Training Examples for EnjoySport

<table>
<thead>
<tr>
<th>Sky</th>
<th>Temp</th>
<th>Humid</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySpt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Warm</td>
<td>Normal</td>
<td>Strong</td>
<td>Warm</td>
<td>Same</td>
<td>Yes</td>
</tr>
<tr>
<td>Sunny</td>
<td>Warm</td>
<td>High</td>
<td>Strong</td>
<td>Warm</td>
<td>Same</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cold</td>
<td>High</td>
<td>Strong</td>
<td>Warm</td>
<td>Change</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Warm</td>
<td>High</td>
<td>Strong</td>
<td>Cool</td>
<td>Change</td>
<td>Yes</td>
</tr>
</tbody>
</table>

What is the general concept?
Representing Hypotheses

Many possible representations

Here, \( h \) is conjunction of constraints on attributes

Each constraint can be

- a specific value (e.g., \( Water = Warm \))
- don’t care (e.g., \( “Water =?” \))
- no value allowed (e.g., \( “Water=∅” \))

For example,

\[
\text{Sky} \quad \text{AirTemp} \quad \text{Humid} \quad \text{Wind} \quad \text{Water} \quad \text{Forecst}
\]

\[
\langle Sunny ? ? Strong ? Same \rangle
\]
Prototypical Concept Learning Task

• Given:
  – Instances $X$: Possible days, each described by
    the attributes $Sky$, $AirTemp$, $Humidity$, $Wind$, $Water$, $Forecast$
  – Target function $c$: $EnjoySport : X \rightarrow \{0, 1\}$
  – Hypotheses $H$: Conjunctions of literals. E.g.
    \[ \langle ?, Cold, High, ?, ?, ?, \rangle. \]
  – Training examples $D$: Positive and negative examples of the target function
    \[ \langle x_1, c(x_1) \rangle, \ldots \langle x_m, c(x_m) \rangle \]

• **Determine:** A hypothesis $h$ in $H$ such that
  $h(x) = c(x)$ for all $x$ in $D$. 
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.
Instance, Hypotheses, and More-General-Than

Instances $X$

$$ x_1 = \langle \text{Sunny, Warm, High, Strong, Cool, Same} \rangle $$

$$ x_2 = \langle \text{Sunny, Warm, High, Light, Warm, Same} \rangle $$

Hypotheses $H$

$$ h_1 = \langle \text{Sunny, ?, ?, Strong, ?, ?} \rangle $$

$$ h_2 = \langle \text{Sunny, ?, ?, ?, ?, ?} \rangle $$

$$ h_3 = \langle \text{Sunny, ?, ?, ?, Cool, ?} \rangle $$
Find-S Algorithm

1. Initialize $h$ to the most specific hypothesis in $H$
2. For each positive training instance $x$
   • For each attribute constraint $a_i$ in $h$
     If the constraint $a_i$ in $h$ is satisfied by $x$
     Then do nothing
     Else replace $a_i$ in $h$ by the next more general constraint that is satisfied by $x$
3. Output hypothesis $h$
Hypothesis Space Search by Find-S

Instances $X$

Hypotheses $H$

$x_1 = \langle\text{Sunny Warm Normal Strong Warm Same}\rangle$, +
$x_2 = \langle\text{Sunny Warm High Strong Warm Same}\rangle$, +
$x_3 = \langle\text{Rainy Cold High Strong Warm Change}\rangle$, -
$x_4 = \langle\text{Sunny Warm High Strong Cool Change}\rangle$, +

$h_0 = \langle\emptyset, \emptyset, \emptyset, \emptyset, \emptyset\rangle$
$h_1 = \langle\text{Sunny Warm Normal Strong Warm Same}\rangle$
$h_2 = \langle\text{Sunny Warm ? Strong Warm Same}\rangle$
$h_3 = \langle\text{Sunny Warm ? Strong Warm Same}\rangle$
$h_4 = \langle\text{Sunny Warm ? Strong ? ?}\rangle$
Complaints about $\text{Find-S}$

- Can’t tell whether it has learned concept
- Can’t tell when training data inconsistent
- Picks a maximally specific $h$ (why?)
- Depending on $H$, there might be several!
Version Spaces

A hypothesis $h$ is **consistent** with a set of training examples $D$ of target concept $c$ if and only if $h(x) = c(x)$ for each training example $\langle x, c(x) \rangle$ in $D$.

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

The **version space**, $VS_{H,D}$, with respect to hypothesis space $H$ and training examples $D$, is the subset of hypotheses from $H$ consistent with all training examples in $D$.

$$VS_{H,D} \equiv \{ h \in H | Consistent(h, D) \}$$
The List-Then-Eliminate Algorithm:

1. \( \text{VersionSpace} \leftarrow \) a list containing every hypothesis in \( H \)
2. For each training example, \( \langle x, c(x) \rangle \)
   remove from \( \text{VersionSpace} \) any hypothesis \( h \) for which \( h(x) \neq c(x) \)
3. Output the list of hypotheses in \( \text{VersionSpace} \)
Example Version Space

\[ S: \{ <\text{Sunny, Warm, ?}, \text{Strong}, ?, ?> \} \]

\[ G: \{ <\text{Sunny, ?, ?}, \text{?, ?}, ?, ?>, <\text{?, Warm, ?}, \text{?, Strong}, ?, ?> \} \]
Representing Version Spaces

The **General boundary**, G, of version space $V S_{H,D}$ is the set of its maximally general members.

The **Specific boundary**, S, of version space $V S_{H,D}$ is the set of its maximally specific members.

Every member of the version space lies between these boundaries

$$V S_{H,D} = \{ h \in H | (\exists s \in S)(\exists g \in G)(g \geq h \geq s) \}$$

where $x \geq y$ means $x$ is more general or equal to $y$. 

Candidate Elimination Algorithm

\[ G \leftarrow \text{maximally general hypotheses in } H \]
\[ S \leftarrow \text{maximally specific hypotheses in } H \]

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
      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 Trace

\[ S_0 : \{ \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \} \]

\[ G_0 : \{ ?, ?, ?, ?, ?, ?, ? \} \]
What Next Training Example?

\[ S: \{ <\text{Sunny, Warm, }?, \text{ Strong, }?, \text{ ?>}> \} \]

\[ G: \{ <\text{Sunny, }?, \text{ }?, \text{ Strong, }?, \text{ }?>>, <\text{?, Warm, }?, \text{ }?, \text{ }?>> \} \]
How Should These Be Classified?

\[
S: \{ <\text{Sunny}, \text{Warm}, ?, \text{Strong}, ?, ?> \}
\]

\[
\]

\langle \text{Sunny \ Warm \ Normal \ Strong \ Cool \ Change} \rangle

\langle \text{Rainy \ Cool \ Normal \ Light \ Warm \ Same} \rangle

\langle \text{Sunny \ Warm \ Normal \ Light \ Warm \ Same} \rangle
What Justifies this Inductive Leap?

+ \langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change\ \rangle
+ \langle Sunny\ Warm\ Normal\ Light\ Warm\ Same\ \rangle

\[ S:\ \langle Sunny\ Warm\ Normal\ ?\ ?\ ?\ \rangle \]

Why believe we can classify the unseen
\[ \langle Sunny\ Warm\ Normal\ Strong\ Warm\ Same\ \rangle \]
An UNBiased Learner

Idea: Choose $H$ that expresses every teachable concept (i.e., $H$ is the power set of $X$)

Consider $H' =$ disjunctions, conjunctions, negations over previous $H$. E.g.,

$\langle Sunny\ Warm\ Normal\ ?\ ?\ ?\ \rangle \lor \neg \langle ?\ ?\ ?\ ?\ Change\ \rangle$

What are $S$, $G$ in this case?

$S \leftarrow$

$G \leftarrow$
Inductive Bias

Consider

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

Definition:

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

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

where $A \vdash B$ means $A$ logically entails $B$
**Inductive Systems and Equivalent Deductive Systems**

---

**Inductive system**

- Training examples
- New instance

**Candidate Elimination Algorithm**

Using Hypothesis Space $H$

Classification of new instance, or "don’t know"

---

**Equivalent deductive system**

- Training examples
- New instance
- Assertion "$H$ contains the target concept"

**Theorem Prover**

Classification of new instance, or "don’t know"

*Inductive bias made explicit*
Three Learners with Different Biases

1. *Rote learner*: Store examples, Classify $x$ iff it matches previously observed example.
2. *Version space candidate elimination algorithm*
3. *Find-$S$*
Summary Points

1. Concept learning as search through $H$
2. General-to-specific ordering over $H$
3. Version space candidate elimination algorithm
4. $S$ and $G$ boundaries characterize learner’s uncertainty
5. Learner can generate useful queries
6. Inductive leaps possible only if learner is biased
7. Inductive learners can be modelled by equivalent deductive systems