# CSC3022H: Machine Learning:

# Concept Learning

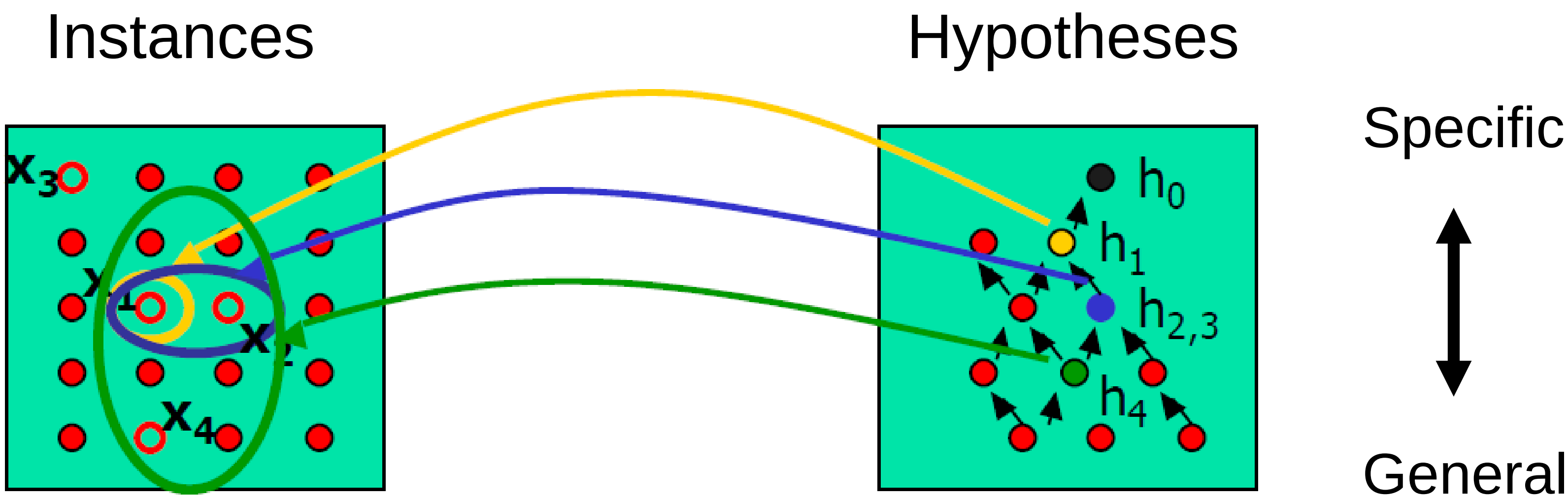
Geoff Nitschke

Department of Computer Science
University of Cape Town, South Africa

# Examples of Machine Learning Types

- **Supervised Learning:**
  - **Classification.**
  - Regression.
- **Unsupervised Learning:**
  - Clustering.
  - Dimensionality reduction.
- Reinforcement Learning:
  - Value and policy iteration.
  - Q Learning.

# Recap: Hypothesis Space Search by Find-S




**h0 = < Ø, Ø, Ø, Ø, Ø, Ø >**

**x1 = < Sunny, Warm, Normal, Strong, Warm, Same >, +**

**h1 = < Sunny, Warm, Normal, Strong, Warm, Same >**

**x2 = <Sunny, Warm, High, Strong, Warm, Same>, +**

**h2 =< Sunny, Warm, ?, Strong, Warm, Same >**

**x3 = < Rainy, Cold, High, Strong, Warm, Change >, -**

**h3 =< Sunny, Warm, ?, Strong, Warm, Same >**

**x4 = < Sunny, Warm, High, Strong, Cool, Change >, +**

**h4 = < Sunny, Warm, ?, Strong, ?, ? >

# Version Spaces

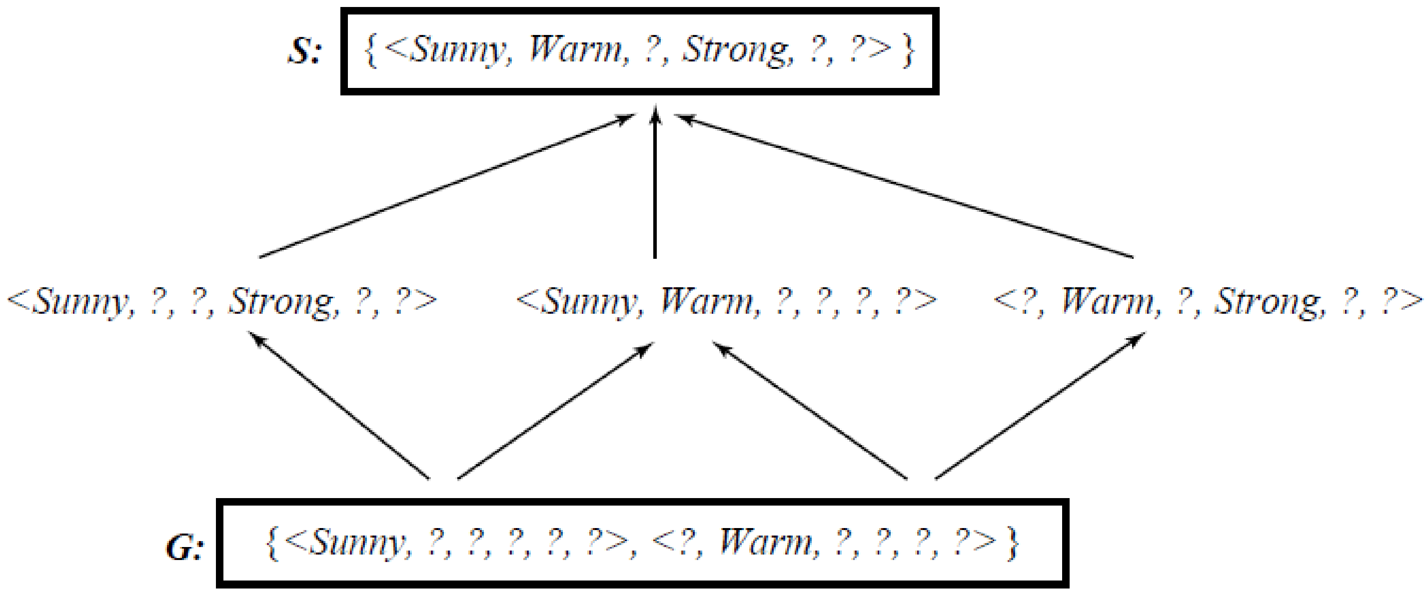
- ***Hypothesis h*** :
  - Is **consistent** with a set of training examples *D* of target concept *c if and only if h( x ) = c( x )* for each training example < *x, c( x )* > in *D*.

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

- ***Version space*** $VS_{H,D}$ :
  - With respect to hypothesis space *H*, and training set *D*, $VS_{HD}$ is the subset of hypotheses from *H* that are consistent with all training examples (in D):

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

# Example Version Space



# List-Then Eliminate Algorithm

- *VersionSpace* → List containing every hypothesis in *H*
- For each training example < *x*, *c( x )* >
  - Remove from VersionSpace any hypothesis that is inconsistent with the training example $h(x) \neq c(x)$.
- Output the list of hypotheses in *VersionSpace*.
- Advantages? – Disadvantages?

# Representing Version Spaces

- ***General boundary G:*** Of version space $VS_{H,D}$ is the set of *maximally general members*.

- ***Specific boundary S:*** Of version space $VS_{H,D}$ is the set of *maximally specific members*.

- Every member of the version space lies between these boundaries:

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

- Where: ***x* ≥ *y*** means ***x*** is more general or equal to ***y***.

# Candidate Elimination Algorithm

*G* ← Maximally general hypotheses in *H*

*S* ← Maximally specific hypotheses in *H*

**FOR** each training example *d*

**IF** *d* is a +ve example **THEN**

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 generalisations *h* of *s* such that:

*h* is consistent with *d* **and** a member of *G* is more general than *h*

**Remove from *S*:** Any hypothesis that is *more general than* another hypothesis in *S*

**END FOR**

**END IF**

**END FOR**

# Candidate Elimination Algorithm

**IF** *d* is a -ve example **THEN:**

Remove from *S* any hypothesis that is 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:

*h* is consistent with *d* ***and***

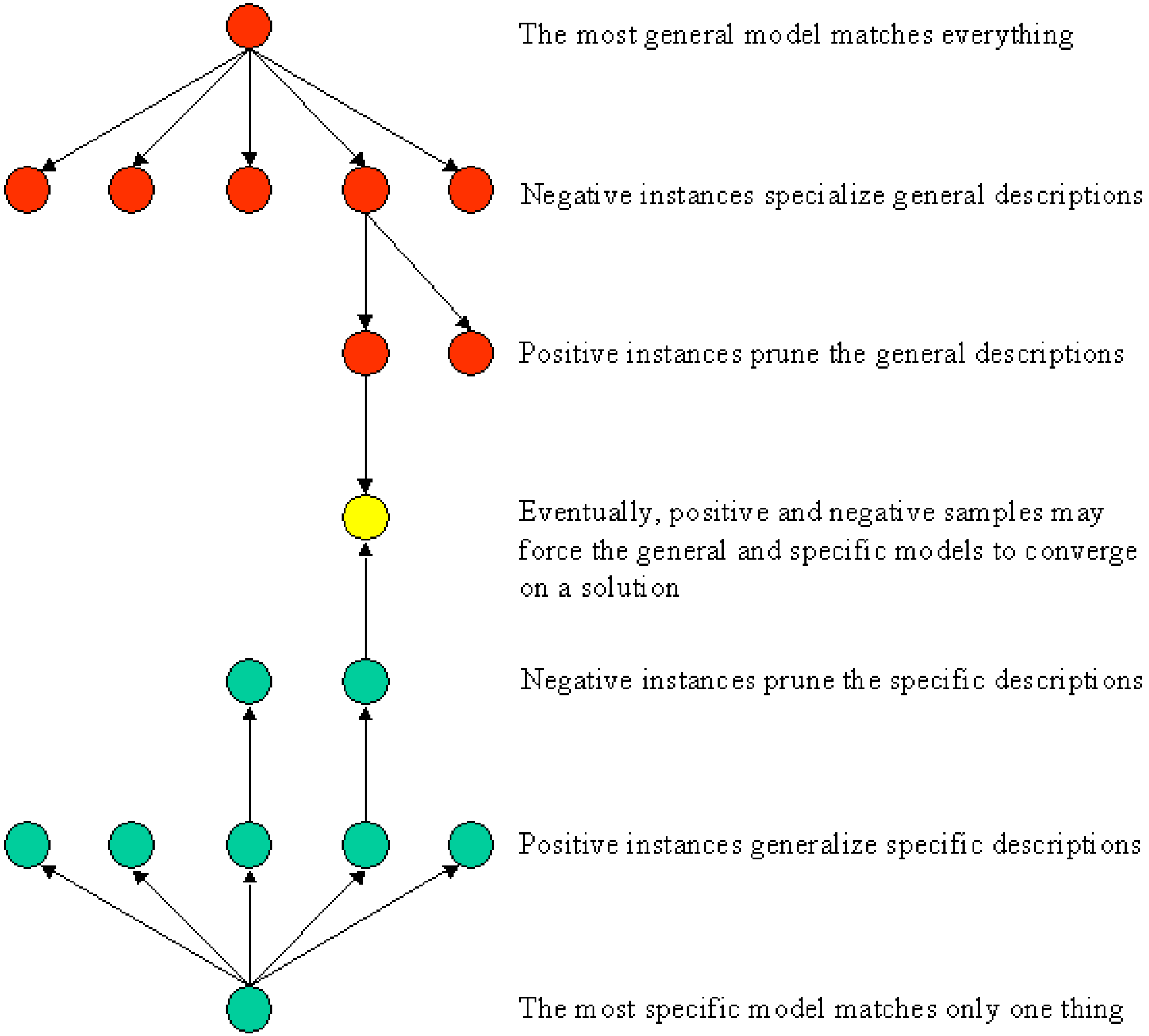
some member of *S* is more specific than *h*

Remove from *G* any hypothesis that is less general than another hypothesis in *G*

**END FOR**

**END IF**

# Version Space Diagram



# Learn concept: "Japanese Economy Car"

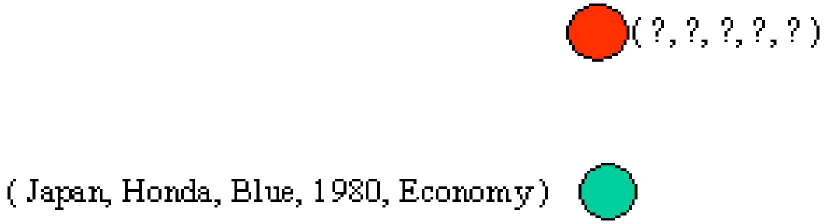
**Features:** ( Country of Origin, Manufacturer, Color, Decade, Type )

| *Origin* | *Manufacturer* | *Colour* | *Decade* | *Type* | *Example Type* |
|---|---|---|---|---|---|
| Japan | Honda | Blue | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1970 | Sports | Negative |
| Japan | Toyota | Blue | 1990 | Economy | Positive |
| USA | Chrysler | Red | 1980 | Economy | Negative |
| Japan | Honda | White | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1980 | Economy | Positive |
| Japan | Honda | Red | 1990 | Economy | Negative |

# Learn concept: "Japanese Economy Car"

1. **Positive Example:** ( Japan, Honda, Blue, 1980, Economy ):

| | |
|---|---|
| Initialise G to a singleton set that includes everything. | G = { (?, ?, ?, ?, ?) } |
| Initialise S to a singleton set that includes the first positive example. | S = { (Japan, Honda, Blue, 1980, Economy) } |

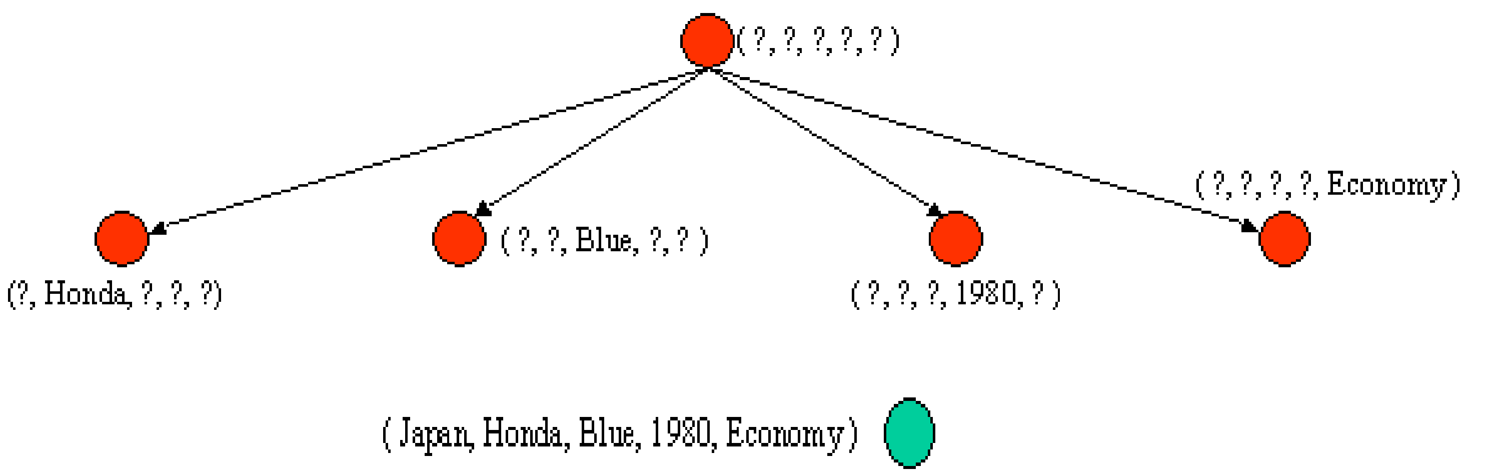


# Learn the concept: "Japanese Economy Car"

**2. Negative Example:** ( Japan, Toyota, Green, 1970, Sports ):

Specialise G to exclude the -ve example.

| | |
|---|---|
| G = | { (?, Honda, ?, ?, ?),<br>(?, ?, Blue, ?, ?),<br>(?, ?, ?, 1980, ?),<br>(?, ?, ?, ?, Economy) } |
| S = | { (Japan, Honda, Blue, 1980, Economy) } |



# Learn concept: "Japanese Economy Car"

**Features:** ( Country of Origin, Manufacturer, Color, Decade, Type )

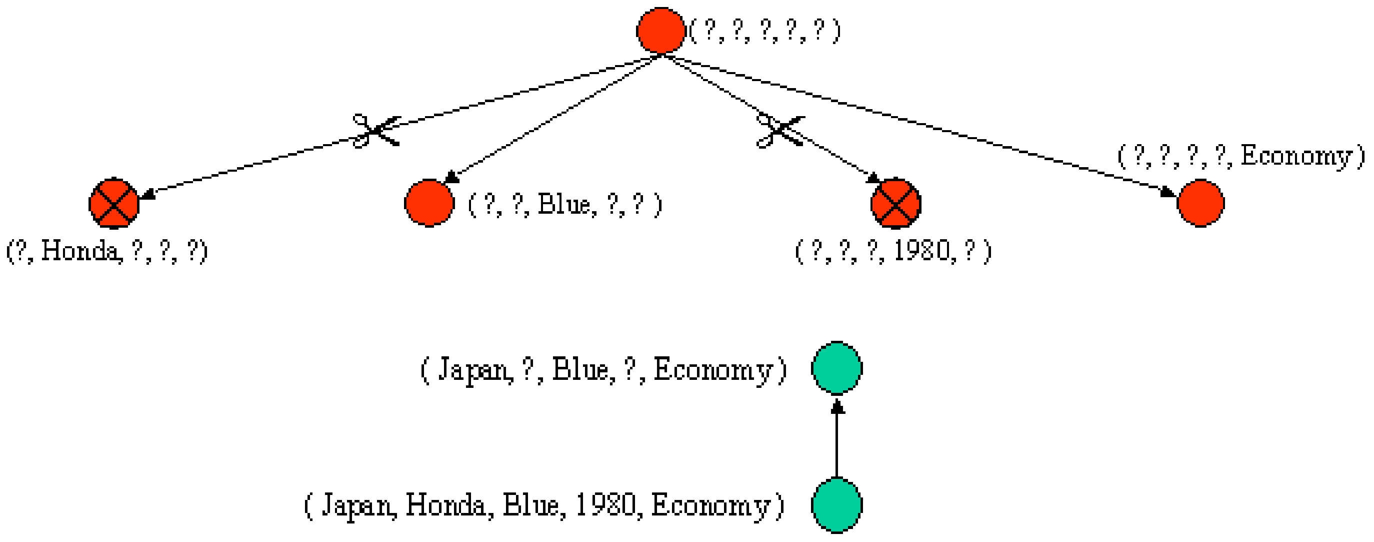
| *Origin* | *Manufacturer* | *Colour* | *Decade* | *Type* | *Example Type* |
|---|---|---|---|---|---|
| Japan | Honda | Blue | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1970 | Sports | Negative |
| Japan | Toyota | Blue | 1990 | Economy | Positive |
| USA | Chrysler | Red | 1980 | Economy | Negative |
| Japan | Honda | White | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1980 | Economy | Positive |
| Japan | Honda | Red | 1990 | Economy | Negative |

# Learn concept: "Japanese Economy Car"

**3. Positive Example:** ( Japan, Toyota, Blue, 1990, Economy ):

- Prune G to exclude descriptions inconsistent with +ve example.
- Generalise S to include the positive example.

| | |
|---|---|
| G = | { (?, ?, Blue, ?, ?),<br>(?, ?, ?, ?, Economy) } |
| S = | { (Japan, ?, Blue, ?, Economy) } |



# Learn the concept: "Japanese Economy Car"

Features: ( Country of Origin, Manufacturer, Color, Decade, Type )

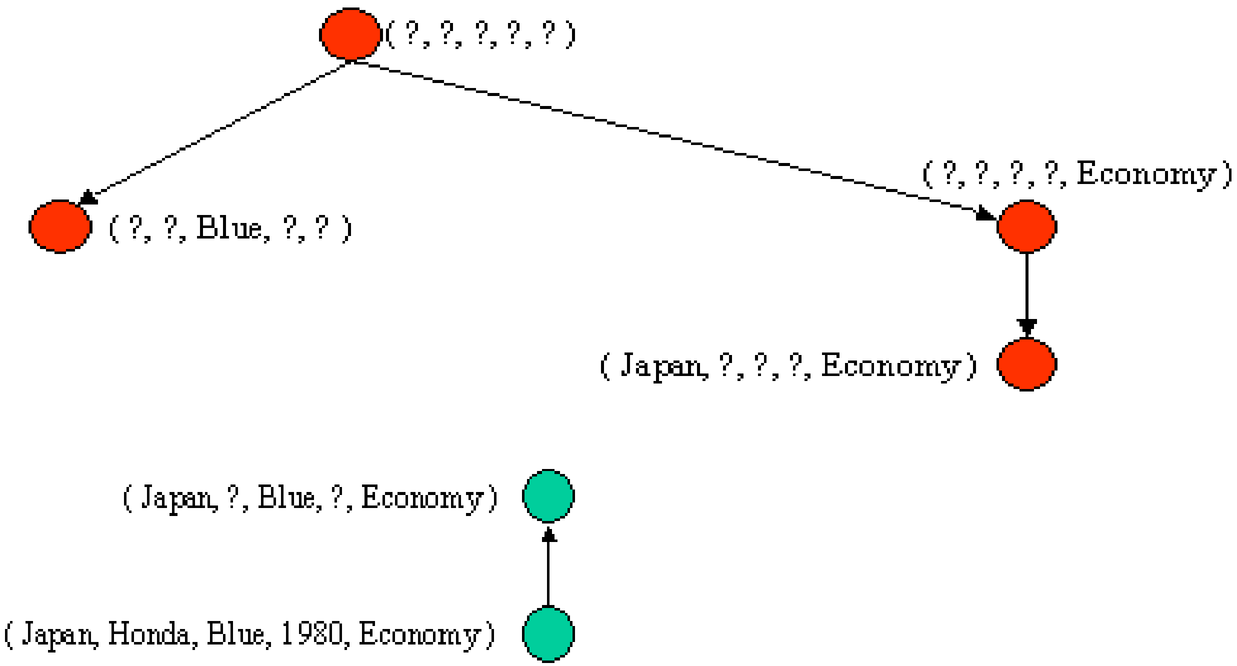
| ***Origin*** | ***Manufacturer*** | ***Colour*** | ***Decade*** | ***Type*** | ***Example Type*** |
|---|---|---|---|---|---|
| Japan | Honda | Blue | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1970 | Sports | Negative |
| Japan | Toyota | Blue | 1990 | Economy | Positive |
| USA | Chrysler | Red | 1980 | Economy | Negative |
| Japan | Honda | White | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1980 | Economy | Positive |
| Japan | Honda | Red | 1990 | Economy | Negative |

# Learn the concept: "Japanese Economy Car"

**4. Negative Example:** ( USA, Chrysler, Red, 1980, Economy ):

- Specialise *G* to exclude *-ve* example ( but stay consistent with *S* ).

| | |
|---|---|
| G = | { (?, ?, Blue, ?, ?),<br>(Japan, ?, ?, ?, Economy) } |
| S = | { (Japan, ?, Blue, ?, Economy) } |



# Learn the concept: "Japanese Economy Car"

Features: ( Country of Origin, Manufacturer, Color, Decade, Type )

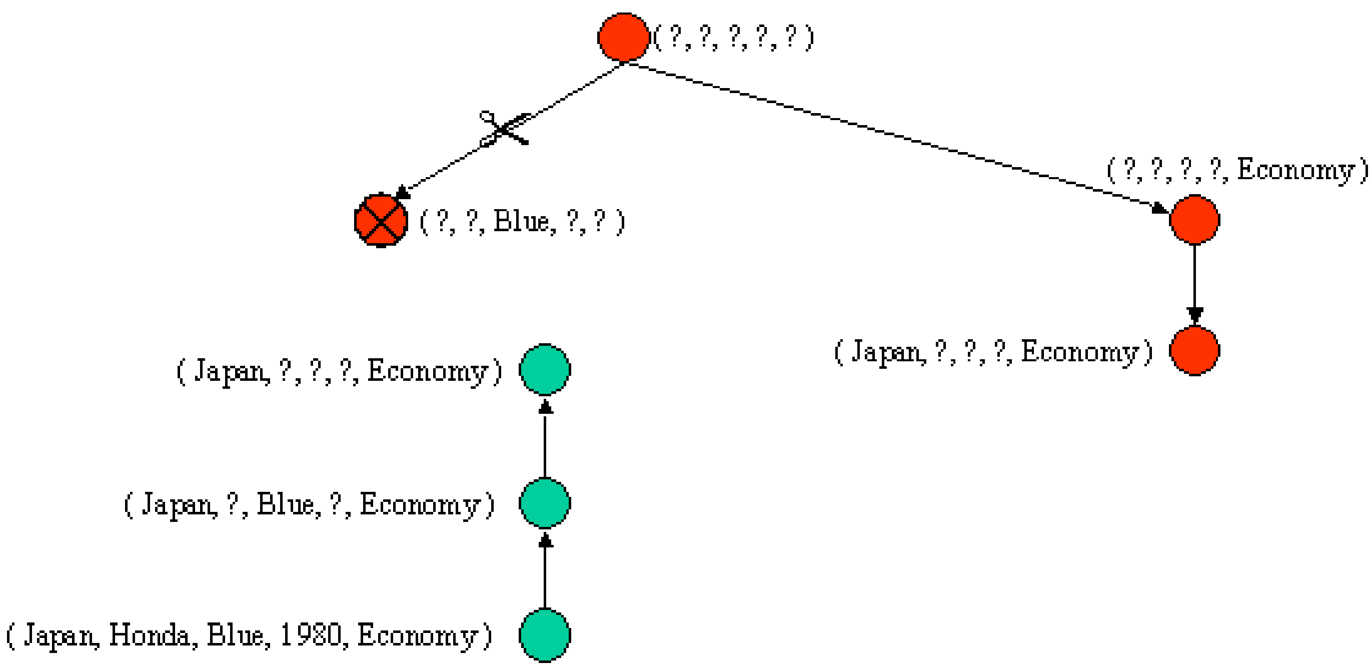
| *Origin* | *Manufacturer* | *Colour* | *Decade* | *Type* | *Example Type* |
|---|---|---|---|---|---|
| Japan | Honda | Blue | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1970 | Sports | Negative |
| Japan | Toyota | Blue | 1990 | Economy | Positive |
| USA | Chrysler | Red | 1980 | Economy | Negative |
| Japan | Honda | White | 1980 | Economy | Positive |
| Japan | Toyota | Green | 1980 | Economy | Positive |
| Japan | Honda | Red | 1990 | Economy | Negative |

# Learn the concept: "Japanese Economy Car"

**5. Positive Example:** ( Japan, Honda, White, 1980, Economy ):

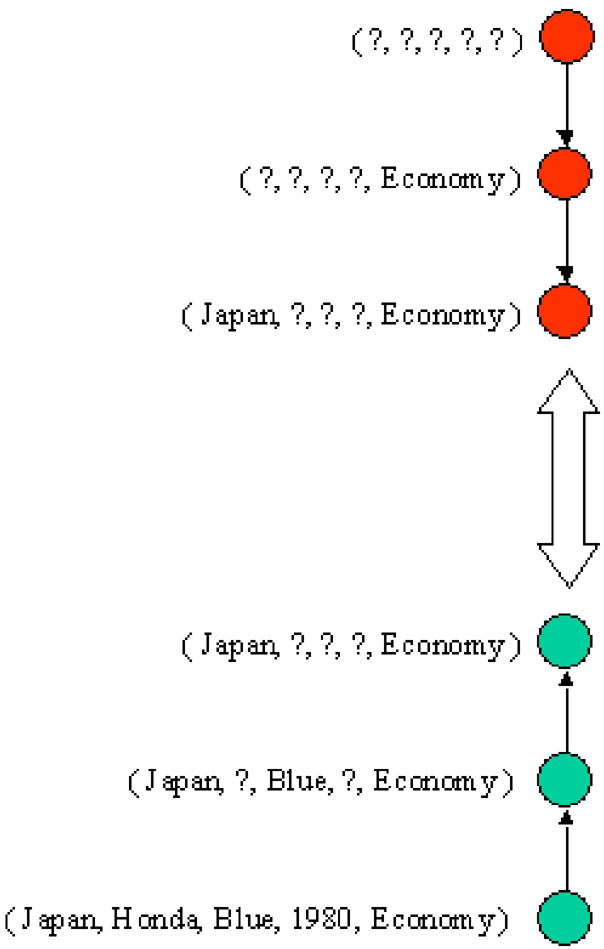
- Prune *G* to exclude descriptions inconsistent with +*ve* example.
- Generalise *S* to include positive example.

| | |
|---|---|
| G = | (Japan, ?, ?, ?, Economy) } |
| S = | { (Japan, ?, ?, ?, Economy) } |



# Learn concept: "Japanese Economy Car"

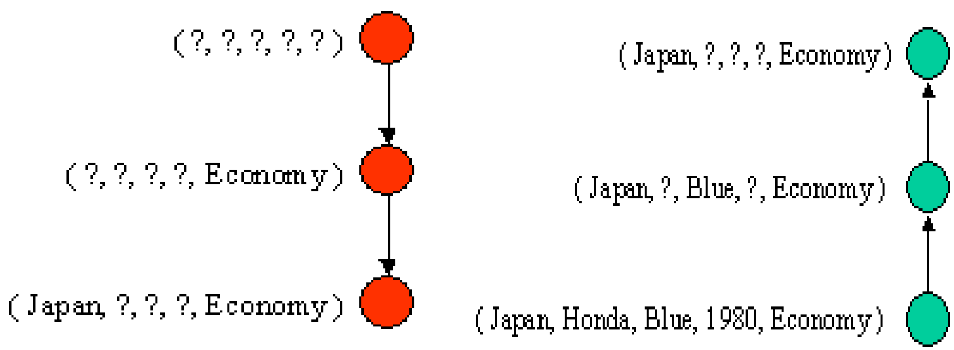
- *G* and *S* are singleton sets and *S* = *G*.



# Learn concept: "Japanese Economy Car"

**6. Positive Example:** ( Japan, Toyota, Green, 1980, Economy ):

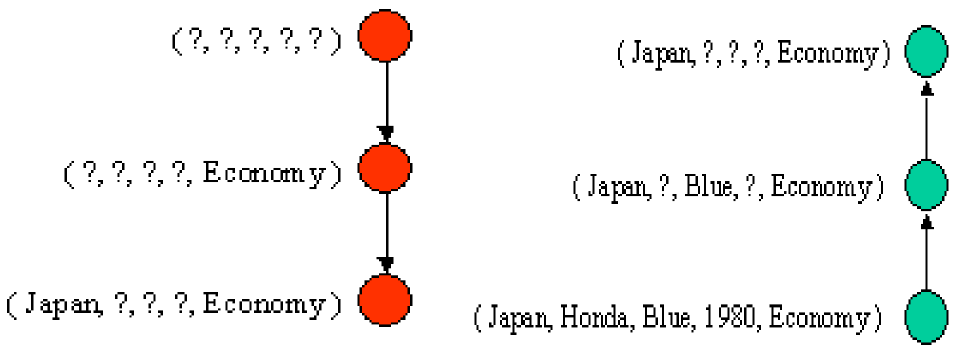
| | |
|---|---|
| New example is consistent with version-space, so no change is made. | G = { (Japan, ?, ?, ?, Economy) }<br>S = { (Japan, ?, ?, ?, Economy) } |




**7. Negative Example:** ( Japan, Honda, Red, 1990, Economy ):

**???**

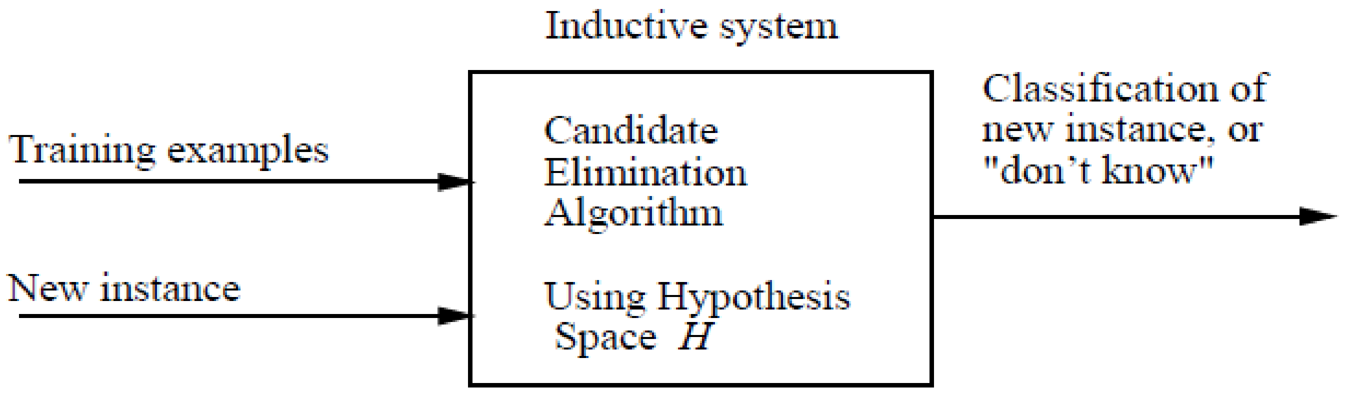
# Learn concept: "Japanese Economy Car"




**7. Negative Example:** ( Japan, Honda, Red, 1990, Economy ):

| | |
|---|---|
| Example is inconsistent with the version-space. | G cannot be specialised. S cannot be generalised. |

# Problems with Candidate Elimination Algorithm?



# CEA Learner Properties

- If there is a consistent hypothesis then the algorithm will converge to $S = G = \{ h \}$ when enough examples are provided.
- "Noisy" examples may cause the removal of the correct *h*.
- If too many examples are inconsistent, *S* and *G* become empty.
- This can also happen, when concept to be learned is not in *H*.
- **Assumes:**
  - There are no errors in the training examples.
  - There is some h in H that correctly describes the target concept.

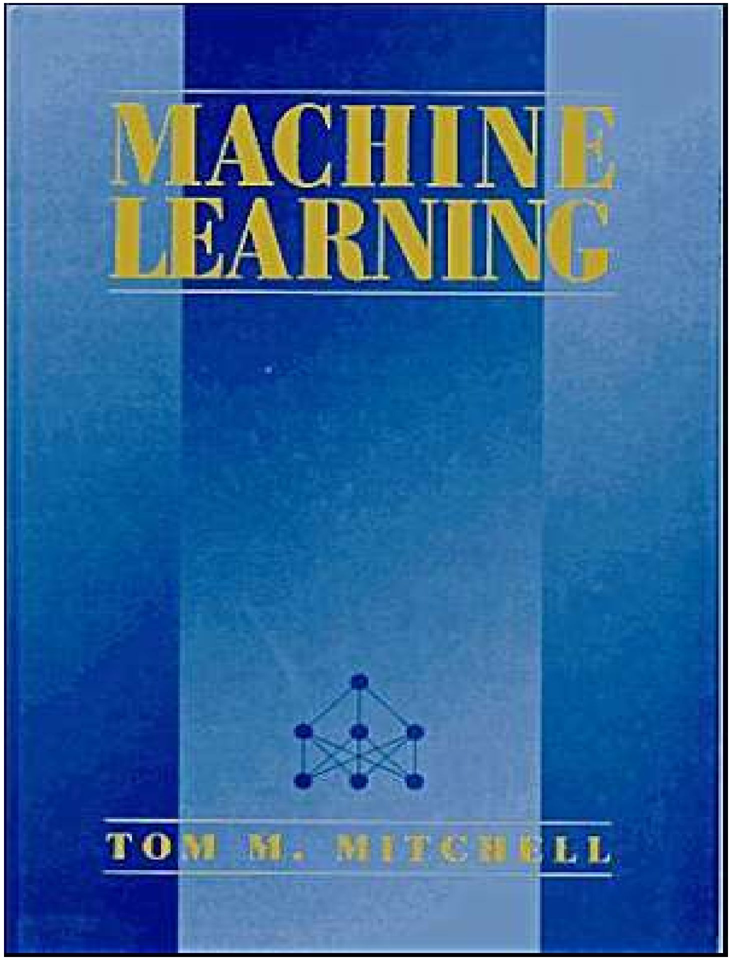
# Inductive Bias

- Concept learners classify unseen examples because of implicit inductive bias.
- Bias is that the target concept can be found in the hypothesis space ( $c \in H$ ).
- More bias implies more generality in classifying unseen examples.
- If there is a hypothesis corresponding to every possible instance, this removes inductive bias from *Candidate-Elimination*.
- This also removes the ability to classify any instance beyond the observed training examples.
- Unbiased learner cannot classify unseen examples (no induction!).

# Concept Learners → Bias !

- Inductive bias of:
  - **Rote learner:** Store examples, Classify x *iff* it matches previously observed example.
    - No inductive bias ( → no generalisation! )
  - **Candidate Elimination Algorithm:**
    - *c* is in *H*.
  - **Find-S:**
    - *c* is in *H*.
- *Inductive leaps possible only if learner is biased*.

# ML: Reading




***Chapter 2: Concept Learning***