Introduction to Artificial Intelligence

Learning from Oberservations

Bernhard Beckert

UNIVERSITÄT KOBLENZ-LANDAU

Wintersemester 2003/2004

Outline

- **Learning agents**
- **Inductive learning**
- **Decision tree learning**

Learning

Reasons for learning

- **Learning is essential for unknown environments,**

  – when designer lacks omniscience –

Learning

Reasons for learning

- **Learning is essential for unknown environments,**

  – when designer lacks omniscience –

- **Learning is useful as a system construction method,**

  – expose the agent to reality rather than trying to write it down –

Learning

Reasons for learning

- **Learning is essential for unknown environments,**

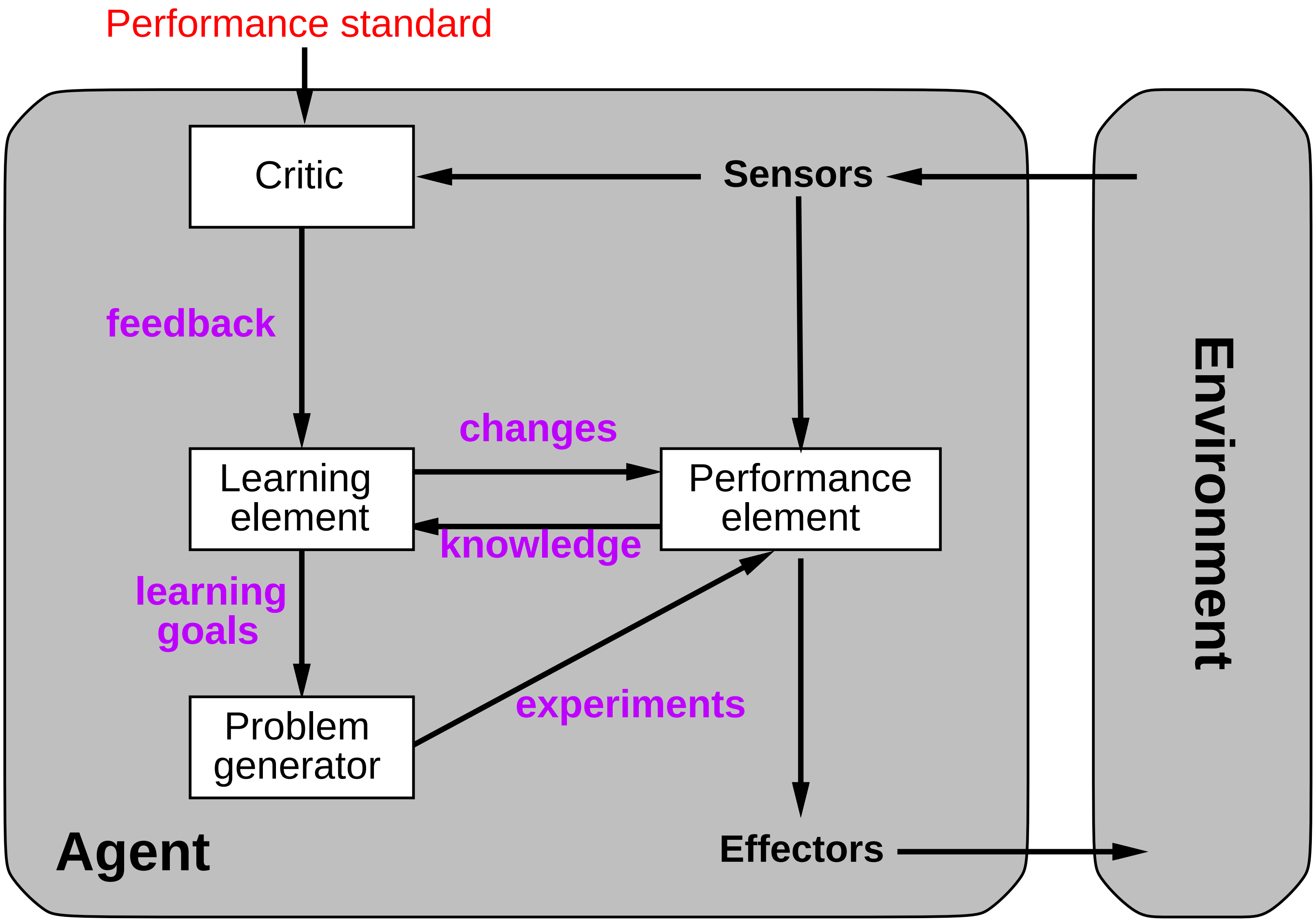
  – when designer lacks omniscience –

- **Learning is useful as a system construction method,**

  – expose the agent to reality rather than trying to write it down –

- **Learning modifies the agent's decision mechanisms to improve performance**

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**

Types of Learning

Supervised learning

Correct answers for each example instance known

Requires “teacher”

Types of Learning

Supervised learning

Correct answers for each example instance known

Requires “teacher”

Reinforcement learning

Occasional rewards

Learning is harder

Requires no teacher

Inductive Learning (a.k.a. Science)

Simplest form

Learn a function f from examples (tabula rasa), i.e.,
find an hypothesis h such that $h \approx f$ given a training set of examples

f is the target function

An example is a pair $x,\ f(x)$

Inductive Learning (a.k.a. Science)

Simplest form

Learn a function f from examples (tabula rasa), i.e.,
find an hypothesis h such that $h \approx f$ given a training set of examples

f is the target function

An example is a pair $x,\ f(x)$

Example (for an example)

$$\begin{array}{c|c|c} O & O & X \\ \hline & X & \\ \hline X & & \end{array} \quad , \quad +1$$

Inductive Learning Method

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

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting

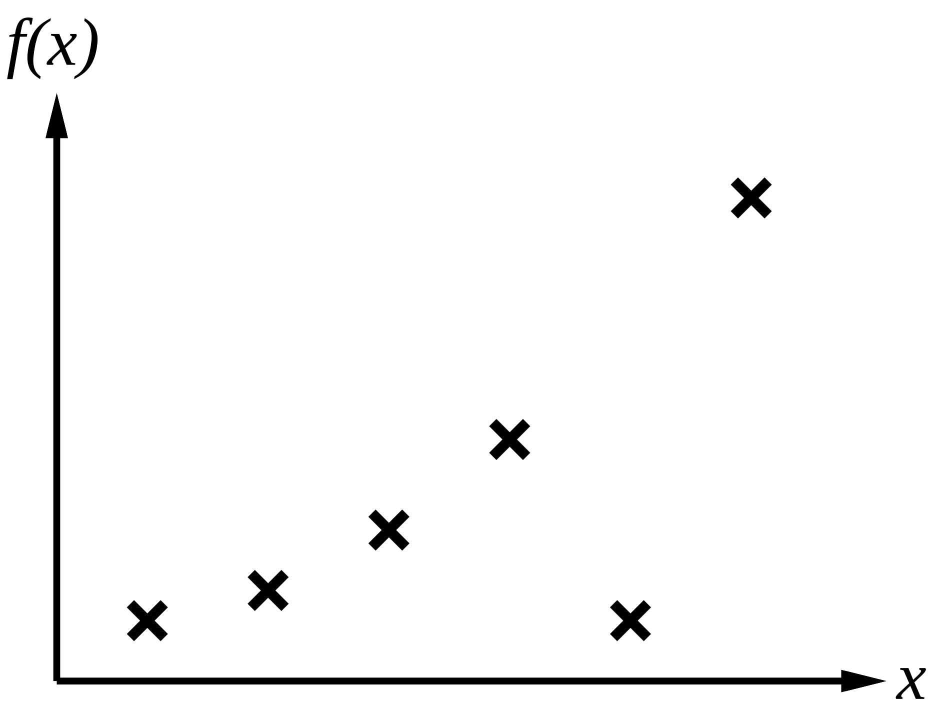
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



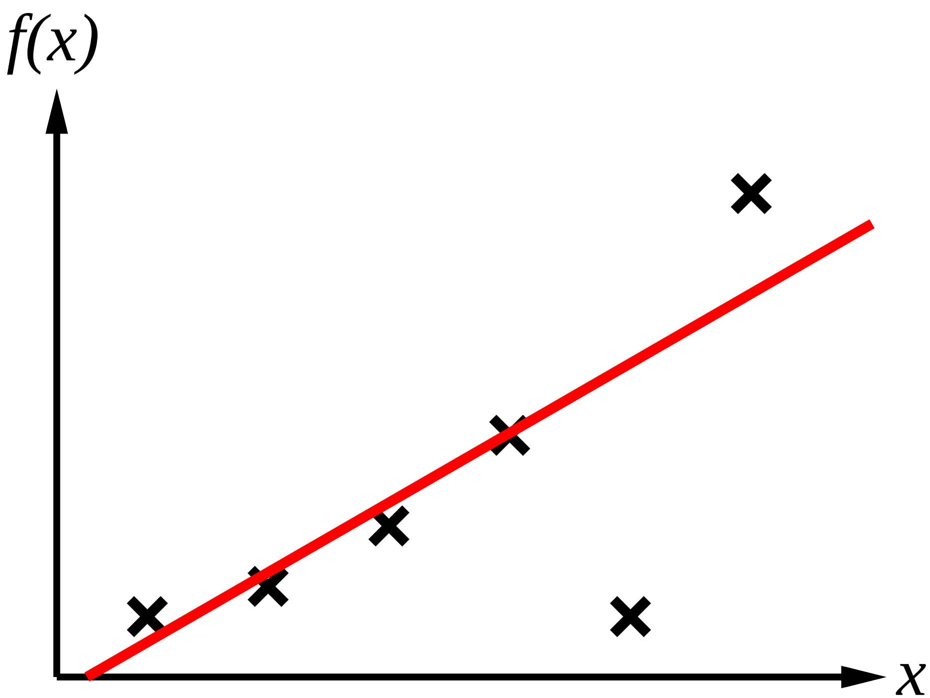
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



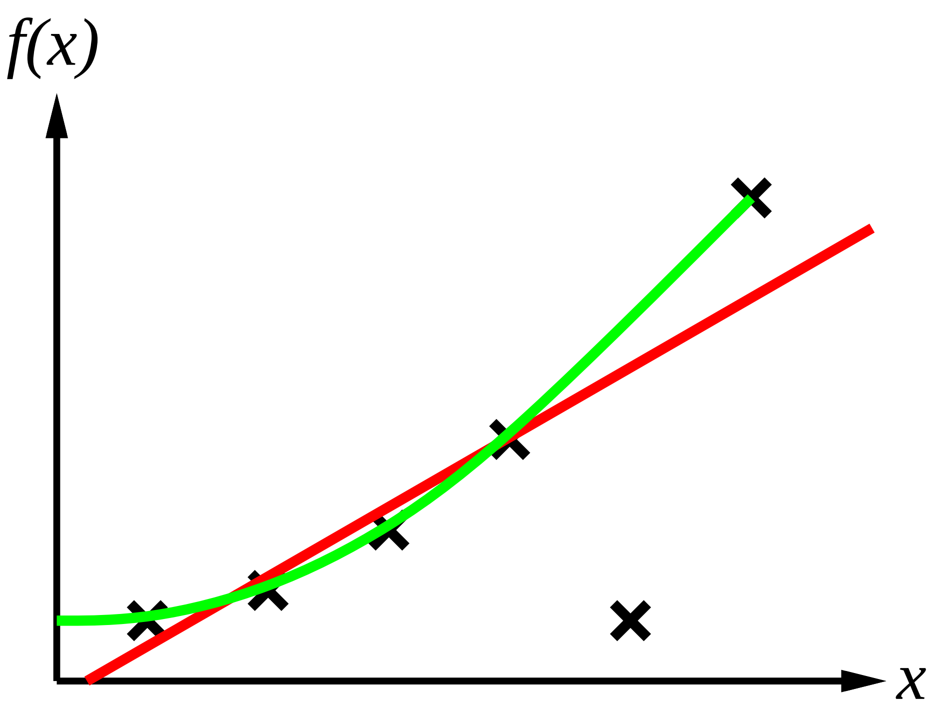
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



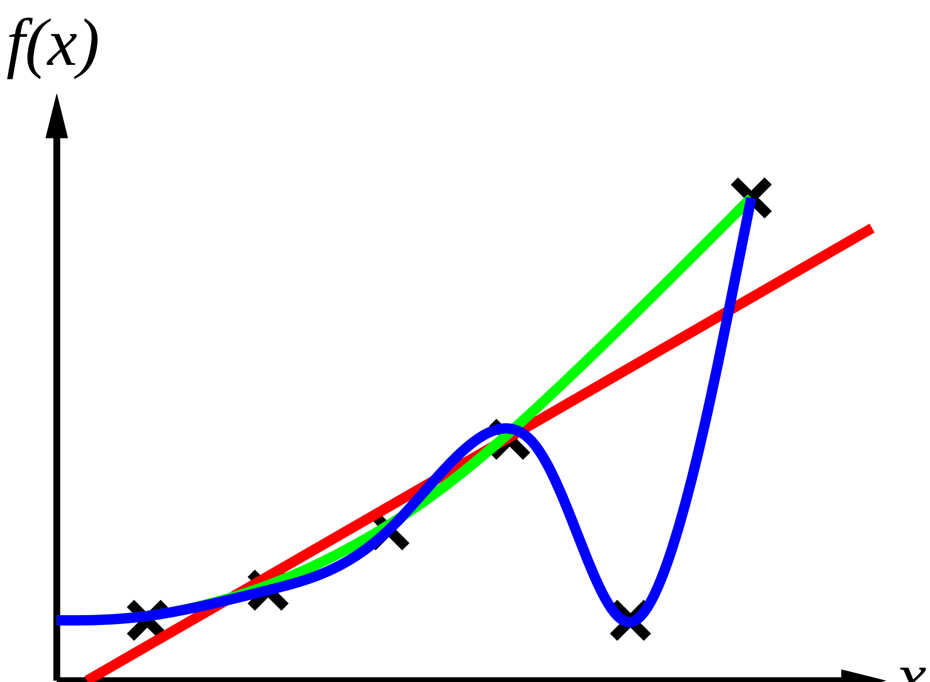
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



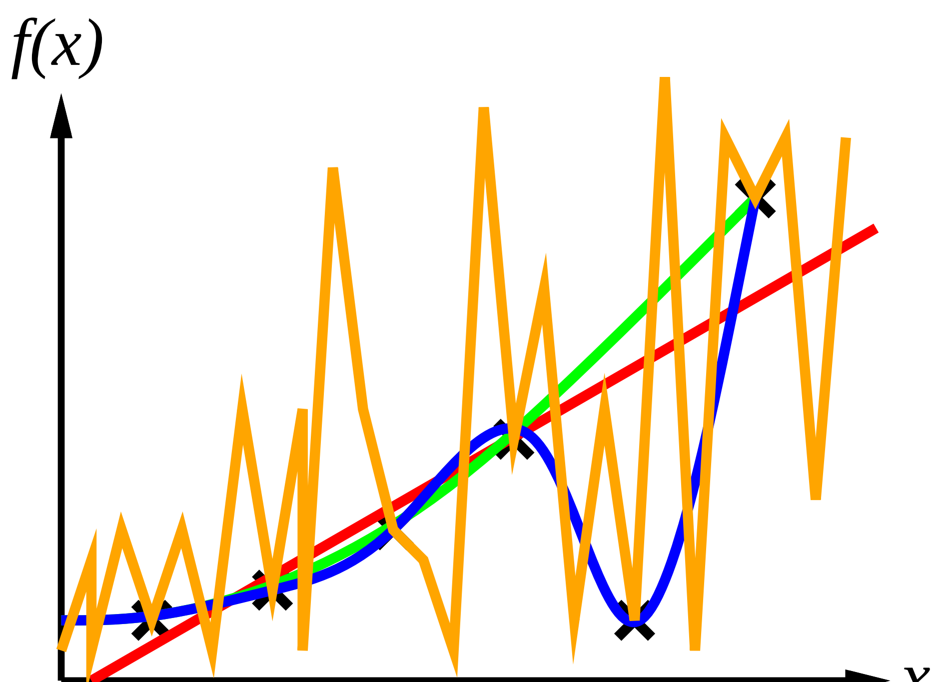
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



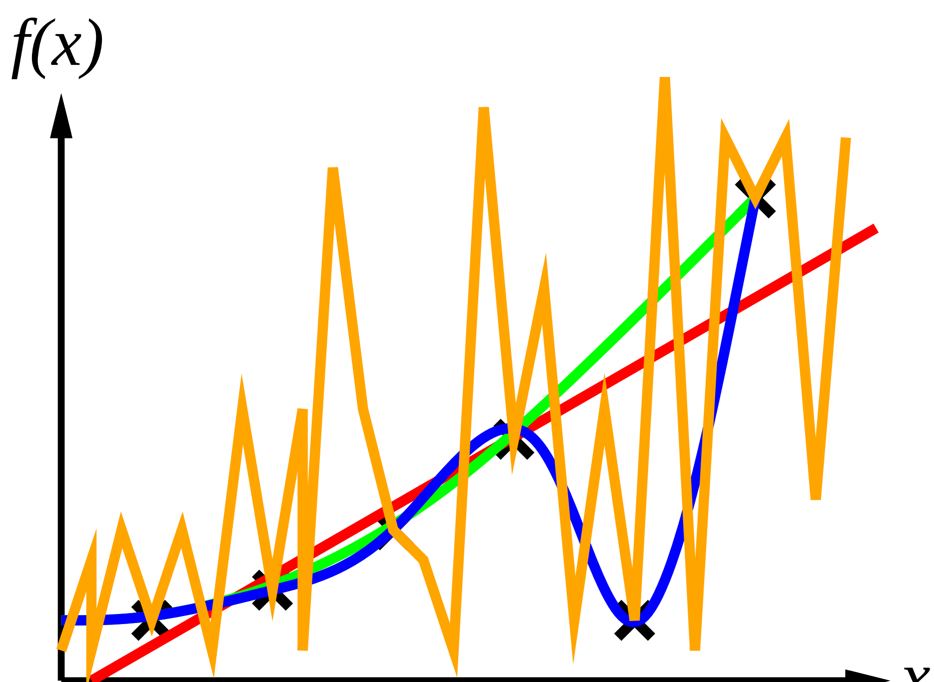
Inductive Learning Method

Idea

Construct/adjust h to agree with f on training set

h is consistent if it agrees with f on all examples

Example: Curve fitting



Ockham's razor

Maximize a combination of consistency and simplicity

Attribute-based Representations

Example description consists of

- **Attribute values** **(boolean, discrete, continuous, etc.)**
- **Target value**

Attribute-based Representations

Example

Situations where I will/won't wait for a table in a restaurant

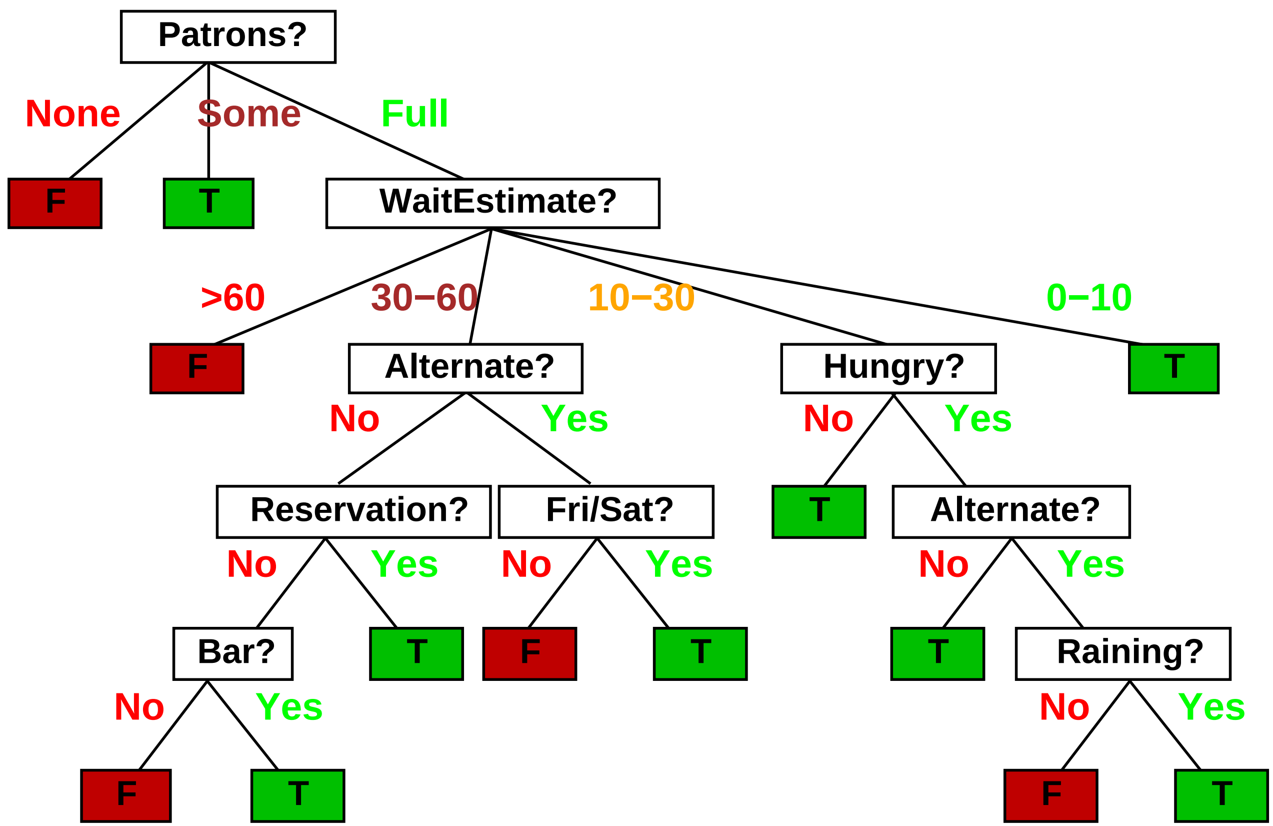
Exmpl.	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	WillWait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	$>$60	F
X_6	F	T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

Decision Trees

A possible representation for hypotheses

Example

The “correct” tree for deciding whether to wait



Decision Trees

Properties

- **Decision trees can approximate any function of the input attributes (“correct” decision tree may be infinite)**

Decision Trees

Properties

- **Decision trees can approximate any function of the input attributes ("correct" decision tree may be infinite)**

- **Trivially, there is a consistent decision tree for any training set with one path to leaf for each example (unless f nondeterministic)**

Decision Trees

Properties

- Decision trees can approximate any function of the input attributes
  (“correct” decision tree may be infinite)
- Trivially, there is a consistent decision tree for any training set
  with one path to leaf for each example
  (unless f nondeterministic)
- Decision tree for training examples probably won’t generalize
  to new examples

Decision Trees

Properties

- Decision trees can approximate any function of the input attributes ("correct" decision tree may be infinite)
- Trivially, there is a consistent decision tree for any training set with one path to leaf for each example (unless f nondeterministic)
- Decision tree for training examples probably won't generalize to new examples
- Compact decision trees are preferable

Decision Trees

Properties

- Decision trees can approximate any function of the input attributes ("correct" decision tree may be infinite)
- Trivially, there is a consistent decision tree for any training set with one path to leaf for each example (unless f nondeterministic)
- Decision tree for training examples probably won't generalize to new examples
- Compact decision trees are preferable
- More expressive hypothesis space
  - increases chance that target function can be expressed
  - increases number of hypotheses consistent with training set

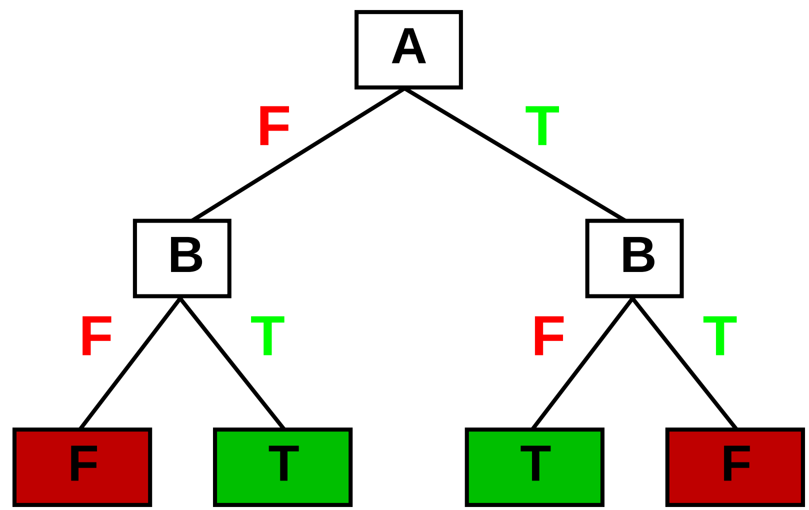
    $\Rightarrow$ may get worse predictions

Decision Trees

Example

For Boolean functions: truth-table row = path to leaf in decision tree

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



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}

Example

With 6 Boolean attributes, there are

$$18{,}446{,}744{,}073{,}709{,}551{,}616 \textbf{ trees}$$

Decision Tree Learning

Aim

Find a small tree consistent with the training examples

Idea

(Recursively) choose “most significant” attribute as root of (sub)tree

Choosing an Attribute

Idea

A good attribute splits the examples into subsets that are (ideally) "all positive" or "all negative", i.e.,

gives much information about the classification

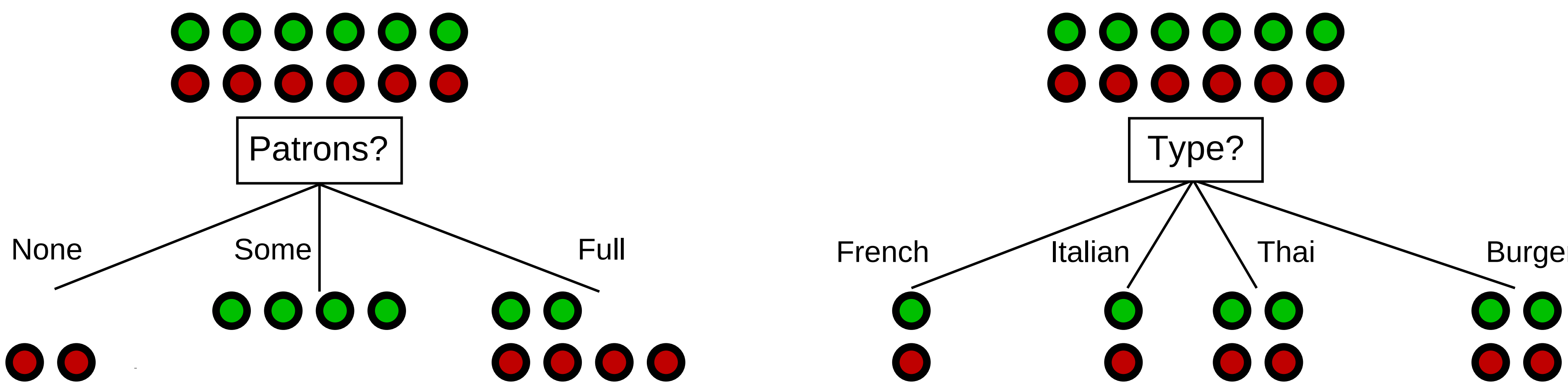
Choosing an Attribute

Idea

A good attribute splits the examples into subsets that are (ideally) “all positive” or “all negative”, i.e.,

gives much information about the classification

Example



Patrons **is a better choice**

Decision Tree Learning: Algorithm

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** MAJORITY-VALUE(*examples*)

else

best $\leftarrow$ CHOOSE-ATTRIBUTE(*attributes*,*examples*)

tree $\leftarrow$ a new decision tree with root test *best*

m $\leftarrow$ MAJORITY-VALUE(*examples*)

for each value v_i of *best* **do**

examples$_i$ $\leftarrow$ $\{$elements of *examples* with *best* = $\mathrm{v}_i\}$

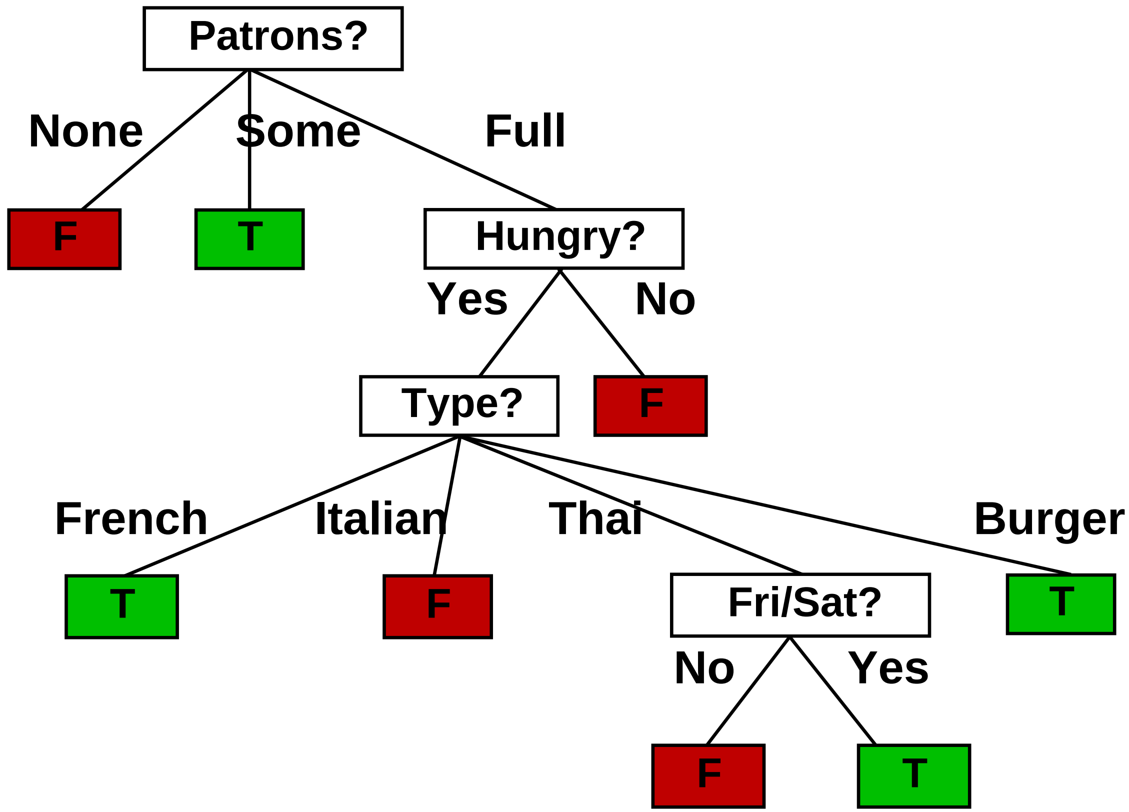
subtree $\leftarrow$,DTL(*examples*$_i$,*attributes* $-$ *best*,*m*)

add a branch to *tree* with label v_i and subtree *subtree*

return *tree*

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

Hume's *Problem of Induction*

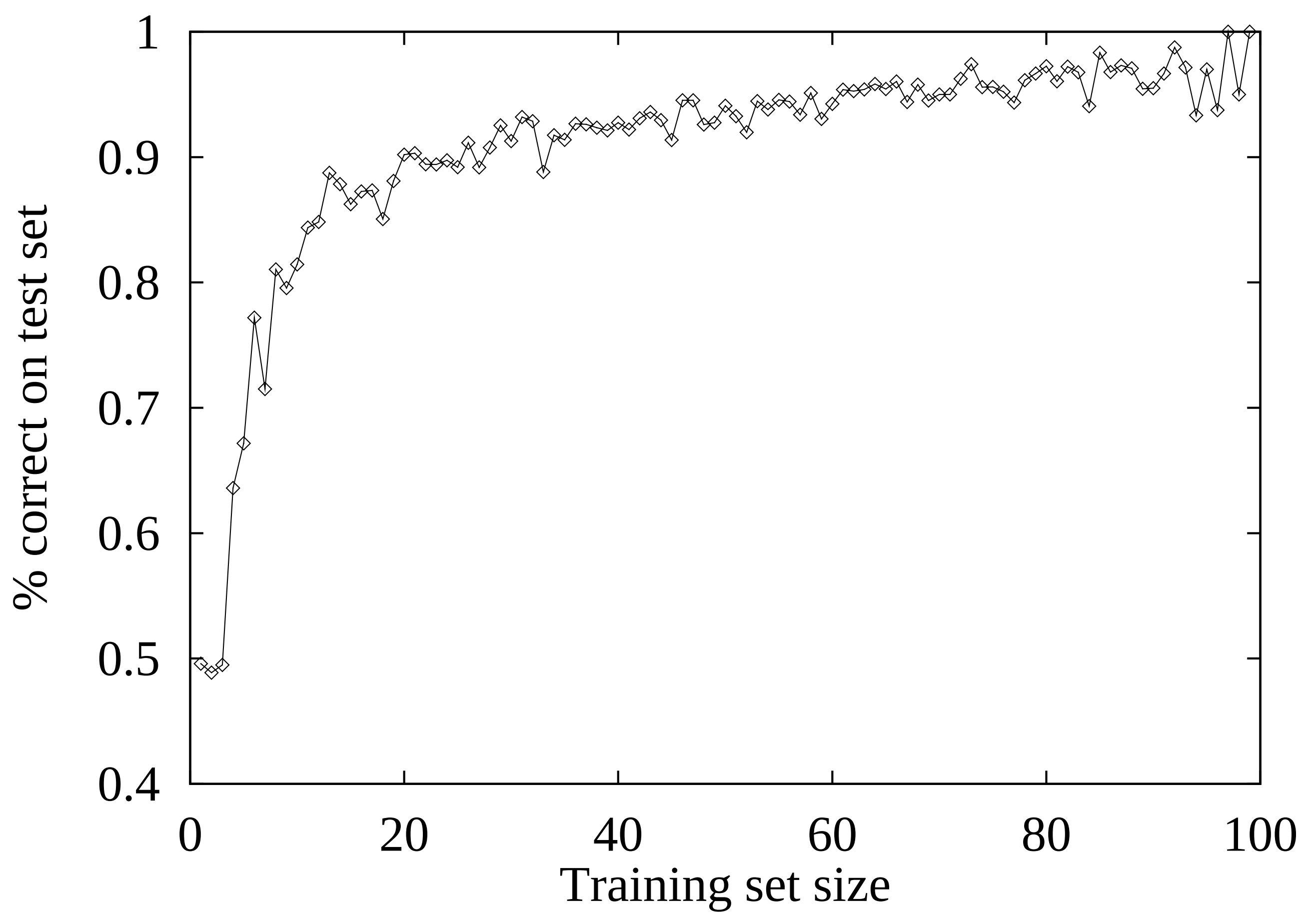
How do we know that $h \approx f$?

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

Performance Measurement

Learning curve

% correct on test set as a function of training set size

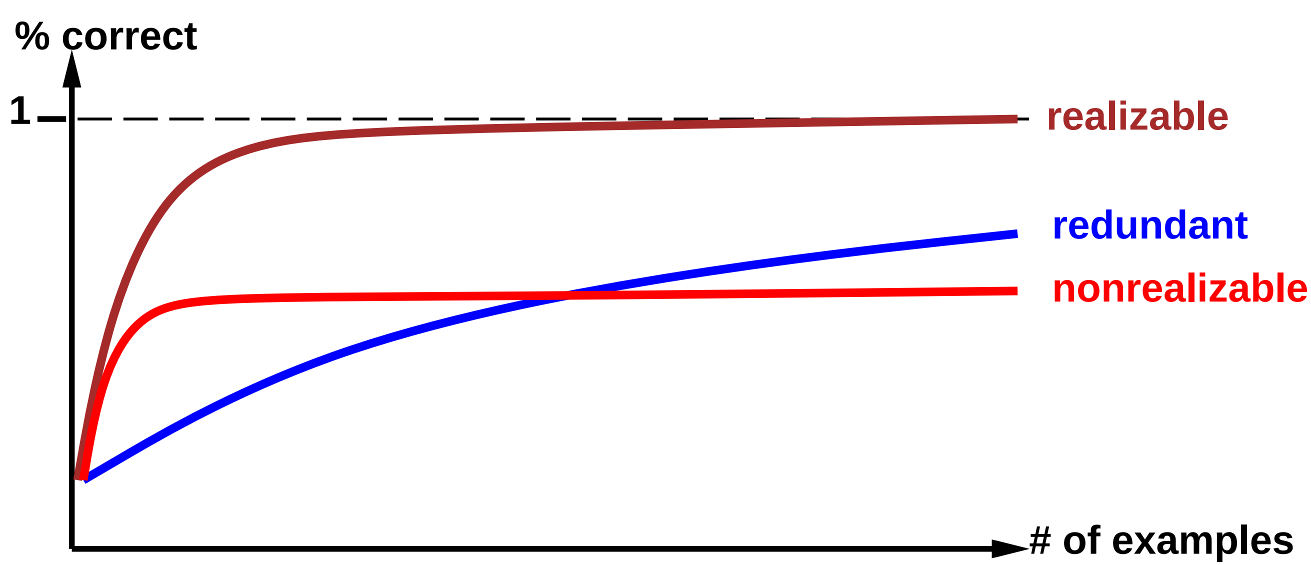


Performance Measurement (cont.)

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)



Summary

- Learning needed for unknown environments, lazy designers
- Learning agent = performance element + learning element
- Learning method depends on type of performance element, available feedback, type of component to be improved
- For supervised learning, the aim is to find a simple hypothesis approximately consistent with training examples
- Decision tree learning using information gain
- Learning performance = prediction accuracy measured on test set