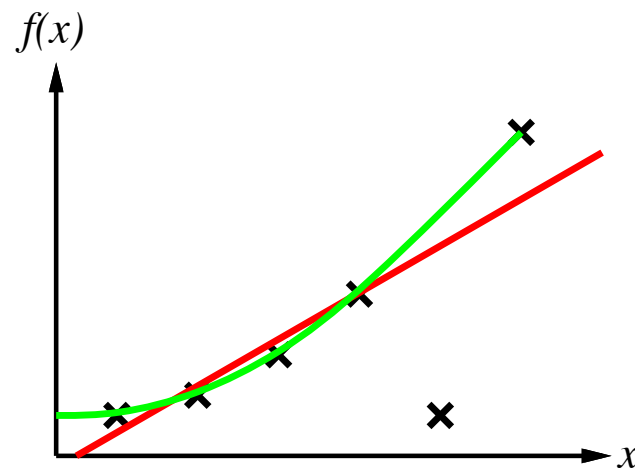
E.g., curve fitting:



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

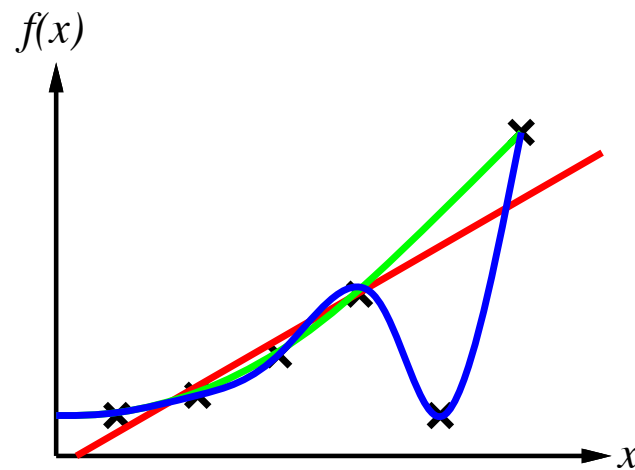
E.g., curve fitting:



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

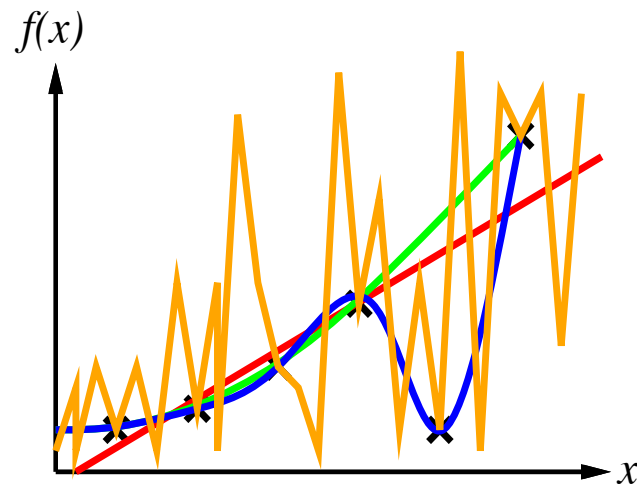
E.g., curve fitting:



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

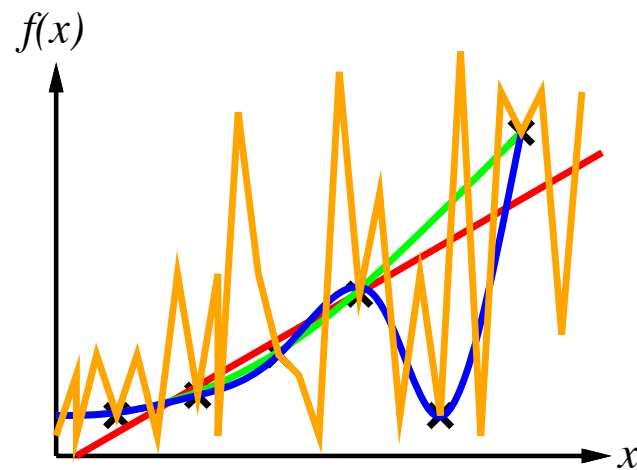
E.g., curve fitting:



Inductive learning method

Construct/adjust h to agree with f on training set
(h is **consistent** if it agrees with f on all examples)

E.g., curve fitting:



Ockham's razor: maximize a combination of consistency and simplicity



Machine Learning: many different techniques

Supervised learning. The agent observes a training set containing a set of (I,O) pairs and learn a function mapping Input to Output

Classification: output has discrete values

Regression: output has continue value

Input provided in terms of **features** which represent the data. **Labeled** data.

Example: price of house in an area

Reinforcement learning. The agent learn from rewards or punishment.

Unsupervised learning. The agent learn pattern in the input without any feedback from the output

Clustering

Example: Google news.



Machine Learning: many different techniques

Semi-supervised learning. A few known labeled examples in a larger set of unlabelled data.

Example: Recognizing age of persons from pictures.

Inductive learning in logic. To find a hypothesis, expressed in logical terms, that classifies the examples well and generalizes well to new examples.

Process of gradually eliminating hypotheses that are inconsistent with the examples, narrowing down the possibilities.

Inductive Logic Programming

- automated discovery of new rules for protein folding (PROGOL 2001)
- new knowledge about the functional genomics of yeast (robot performing biological experiments (2009)
- extract complex relational information from text



Unsupervised vs Supervised Learning

Supervised learning methods

(regression, classification ...)

We observe both a set of features X_1, X_2, \dots, X_p for each object, as well as a response or outcome variable Y . The goal is then to predict Y using X_1, X_2, \dots, X_p .

Unsupervised learning,

We observe only the features X_1, X_2, \dots, X_p . We are not interested in prediction, because we do not have an associated response variable Y .



Supervised learning: linear regression

Given some I/O data
(prices of house)

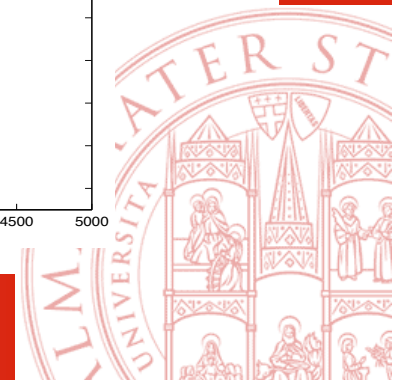
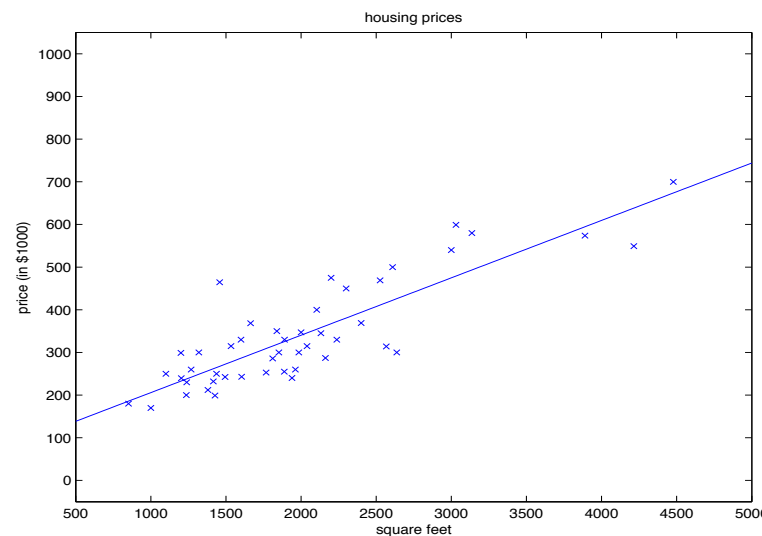
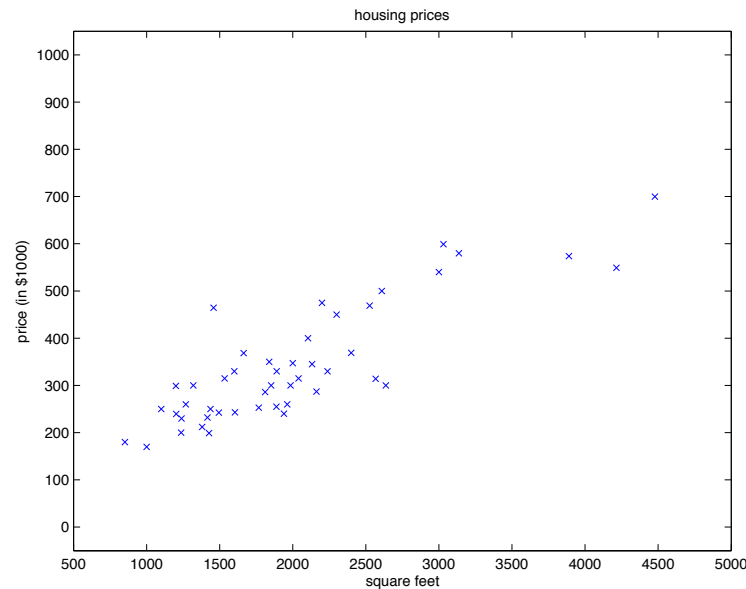
Try to learn the hypothesis function h

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

By minimizing the cost function

$$J(\theta) = \frac{1}{2} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Using gradient descent
one obtain



Supervised learning: classification



Goals of Unsupervised Learning

The goal of unsupervised learning is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?

Two main methods:

Principal components analysis, a tool used for data visualization or data pre-processing before supervised techniques are applied. Produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.

Clustering, a broad class of methods for discovering unknown subgroups in data. We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other,

- **k-means clustering**: we partition the data into k clusters.
- **hierarchical clustering**: we do not know how many clusters we want;



Advantages of Unsupervised Learning

Techniques for unsupervised learning are used in many fields:

- subgroups of breast cancer patients grouped by their gene expression measurements,
- groups of shoppers characterized by their browsing and purchase histories,
- movies grouped by the ratings assigned by movie viewers.

It is often easier to obtain unlabeled data — from a lab instrument or a computer — than labeled data, which can require human intervention.



Attribute-based representations

Examples described by **attribute values** (Boolean, discrete, continuous, etc.)
 E.g., situations where I will/won't wait for a table:

Example	Attributes										Target
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	<i>WillWait</i>
X_1	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>0–10</i>	<i>T</i>
X_2	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>30–60</i>	<i>F</i>
X_3	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>Some</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>0–10</i>	<i>T</i>
X_4	<i>T</i>	<i>F</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>10–30</i>	<i>T</i>
X_5	<i>T</i>	<i>F</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>French</i>	<i>>60</i>	<i>F</i>
X_6	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Italian</i>	<i>0–10</i>	<i>T</i>
X_7	<i>F</i>	<i>T</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>0–10</i>	<i>F</i>
X_8	<i>F</i>	<i>F</i>	<i>F</i>	<i>T</i>	<i>Some</i>	<i>\$\$</i>	<i>T</i>	<i>T</i>	<i>Thai</i>	<i>0–10</i>	<i>T</i>
X_9	<i>F</i>	<i>T</i>	<i>T</i>	<i>F</i>	<i>Full</i>	<i>\$</i>	<i>T</i>	<i>F</i>	<i>Burger</i>	<i>>60</i>	<i>F</i>
X_{10}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$\$\$</i>	<i>F</i>	<i>T</i>	<i>Italian</i>	<i>10–30</i>	<i>F</i>
X_{11}	<i>F</i>	<i>F</i>	<i>F</i>	<i>F</i>	<i>None</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Thai</i>	<i>0–10</i>	<i>F</i>
X_{12}	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>	<i>Full</i>	<i>\$</i>	<i>F</i>	<i>F</i>	<i>Burger</i>	<i>30–60</i>	<i>T</i>

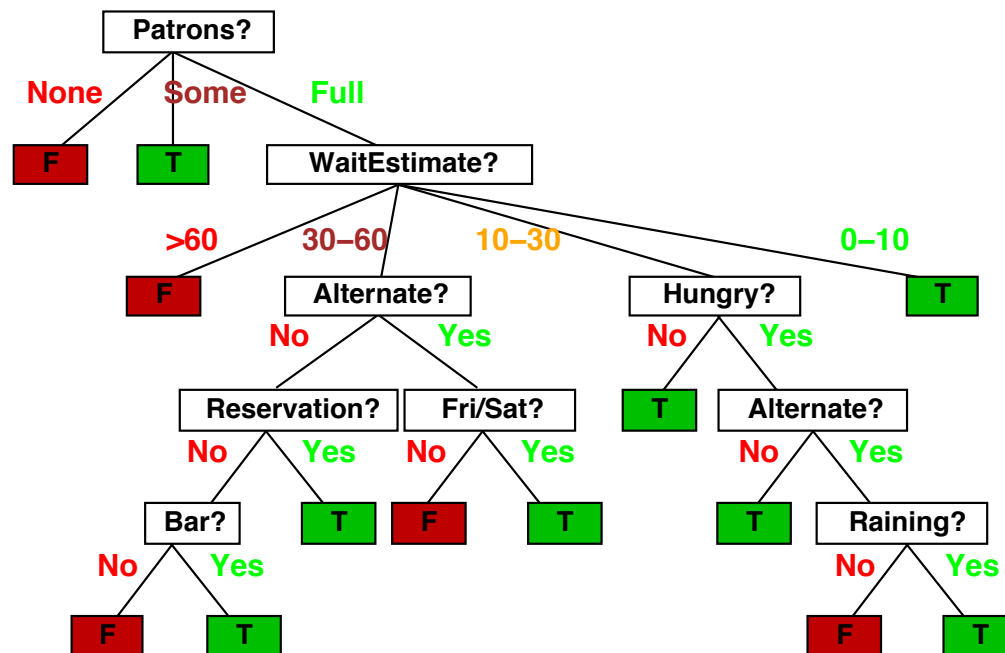
Classification of examples is **positive** (T) or **negative** (F)



Decision trees

One possible representation for hypotheses

E.g., here is the “true” tree for deciding whether to wait:



Decision trees

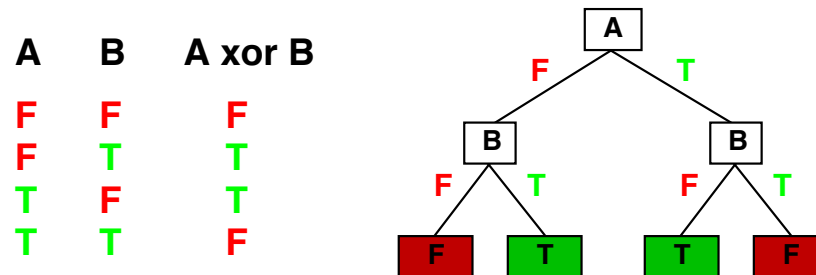
A Decision Tree (DT) represents a function that takes as input a vector of attribute values and returns a “decision”

Each internal node in a DT correspond to a test on one of the attributes



Expressiveness

Decision trees can express any function of the input attributes.
E.g., for Boolean functions, truth table row \rightarrow path to leaf:



Trivially, there is a consistent decision tree for any training set
w/ one path to leaf for each example (unless f nondeterministic in x)
but it probably won't generalize to new examples

Prefer to find more **compact** decision trees



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions

= number of distinct truth tables with 2^n rows



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions

= number of distinct truth tables with 2^n rows = 2^{2^n}



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions

= 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 trees



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions

= 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 trees

How many purely conjunctive hypotheses (e.g., $Hungry \wedge \neg Rain$)??



Hypothesis spaces

How many distinct decision trees with n Boolean attributes??

= number of Boolean functions

= 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 trees

How many purely conjunctive hypotheses (e.g., $Hungry \wedge \neg Rain$)??

Each attribute can be in (positive), in (negative), or out

$\Rightarrow 3^n$ distinct conjunctive hypotheses

More expressive hypothesis space

- increases chance that target function can be expressed 😊
- increases number of hypotheses consistent w/ training set
 \Rightarrow may get worse predictions 😞



Decision tree learning

Aim: find a small tree consistent with the training examples

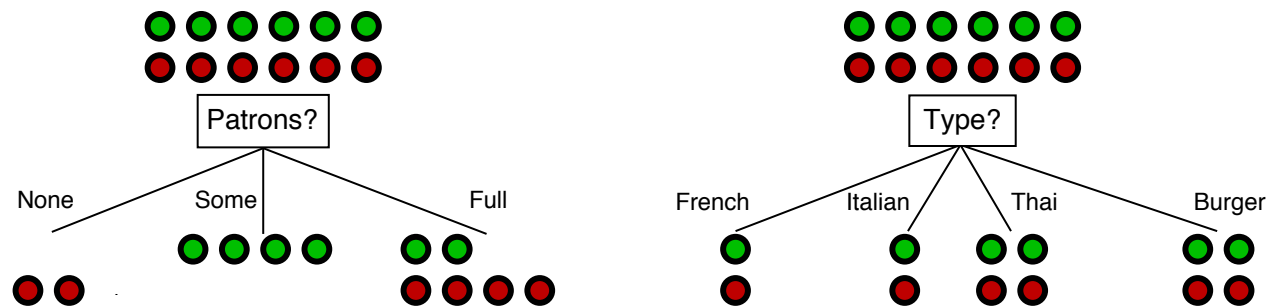
Idea: (recursively) choose “most significant” attribute as root of (sub)tree

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



Choosing an attribute

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



Patrons? is a better choice—gives **information** about the classification



Information

Information answers questions

The more clueless I am about the answer initially, the more information is contained in the answer

Scale: 1 bit = answer to Boolean question with prior $\langle 0.5, 0.5 \rangle$

Information in an answer when prior is $\langle P_1, \dots, P_n \rangle$ is

$$H(\langle P_1, \dots, P_n \rangle) = \sum_{i=1}^n -P_i \log_2 P_i$$

(also called **entropy** of the prior)



Information contd.

Suppose we have p positive and n negative examples at the root

$\Rightarrow H(\langle p/(p+n), n/(p+n) \rangle)$ bits needed to classify a new example

E.g., for 12 restaurant examples, $p = n = 6$ so we need 1 bit

An attribute splits the examples E into subsets E_i , each of which (we hope) needs less information to complete the classification

Let E_i have p_i positive and n_i negative examples

$\Rightarrow H(\langle p_i/(p_i+n_i), n_i/(p_i+n_i) \rangle)$ bits needed to classify a new example

\Rightarrow **expected** number of bits per example over all branches is

$$\sum_i \frac{p_i + n_i}{p + n} H(\langle p_i/(p_i + n_i), n_i/(p_i + n_i) \rangle)$$

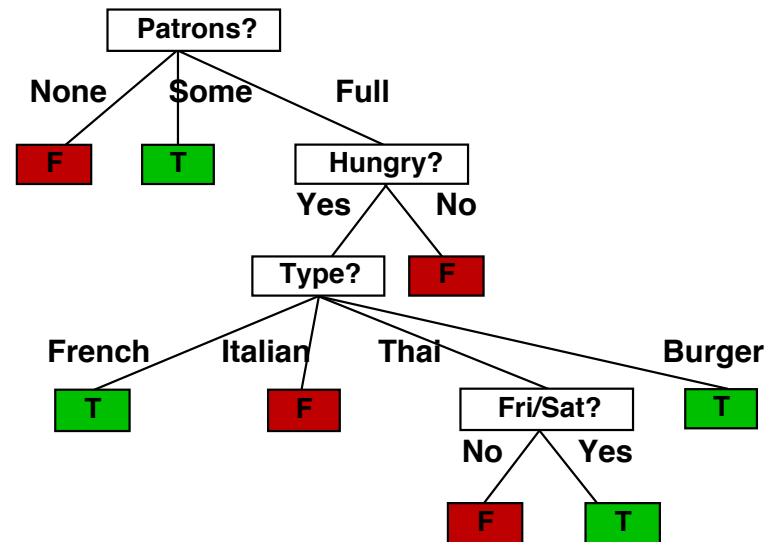
For *Patrons?*, this is 0.459 bits, for *Type* this is (still) 1 bit

\Rightarrow choose the attribute that minimizes the remaining information needed



Example contd.

Decision tree learned from the 12 examples:



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

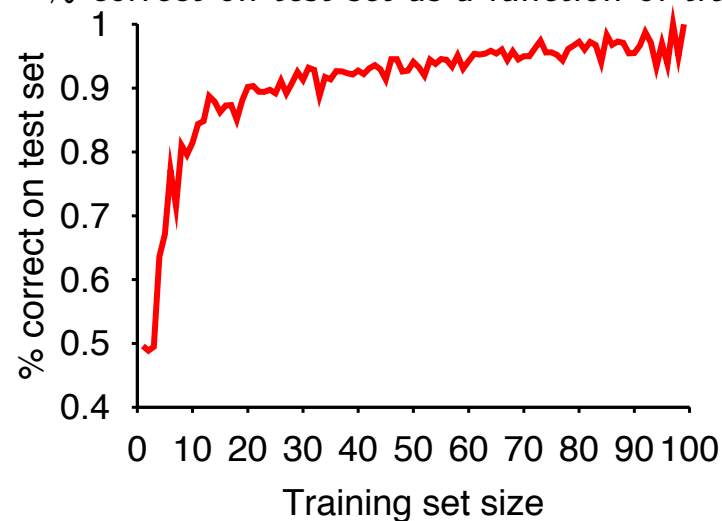


Performance measurement

How do we know that $h \approx f$? (Hume's **Problem of Induction**)

- 1) Use theorems of computational/statistical learning theory
- 2) Try h on a new **test set** of examples
(use **same distribution over example space** as training set)

Learning curve = % correct on test set as a function of training set size



Performance measurement contd.

Learning curve depends on

- **realizable** (can express target function) vs. **non-realizable**
non-realizability can be due to missing attributes
or restricted hypothesis class (e.g., thresholded linear function)
- redundant expressiveness (e.g., loads of irrelevant attributes)

