

# Foundations of AI

## Machine Learning

Luc De Raedt

1

## Learning

- **What is learning ?**  
An agent learns, when it improves its performance at specific tasks through experience  
→ e.g. Game playing programs
  - **Why learning ?**  
→ Engineering, Philosophy of Science, Cognitive Sciences  
→ Knowledge discovery in data bases and data mining
- No intelligence without learning !**

3

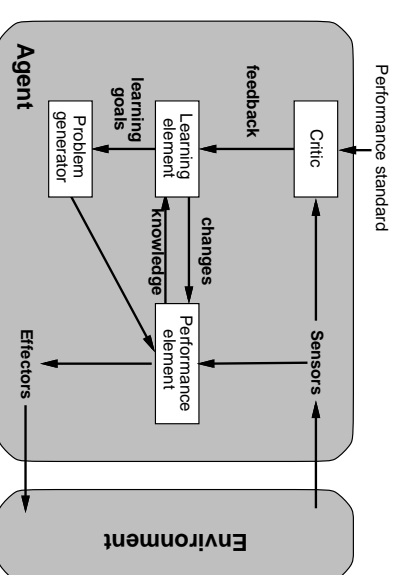
## Overview

- Learning agents
- Inductive learning
- Decision tree learning
- Inducing hypotheses
- Why learning works

2

## Learning agents

Up till now percepts were the basis for acting. Now they should also be used for learning, for improving future behaviour.



4

## Components of learning agents

---

**Performance-Element:** Select actions on the basis of percepts  $\rightsquigarrow$  cf. our previous agent model.

**Learning-Element:** Implementing improvements  $\rightsquigarrow$  requires knowledge and representation of the agent itself and its behaviour in the world.

**Critic:** Evaluates the actions and performance of the agent on the basis of external or internal feedback.

**Problem-Generator:** Proposes explorative actions that may lead to new experience.

5

## The learning element

---

Key issues

1. Which parts of the Performance-Element should be improved ?
2. What representation is used ?
3. What form of feedback / learning experience is available ?
4. What knowledge is available to the learner?

6

## Learning Experience

---

**Input:** Information about the environment

**Output:** Effects of the actions of the agent

Effects that arise in the real world are often different from the desired effects (the target) of the agent.

**Goal of learning :** Approximating the target function

**Supervised Learning:** Input/Outputs are available. A teacher tells the system the which action to execute

**Reinforcement Learning:** The agent is rewarded or punished according to its behaviour.

**Unsupervised Learning:** The agent can detect regularity in the domain through observations, but does *not* know which actions to execute.

7

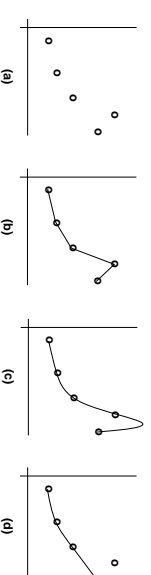
## Inductive Learning

---

Almost all forms of learning can be understood as **learning the representation of a function.**

An **example** is a tuple  $(x, f(x))$ .

**Inductive Inference:** Given a set of examples of a function  $f$  find a function  $h$  (a hypothesis) that approximates  $f$ .



**Bias:** Anything that influences the selection of hypotheses and that is not based on the examples.

8

## Decision trees

**Input:** Description of situation using a set of propositions or first order literals.

**Output:** Yes/No decision (class) w.r.t. a certain target predicate.

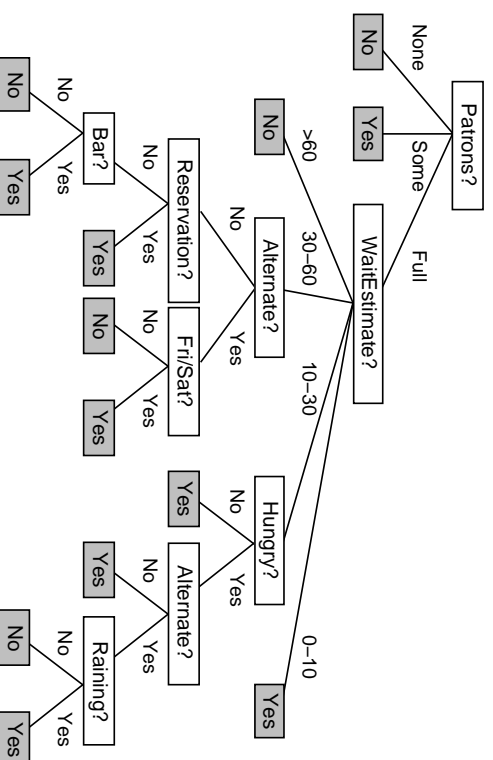
Decision trees represent boolean functions.

An internal node in a decision tree represents a test for a certain property or attribute. For each possible value of the attribute there will be a subtree. Leaf nodes contain a class value.

**Goal of the learning :** Definition of the target predicate as a decision tree.

9

## Restaurantbeispiel (Decision tree)



11

## Restaurant example (Tests)

*Patrons*: how many guests ? (none, some, full)

*WaitEstimate*: how long should we wait ? (0-10, 10-30, 30-60, >60)

*Alternate*: Is there an alternative ? (T/F)

*Hungry*: Am I hungry ? (T/F)

*Reservation*: Did I make a reservation ? (T/F)

*Bar*: Is there a bar (to better enjoy the waiting) ? (T/F)

*Fri/Sat*: Is it Friday or Saturday (T/F)

*Raining*: Is it raining ? (T/F)

*Price*: How expensive is the food ? (\$, \$\$, \$\$\$)

*Type*: Type of restaurant ? (french, italian, Thai, Burger)

10

## Representation of the target predicate

A decision tree can be represented as a **set of clauses/rules**. A rule represents one path ending in the class YES.

WillWait(*r*) :- Patrons(*r*,Full), WaitEstimate(*r*,10-30), Hungry(*r*,No)

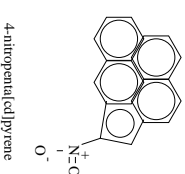
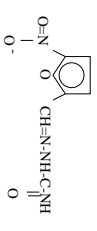
WillWait(*r*) :- Patrons(*r*,Some)

...

12

## Example: Mutagenicity

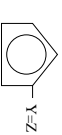
Active



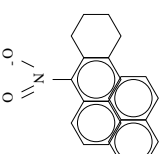
nitrofurzone

4-nitroindole

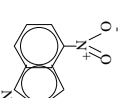
Structural alert:



Inactive



6-nitro-7,8,9,10-tetrahydrobenzo[a]pyrene



4-nitroindole

14

## Expressiveness of decision trees

**Theorem 1** All boolean functions can be represented as decision trees.

Can a decision tree represent complex structured objects ?

~> Traditional decision trees **cannot**. All tests concern the same simple object (here: Restaurant  $r$ ) and the language of traditional decision trees is inherently propositional.

For instance, the following condition cannot be used as test/rule in traditional decision trees.

$$\exists r_2 \text{ Near}(r_2, r) \wedge \text{Price}(r, p) \wedge \text{Price}(r_2, p_2) \wedge \text{Cheaper}(p_2, p)$$

~> Extensions exist (inductive logic programming), e.g. (Blockeel and De Raedt, *Artificial Intelligence*, 1998)

13

## Compact representations

In theory, one can represent any boolean function as a decision tree. This is done by letting each row in the truth-table correspond to one path in the tree.

This leads however to exponentially large trees (in the number of attributes).

Functions that require decision trees of exponential size:

*Parity Function:*

$$p(x) = \begin{cases} 1 & \text{even number of inputs that are 1s} \\ 0 & \text{otherwise} \end{cases}$$

*Majority Function:*

$$m(x) = \begin{cases} 1 & \text{at least half of the inputs is 1} \\ 0 & \text{otherwise} \end{cases}$$

But: there is no compact representation for all possible boolean functions.

15

## The training set

Classification of an example = value of the target predicate

TRUE ~> positive Example  
FALSE ~> negative Example

Example	Attributes										Goal	
	All	Bar	Fri	Hun	Pal	Price	Rain	Res	Res	Type		Est
X <sub>1</sub>	Yes	No	No	Yes	Some	\$\$\$	No	Yes	Yes	French	0-10	Yes
X <sub>2</sub>	Yes	No	No	Yes	Full	\$\$\$	No	No	No	Thai	30-60	No
X <sub>3</sub>	No	Yes	No	No	Some	\$	No	No	No	Burger	0-10	Yes
X <sub>4</sub>	Yes	No	Yes	Yes	Full	\$	No	No	No	Thai	10-30	Yes
X <sub>5</sub>	Yes	No	Yes	No	Full	\$\$\$	Yes	Yes	Yes	French	>60	No
X <sub>6</sub>	No	Yes	No	Yes	Some	\$\$\$	Yes	Yes	Yes	Italian	0-10	Yes
X <sub>7</sub>	No	Yes	No	No	None	\$	Yes	No	No	Burger	0-10	No
X <sub>8</sub>	No	No	No	Yes	Some	\$\$	Yes	Yes	Yes	Thai	>60	Yes
X <sub>9</sub>	No	Yes	Yes	Yes	Full	\$	Yes	No	No	Burger	10-30	No
X <sub>10</sub>	No	No	No	Yes	Full	\$\$\$	No	Yes	Yes	Italian	0-10	No
X <sub>11</sub>	No	No	No	No	None	\$	No	No	No	Thai	0-10	No
X <sub>12</sub>	Yes	Yes	Yes	Yes	Full	\$	No	No	No	Burger	30-60	Yes

16



## The Algorithm

```

function DECISION-TREE-LEARNING(examples, attributes, default) returns a decision tree
  inputs: examples, set of examples
           attributes, set of attributes
           default, default value for the goal predicate

  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 ← 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 =  $v_i$ }
      subtreei ← DECISION-TREE-LEARNING(examplesi, attributes - best,
                                         MAJORITY-VALUE(examplesi))
      add a branch to tree with label  $v_i$  and subtree subtreei
    end
  return tree
  
```

21

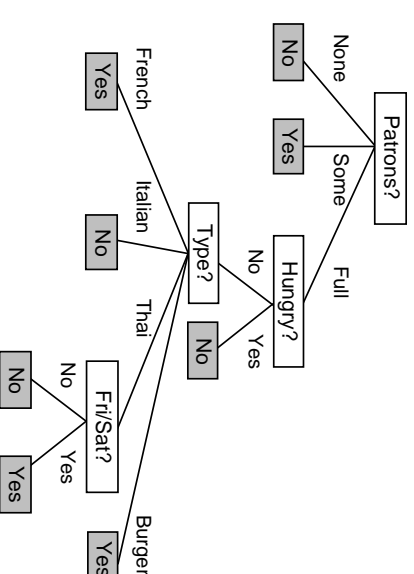
## Evaluation of learning

Determining the predictive accuracy of the hypothesis:

- Determine a (large) number of examples.
- Divide the training examples in two *disjunct* subsets : **training-** and **testset**.
- Use the trainingset to construct the hypothesis  $H$ .
- Use the testset to compute the accuracy, i.e. the percentage of correctly classified examples.
- Repeat the procedure for randomly selected training sets of different size.

23

## Restaurant example



22

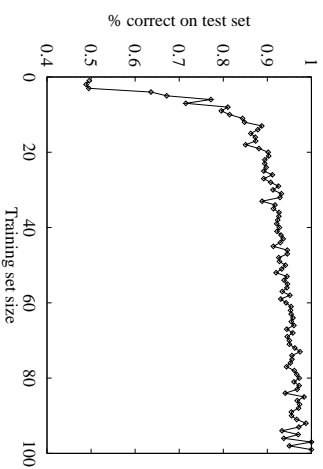
## Important points

- Training- and testset **MUST** be disjunct.
- Frequent mistake: use training set for testing !

24

## Learning curve in the Restaurant example

---



The predictive accuracy increases with the size of the trainingset.

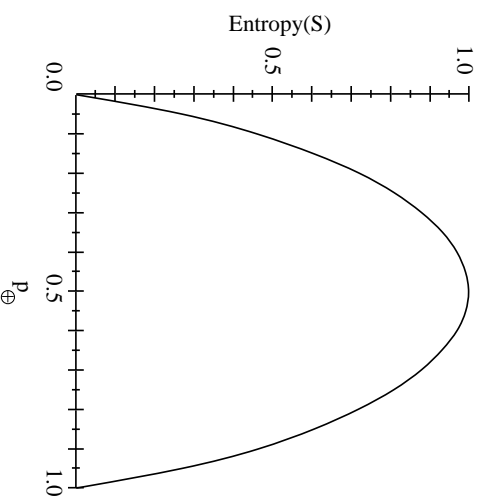
### Example applications:

- Behaviour cloning (e.g. flight simulators)
- Decision support systems
- Data Mining

25

---

## Entropy



27

---

## Selecting Attributes

Toss a coin : what is the value of information if we are tossing for 1\$ ?

- Biases coin 99% head and 1% tail.  
( $\implies$  average gain for winning is 0.98)  
 $\rightsquigarrow$  Value of information about the result is less 0.02\$.
- Fair coin .  
 $\rightsquigarrow$  Value of information is less than 1\$.  
 $\rightsquigarrow$  The less you know about the outcome, the more important is information about the result.

26

---

## Information-theory

Measures the value of information in bits.

Given: Possible values  $v_i$  with corresponding probabilities  $P(v_i)$

$$I(P^{(v_1)}, \dots, P^{(v_n)}) = \sum_{i=1}^n P^{(i)} \log_2 \left( \frac{1}{P^{(i)}} \right)$$

Coins :

$$I\left(\frac{1}{2}, \frac{1}{2}\right) = \frac{1}{2} \times 1 + \frac{1}{2} \times 1 = 1$$

$$I(0.99, 0.01) = 0.08$$

28

## Selecting attributes

Attribute  $A$  divides set of examples  $E$  in  $p$  positive and  $n$  negative examples:

$$I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = \frac{p}{p+n} \log_2\left(\frac{p+n}{p}\right) + \frac{n}{p+n} \log_2\left(\frac{p+n}{n}\right)$$

Quality of  $A$  also depends on the information that is still needed.

Assumption :  $A$  divides training set  $E$  in subsets  $E_i, i = 1, \dots, v$ .

Each  $E_i$  has again  $I\left(\frac{p_i}{p_i+n_i}, \frac{n_i}{p_i+n_i}\right)$

Random example has value  $i$  with probability  $\frac{p_i+n_i}{p+n}$

29

## Example

$$\text{GAIN}(\text{Patrons?}) = 1 - \left[ \frac{2}{12} I(0, 1) + \frac{4}{12} I(0, 1) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right) \right]$$

$$\approx 0.541$$

$$\begin{aligned} \text{GAIN}(\text{Type?}) &= 1 - \left[ \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{2}{12} I\left(\frac{1}{2}, \frac{1}{2}\right) \right. \\ &\quad \left. + \frac{4}{12} I\left(\frac{2}{4}, \frac{2}{4}\right) + \frac{6}{12} I\left(\frac{2}{6}, \frac{4}{6}\right) \right] \\ &= 0 \end{aligned}$$

31

## Selecting Attributes (2)

↪ Average information content after knowing  $A$  is

$$R(A) = \sum_{i=1}^v \frac{p_i + n_i}{p + n} \times I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

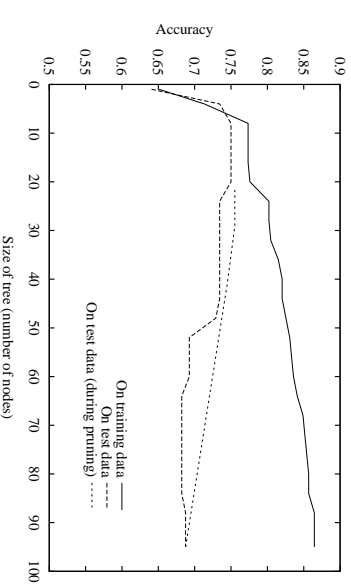
↪ Information gain by choosing  $A$

$$\text{Gain}(A) = I\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - R(A)$$

30

## Noise

- **What is noise ?** Random errors in the data
- **Effect:** Larger trees are induced whose predictive accuracy is smaller. (Overfitting)
- **Overfitting can be avoided using a „Validation Set“:** Divide trainingsset in two subsets ; 70 % to build the tree and 30% to determine the right size of the tree („Pruning“)



32

## Lernen logical hypotheses

Target predicate of arity 1  $Q = WillWait(r)$

The **hypothesis space**  $H$  is the set of all possible logical definitions of  $Q$ . If  $C_i$  is a possible definition then it then it has the form

$$\forall x Q(x) \Leftrightarrow C_i(x)$$

Restaurant-Example: Hypothesis  $H_r$ . (One might also use Prolog here.)

$$\forall r \quad WillWait(r) \Leftrightarrow$$

$$Patrons(r, some) \vee$$

$$Patrons(r, full) \wedge \neg Hungry(r) \wedge Type(r, French) \vee$$

$$Patrons(r, full) \wedge \neg Hungry(r) \wedge Type(r, Thai) \wedge Fri/Sat(r) \vee$$

$$Patrons(r, full) \wedge \neg Hungry(r) \wedge Type(r, Burger)$$

The disjunction of the hypotheses  $H_1 \vee H_2 \vee \dots \vee H_n$  is true. This means that the correct hypothesis is in the hypothesis space.

**Extension** of a hypothesis: Set of examples that satisfy  $C_i$

## Summarizing decision trees

- One way to represent boolean functions
- Decision trees can have exponential size (in the number of attributes)
- It is hard to find the smallest decision trees.
- One way of selecting attributes uses information theory.
- Overfitting occurs because of noise.

33

## Trainingsexample and hypothesis

$D_i(X_i)$  is the representation of the  $i$ -th example.

$$Alternate(X_1) \wedge \neg Bar(X_1) \wedge \neg Fri/Sat(X_1) \wedge Hungry(X_1) \dots \wedge WillWait(X_1)$$

The class of a positive example  $X_i$  is  $Q(X_i)$ , of a negative one  $\neg Q(X_i)$ .

The complete set of examples denotes the conjunction of the examples.

A hypothesis is consistent when it correctly classifies all examples in the training set.

Hypotheses can be inconsistent (w.r.t. specific examples)

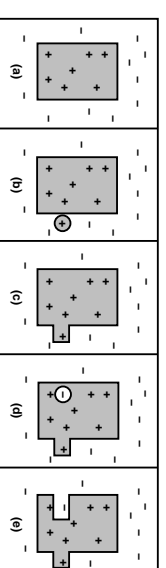
1. **false negative**:  $H$  classifies  $X$  as negative whereas it is positive
2. **false positive**:  $H$  classifies  $X$  as positive whereas it is negative.

35

## Search algorithm: Current-best Hypothesis

Keeps track of one hypothesis that is modified when one discovers an inconsistency with a specific example.

- **Generalisation** : for a **false negative** (b+c)
- **Spezialisaton**: for a **false positive** (d+e)



After all modifications we have to check for consistency, w.r.t the already seen examples.

36

## The algorithm

**function** CURRENT-BEST-LEARNING(*examples*) **returns** a hypothesis  
 $H \leftarrow$  any hypothesis consistent with the first example in *examples*  
**for each** remaining example in *examples* **do**  
   **if**  $e$  is false positive for  $H$  **then**  
      $H \leftarrow$  **choose** a specialization of  $H$  consistent with *examples*  
   **else if**  $e$  is false negative for  $H$  **then**  
      $H \leftarrow$  **choose** a generalization of  $H$  consistent with *examples*  
   **if** no consistent specialization/generalization can be found **then fail**  
**end**  
**return**  $H$

$H_1$  is a generalisation of  $H_2: \forall x C_2(x) \Rightarrow C_1(x)$   
 $\rightsquigarrow$  **dropping conditions** (deletion of conjunctive literals )

$\rightsquigarrow$  more examples satisfy weaker definition  
 Specialization by adding conjunctive literals.

Specialization and generalization operators depend on the language considered.

37

## Illustration (1)

Example	Attributes										Goal
	Alt	Bur	Fri	Hun	Par	Price	Rain	Res	Type	Est	
$X_1$	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	Yes
$X_2$	Yes	No	No	No	Full	\$	No	No	Thai	30-60	No
$X_3$	No	Yes	No	No	Some	\$	No	No	Burger	0-10	Yes
$X_4$	Yes	No	Yes	Yes	Full	\$\$\$	No	Yes	Thai	10-30	Yes
$X_5$	No	No	No	Yes	Full	\$\$\$	No	Yes	French	>60	No
$X_6$	No	Yes	No	No	Some	\$\$	Yes	Yes	Indian	0-10	Yes
$X_7$	No	Yes	No	No	None	\$\$	Yes	No	Burger	0-10	No
$X_8$	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	>60	Yes
$X_9$	No	Yes	Yes	No	Full	\$\$\$	No	Yes	Burger	10-30	No
$X_{10}$	Yes	Yes	Yes	Yes	None	\$	No	No	Thai	0-10	No
$X_{11}$	Yes	No	No	Yes	Full	\$	No	No	Thai	0-10	No
$X_{12}$	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	Yes

1. Example  $X_1$  is positive. *Alternate*( $X_1$ ) is true.

$$H_1 : \forall x \text{ WillWait}(x) \Leftrightarrow \text{Alternate}(x).$$

2. Example  $X_2$  is a false positive:  $H_1$  specialized

$$H_2 : \forall x \text{ WillWait}(x) \Leftrightarrow \text{Alternate}(x) \wedge \text{Patrons}(x, \text{some})$$

38

## Illustration (2)

3.  $X_3$  is a false negative:  $H_2$  generalized

$$H_3 : \forall x \text{ WillWait}(x) \Leftrightarrow \text{Patrons}(x, \text{some})$$

4.  $X_4$  is a false negative:  $H_3$  is generalized

Deleting *Patrons*( $x$ , some) is inconsistent with  $X_2$ . Adding disjunction e.g.

$$H_4 : \forall x \text{ WillWait}(x) \Leftrightarrow \text{Patrons}(x, \text{some}) \vee [\text{Patrons}(x, \text{full}) \wedge \text{Fri} / \text{Sat}(x)]$$

## Disadvantages of current-best Hypothesis

- All previous examples must be considered again.
- No guarantee of finding the simplest hypothesis.
- Good heuristics are hard to find.
- Search can lead to dead-ends, where no modification of the present hypothesis leads to a consistent hypothesis.

*Backtracking* is necessary, because we only keep track of one hypothesis at a time. The hypothesis space is however a disjunction !

$\rightsquigarrow$  *Version Space*

39

40

## Version-Space/Candidate-Elimination Learning

**function** VERSION-SPACE-LEARNING(*examples*) **returns** a version space  
**local variables:** *V*, the version space: the set of all hypotheses

```

V ← the set of all hypotheses
for each example e in examples do
  if V is not empty then V ← VERSION-SPACE-UPDATE(V, e)
end
return V

```

**function** VERSION-SPACE-UPDATE(*V*, *e*) **returns** an updated version space

```

V ← {h ∈ V : h is consistent with e}

```

For each inconsistent example, the hypothesis space is reduced. The procedure is *incremental* and also an illustration of the principle of *least commitment*.

↪ compact representation of the hypothesis space ?

41

## Updating the boundary sets (1)

Let  $G = \{G_1, \dots, G_n\}$  and  $S = \{S_1, \dots, S_n\}$ .

Let  $X$  be a new example

- $X$  is a false positive for  $S_i$ .  
 ↪  $S_i$  is too general :  $S' = S \setminus \{S_i\}$
- $X$  is a false negative for  $S_i$ .  
 ↪  $S_i$  too specific : generalise  $S_i$ .
- $X$  is a false positive for  $G_i$ .  
 ↪  $G_i$  too general : specialise  $G_i$ .
- $X$  is a false negative for  $G_i$ .  
 ↪  $G_i$  too specific :  $G' = G \setminus \{G_i\}$

43

## Boundary Set

Generalisation and specialization induce a *partial order* on the set of hypotheses.

The hypothesis space can be characterized by

- the (*most general*) hypotheses G-Set and
- the (*most specific*) hypotheses S-Set

**Theorem 2** A hypothesis  $H$  is consistent iff  $H$  is a generalization of an element in  $S$  and a specialization of an element in  $G$ .

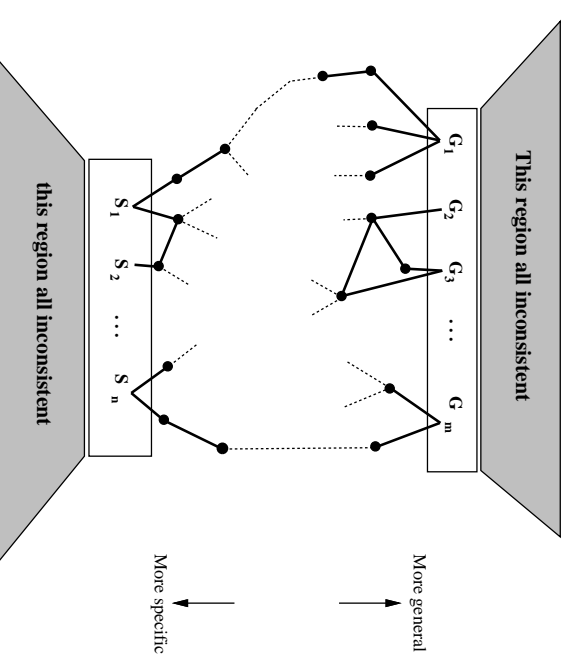
G-Set = {TRUE}

S-Set = {FALSE}

TRUE and FALSE are boolean functions (constants).

42

## Updating the Boundary Sets (2)



44

## Candidate Elimination Algorithm

---

$G$  = maximally general hypotheses in  $H$

$S$  = maximally specific hypotheses in  $H$

For each training example  $d$ , do

- If  $d$  is a positive example
  - Remove from  $G$  any hypothesis inconsistent with  $d$
  - For each hypothesis  $s$  in  $S$  that is not consistent with  $d$ 
    - \* Remove  $s$  from  $S$
    - \* Add to  $S$  all minimal generalizations  $h$  of  $s$  such that  $h$  is consistent with  $d$ , and some member of  $G$  is more general than  $h$
  - Remove from  $S$  any hypothesis that is more general than another hypothesis in  $S$

45

## Illustration

---

Sky	Temp	Humid	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Warm	Same	Yes
Sunny	Warm	High	Warm	Same	Yes
Rainy	Cold	High	Warm	Change	No
Sunny	Warm	High	Cool	Change	Yes

$S_0 = \{false\}; G_0 = \{true\}.$

$S_1 = \{Skyl(r, sunny) \wedge Temp(r, warm) \wedge Humid(r, normal) \wedge$

$Water(r, warm) \wedge Forecst(r, same)\};$

$G_1 = \{true\};$

$S_2 = \{Skyl(r, sunny) \wedge Temp(r, warm) \wedge Water(r, warm) \wedge Forecst(r, same)\};$

$G_2 = \{true\};$

$S_3 = \{Skyl(r, sunny) \wedge Temp(r, warm) \wedge Water(r, warm) \wedge Forecst(r, same)\};$

$G_3 = \{Skyl(r, sunny); Temp(r, warm); Forecst(r, same)\};$

47

## Candidate Elimination Algorithm (ctd.)

---

- If  $d$  is a negative example
  - Remove from  $S$  any hypothesis inconsistent with  $d$
  - For each hypothesis  $g$  in  $G$  that is not consistent with  $d$ 
    - \* Remove  $g$  from  $G$
    - \* Add to  $G$  all minimal specializations  $h$  of  $g$  such that  $h$  is consistent with  $d$ , and some member of  $S$  is more specific than  $h$
  - Remove from  $G$  any hypothesis that is less general than another hypothesis in  $G$

46

$S_4 = \{Skyl(r, sunny) \wedge Temp(r, warm)\};$

$G_4 = \{Skyl(r, sunny); Temp(r, warm)\};$

48

## Termination (1)

3 Possibilities:

1. Exactly one remaining hypothesis.
  - ↪ Solution (Convergence)
2. The hypothesis space collapses ( $S$  or  $G$  empty).
  - ↪ no consistent hypothesis/solution within the space
3. A set of remaining hypotheses.
  - ↪ Result is their disjunction. (Classification of an example might be done by a majority vote among the remaining hypotheses).

49

## Why Learning works

How to decide that  $h$  closely approximates  $f$ , when  $f$  is unknown ?

↪ *Probably Approximately Correct*

*Stationarity* as assumption in PAC-Learning: Training and testset are selected using the same (unknown) probability distribution.

How many examples do we need ?

- $X$  Set of examples
- $D$  Distribution
- $H$  Hypothesis space ( $f \in H$ )
- $m$  Number of example in training set

$$error(h) = P(h(x) \neq f(x)) \leq \epsilon$$

51

## Termination (2)

Problems:

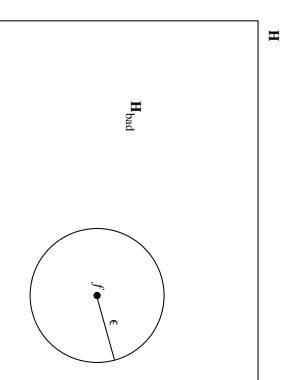
- If the data are noisy, then the version space is likely to collapse.
- If all forms of disjunctions are allowed, then  $S$  contains a unique element, the disjunction of all positives.

50

## PAC Learning

A hypothesis  $h$  is *probably approximately correct* iff  $error(h) \leq \epsilon$ .

To prove: after training with  $m$  examples every consistent hypothesis is approximately correct with high probability.



What is the probability that a false hypothesis  $h_b \in H_{train}$  is consistent with the first  $m$  examples ?

52

## Sample Complexity

Assumption:  $error(h_b) > \epsilon. \Rightarrow$

$$\begin{aligned} P(h_b \text{ cons. with 1 example}) &\leq (1 - \epsilon) \\ P(h_b \text{ cons. with } m \text{ examples}) &\leq (1 - \epsilon)^m \\ P(H_{bad} \text{ contains consistent hypothesis } h) &\leq |H_{bad}|(1 - \epsilon)^m \end{aligned}$$

In all cases,  $|H_{bad}| \leq |H|$  and therefore

$$\leq |H|(1 - \epsilon)^m$$

Requirement:

$$|H|(1 - \epsilon)^m < \delta$$

because  $\ln(1 - \epsilon) \leq -\epsilon$  this holds when

$$m \geq \frac{1}{\epsilon} (\ln \frac{1}{\delta} + \ln |H|)$$

**Sample complexity:** number of needed examples as a function of  $\epsilon$  and  $\delta$ .

53

## The estimate

$$\begin{aligned} |H|(1 - \epsilon)^m &\leq \delta \\ \ln |H|(1 - \epsilon)^m &\leq \ln \delta \\ \ln |H| + m \ln(1 - \epsilon) &\leq \ln \delta \end{aligned}$$

Because  $\ln(1 + \alpha) \leq \alpha$  and with  $\alpha = -\epsilon$  we obtain  $\ln(1 - \epsilon) \leq -\epsilon$ .

$$\begin{aligned} \ln |H| - m\epsilon &\leq \ln \delta \\ \frac{1}{\epsilon} \ln |H| - m &\leq \frac{1}{\epsilon} \ln \delta \\ \frac{1}{\epsilon} \ln |H| &\leq m + \frac{1}{\epsilon} \ln \delta \\ \frac{1}{\epsilon} (\ln |H| - \ln \delta) &\leq m \end{aligned}$$

↪ For boolean functions  $m$  grows exponentially because  $H$  grows double exponentially.

Dilemma: if one does not restrict the hypothesis space one cannot learn. If one restricts it, one might eliminate the true target function from the space.

55

## Sample Complexity (2)

Sample complexity is a measure to estimate the difficulty of training .

If a hypothesis is consistent with  $m \geq \frac{1}{\epsilon} (\ln \frac{1}{\delta} + \ln |H|)$  examples, then it is highly probable ( $1 - \delta$ ) that the error is less than  $\epsilon$ .

Problem: Estimate the size of the hypothesis space

**Illustration:** boolean functions

Number of boolean functions with  $n$  attributes is  $H = 2^{2^n}$ .

The sample complexity grows with  $2^n$ .

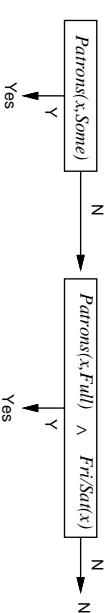
Because  $2^n$  is exactly the number of possible examples, there is no learning algorithm that performs better than a look up table and that is consistent with all known examples.

54

## Learning decision lists

As compared to decision trees

- the total structure is simpler
- single tests are more complex



This corresponds to the hypothesis

$$H_4 : \forall x \quad WillWait(x) \Leftrightarrow Patrons(x, some) \vee [Patrons(x, full) \wedge Fri/Sat(x)]$$

If tests of arbitrary length are allowed, then one can represent any boolean function.

**k-DL:** Language with tests of length  $\leq k$ .

$$\mathbf{k-DT} \subseteq \mathbf{k-DL}$$

56

## Lernability of k-DL

**function** DECISION-LIST-LEARNING(*examples*) **returns** a decision list, *No* or failure

**if** *examples* is empty **then return** the value *No*

$t \leftarrow$  a test that matches a nonempty subset *examples<sub>t</sub>* of *examples*

such that the members of *examples<sub>t</sub>* are all positive or all negative

**if** there is no such  $t$  **then return** failure

**if** the examples in *examples<sub>t</sub>* are positive **then**  $o \leftarrow$  *Yes*

**else**  $o \leftarrow$  *No*

**return** a decision list with initial test  $t$  and outcome  $o$

and remaining elements given by DECISION-LIST-LEARNING(*examples* – *examples<sub>t</sub>*)

$$\begin{aligned}
 |k\text{-DL}(n)| &\leq 3^{|Conj(n,k)|} \times |Conj(n,k)|! && \text{(Yes,No,no-test, all Permutations)} \\
 |Conj(n,k)| &= \sum_{i=0}^k \binom{2n}{i} && \text{(Combination without repeating pos/neg Attribut)} \\
 &= O(n^k) \\
 |k\text{-DL}(n)| &= 2^{\mathcal{O}(n^k \text{ld}(n^k))} && \text{(with Euler Sum)} \\
 m &\geq \frac{1}{\epsilon} \left( m \frac{1}{\delta} + O(n^k \text{ld}(n^k)) \right)
 \end{aligned}$$

57

## Summary

Inductive learning is concerned with the learning of a representation of a function on the basis of examples.

- **Decision trees** represent boolean functions
- **Current-best Hypothesis** is a method to induce logical theories
- **Version Space/Candidate Elimination** is a method to learn logical theories, it is based on a compact representation of the version space.
- **PAC Learning** is considered with the complexity of the learning.
- **Decision lists** can be learned easily

58