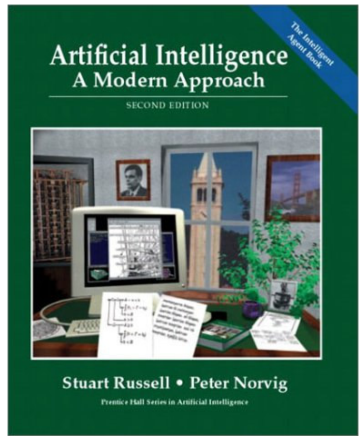
Inductive Learning (continued)

Chapter 19



Slides for Ch. 19 by J.C. Latombe

Learning logical descriptions

- The process of constructing a decision tree can be seen as searching the **hypothesis space H.** The goal is to construct an hypothesis *H* that explains the data in the training set.
- The hypothesis *H* is a logical description of the form:

$$H: \quad H_1 \vee H_2 \vee \ldots \vee H_n$$

$$H_i: \quad \forall x \; Q(x) \iff C_i(x)$$

where $Q(x)$ is the goal predicate and $C_i(x)$ are candidate definitions.

Hypotheses and types of examples

- Let X_i be an example, and $D_i(X_i)$ its logical description, and $Q(X)$ the current hypothesis:
  - X_i is a <u>positive</u> example iff $D_i(X_i)$ and $Q(X_i)$ are both true.
  - X_i is a <u>negative</u> example iff $D_i(X_i)$ and $Q(X_i)$ are both false.
  - X_i is a <u>false positive</u> example iff $Q(X_i)$ is false and $D_i(X_i)$ is true.
  - X_i is a <u>false negative</u> example iff $Q(X_i)$ is true and $D_i(X_i)$ is false.

Dealing with inconsistent examples

- If an example is false positive or false negative, the hypothesis must be revised by eliminating the inconsistent hypothesis clause:
  - if example X_i described by sentence I_i is inconsistent with H_i in $H_1 \vee H_2 \vee \ldots \vee H_n$, then eliminate H_i from H.
- The learning process can be seen as a process of eliminating hypotheses that are inconsistent with the examples. The question is how to do this efficiently.

Current-best-hypothesis search

- Incrementally build the hypothesis *H*, adjusting it as new examples arrive:
  - if new example is consistent (true positive or true negative) with *H*, do nothing.
  - if **<u>false negative</u>**, **<u>generalize</u>** *H* to *H'* by extending *H* to *H'* to make the example true.
  - if **<u>false positive</u>**, **<u>specialize</u>** *H* to *H'* by restricting *H* to *H'* to make the example false in *H'*.
  - in both cases, check consistency of all previous examples!

Specialization and generalization

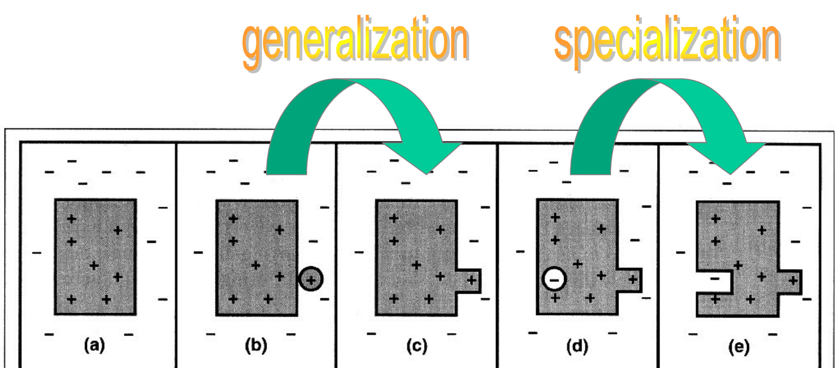


Figure 18.10 (a) A consistent hypothesis. (b) A false negative. (c) The hypothesis is generalized. (d) A false positive. (e) The hypothesis is specialized.

+ indicates positive examples
– indicates negative examples

Circled + and – indicates the example being added

Generalization and specialization

- **Relations:**
  - a is a generalization of $(a \wedge b)$ $(a \wedge b) \Rightarrow a$
  - $(a \wedge b)$ is a specialization of a $(a \wedge b) \Rightarrow a$
  - a is a specialization of $(a \vee b)$ $a \Rightarrow (a \vee b)$
  - $(a \vee b)$ is a generalization of a $a \Rightarrow (a \vee b)$
  - Note: b can also be negated $(a \wedge \sim b) \Rightarrow a$
- Specialization and generalization are symmetric, transitive.
- Might have several choices for each operation

Current best learning algorithm

```
function Current-Best-Learning(examples) returns hypothesis H
   H := hypothesis consistent with first example
  for each  remaining example e in  examples do
     if  e is false positive for H then
          H := choose a specialization of H consistent with examples
    else if e is false negative for H then
          H := choose a generalization of H consistent with examples
    if no consistent generalization/specialization found then fail
   end
  return H
```

Note: choose operations are nondeterministic
and indicate backtracking points.

Version spaces

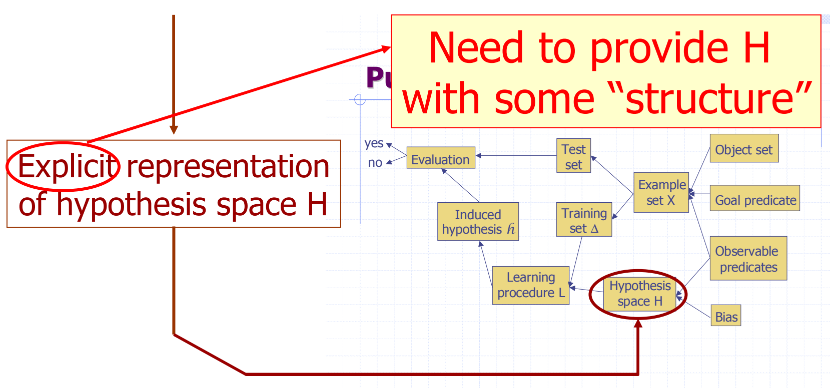
- READING:
  Russell & Norvig, 19.1
  alt: Mitchell, *Machine Learning*, Ch. 2 (through section 2.5)

- Hypotheses are represented by a set of logical sentences.
- Incremental construction of hypothesis.
- Prior "domain" knowledge can be included/used.
- Enables using the full power of logical inference.

Version space slides adapted from Jean-Claude Latombe

Predicate-Learning Methods

- Decision tree
- Version space



Version Spaces

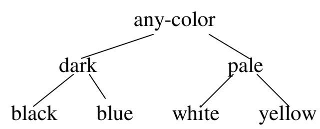
- The "version space" is the set of all hypotheses that are consistent with the training instances processed so far.
- An algorithm:
  - V := H ;; the version space V is ALL hypotheses H
  - For each example e:
    - Eliminate any member of V that disagrees with e
    - If V is empty, FAIL
  - Return V as the set of consistent hypotheses

Version Spaces: The Problem

- PROBLEM: V is huge!!
- Suppose you have N attributes, each with k possible values
- Suppose you allow a hypothesis to be any disjunction of instances
- There are k^N possible instances $\rightarrow$ $|H| = 2^{k^N}$
- If N=5 and k=2, $|H| = 2^{32}$!!

Version Spaces: The Tricks

- First Trick: Don't allow arbitrary disjunctions
  - Organize the feature values into a hierarchy of allowed disjunctions, e.g.



  - Now there are only 7 "abstract values" instead of 16 disjunctive combinations (e.g., "black or white" isn't allowed)
- Second Trick: Define a partial ordering on H ("general to specific") and only keep track of the upper bound and lower bound of the version space
- RESULT: An incremental, efficient algorithm!

Rewarded Card Example

(r=1) v … v (r=10) v (r=J) v (r=Q) v (r=K) ⇔ ANY-RANK(r)
(r=1) v … v (r=10) ⇔ NUM(r)
(r=J) v (r=Q) v (r=K) ⇔ FACE(r)
(s=♠) v (s=♣) v (s=♦) v (s=♥) ⇔ ANY-SUIT(s)
(s=♠) v (s=♣) ⇔ BLACK(s)
(s=♦) v (s=♥) ⇔ RED(s)

A hypothesis is any sentence of the form:

R(r) ∧ S(s)

where:
- R(r) is ANY-RANK(r), NUM(r), FACE(r), or (r=x)
- S(s) is ANY-SUIT(s), BLACK(s), RED(s), or (s=y)

Simplified Representation

For simplicity, we represent a concept by rs, with:
- r ∈ {a, n, f, 1, …, 10, j, q, k}
- s ∈ {a, b, r, ♣, ♠, ♦, ♥}

For example:
- n♠ represents:
  NUM(r) ∧ (s=♠)
- aa represents:
  ANY-RANK(r) ∧ ANY-SUIT(s)

Extension of a Hypothesis

The extension of a hypothesis h is the set of objects that satisfies h

Examples:
- The extension of f♠ is: {j♠, q♠, k♠}
- The extension of aa is the set of all cards

More General/Specific Relation

- Let h_1 and h_2 be two hypotheses in H
- h_1 is more general than h_2 iff the extension of h_1 is a proper superset of the extension of h_2

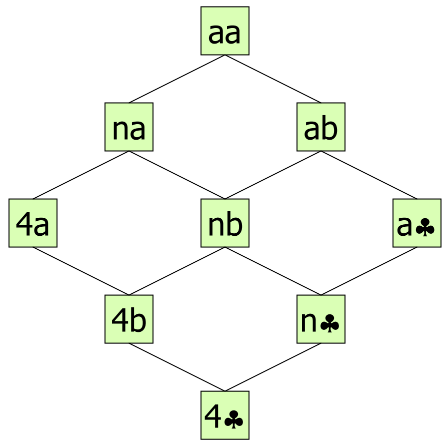
Examples:

- aa is more general than f♦
- f♥ is more general than q♥
- fr and nr are not comparable

More General/Specific Relation

- Let h_1 and h_2 be two hypotheses in H
- h_1 is more general than h_2 iff the extension of h_1 is a proper superset of the extension of h_2
- The inverse of the "more general" relation is the "more specific" relation
- The "more general" relation defines a partial ordering on the hypotheses in H

Example: Subset of Partial Order

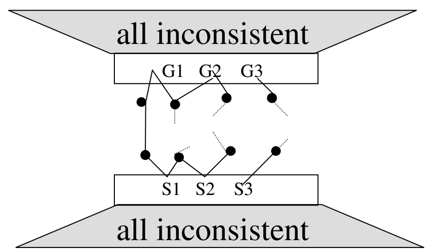


G-Boundary / S-Boundary of V

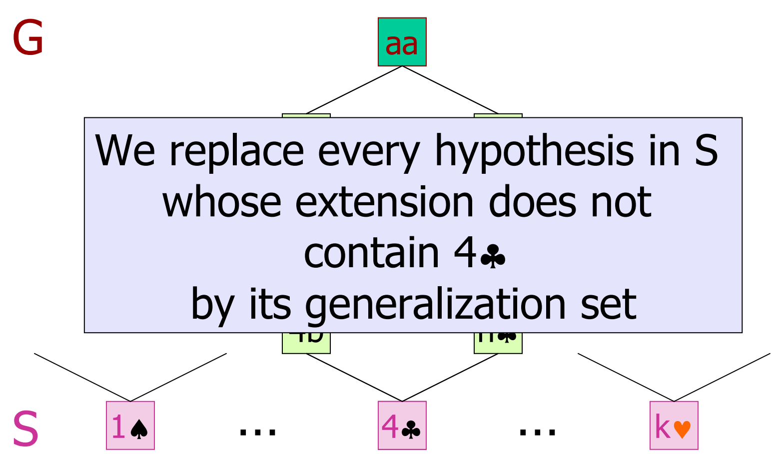
- A hypothesis in V is most general iff no hypothesis in V is more general
- G-boundary G of V: Set of most general hypotheses in V

G-Boundary / S-Boundary of V

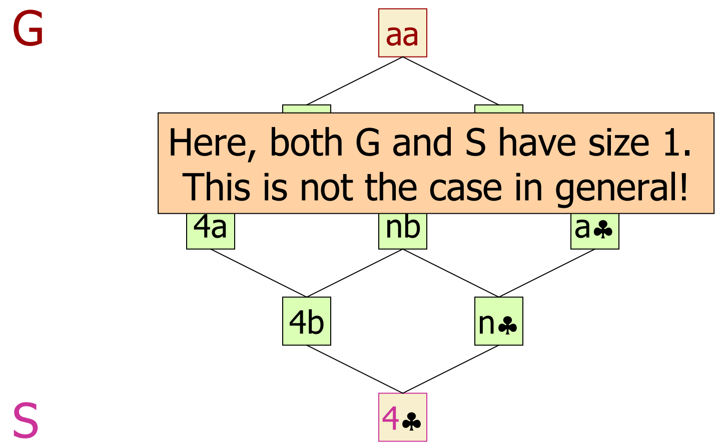
- A hypothesis in V is most general iff no hypothesis in V is more general
- G-boundary G of V: Set of most general hypotheses in V
- A hypothesis in V is most specific iff no hypothesis in V is more specific
- S-boundary S of V: Set of most specific hypotheses in V



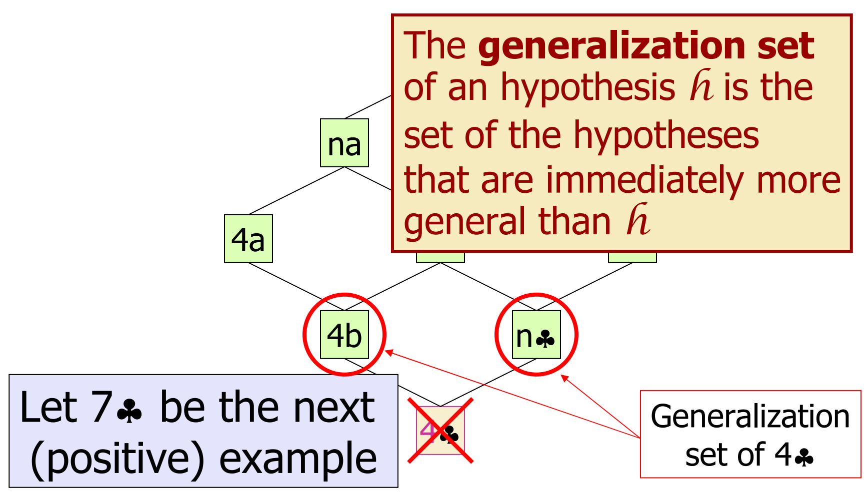
Example: G-/S-Boundaries of V



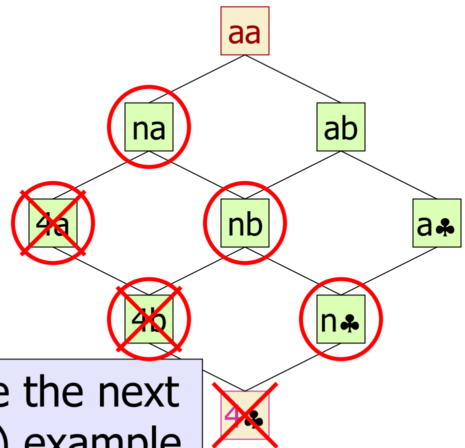
Example: G-/S-Boundaries of V



Example: G-/S-Boundaries of V

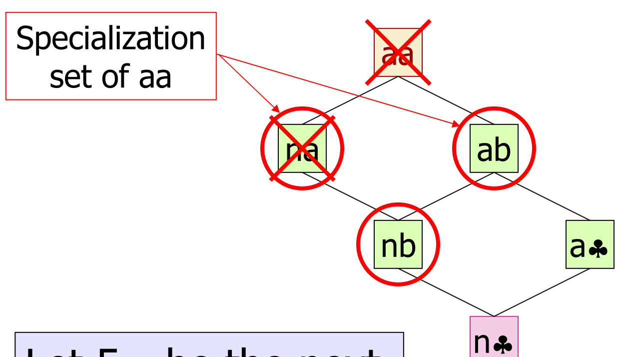


Example: G-/S-Boundaries of V



Let 7♣ be the next (positive) example

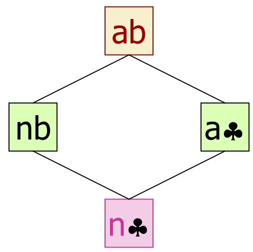
Example: G-/S-Boundaries of V



Let 5♥ be the next (negative) example

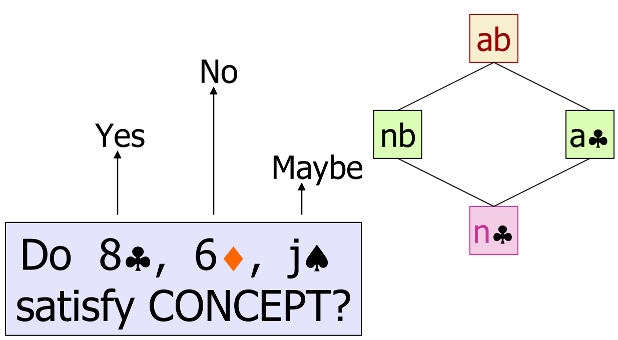
Example: G-/S-Boundaries of V

G and S, and all hypotheses in between
form exactly the version space

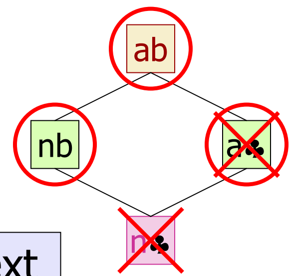


Example: G-/S-Boundaries of V

At this stage …

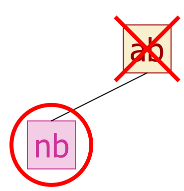


Example: G-/S-Boundaries of V



Let 2♠ be the next (positive) example

Example: G-/S-Boundaries of V



Let j♠ be the next (negative) example

Example: G-/S-Boundaries of V

+ 4♣ 7♣ 2♠
– 5♥ j♠

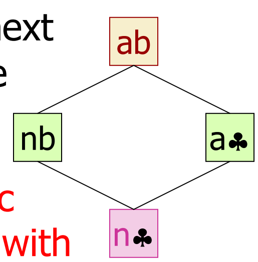
nb

NUM(r) ∧ BLACK(s)

Example: G-/S-Boundaries of V

Let us return to the version space …

… and let 8♣ be the next (negative) example

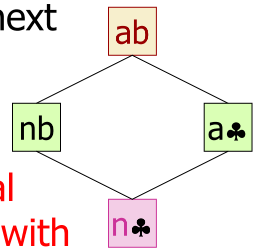


The only most specific hypothesis disagrees with this example, so no hypothesis in H agrees with all examples

Example: G-/S-Boundaries of V

Let us return to the version space …

… and let j♥ be the next (positive) example



The only most general hypothesis disagrees with this example, so no hypothesis in H agrees with all examples

Version Space Update

1. x ← new example
2. If x is positive then
   (G,S) ← POSITIVE-UPDATE(G,S,x)
3. Else
   (G,S) ← NEGATIVE-UPDATE(G,S,x)
4. If G or S is empty then return failure

POSITIVE-UPDATE(G,S,x)

1. Eliminate all hypotheses in G that do not agree with x

POSITIVE-UPDATE(G,S,x)

1. Eliminate all hypotheses in G that do not agree with x
2. Minimally generalize all hypotheses in S until they are consistent with x

Using the generalization sets of the hypotheses

POSITIVE-UPDATE(G,S,x)

1. Eliminate all hypotheses in G that do not agree with x
2. Minimally generalize all hypotheses in S until they are consistent with x
3. Remove from S every hypothesis that is neither more specific than nor equal to a hypothesis in G

This step was not needed in the card example

POSITIVE-UPDATE(G,S,x)

1. Eliminate all hypotheses in G that do not agree with x
2. Minimally generalize all hypotheses in S until they are consistent with x
3. Remove from S every hypothesis that is neither more specific than nor equal to a hypothesis in G
4. Remove from S every hypothesis that is more general than another hypothesis in S
5. Return (G,S)

NEGATIVE-UPDATE(G,S,x)

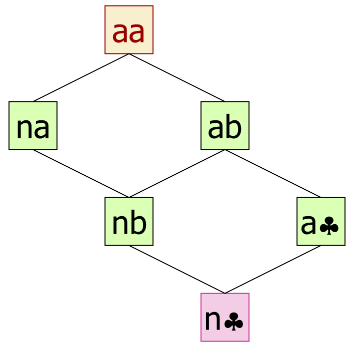
1. Eliminate all hypotheses in S that do agree with x
2. Minimally specialize all hypotheses in G until they are consistent with (exclude) x
3. Remove from G every hypothesis that is neither more general than nor equal to a hypothesis in S
4. Remove from G every hypothesis that is more specific than another hypothesis in G
5. Return (G,S)

Example-Selection Strategy

- Suppose that at each step the learning procedure has the possibility to select the object (card) of the next example
- Let it pick the object such that, whether the example is positive or not, it will eliminate one-half of the remaining hypotheses
- Then a single hypothesis will be isolated in O(log |H|) steps

Example

• 9♣?
• j♥?
• j♣?



Example-Selection Strategy

- Suppose that at each step the learning procedure has the possibility to select the object (card) of the next example
- Let it pick the object such that, whether the example is positive or not, it will eliminate one-half of the remaining hypotheses
- Then a single hypothesis will be isolated in O(log |H|) steps
- But picking the object that eliminates half the version space may be expensive

Noise

- If some examples are misclassified, the version space may collapse
- Possible solution:
  Maintain several G- and S-boundaries, e.g., consistent with all examples, all examples but one, etc…

VSL vs DTL

- Decision tree learning (DTL) is more efficient if all examples are given in advance; else, it may produce successive hypotheses, each poorly related to the previous one
- Version space learning (VSL) is incremental
- DTL can produce simplified hypotheses that do not agree with all examples
- DTL has been more widely used in practice