II. Machine Learning Basics

- Linear Regression
- Concept Learning: Search in Hypothesis Space
- Concept Learning: Version Space
- Evaluating Effectiveness
Concept Learning: Search in Hypothesis Space
Simple Classification Problems

Setting:
- \( X \) is a set of feature vectors.
- \( C \) is a set of two classes: \( \{0, 1\} \), \( \{\text{yes, no}\} \), “belongs to a concept or not”
- \( c : X \rightarrow C \) is the (unknown) ideal classifier for \( X \).
- \( D = \{(x_1, c(x_1)), \ldots, (x_n, c(x_n))\} \subseteq X \times C \) is a set of examples.

Todo:
- Approximate \( c(x) \), which is implicitly given via \( D \), with a feature-value pattern.
Concept Learning: Search in Hypothesis Space
An Exemplary Learning Task

The set $X$ is comprised of vectors that describe the weather condition in the six dimensions “Sky”, “Temperature”, “Humidity”, “Wind”, “Water”, and “Forecast”. The set $D$ are examples of weather conditions $x \in X$ along with a statement whether or not our friend will enjoy her favorite sport:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>warm</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>rainy</td>
<td>cold</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>change</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<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 concept behind “EnjoySport”?
- What are possible hypotheses to formalize the concept “EnjoySport”?

Similarly: What are the elements of the set or class “EnjoySport”? 
Remarks:

- Domains of the features in the learning task:

<table>
<thead>
<tr>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
</tr>
<tr>
<td>rainy</td>
<td>cold</td>
<td>high</td>
<td>light</td>
<td>cool</td>
<td>change</td>
</tr>
<tr>
<td>cloudy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- A concept is a subset of a larger set of objects. In the exemplary learning task the larger object set is comprised of all possible weather conditions, while the subset (= concept) contains those weather conditions when surfing can be enjoyed.

- A hypothesis is expected to “capture a (target) concept”, to “explain a (target) concept”, or to “predict a (target) concept” in terms of the feature expressions of the objects.

- The “quality”, the “persuasiveness”, or the “power” of a hypothesis depends on its capability to represent (= to explain) a given set of observations, which are called examples here.

- A hypothesis cannot be inferred or proven by deductive reasoning. A hypothesis is a finding or an insight gained by inductive reasoning.
Concept Learning: Search in Hypothesis Space

**Definition 1 (Concept, Hypothesis, Hypothesis Space)**

A concept is a subset of an object set $O$ and hence determines a subset of $X = \alpha(O)$. Concept learning is the approximation of the ideal classifier $c : X \rightarrow \{0, 1\}$, where

$$c(x) = \begin{cases} 1 & \text{if } \alpha^{-1}(x) \text{ belongs to the concept,} \\ 0 & \text{otherwise.} \end{cases}$$

Objects $O$ \quad Classes $\{1, 0\}$

$\gamma(o)$

Feature vectors $X$ \quad $c(x) \approx h(x)$

[ML: I Specification of Learning Problems]
**Definition 1 (Concept, Hypothesis, Hypothesis Space)**

A concept is a subset of an object set \( O \) and hence determines a subset of \( X = \alpha(O) \). Concept learning is the approximation of the ideal classifier \( c : X \rightarrow \{0, 1\} \), where

\[
c(x) = \begin{cases} 
1 & \text{if } \alpha^{-1}(x) \text{ belongs to the concept}, \\
0 & \text{otherwise}.
\end{cases}
\]

A hypothesis is a function \( h(x) \), \( h : X \rightarrow \{0, 1\} \), to approximate \( c(x) \). A hypothesis space is a set \( H \) of hypotheses among which \( h(x) \) is searched.

[ML: I Specification of Learning Problems]
Remarks:

- A hypothesis may also be called *model function* or *model*. Note however, that it is common practice to designate only the *parameters* of a model function, $\theta$ or $w$, as hypothesis (and not the function itself), especially if the setting focuses on a certain class of models, such as linear models, polynomials of a fixed degree, or Gaussian distributions.

- The subtle semantic distinction between the terms “model function” and “hypothesis” made in machine learning is that the former term is typically used to denote a function class or a particular class of computational approaches, while the latter term refers to a specific instance of that class.

- Depending on the learning task—more specifically: on the *structure of the feature space*—a hypothesis (model function) can take different forms and, accordingly, is denoted differently: $h(x)$ (as done here), $y(x)$ (in regression settings), $T$ (for decision trees), $\prod P(A \mid B)$ (within statistical learning), etc.
Concept Learning: Search in Hypothesis Space

The example set $D$, $D = \{(x_1, c(x_1)), \ldots, (x_n, c(x_n))\}$, contains usually both positive ($c(x) = 1$) and negative ($c(x) = 0$) examples. [learning task]

**Definition 2 (Hypothesis-Fulfilling, Consistency)**

An example $(x, c(x))$ fulfills a hypothesis $h(x)$ iff $h(x) = 1$.

A hypothesis $h(x)$ is consistent with an example $(x, c(x))$ iff $h(x) = c(x)$.

A hypothesis $h(x)$ is consistent with a set $D$ of examples, denoted as $\text{consistent}(h, D)$, iff:

$$\forall (x, c(x)) \in D : h(x) = c(x)$$
Remarks:

- The string “Iff” or “iff” is an abbreviation for “If and only if”, which means “necessary and sufficient”. It is a textual representation for the logical biconditional, also known as material biconditional or iff-connective. The respective symbol is “$\leftrightarrow$”. [Wolfram] [Wikipedia]

- The following terms are used synonymously: target concept, target function, ideal classifier.

- The fact that a hypothesis is consistent with an example can also be described the other way round: an example is consistent with a hypothesis.

- Given an example $(x, c(x))$, notice the difference between (1) hypothesis-fulfilling and (2) being consistent with a hypothesis. The former asks for $h(x) = 1$, disregarding the actual target concept value $c(x)$. The latter asks for the identity between the target concept $c(x)$ and the hypothesis $h(x)$.

- The consistency of $h(x)$ can be analyzed for a single example as well as for a set $D$ of examples. Given the latter, consistency requires for all elements in $D$ that $h(x) = 1$ iff $c(x) = 1$. This is equivalent with the condition that $h(x) = 0$ iff $c(x) = 0$ for all $x \in D$.

- Learning means to determine a hypothesis $h(x) \in H$ that is consistent with $D$. Similarly: Machine learning means to systematically search the hypothesis space.
Concept Learning: Search in Hypothesis Space
Simple Classification Problems (continued)

Structure of a hypothesis $h(x)$:

1. conjunction of feature-value pairs
2. three kinds of values: literal, ? (wildcard), ⊥ (contradiction)

A hypothesis for **EnjoySport** [learning task] : ⟨ sunny, ?, ?, strong, ?, same ⟩
Concept Learning: Search in Hypothesis Space

Simple Classification Problems (continued)

Structure of a hypothesis $h(x)$:

1. conjunction of feature-value pairs
2. three kinds of values: literal, $?$ (wildcard), $\bot$ (contradiction)

A hypothesis for EnjoySport [learning task]: $\langle \text{sunny, ?, ?, strong, ?, same} \rangle$

**Definition 3 (Maximally Specific / General Hypothesis)**

The hypotheses $s_0(x) \equiv 0$ and $g_0(x) \equiv 1$ are called maximally specific and maximally general hypothesis respectively. No $x \in X$ fulfills $s_0(x)$, and all $x \in X$ fulfill $g_0(x)$.

Maximally specific / general hypothesis in the example [learning task]:

- $s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$ (never enjoy sport)
- $g_0 = \langle ?, ?, ?, ?, ?, ? \rangle$ (always enjoy sport)
Concept Learning: Search in Hypothesis Space

Order of Hypotheses

Feature vectors $X$

Hypothesis space $H$

$x_1 = (sunny, warm, normal, strong, warm, same)$

$h_1 = \langle sunny, ?, normal, ?, ?, ? \rangle$

$h_2 = \langle sunny, ?, ?, ?, warm, ? \rangle$

$x_4 = (sunny, warm, high, strong, cool, change)$

$h_4 = \langle sunny, ?, ?, ?, ?, ? \rangle$
Concept Learning: Search in Hypothesis Space

Order of Hypotheses

**Definition 4 (More General Relation)**

Let $X$ be a set of feature vectors and let $h_1(x)$ and $h_2(x)$ be two boolean-valued functions with domain $X$. Then $h_1(x)$ is called more general than $h_2(x)$, denoted as $h_1(x) \geq_g h_2(x)$, iff:

$$\forall x \in X : ( h_2(x) = 1 \text{ implies } h_1(x) = 1 )$$

$h_1(x)$ is called strictly more general than $h_2(x)$, denoted as $h_1(x) >_g h_2(x)$, iff:

$$(h_1(x) \geq_g h_2(x)) \text{ and } (h_2(x) \not\geq_g h_1(x))$$
Concept Learning: Search in Hypothesis Space
Order of Hypotheses

Definition 4 (More General Relation)

Let $X$ be a set of feature vectors and let $h_1(x)$ and $h_2(x)$ be two boolean-valued functions with domain $X$. Then $h_1(x)$ is called more general than $h_2(x)$, denoted as $h_1(x) \geq_g h_2(x)$, iff:

$$\forall x \in X : ( h_2(x) = 1 \text{ implies } h_1(x) = 1 )$$

$h_1(x)$ is called strictly more general than $h_2(x)$, denoted as $h_1(x) >_g h_2(x)$, iff:

$$(h_1(x) \geq_g h_2(x)) \text{ and } (h_2(x) \not\geq_g h_1(x))$$

About the maximally specific / general hypothesis:

- $s_0(x)$ is minimum and $g_0(x)$ is maximum with regard to $\geq_g$: no hypothesis is more specific wrt. $s_0(x)$, and no hypothesis is more general wrt. $g_0(x)$.
- We will consider only hypothesis spaces that contain $s_0(x)$ and $g_0(x)$.
If $h_1(x)$ is more general than $h_2(x)$, then $h_2(x)$ can also be called being more specific than $h_1(x)$.

$\geq_g$ and $>_g$ are independent of a target concept $c(x)$. They depend only on the fact that examples fulfill a hypothesis, i.e., whether $h(x) = 1$. It is not required that $c(x) = 1$.

The $\geq_g$-relation defines a partial order on the hypothesis space $H : \geq_g$ is reflexive, anti-symmetric, and transitive. The order is partial since (unlike in a total order) not all hypothesis pairs stand in the relation. [Wikipedia partial, total]

I.e., we are given hypotheses $h_i(x)$, $h_j(x)$, for which neither $h_i(x) \geq_g h_j(x)$ nor $h_j(x) \geq_g h_i(x)$ holds, such as the hypotheses $h_1(x)$ and $h_2(x)$ in the hypothesis space.
The semantics of the implication, in words “$a$ implies $b$”, denoted as $a \rightarrow b$, is as follows. $a \rightarrow b$ is true if either (1) $a$ is true and $b$ is true, or (2) if $a$ is false and $b$ is true, or (3) if $a$ is false and $b$ is false—in short: “if $a$ is true then $b$ is true as well”, or, “the truth of $a$ implies the truth of $b$”.

“$\rightarrow$” can be understood as “causality connective”: Let $a$ and $b$ be two events where $a$ is a cause for $b$. If we interpret the occurrence of an event as true and its non-occurrence as false, we will observe only occurrence combinations such that the formula $a \rightarrow b$ is true. The connective is also known as material conditional, material implication, material consequence, or simply, implication or conditional.

Note in particular that the connective “$\rightarrow$” does not mean “entails”, which would be denoted as either $\Rightarrow$ or $\models$. Logical entailment (synonymously: logical inference, logical deduction, logical consequence) allows to infer or to prove a fact. Consider for instance the More-General-Definition: From the fact $h_2(x) = 1$ we cannot infer or prove the fact $h_1(x) = 1$.

In the More-General-Definition the implication specifies a condition that is to be fulfilled by the definiendum (= the thing to be defined). The implication is used to check whether or not a thing belongs to the set of things specified by the definiens (= the expression that defines): Each pair of functions, $h_1(x), h_2(x)$, is a thing that belongs to the set of things specified by the definition of the $\geq_g$-relation (i.e., stands in the $\geq_g$-relation) if and only if the implication $h_2(x) = 1 \rightarrow h_1(x) = 1$ is true for all $x \in X$. 


Remarks: (continued)

- In a nutshell: distinguish carefully between “α requires β”, denoted as $α \rightarrow β$, on the one hand, and “from α follows β”, denoted as $α \Rightarrow β$, on the other hand. $α \rightarrow β$ is considered as a sentence from the object language (language of discourse) and stipulates a computing operation, whereas $α \Rightarrow β$ is a sentence from the meta language and makes an assertion about the sentence $α \rightarrow β$, namely: “$α \rightarrow β$ is a tautology”.

- Finally, consider the following sentences from the object language, which are synonymous: “$α \rightarrow β$”, “α implies β”, “if α then β”, “α causes β”, “α requires β”, “α is sufficient for β”, “β is necessary for α”.

“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.”  [p.23, Mitchell 1997]
Concept Learning: Search in Hypothesis Space

Find-S Algorithm

1. \( h(x) = s_0(x) \) // \( h(x) \) is a maximally specific hypothesis in \( H \).
2. \textbf{FOREACH} \( (x, c(x)) \in D \) \textbf{DO}
   \hspace{1em} \textbf{IF} \( c(x) = 1 \) \textbf{THEN} // Use only positive examples.
   \hspace{1em} \hspace{1em} \textbf{IF} \( h(x) = 0 \) \textbf{DO}
   \hspace{1em} \hspace{1em} \hspace{1em} \textit{h = min{generalization(h, x)}} // Relax \( h(x) \) wrt. \( x \).
   \hspace{1em} \hspace{1em} \textbf{ENDIF}
   \hspace{1em} \textbf{ENDIF}
\textbf{ENDDO}
3. \textit{return(h(x))}
Remarks:

- Another term for “generalization” is “relaxation”.

- The function $\text{min}\_\text{generalization}(h, x)$ returns a hypothesis $h'(x)$ that is minimally generalized wrt. $h(x)$ and that is consistent with $(x, 1)$. Denoted formally: $h'(x) \geq_g h(x)$ and $h'(x) = 1$, and there is no $h''(x)$ with $h'(x) > h''(x) \geq_g h(x)$ with $h''(x) = 1$.

- For more complex hypothesis structures the relaxation of $h(x)$, $\text{min}\_\text{generalization}(h, x)$, may not be unique. In such a case one of the alternatives has to be chosen.

- If a hypothesis $h(x)$ needs to be relaxed towards some $h'(x)$ with $h'(x) \not\in H$, the maximally general hypothesis $g_0 \equiv 1$ can be added to $H$.

- Similar to $\text{min}\_\text{generalization}(h, x)$, a function $\text{min}\_\text{specialization}(h, x)$ can be defined, which returns a minimally specialized, consistent hypotheses for negative examples.
See the example set $D$ for the concept $EnjoySport$.

$$h_0 = s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$$

$$x_1 = \langle \text{sunny}, \text{warm}, \text{normal}, \text{strong}, \text{warm}, \text{same} \rangle$$

$$h_1 = \langle \text{sunny}, \text{warm}, \text{normal}, \text{strong}, \text{warm}, \text{same} \rangle$$
Concept Learning: Search in Hypothesis Space

Find-S Algorithm

See the example set $D$ for the concept *EnjoySport*.

Feature vectors $X$

Hypothesis space $H$

$h_0 = s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$

$x_1 = \langle \text{sunny, warm, normal, strong, warm, same} \rangle$

$x_2 = \langle \text{sunny, warm, high, strong, warm, same} \rangle$

$h_1 = \langle \text{sunny, warm, normal, strong, warm, same} \rangle$

$h_2 = \langle \text{sunny, warm, ?, strong, warm, same} \rangle$
Concept Learning: Search in Hypothesis Space

Find-S Algorithm

See the example set $D$ for the concept $EnjoySport$.

Feature vectors $X$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny, warm, normal, strong, warm, same</td>
<td>sunny, warm, high, strong, warm, same</td>
<td>rainy, cold, high, strong, warm, change</td>
</tr>
</tbody>
</table>

Hypothesis space $H$

<table>
<thead>
<tr>
<th>$h_0$</th>
<th>$h_1$</th>
<th>$h_2, 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$</td>
<td>$\langle \text{sunny, warm, normal, strong, warm, same} \rangle$</td>
<td>$\langle \text{sunny, warm, ?, strong, warm, same} \rangle$</td>
</tr>
</tbody>
</table>

$h_0 = s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$

$h_1 = \langle \text{sunny, warm, normal, strong, warm, same} \rangle$

$h_2 = \langle \text{sunny, warm, ?, strong, warm, same} \rangle$

$h_3 = \langle \text{sunny, warm, ?, strong, warm, same} \rangle$
Concept Learning: Search in Hypothesis Space

Find-S Algorithm

See the example set $D$ for the concept $EnjoySport$. 

Feature vectors $X$

$$x_1 = \langle \text{sunny, warm, normal, strong, warm, same} \rangle$$

$$x_2 = \langle \text{sunny, warm, high, strong, warm, same} \rangle$$

$$x_3 = \langle \text{rainy, cold, high, strong, warm, change} \rangle$$

$$x_4 = \langle \text{sunny, warm, high, strong, cool, change} \rangle$$

Hypothesis space $H$

$$h_0 = s_0 = \langle \bot, \bot, \bot, \bot, \bot, \bot \rangle$$

$$h_1 = \langle \text{sunny, warm, normal, strong, warm, same} \rangle$$

$$h_2 = \langle \text{sunny, warm, ?, strong, warm, same} \rangle$$

$$h_3 = \langle \text{sunny, warm, ?, strong, warm, same} \rangle$$

$$h_4 = \langle \text{sunny, warm, ?, strong, ?, ?} \rangle$$
1. Did we learn the only concept—or are there others?
2. Why should one pursue the maximally specific hypothesis?
3. What if several maximally specific hypotheses exist?
4. Inconsistencies in the example set $D$ remain undetected.
5. An inappropriate hypothesis structure or space $H$ remains undetected.
Concept Learning: Version Space

**Definition 5 (Version Space)**

The version space $V_{H,D}$ of a hypothesis space $H$ and a example set $D$ is comprised of all hypotheses $h(x) \in H$ that are consistent with a set $D$ of examples:

$$V_{H,D} = \{ h(x) \mid h(x) \in H \land \left( \forall (x, c(x)) \in D : h(x) = c(x) \right) \}$$
Concept Learning: Version Space

Definition 5 (Version Space)
The version space $V_{H,D}$ of a hypothesis space $H$ and a example set $D$ is comprised of all hypotheses $h(x) \in H$ that are consistent with a set $D$ of examples:

$$V_{H,D} = \{ h(x) \mid h(x) \in H \land (\forall (x, c(x)) \in D : h(x) = c(x)) \}$$

Illustration of $V_{H,D}$ for the example set $D$:

```
S  { < sunny, warm, ?, strong, ?, ? > }


```
Remarks:

- The term “version space” reflects the fact that $V_{H,D}$ represents the set of all consistent versions of the target concept that are encoded in $D$.

- A naive approach for the construction of the version space is the following: (1) enumeration of all members of $H$, and, (2) elimination of those $h(x) \in H$ for which $h(x) \neq c(x)$ holds. This approach presumes a finite hypothesis space $H$ and is feasible only for toy problems.
Concept Learning: Version Space

Definition 6 (Boundary Sets of a Version Space)
Let $H$ be hypothesis space and let $D$ be set of examples. Then, based on the $\geq_g$-relation, the set of maximally general hypotheses, $G$, is defined as follows:

$$G = \{ g(x) \mid g(x) \in H \land \text{consistent}(g, D) \land$$
$$\left( \forall g'(x) : g'(x) \in H \land g'(x) \geq_g g(x) \land \text{consistent}(g', D) \right) \}$$

Similarly, the set of maximally specific (i.e., minimally general) hypotheses, $S$, is defined as follows:

$$S = \{ s(x) \mid s(x) \in H \land \text{consistent}(s, D) \land$$
$$\left( \forall s'(x) : s'(x) \in H \land s(x) \geq_g s'(x) \land \text{consistent}(s', D) \right) \}$$
Theorem 7 (Version Space Representation)

Let $X$ be a set of feature vectors and $H$ a set of boolean-valued functions with domain $X$. Moreover, let $c(x) : X \rightarrow \{0, 1\}$ be a target concept and $D$ a set of examples of the form $(x, c(x))$.

Then, based on the $\geq_g$-relation, each member of the version space $V_{H,D}$ lies in between two members of $G$ and $S$ respectively:

$$V_{H,D} = \{ h(x) \mid h(x) \in H \land \hspace{1cm} ( \exists g(x) \in G \exists s(x) \in S : g(x) \geq_g h(x) \geq_g s(x) ) \}$$
Candidate Elimination Algorithm  [Mitchell 1997]

1. Initialization:  \( G = \{g_0\} \),  \( S = \{s_0\} \)

2. If \( x \) is a **positive** example
   - Remove from \( G \) any hypothesis that is not consistent with \( x \)
   - For each hypothesis \( s \) in \( S \) that is not consistent with \( x \)
     - Remove \( s \) from \( S \)
     - Add to \( S \) all minimal **generalizations** \( h \) of \( s \) such that
       1. \( h \) is consistent with \( x \) and
       2. some member of \( G \) is more general than \( h \)
   - Remove from \( S \) any hypothesis that is less specific than another hypothesis in \( S \)
Concept Learning: Version Space  
Candidate Elimination Algorithm  [Mitchell 1997]

1. Initialization: \( G = \{g_0\}, \quad S = \{s_0\} \)

2. If \( x \) is a **positive** example
   - Remove from \( G \) any hypothesis that is not consistent with \( x \)
   - For each hypothesis \( s \) in \( S \) that is not consistent with \( x \)
     - Remove \( s \) from \( S \)
     - Add to \( S \) all minimal generalizations \( h \) of \( s \) such that
       1. \( h \) is consistent with \( x \) and
       2. some member of \( G \) is more general than \( h \)
   - Remove from \( S \) any hypothesis that is less specific than another hypothesis in \( S \)

3. If \( x \) is a **negative** example
   - Remove from \( S \) any hypothesis that is not consistent with \( x \)
   - For each hypothesis \( g \) in \( G \) that is not consistent with \( x \)
     - Remove \( g \) from \( G \)
     - Add to \( G \) all minimal specializations \( h \) of \( g \) such that
       1. \( h \) is consistent with \( x \) and
       2. some member of \( S \) is more specific than \( h \)
   - Remove from \( G \) any hypothesis that is less general than another hypothesis in \( G \)
Remarks:

- The basic idea of Candidate Elimination is as follows:
  - A maximally specific hypothesis \( s(x) \in S \) restricts the positive examples in first instance. Hence, \( s(x) \) needs to be relaxed (= generalized) with regard to each positive example that is not consistent with \( s(x) \).
  - Conversely, a maximally general hypothesis \( g(x) \in G \) tolerates the negative examples in first instance. Hence, \( g(x) \) needs to be constrained (= specialized) with regard to each negative example that is not consistent with \( g(x) \).
1. $G = \{g_0\}$ // $G$ is the set of maximally general hypothesis in $H$.  
   $S = \{s_0\}$ // $S$ is the set of maximally specific hypothesis in $H$.  

2. FOREACH $(x, c(x)) \in D$ DO  
   IF $c(x) = 1$ THEN // $x$ is a positive example.  
      FOREACH $g \in G$ DO IF $g(x) \neq 1$ THEN $G = G \setminus \{g\}$ ENDDO  
      FOREACH $s \in S$ DO  
         IF $s(x) \neq 1$ THEN  
            $S = S \setminus \{s\}$, $S^+ = \text{min}_{\text{generalizations}}(s, x)$  
            FOREACH $s \in S^+$ DO IF $(\exists g \in G : g \geq_g s)$ THEN $S = S \cup \{s\}$ ENDDO  
            FOREACH $s \in S$ DO IF $(\exists s' \in S : s' \neq s \land s \geq_g s')$ THEN $S = S \setminus \{s\}$ ENDDO  
         ENDIF  
      ENDIF  
   ENDIF  

ELSE // $x$ is a negative example.  

ENDIF  
ENDDO  

3. return$(G, S)$
1. \( G = \{ g_0 \} \) // \( G \) is the set of maximally general hypothesis in \( H \).
   \( S = \{ s_0 \} \) // \( S \) is the set of maximally specific hypothesis in \( H \).

2. \textbf{FOREACH} \((x, c(x)) \in D\) \textbf{DO}
   \hspace{1em} \textbf{IF} \( c(x) = 1 \) \textbf{THEN} // \( x \) is a positive example.
   \hspace{2em} \textbf{FOREACH} \( g \in G \) \textbf{DO} \textbf{IF} \( g(x) \neq 1 \) \textbf{THEN} \( G = G \setminus \{g\} \) \textbf{ENDDO}
   \hspace{2em} \textbf{FOREACH} \( s \in S \) \textbf{DO}
   \hspace{3em} \textbf{IF} \( s(x) \neq 1 \) \textbf{THEN}
   \hspace{4em} \( S = S \setminus \{s\}, \quad S^+ = \text{min\_generalizations}(s, x) \)
   \hspace{3em} \textbf{FOREACH} \( s \in S^+ \) \textbf{DO} \textbf{IF} \( \exists g \in G : g \geq_g s \) \textbf{THEN} \( S = S \cup \{s\} \) \textbf{ENDDO}
   \hspace{4em} \textbf{FOREACH} \( s \in S \) \textbf{DO} \textbf{IF} \( \exists s' \in S : s' \neq s \land s \geq_g s' \) \textbf{THEN} \( S = S \setminus \{s\} \) \textbf{ENDDO}
   \hspace{2em} \textbf{ENDDO}
   \hspace{1em} \textbf{ELSE} // \( x \) is a negative example.
   \hspace{2em} \textbf{FOREACH} \( s \in S \) \textbf{DO} \textbf{IF} \( s(x) \neq 0 \) \textbf{THEN} \( S = S \setminus \{s\} \) \textbf{ENDDO}
   \hspace{2em} \textbf{FOREACH} \( g \in G \) \textbf{DO}
   \hspace{3em} \textbf{IF} \( g(x) \neq 0 \) \textbf{THEN}
   \hspace{4em} \( G = G \setminus \{g\}, \quad G^- = \text{min\_specializations}(g, x) \)
   \hspace{3em} \textbf{FOREACH} \( g \in G^- \) \textbf{DO} \textbf{IF} \( \exists s \in S : g \geq_g s \) \textbf{THEN} \( G = G \cup \{g\} \) \textbf{ENDDO}
   \hspace{4em} \textbf{FOREACH} \( g \in G \) \textbf{DO} \textbf{IF} \( \exists g' \in G : g' \neq g \land g' \geq_g g \) \textbf{THEN} \( G = G \setminus \{g\} \) \textbf{ENDDO}
   \hspace{2em} \textbf{ENDDO}
   \hspace{2em} \textbf{ENDIF}
 \hspace{1em} \textbf{ENDDO}

3. \textit{return}(G, S)
Concept Learning: Version Space
Candidate Elimination Algorithm (illustration)

\[ \{ < \bot, \bot, \bot, \bot, \bot, \bot > \} \quad S_0 \]

\[ \{ < ?, ?, ?, ?, ?, ? > \} \quad G_0, \]
Concept Learning: Version Space
Candidate Elimination Algorithm (illustration)

\[ S_0 = \{ < ?, ?, ?, ?, ?, ? > \} \]


\[ G_0, \ G_1, \]

\[ x_1 = (sunny, warm, normal, strong, warm, same) \]

\[ EnjoySport(x_1) = 1 \]
Concept Learning: Version Space
Candidate Elimination Algorithm (illustration)

\[ S_0 \]
\[ \{ < \bot, \bot, \bot, \bot, \bot, \bot > \} \]
\[ S_1 \]
\[ \{ < \text{sunny, warm, normal, strong, warm, same} > \} \]
\[ S_2, \]
\[ \{ < \text{sunny, warm, ?, strong, warm, same} > \} \]
\[ G_0, G_1, G_2 \]

\[ x_1 = (\text{sunny, warm, normal, strong, warm, same}) \]
\[ x_2 = (\text{sunny, warm, high, strong, warm, same}) \]

\[ \text{EnjoySport}(x_1) = 1 \]
\[ \text{EnjoySport}(x_2) = 1 \]
Concept Learning: Version Space

Candidate Elimination Algorithm (illustration)

\[
\begin{align*}
S_0 & = \{ < \bot, \bot, \bot, \bot, \bot, \bot > \} \\
S_1 & = \{ < \text{sunny}, \text{warm}, \text{normal}, \text{strong}, \text{warm}, \text{same} > \} \\
S_2 & = \{ < \text{sunny}, \text{warm}, \bot, \text{strong}, \text{warm}, \text{same} > \} \\
S_3 & = \{ < \text{rainy}, \text{cold}, \bot, \text{strong}, \text{warm}, \text{change} > \}
\end{align*}
\]

\[
\begin{align*}
G_0 & = \{ < \bot, \bot, \bot, \bot, \bot, \bot > \} \\
G_1 & = \{ < \text{sunny}, \text{warm}, \bot, \text{normal}, \text{strong}, \text{warm}, \text{same} > \} \\
G_2 & = \{ < \text{sunny}, \text{?, ?, ?, ?, ?, ?} >, < \text{?, warm}, \text{?, ?, ?, ?, ?} >, < \text{?, ?, ?, ?, ?, same} > \} \\
G_3 & = \{ < \text{sunny}, \text{?, ?, ?, ?, ?, ?} >, < \text{?, warm}, \text{?, ?, ?, ?, ?} >, < \text{?, ?, ?, ?, ?, same} > \}
\end{align*}
\]

\[
\begin{align*}
\text{x}_1 & = (\text{sunny, warm, normal, strong, warm, same}) & \text{EnjoySport(x}_1) & = 1 \\
\text{x}_2 & = (\text{sunny, warm, high, strong, warm, same}) & \text{EnjoySport(x}_2) & = 1 \\
\text{x}_3 & = (\text{rainy, cold, high, strong, warm, change}) & \text{EnjoySport(x}_3) & = 0
\end{align*}
\]
Concept Learning: Version Space

Candidate Elimination Algorithm (illustration)

\[
\begin{align*}
S_0 & \rightarrow \{< \bot, \bot, \bot, \bot, \bot, \bot>\} \\
S_1 & \rightarrow \{< \text{sunny}, \text{warm}, \text{normal}, \text{strong}, \text{warm}, \text{same}>\} \\
S_2 & \rightarrow \{< \text{sunny}, \text{warm}, ?, \text{strong}, \text{warm}, \text{same}>\} \\
S_3 & \rightarrow \{< \text{sunny}, ?, ?, \text{strong}, ?, \text{same}>\} \\
S_4 & \rightarrow \{< \text{sunny}, ?, ?, \text{strong}, ?, ?, ?>\} \\
\end{align*}
\]

\[
\begin{align*}
x_1 & = (\text{sunny}, \text{warm}, \text{normal}, \text{strong}, \text{warm}, \text{same}) & \text{EnjoySport}(x_1) & = 1 \\
x_2 & = (\text{sunny}, \text{warm}, \text{high}, \text{strong}, \text{warm}, \text{same}) & \text{EnjoySport}(x_2) & = 1 \\
x_3 & = (\text{rainy}, \text{cold}, \text{high}, \text{strong}, \text{warm}, \text{change}) & \text{EnjoySport}(x_3) & = 0 \\
x_4 & = (\text{sunny}, \text{warm}, \text{high}, \text{strong}, \text{cool}, \text{change}) & \text{EnjoySport}(x_4) & = 1
\end{align*}
\]
Concept Learning: Version Space
Discussion of the Candidate Elimination Algorithm

1. What about selecting examples from $D$ according to a certain strategy? Keyword: active learning

2. What are partially learned concepts and how to exploit them? Keyword: ensemble classification

3. The version space as defined here is “biased”. What does this mean?

4. Will Candidate Elimination converge towards the correct hypothesis?

5. When does one end up with an empty version space?
Question 1: Selecting Examples from $D$

An example from which we can “maximally” learn:

$x_7 = (\text{sunny, warm, normal, light, warm, same})$
Concept Learning: Version Space

Question 1: Selecting Examples from $D$

An example from which we can “maximally” learn:

$x_7 = (sunny, warm, normal, light, warm, same)$

I.e., irrespective the value of $c(x_7)$, the example $(x_7, c(x_7))$ will be consistent with three of the six hypotheses. It follows:

- If $EnjoySport(x_7) = 1$  
  $S$ can be further generalized.
- If $EnjoySport(x_7) = 0$  
  $G$ can be further specialized.
Concept Learning: Version Space

Question 2: Partially Learned Concepts

Combine the six classifiers in the version space:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cold</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>cold</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
</tbody>
</table>
Concept Learning: Version Space

Question 2: Partially Learned Concepts

\[ S \{ < \text{sunny, warm, ?, strong, ?, ?} > \} \]

\[ G \{ < \text{sunny, ?, ?, ?, ?, ?} >, < \text{?, warm, ?, ?, ?, ?} > \} \]

Combine the six classifiers in the version space:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>6+ : 0–</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cold</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>cold</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
</tbody>
</table>
Concept Learning: Version Space

Question 2: Partially Learned Concepts

Combine the six classifiers in the version space:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>6+ : 0–</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cold</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td>0+ : 6–</td>
</tr>
<tr>
<td>7</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>cold</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
</tbody>
</table>
Concept Learning: Version Space

Question 2: Partially Learned Concepts

Combine the six classifiers in the version space:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>6+ : 0–</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cold</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td>0+ : 6–</td>
</tr>
<tr>
<td>7</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td>3+ : 3–</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>cold</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td></td>
</tr>
</tbody>
</table>
Concept Learning: Version Space

Question 2: Partially Learned Concepts

Combine the six classifiers in the version space:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>6+ : 0–</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cold</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td>0+ : 6–</td>
</tr>
<tr>
<td>7</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>light</td>
<td>warm</td>
<td>same</td>
<td>3+ : 3–</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>cold</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>2+ : 4–</td>
</tr>
</tbody>
</table>
Concept Learning: Version Space

Question 3: Inductive Bias

A new set of training examples $D$:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>cloudy</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
</tbody>
</table>

$S = \{ \langle ?, \text{warm}, \text{normal}, \text{strong}, \text{cool}, \text{change} \rangle \}$
A new set of training examples $D$:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>cloudy</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>rainy</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>no</td>
</tr>
</tbody>
</table>

$\Rightarrow S = \{ \langle ?, \text{warm}, \text{normal}, \text{strong}, \text{cool}, \text{change} \rangle \}$

$\Rightarrow S = \{ \}$

Discussion:

- What assumptions about the target concept are met by the learner a-priori?
Question 3: Inductive Bias

A new set of training examples $D$:

<table>
<thead>
<tr>
<th>Example</th>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>EnjoySport</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>cloudy</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>rainy</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>no</td>
</tr>
</tbody>
</table>

$S = \{ \langle ?, \text{warm}, \text{normal}, \text{strong}, \text{cool}, \text{change} \rangle \}$

$\vdash \vdash$

$S = \{ \}$

Discussion:

- What assumptions about the target concept are met by the learner a-priori?

Consequence: The hypothesis space $H$ may be designed to contain more complex concepts, e.g., $\langle \text{sunny}, ?, ?, ?, ?, ? \rangle \lor \langle \text{cloudy}, ?, ?, ?, ?, ?, ? \rangle$. 
In a binary classification problem the unrestricted (= unbiased) hypothesis space contains $|\mathcal{P}(X)| = 2^{|X|}$ elements.

A learning algorithm that considers all possible hypotheses as equally likely makes no a-priori assumption with regard to the target concept.

A learning algorithm without a-priori assumptions has no “inductive bias”.

“The policy by which a [learning] algorithm generalizes from observed training examples to classify unseen instances is its inductive bias. […] Inductive bias is the set of assumptions that, together with the training data, deductively justify the classification by the learner to future instances.” [p.63, Mitchell 1997]
In a binary classification problem the unrestricted (= unbiased) hypothesis space contains \(|\mathcal{P}(X)| \equiv 2^{|X|}\) elements.

A learning algorithm that considers all possible hypotheses as equally likely makes no a-priori assumption with regard to the target concept.

A learning algorithm without a-priori assumptions has no “inductive bias”.

“The policy by which a [learning] algorithm generalizes from observed training examples to classify unseen instances is its inductive bias. […] Inductive bias is the set of assumptions that, together with the training data, deductively justify the classification by the learner to future instances.” [p.63, Mitchell 1997]

→ A learning algorithm without inductive bias has no directive to classify unseen examples. Put another way: the learner cannot generalize.

→ A learning algorithm without inductive bias can only memorize.

Which algorithm (Find-S, Candidate Elimination) has a stronger inductive bias?