CS3244 Tutorial 1

Wu Zhaoxuan
wu.zhaoxuan@u.nus.edu

2022.1.28

Concept Learning

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Input **Output**

- A ***concept*** is a boolean-valued function over a set of **input** instances X
- Learned concept $Input \rightarrow Output$, or specifically $c : X \rightarrow \{0,1\}$

Core Definitions

- **Hypothesis space H**: the space that contains candidates for the target c.
  - Each hypothesis $h \in H$ is a conjunction of constraints on input attributes
  - e.g. $h = \langle Sunny, ?, ?, Strong, ?, Same \rangle$
- **"Satisfy"**: An input instance $x \in X$ **satisfies** a hypothesis $h \in H$ iff $h(x) = 1$.
- "**Consistency**": A hypothesis h is **consistent** with a set of training examples D iff $h(x) = c(x)$ for all $\langle x, c(x) \rangle \in D$.
  - "The hypothesis predicts the output correctly for all training examples"

BL 1

Prove Proposition 1 on page 14.

h is consistent with D iff every +ve training instance satisfies h and every –ve training instance does not satisfy h.

Solution

1. h is consistent with D iff $h(x) = c(x)$ for all $\langle x, c(x) \rangle \in D$.

2. h is consistent with D iff $h(x) = 1$ for all $\langle x, 1 \rangle \in D$ and $h(x) = 0$ for all $\langle x, 0 \rangle \in D$.

Note that you need to prove both directions for "iff".

TM 2.1

- Input instances X: Each instance $x \in X$ is represented by the following input attributes describing the day:
  - *Sky* (with possible values *Sunny*, *Cloudy*, and *Rainy*)
  - *AirTemp* (with values *Warm* and *Cold*)
  - *Humidity* (with values *Normal* and *High*)
  - *Wind* (with values *Strong* and *Weak*)
  - *Water* (with values *Warm* and *Cool*)
  - *Forecast* (with values *Same* and *Change*)

Solution

$$|X| = 3 \times 2 \times 2 \times 2 \times 2 \times 2 \times 3 = 288$$

$$|H| = 1 + 4 \times 3 \times 3 \times 3 \times 3 \times 3 \times 4 = 3889$$

$$|X'| = k|X|$$

$$|H'| = (k+1)(|H|-1) + 1 = |H|(k+1) - k$$

- New attribute: $WaterCurrent = \{Light, Moderate, Strong\}$
- Calculate number of possible input instances $|X|$, and the number of possible semantically distinct hypotheses $|H|$.

Find-S Algorithm

- h_j is **more general than or equal to** h_k (denoted by $h_j \geq_g h_k$) iff any input instance x that satisfies h_k also satisfies h_j.
  - $\forall x \in X \quad (h_k(x) = 1) \rightarrow (h_j(x) = 1)$
- h_j is **more general than** h_k (denoted by $h_j >_g h_k$) iff $h_j \geq_g h_k$ and $h_k \ngeq_g h_j$.
- h_j is **more specific than** h_k iff h_k is more general than h_j.

Find-S Algorithm

Idea. Start with most specific hypothesis. Whenever it wrongly classifies a +ve training example as –ve, "minimally" generalize it to satisfy its input instance.

1. Initialize h to most specific hypothesis in H
2. For each positive training instance x
    - For each attribute constraint a_i in h

      If x satisfies constraint a_i in h

      Then do nothing

      Else replace a_i in h by the next more general constraint that is satisfied by x
3. Output hypothesis h

Version Space

- The version space $VS_{H,D}$ w.r.t. hypothesis space H and training examples D, is the subset of hypotheses from H consistent with D:
  - $VS_{H,D} = \{h \in H \mid h \text{ is consistent with } D\}$
- List-then-Eliminate Algorithm
  - $VS \leftarrow$ a list containing every hypothesis in H
  - For each $\langle x, c(x) \rangle \in D$:
    - Remove from VS any h for which $h(x) \neq c(x)$
  - Output the list of hypotheses from VS

Version Space Boundaries

- The **general boundary** G of $VS_{H,D}$ is the set of maximally general members of H consistent with D:

$$G = \{g \in H \,|\, g \text{ consistent with } D \wedge (\neg \exists g' \in H \; g' >_g g \;\wedge\; g' \text{ consistent with } D)\}$$

- The **specific boundary** S of $VS_{H,D}$ is the set of maximally specific members of H consistent with D:

$$S = \{s \in H \,|\, s \text{ consistent with } D \wedge (\neg \exists s' \in H \; s >_g s' \;\wedge\; s' \text{ consistent with } D)\}$$

- Every member of the VS should lie between the boundaries

Candidate-Elimination Algorithm

1. $G \leftarrow$ maximally general hypotheses in H
2. $S \leftarrow$ maximally specific hypotheses in H
3. For each training example d
   - If d is a +ve example
     - Remove from G any hypothesis inconsistent with d
     - For each $s \in S$ not consistent with d
       - Remove s from S
       - Add to S all minimal generalizations h of s s.t.

         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

Candidate-Elimination Algorithm

- If d is a −ve example
  - Remove from S any hypothesis inconsistent with d
  - For each $g \in G$ not consistent with d
    - Remove g from G
    - Add to G all minimal specializations h of g s.t.

      h is consistent with d, and

      some member of S is more specific than h
    - Remove from G any hypothesis that is more specific than another hypothesis in G

Candidate-Elimination Algorithm

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- Init most general $G_0 = \{\langle ?, ?, ?, ?, ?, ? \rangle\}$
- Init most specific $S_0 = \{\langle \varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing \rangle\}$
- eg.4 satisfies $\langle ?, ?, ?, ?, ?, ? \rangle$ and YES, => consistent => $G_1 = G_0$
- eg.4 does not satisfy $\langle \varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing \rangle$ and YES, => inconsistent => Remove
  - $h = \langle Sunny, Warm, High, Strong, Cool, Change \rangle$ is the min. generalisation of $\langle \varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing \rangle$ and eg.4 satisfies h and YES => consistent => $S_1 = \{\langle Sunny, Warm, High, Strong, Cool, Change \rangle\}$

- Remove from G any hypothesis inconsistent with d
- For each $s \in S$ not consistent with d
  - Remove s from S
  - Add to S all minimal generalizations h of s s.t.
    h is consistent with d, and
    some member of G is more general than h

Candidate-Elimination Algorithm

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- $G_1 = \{\langle ?, ?, ?, ?, ?, ? \rangle\}$
- $S_1 = \{\langle Sunny, Warm, High, Strong, Cool, Change \rangle\}$
- eg.3 does not satisfy $\langle Sunny, Warm, High, Strong, Cool, Change \rangle$ and NO, => consistent => $S_2 = S_1$
- eg.3 satisfies $\langle ?, ?, ?, ?, ?, ? \rangle$ and NO, => inconsistent => Remove
  - $h = \langle Sunny, ?, ?, ?, ?, ? \rangle$ is a min. specialization of $\langle ?, ?, ?, ?, ?, ? \rangle$ and also $\langle Sunny, ?, ?, ?, ?, ? \rangle >_g \langle Sunny, WarmHigh, Strong, Cool, Change \rangle$
  - eg.3 does not satisfy h and NO => consistent => G_2 should contain h
- Similarly for $\langle ?, Warm, ?, ?, ?, ? \rangle$, $\langle ?, ?, ?, ?, Cool, ? \rangle$
  - $G_2 = \{\langle Sunny, ?, ?, ?, ?, ? \rangle, \langle ?, Warm, ?, ?, ?, ? \rangle, \langle ?, ?, ?, ?, Cool, ? \rangle\}$

- Remove from S any hypothesis inconsistent with d
- For each $g \in G$ not consistent with d
  - Remove g from G
  - Add to G all minimal specializations h of g s.t. h is consistent with d, and some member of S is more specific than h

Candidate-Elimination Algorithm

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- $G_2 = \{\langle Sunny, ?, ?, ?, ?, ?\rangle, \langle ?, Warm, ?, ?, ?, ?\rangle, \langle ?, ?, ?, ?, Cool, ?\rangle\}$
- $S_2 = \{\langle Sunny, Warm, High, Strong, Cool, Change\rangle\}$
- eg.2 is not consistent with $\langle ?, ?, ?, ?, Cool, ?\rangle$ => Remove
  - $G_3 = \{\langle Sunny, ?, ?, ?, ?, ?\rangle, \langle ?, Warm, ?, ?, ?, ?\rangle\}$
- eg.2 is inconsistent with $\langle Sunny, Warm, High, Strong, Cool, Change\rangle$ => Remove
  - $h = \langle Sunny, Warm, High, Strong, ?, ?\rangle$ is the min. generalisation of $\langle Sunny, Warm, High, Strong, Cool, Change\rangle$ and eg.2 satisfies h and YES => consistent, also, $\langle Sunny, ?, ?, ?, ?, ?\rangle \in G_3 >_g h$ =>
    $S_3 = \{\langle Sunny, Warm, High, Strong, ?, ?\rangle\}$

- Remove from G any hypothesis inconsistent with d
- For each $s \in S$ not consistent with d
  - Remove s from S
  - Add to S all minimal generalizations h of s s.t.
    h is consistent with d, and
    some member of G is more general than h

Candidate-Elimination Algorithm

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- Remove from G any hypothesis inconsistent with d
- For each $s \in S$ not consistent with d
  - Remove s from S
  - Add to S all minimal generalizations h of s s.t. h is consistent with d, and some member of G is more general than h

- $G_3 = \{\langle Sunny, ?, ?, ?, ?, ?\rangle, \langle ?, Warm, ?, ?, ?, ?\rangle\}$
- $S_3 = \{\langle Sunny, Warm, High, Strong, ?, ?\rangle\}$
- eg.1 is consistent with all hypotheses in G_3 => $G_4 = G_3$
- eg.1 is inconsistent with $\langle Sunny, Warm, High, Strong, ?, ?\rangle$ => Remove
  - $h = \langle Sunny, Warm, ?, Strong, ?, ?\rangle$ is the min. generalisation of $\langle Sunny, Warm, High, Strong, ?, ?\rangle$ and eg. satisfies h and YES => consistent, also, $\langle Sunny, ?, ?, ?, ?, ?\rangle \in G_4 >_g h$ => $S_4 = \{\langle Sunny, Warm, ?, Strong, ?, ?\rangle\}$

TM 2.2 (b)

Although the final version space will be the same regardless of the sequence of examples (why?), the sets S and G computed at intermediate stages will, of course, depend on this sequence. Can you come up with ideas of heuristics for ordering the training examples to minimize the sum of the sizes of these intermediate S and G sets for the H used in the $EnjoySport$ task?

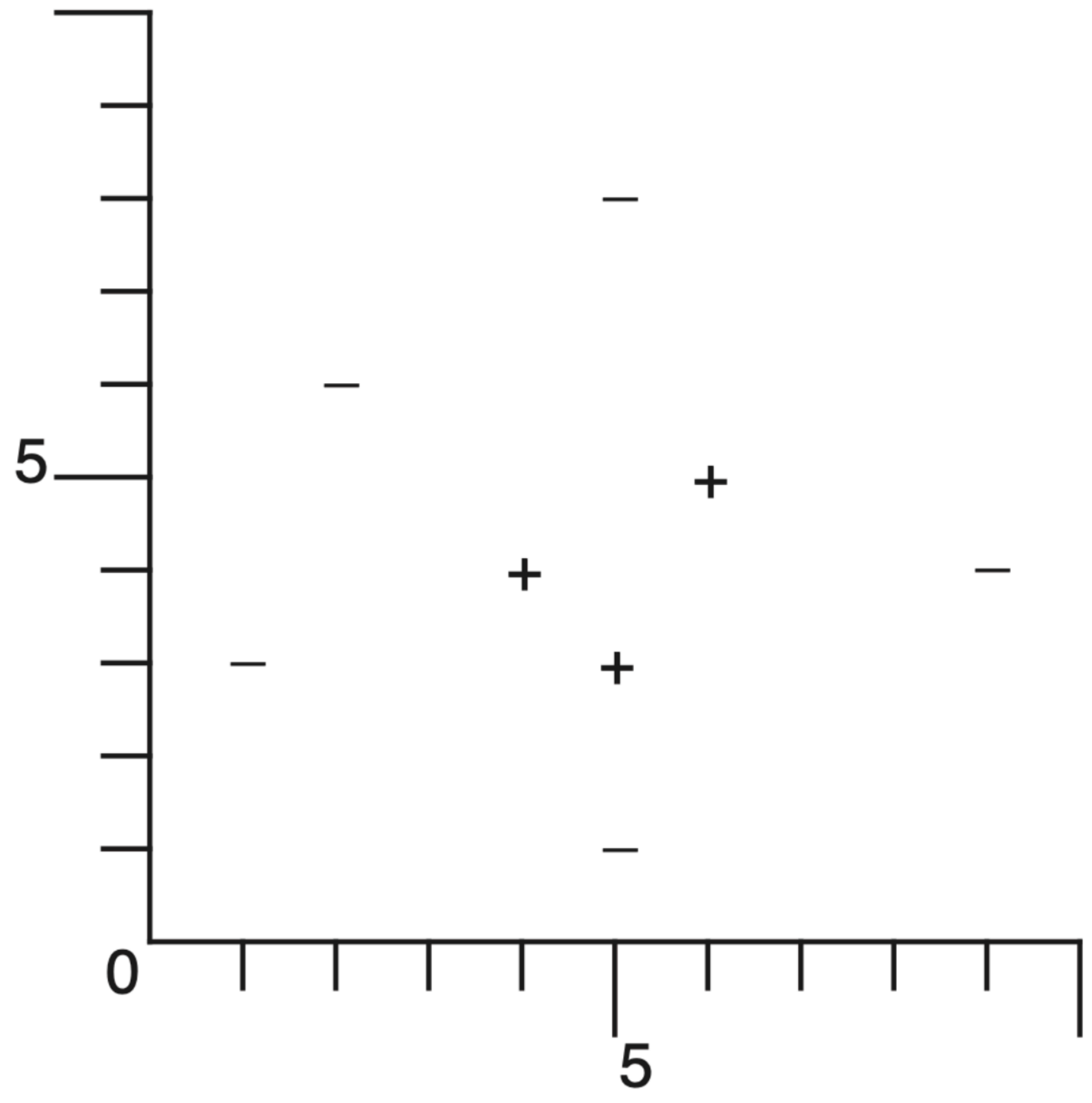
Solution

- The definition of S and G are **independent** of the sequence of the examples
- Unlike Find-S, the algorithm is designed to find all maximally general and specific hypotheses, hence the final version space.
- Heuristic 1: Select all +ve examples first, followed by the -ve examples
  - The set S will always maintain a size of one maximally specific hypothesis in each iteration
  - After all +ve examples are trained, the resulting most specific S can reduce the number of min. specialization of g added to G during -ve examples training.

TM 2.4

Consider the instance space consisting of integer points in the x, y plane and the set of hypotheses H consisting of rectangles. More precisely, hypotheses are of the form $a \leq x \leq b$, $c \leq y \leq d$, where a, b, c and d can be any integers. Consider the version space with respect to the set of positive (+) and negative (–) training examples shown below.

Draw out S and G.

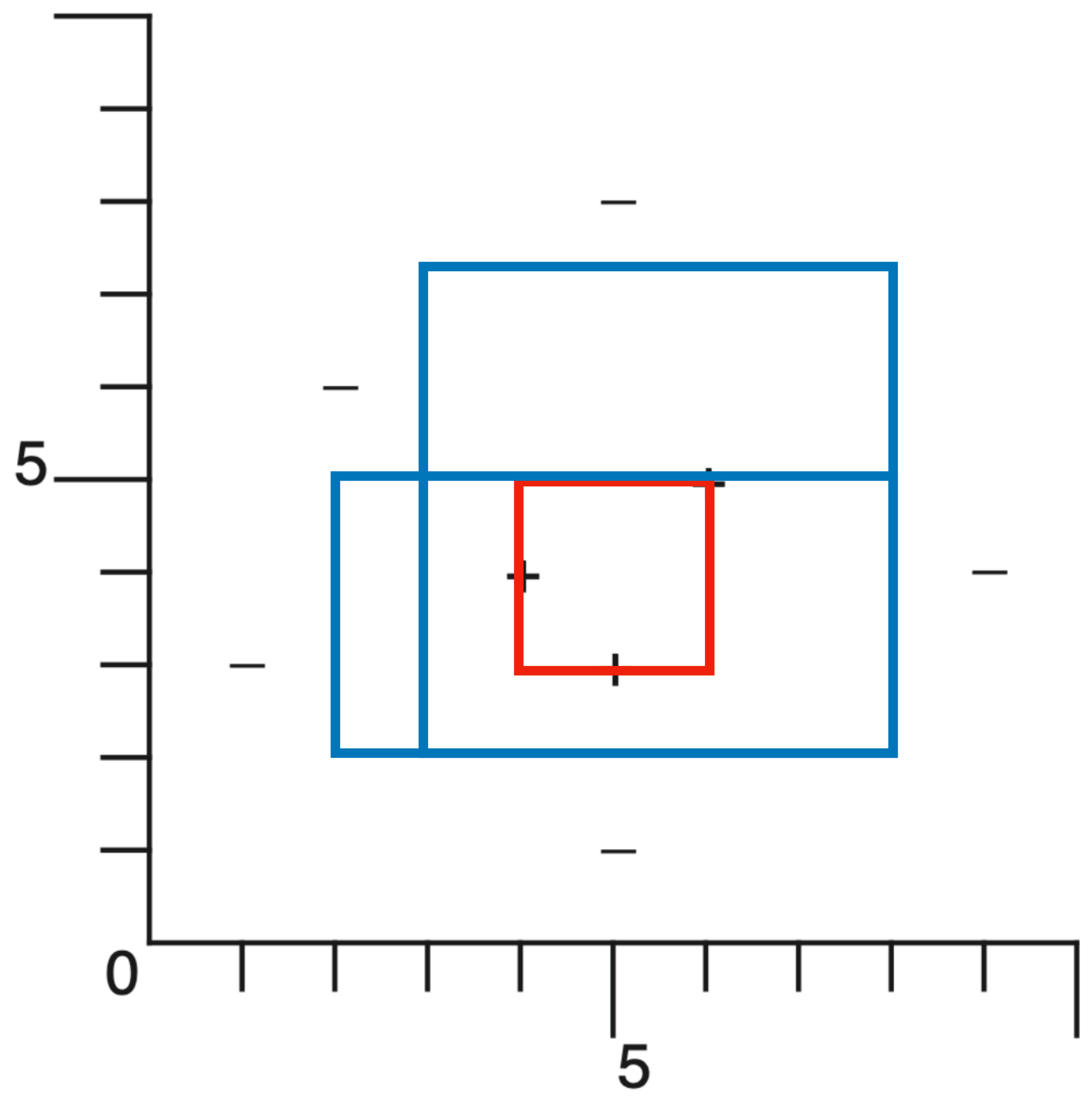


TM 2.4

Solution

(a) $S = \{(4 \leq x \leq 6, 3 \leq y \leq 5)\}$

(b) $G = \{(2 \leq x \leq 8, 2 \leq y \leq 5), (3 \leq x \leq 8, 2 \leq y \leq 7)\}$



TM 2.4

(c) Suppose that the learner may now suggest a new x, y instance and ask the trainer for its classification. Suggest a query guaranteed to reduce the size of the version space, regardless of how the trainer classifies it. Suggest one that will not.

Solution

Any query inside G but outside S will reduce the VS.

Any query inside S or outside G will never reduce the VS.

TM 2.4

Now assume that you are a teacher attempting to teach a particular target concept (e.g., $3 \leq x \leq 5$, $2 \leq y \leq 9$). What is the smallest number of training examples you can provide so that the Candidate-Elimination algorithm will perfectly learn the target concept?

Solution

To learn the concept through Candidate-Elimination, consider two parts: learning S and G both equal to the target concept.

For S, two points at the corner of the rectangle is required.

For G, four points are required: each on one of the side of the rectangle.

Thus, the total number of points required is 6.

Thank you!

- Any questions?