Concept Learning

- Positive and negative examples
- General-to-specific ordering (partial order)
- Inductive learning
- Hypothesis space
- Version space
- Inductive bias

Concept

- Classification of things into discrete categories, or discrete (many times binary) decisions.
- Description of a small subset within a larger set.
- Boolean-valued function from the set to $\{True, False\}$: e.g., an element in the set is mapped to true if bird, and false if not.
- Problem: **automatically infer** the **general definition** of some concept, given **labeled examples**

Concept Learning

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

- Inferring a **boolean-valued function** from training examples consisting of input and output.
- Example: "Days on which my friend enjoys his favorite water sport."
- Task: given a set of examples with **attributes** and **decisions**, want to **predict** the decision for an arbitrary date.

EnjoySport domain

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

- $Sky \in \{Sunny, Cloudy, Rainy\}$
- $AirTemp \in \{Warm, Cold\}$
- $Humidity \in \{Normal, High\}$
- $Wind \in \{Strong, Weak\}$
- $Water \in \{Warm, Cool\}$
- $Forecast \in \{Same, Change\}$
- $EnjoySport \in \{Yes, No\}$

Hypothesis: Conjunction of Constraints

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

- Need to decide the form of hypothesis: conjunction of constraints may be one.
- A six-element vector, where each element can be:
  - Attribute value,
  - "?" (any value is allowed), or
  - "$\emptyset$" (no value is acceptable).
- Sample $x = \langle Sunny, Cold, High, Warm, Same \rangle$ would **satisfy** the hypothesis $h = \langle ?, Cold, High, ?, ?, ? \rangle$: that is, $h(x) = 1$.

Conjunctive Hypothesis

- Given hypothesis $h = \langle h_1, h_2, h_3, h_4, h_5, h_6 \rangle$, and an example $x = \langle x_1, x_2, x_3, x_4, x_5, x_6 \rangle$:

  $$h(x) = [(h_1 = x_1) \vee (h_1 = ?)] \wedge [(h_2 = x_2) \vee (h_2 = ?)] \wedge ...$$

- Note that if any h_i is $\emptyset$, then $h(x) = 0$ for all x.
- Example x satisfies hypothesis h if $h(x) = 1$, regardless of the decision.

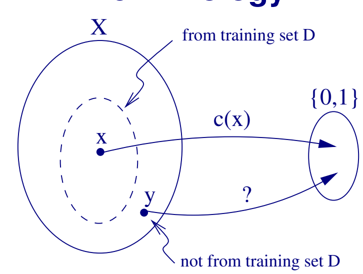
Examples

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

All examples above would:

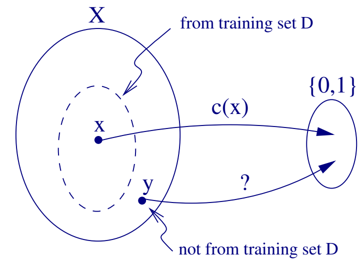
- satisfy the hypothesis $\langle Sunny, Warm, ?, ?, ?, ? \rangle$, except for example 3.
- satisfy the hypothesis $\langle ?, ?, ?, ?, ?, ? \rangle$, regardless of the negative example ($EnjoySport = No$).
- not satisfy the hypothesis $\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$, or any hypothesis containing $\emptyset$.

Terminology



- set of **instances** X: the set from which examples are drawn from. Note that training my happen on a subset of this set.
- **target concept**: concept or function to be learned, given X, a set of instances: $c : X \rightarrow \{0, 1\}$ $c(X) = 1$ if $EnjoySport =$"Yes", and
  $c(X) = 0$ if $EnjoySport =$"No".
- set of **training examples** D: examples x drawn from X, together with the concept value $c(x) \in \{0, 1\}$: $\langle x, c(x) \rangle$
  - Positive example: $c(x) = 1$, Negative example: $c(x) = 0$

Terminology (Cont'd)



- Goal: find $h(x) = c(x)$ **for all** $x \in X$ (i.e., includes y in the figure above).
- What if $domain(D) = X$?
  - Regardless of $c(x)$, learning is trivial: just keep a look-up table for all possible x.
  - $\langle ?, ?, ?, ..., ? \rangle$ may not work, because there may be **negative examples**
  - Note: $D = \{\langle x, c(x)\rangle | x \text{ is a training sample}\}$

Inductive Learning Hypothesis

- Hypothesis h derived from the training set can only fit the given data.
- That is, output hypothesis fits the target concept over the training set, at best.
- Assumption: hypothesis that fits the observed data may also fit unseen data.
- **The inductive learning hypothesis:** "Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples."

Concept Learning as Search

- Given a set of hypotheses $h_i(x)$, can we find one that holds $h_i(x) = c(x)$? This is basically a search problem.
- Choice of representation for hypotheses determine the **hypothesis space**:
  - In $EnjoySport$, $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$ possible combination of attributes (each attribute can take on 3, 2, 2, ... distinct values).
  - The choice of a **conjunction of constraints hypothesis** gives $5 \times 4 \times 4 \times 4 \times 4 \times 4 = 5120$ **syntactically distinct** hypotheses (attribute values plus "?" and "$\emptyset$").
- Only $1 + 4 \times 3 \times 3 \times 3 \times 3 \times 3 = 973$ **semantically distinct** hypotheses (ignore all that contain $\emptyset$, except for $\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$.

General-to-Specific Ordering

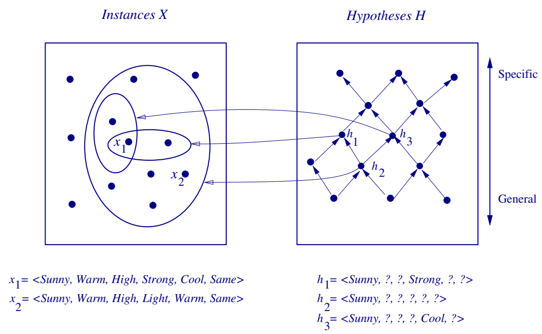
- A **useful structure** exists in concept learning problem: a general-to-specific ordering, which allows you to conduct **efficient search** even in an infinite hypothesis space.
- Example: h_2 is more general than h_1
  - $h_1 = \langle Sunny, ?, ?, Strong, ?, ? \rangle$
  - $h_2 = \langle Sunny, ?, ?, ?, ?, ? \rangle$

  Definition:
- Let h_j and h_k be boolean functions defined over X. Then, h_j is **more_general_than_or_equal_to** h_k (written $h_j \geq_g h_k$) iff

$$(\forall x \in X)\,[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$

- $h_j >_g h_k$: strictly more_general_than
- more specific, etc.

General-to-Specific Ordering



- Within the boundary: positive; Outside the boundary: negative.
- More general: more examples classified as positive.
- $\geq_g$ ($\longrightarrow$) forms a **partial order** over the hypotheses (reflexive, antisymmetric, and transitive).
- Some pairs may have no $\geq_g$ relation at all.

Find-S: Find Maximally Specific Hypothesis

- Initialize h to the most specific hypothesis in H.
- For each positive training instance x
  - For each attribute constraint a_i in h,
    If the constraint is satisfied by x_i
    Then do nothing
    Else replace a_i in h by the next more general constraint that is satisfied by x.
- Output hypothesis h.

Find-S: Example

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

Specific $\emptyset \longrightarrow \{Sunny, Warm, etc.\} \longrightarrow$ to General.

Example:

- Begin with $h = \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$.
- Take example 1 from the table above: let's call this x.
- $h(x) = 0$, so replace each a_i in h with more general ones so that h is satisfied by x (i.e. $h(x) = 1$).

  $$h \leftarrow \langle Sunny, Warm, Normal, Strong, Warm, Same \rangle$$

Find-S: Example

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

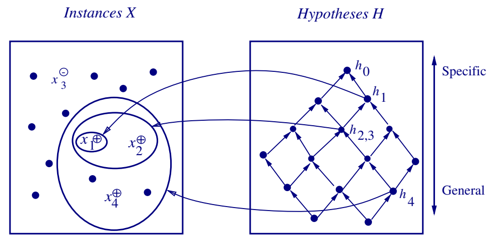
Start with
$\langle Sunny, Warm, Normal, Strong, Warm, Same \rangle$:

- Take example 2 from the table above (call it x).
- $h(x) = 0$, so generalize conflicting constraints.

  $$h \leftarrow \langle Sunny, Warm, ?, Strong, Warm, Same \rangle$$

- Take example 3: nothing happens, because it's a **negative example** (current h is **already consistent**: $h(x_3) = c(x_3)$)
- Take example 4: $h \leftarrow \langle Sunny, Warm, ?, Strong, ?, ? \rangle$

Find-S: Example



h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

x_1 = <Sunny Warm Normal Strong Warm Same>, + | h_1 = <Sunny Warm Normal Strong Warm Same>

x_2 = <Sunny Warm High Strong Warm Same>, + | h_2 = <Sunny Warm ? Strong Warm Same>

x_3 = <Rainy Cold High Strong Warm Change>, - | h_3 = <Sunny Warm ? Strong Warm Same>

x_4 = <Sunny Warm High Strong Cool Change>, + | h_4 = <Sunny Warm ? Strong ? ? >

- Use of the **more_general_than** partial ordering to organize the search: move from hypothesis to hypothesis, from specific to general.
- Each positive example can potentially change the current hypothesis, from specific (top) to general (bottom).

Find-S: Properties and Limitations

Generalize only as far as necessary to account for the examples (the most specific hypothesis that is consistent with the examples).

Limitations: In general,

- There can be other consistent hypotheses.
- Why only the most specific hypothesis?
- Can't tell whether the training data is inconsistent.
- Depending on H, there might be several maximally specific hypotheses.

Version Space and Candidate-Elimination Algorithm

- The set of all hypotheses consistent with the examples is called the **version space**. (Note: h does not need to be maximally specific.)
- Thus, the version space is a **subset** of the hypothesis space H.
- How can we represent the version space?: One way is to enumerate all hypotheses consistent with the examples.
- **Candidate-Elimination algorithm** can output a description of the version space, **without enumerating all the hypotheses.**

Definition of Terms

- **Consistent:** A hypothesis is **consistent** with a set of training examples D iff $h(x) = c(x)$ for all $\langle x, c(x)\rangle \in D$

  $$Consistent(h, D) \equiv (\forall \langle x, c(x)\rangle \in D) h(x) = c(x)$$

  Cf. x **satisfies** h if $h(x) = 1$, whether or not $c(x) = 1$.

- **Version space:** The version space $VS_{H,D}$, with respect to hypothesis space H and training examples D, is:

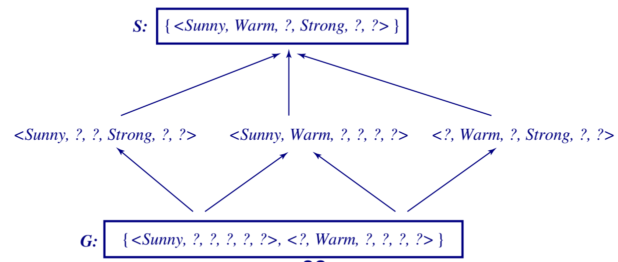
  $$VS_{H,D} \equiv \{h \in H | Consistent(h, D)\}$$

List-Then-Eliminate Algorithm

- Algorithm
  - $VS \leftarrow H$
  - For each $\langle x, c(x) \rangle \in D$
    remove from VS any hypothesis h for which $h(x) \neq c(x)$
  - Output the list of hypotheses in VS.
- Evaluation
  - Problem: Can't cope with infinite hypothesis space. Need to enumerate all.
  - Merit: guaranteed to output **all** consistent hypotheses, if H is finite.

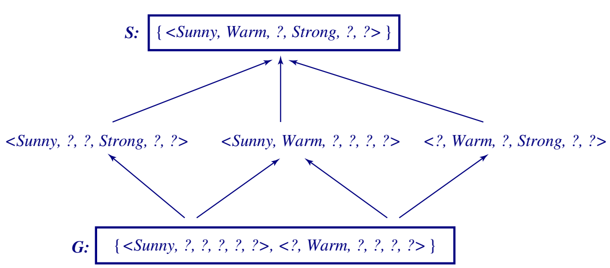
Compact Representation of VS

- Just keep two small hypothesis sets G and S to **mark the boundary** of the version space.
- G: set of most general hypotheses consistent with D
  - Any more general then become **inconsistent**.
- S: set of most specific hypotheses consistent with D.
  - Any more specific then become **inconsistent**.



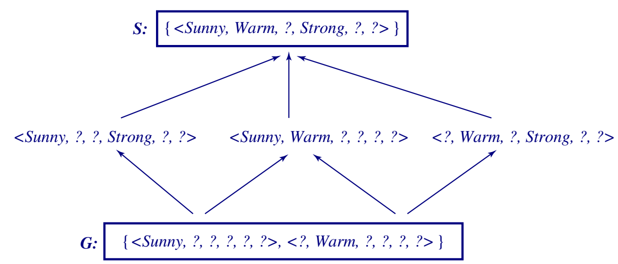
G and S

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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



Note: There can be more than one h in G.

Property of G and S



- Given the two sets G and S, we can easily find all members of the version space by generating the hypotheses that **lie between** G **and** S in the general-to-specific partial ordering.

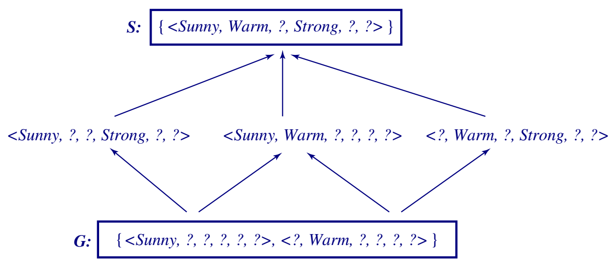
Formal Definition of G and S

- **General Boundary** G, with respect to H and D:

$$G \equiv \{g \in H | Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)]\}$$

- **Specific Boundary** S:

$$S \equiv \{s \in H | Consistent(s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)]\}$$



Version Space Representation Theorem

Let X be an arbitrary set of instances and let H be a set of boolean-valued hypotheses defined over X. Let $c : X \rightarrow \{0, 1\}$ be an arbitrary target concept defined over X, and let D be an arbitrary set of training examples $\{\langle x, c(x) \rangle\}$. For all X, H, c, and D such that S and G are well defined,

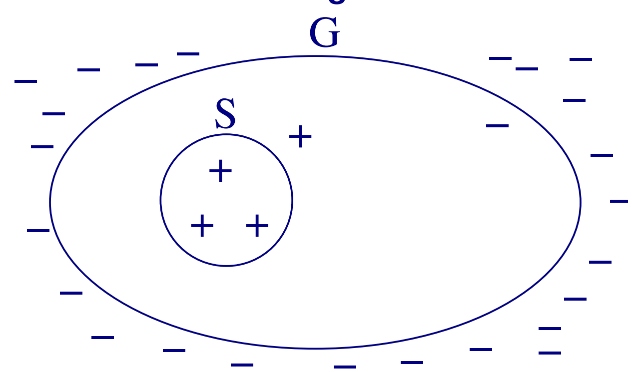
$$VS_{H,D} = \{h \in H | (\exists s \in S)(\exists g \in G)(g \geq_g h \geq_g s)\}$$

Proof hint: $g \in G$ cannot be satisfied by any negative example, and $s \in S$ must be satisfied by all positive examples.

G and S Revisited

- G: set of most general hypotheses consistent with D
  - Any more general then it will become **inconsistent**.
  - It will then **start returning 1 for negative examples**.
  - That is, it will generate error on negative examples.
- S: set of most specific hypotheses consistent with D.
  - Any more specific then it will become **inconsistent**.
  - It will then **start returning 0 for positive examples**.
  - That is, it will generate error on positive examples.

Candidate Elimination Algorithm: Basic Concept



- If negative example is misclassified by hypotheses in G, reduce the scope (make more specific).
- If positive example is misclassified by hypotheses in S, increase the scope (make more general).

Candidate Elimination Algorithm

$G \leftarrow$ maximally general hypotheses in H: $\langle ?, ?, ?, ?, ?, ? \rangle$
$S \leftarrow$ maximally specific hypotheses in H: $\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$

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
      1. h is consistent with d, and
      2. some member of G is more general than h
    - * Remove from S any hypothesis that is more general than another hypothesis in S
- 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
      1. h is consistent with d, and
      2. some member of S is more specific than h
    - * Remove from G any hypothesis that is less general than another hypothesis in G

Example

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

Initial $S_0 = \{\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset\}$, $G_0 = \{\langle ?, ?, ?, ?, ?, ? \rangle\}$.

- After example 1:

$$S_1 = \{\langle Sunny, Warm, Normal, Strong, Warm, Same \rangle\}$$

- After example 2:

$$S_2 = \{\langle Sunny, Warm, ?, Strong, Warm, Same \rangle\}$$

Note that $G_0 = G_1 = G_2$, i.e., no change.

Example (cont'd)

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

$S_2 = \{\langle Sunny, Warm, ?, Strong, Warm, Same \rangle\}$, and
$G_2 = \{\langle ?, ?, ?, ?, ?, ? \rangle\}$.

- After example 3, a negative example, G gets updated:

$$G_3 = \{\langle Sunny, ?, ?, ?, ?, ? \rangle, \langle ?, Warm, ?, ?, ?, ? \rangle \\ \langle ?, ?, ?, ?, ?, Same \rangle\}$$

- Question: Why didn't $\langle ?, ?, Normal, ?, ?, ? \rangle$ etc. get added to G?: See the **"more specific than"** condition.

Example (cont'd)

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
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

$S_3 = \{\langle Sunny, Warm, ?, Strong, Warm, Same \rangle\}$, and
$G_3 = \{\langle Sunny, ?, ?, ?, ?, ? \rangle, \langle ?, Warm, ?, ?, ?, ? \rangle, \langle ?, ?, ?, ?, ?, Same \rangle\}$.

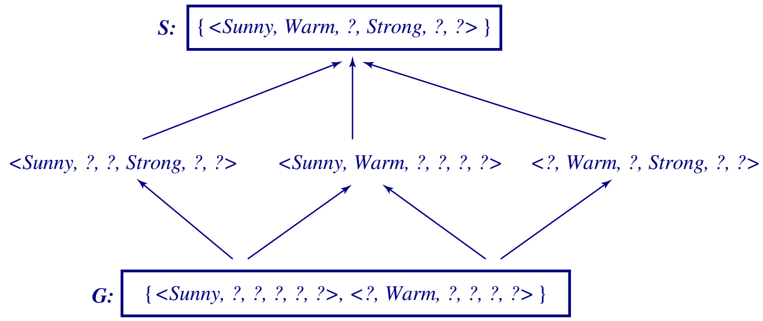
- After example 4:

$$S_4 = \{\langle Sunny, Warm, ?, Strong, ?, ? \rangle\}$$

$$G_4 = \{\langle Sunny, ?, ?, ?, ?, ? \rangle, \langle ?, Warm, ?, ?, ?, ? \rangle\}$$

- Both S and G got updated.

Example: Resulting Version Space



- S_4 and G_4 specify the tightest boundary.

Candidate-Elimination: Discussion

- Convergence: Algorithm converges if
  - Training examples are correct.
  - H includes the target concept represented by the training examples.
- Generating further training examples (exploration by the learner):
  - Want to test with a training example that can effectively narrow down the represented version space.
- Use of partially learned concepts in classifying new instances?
  - If every h in VS classifies an instance as positive, then say "Yes" (test if consistent with every h in S).
  - If every h in VS classifies an instance as negative, then say "No" (test if inconsistent with every h in G).

Inductive Bias

- What if the target concept is not included in H?
- How can we extend H in such a case?
- How does increasing the size of H affect generalization?

Biased Hypothesis Space

- If target concept is not in H, **enrich** the hypothesis space.
- The conjunctions-of-attributes hypothesis space is very limited, so it cannot include **disjunctive** target concepts such as "$Sky = Sunny \vee Sky = Cloudy$".

Ex Num	Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
1	Sunny	Warm	Normal	Strong	Cool	Change	Yes
2	Cloudy	Warm	Normal	Strong	Cool	Change	Yes
3	Rainy	Warm	Normal	Strong	Cool	Change	No

- Most specific h in conjunctive H given examples 1 and 2 will misclassify example 3.
- **Bias** in the selection of H led to this problem.
- In other words, learning can be **limited** by the bias.

An Unbiased Learner

- Can be unbiased if H is large enough, to include **any subset of the instances of X**.
- $|X|$ = 96, for conjunction of attributes. The number of subsets $= 2^{|X|} = 2^{96} \sim 10^{28}$.
- Allow arbitrary use of $\vee$, $\wedge$, and $\neg$.
- Problem: learned concept **cannot be generalized**—learned hypothesis will exactly represent the given instances and no other.

Futility of Bias-Free Learning

"... a learner that makes no a priori assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances."

- Concept learning algorithm L; instances X, target concept c
- training examples $D_c = \{\langle x, c(x) \rangle\}$
- let $L(x_i, D_c)$ denote the classification assigned to the instance x_i by L after training on data D_c.

The **inductive bias** of L is any minimal set of assertions B such that for any target concept c and corresponding training examples D_c

$$(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$$

where $A \vdash B$ means B logically entails from A (B is provable from A).
Note: $(D_c \wedge x_i) \vdash L(x_i, D_c)$ may not always be the case.

Comparing Inductive Bias

By comparing inductive biases, learning algorithms can be categorized:

- Rote-Learner: no inductive bias
- Candidate-Elimination: target concept c is contained in H.
- Find-S: on top of $c \in H$, all instances are negative unless the opposite is entailed by its other knowledge.