CS 584 – Machine Learning

Topic: Concept Learning

Mustafa Bilgic

http://www.cs.iit.edu/~mbilgic

https://twitter.com/bilgicm

Acknowledgements

- Several figures are from http://www.cs.cmu.edu/~tom/mlbook.html
- When I copied a Table/Figure from the book, I also copied the Table/Figure number so that you can find its location in the book

Simple Exercise

You are training a model for loan approval/rejection. Inputs are: MissedPayments, Income, Wealth. The output is the decision to approve or reject the loan application. Here is a training dataset.

MissedPayments	Income	Wealth	Decision
Yes	Low	Low	Reject
No	High	High	Accept
No	Low	Low	Reject
No	Low	High	Accept

Here are the test cases. How should your model decide on these cases? Why?

MissedPayments	Income	Wealth	Decision
Yes	Low	High	??
Yes	High	Low	??
Yes	High	High	??
No	High	Low	??

Motivation

- Induce a general function from specific training examples
  - Concept: spam; training examples: emails labeled as spam/~spam
  - Concept: heartDisease; training examples: patient records labeled as heartDisease/~heartDisease
  - Concept: positive; training examples: reviews labeled as positive/negative
  - …
- Goal: induce a general function that fits to the training data well *and* generalizes well to unseen/future data

Problem Formulation

- Define a space of hypotheses / functions
- Search for a hypothesis/function that fits well to the training data
- To search efficiently, utilize the structure of the hypothesis space
  - In this chapter, we will utilize *general-to-specific ordering* of hypotheses

Reading

- Tom Mitchell, Machine Learning
  - Chapter 2

The Million-Dollar Question

- What does the hypothesis (the classification function) even look like? What is the hypothesis space?
- Is it
  - a logical formula of conjunctions (ANDs) and disjunctions (ORs)?
  - a set of rules?
  - a decision tree?
  - a linear additive function where each feature has its own weight, and it contributes one way or the other?
  - a non-linear function?

Simple Exercise

You are training a model for loan approval/rejection. Inputs are: MissedPayments, Income, Wealth. The output is the decision to approve or reject the loan application. Here is a training dataset.

MissedPayments	Income	Wealth	Decision
Yes	Low	Low	Reject
No	High	High	Accept
No	Low	Low	Reject
No	Low	High	Accept

What does the decision function look like?

- A set of rules?
- A decision tree?
- An additive function where each feature has a positive/negative contribution?
- A nonlinear function?

How Many Functions to Choose From?

- Assume one input variable, and one output variable
  - MissedPayments: Yes/No
  - Decision: Approve/Reject
- How many possible decision functions are there?
- Add one more binary variable; how many functions are there?
- Add the third binary variable; how many functions are there now?
- See OneNote

ENJOYSPORT EXAMPLE

3 x 2 x 2 x 2 x 2 x 2

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

TABLE 2.1
Positive and negative training examples for the target concept *EnjoySport*.

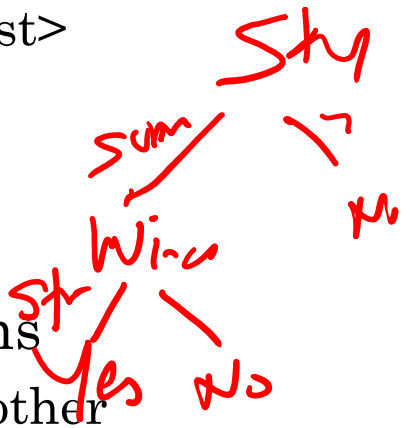
Sky: Sunny, Cloudy, Rainy
AirTemp: Warm, Cold
Humidity: Normal, High
Wind: Strong, Weak
Water: Warm, Cold
Forecast: Same, Change

Instance
2

1. What do you think the decision function is?
2. What is the size of the instance space?
3. How many possible decision functions are there?

Hypothesis Representation

- $h(x)$ = 1 if EnjoySport is Yes and 0 otherwise
- How should we represent $h(.)$?
- Let's start with a simple one
- A conjunction (and) of constraints on the attributes
  - <Sky, AirTemp, Humidity, Wind, Water, Forecast>
  - ? indicates any value is acceptable
  - A specific value means it must be that value
  - ϕ means no value is acceptable
- For example <Sunny, ?, ?, Strong, ?, ?> means
  - Sky must be Sunny, Wind must be Strong, and other attributes can be any value
- $c(x)$ is the true function that is not given, except on training data



Propositional Logic Review

Propositional Logic Language

- Propositional symbols
  - P, Q, R, …
  - True, False
- Connectives
  - $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftarrow$, $\Leftrightarrow$

More on the Connectives

- $\neg$ (negation)
- $\wedge$ (and): P $\wedge$ Q is called a **conjunction** and P and Q are **conjuncts**
- $\vee$ (or): P $\vee$ Q is called a **disjunction** and P and Q are **disjuncts**
- $\Rightarrow$ (implies): P $\Rightarrow$ Q is called an **implication**. P is the **premise** or **antecedent** and Q is its **conclusion** or **consequent**
- $\Leftrightarrow$ (if and only if): P $\Leftrightarrow$ Q is called a **biconditional**

Syntax

- Sentence → AtomicSentence | ComplexSentence
- AtomicSentence → True | False | P | Q | R | …
- ComplexSentence →
  - ¬ sentence
  - sentence ∧ sentence
  - sentence ∨ sentence
  - sentence ⇒ sentence
  - sentence ⇔ sentence
- Operator precedence: ¬, ∧, ∨, ⇒, ⇔

Semantics

- Semantics define the rules for determining the truth of a sentence with respect to a **model**
- A model fixes the truth value of for every proposition
- In other words, a model assigns a value (True or False) to every predicate in propositional logic

Semantics of Propositional Logic

- It specifies how to determine the truth value of any sentence in a model *m*
- The truth value of *True* is *True*
- The truth value of *False* is *False*
- The truth value of each atomic sentence is given by *m*
- The truth value of every other sentence is obtained recursively by using truth tables

Truth Rules

- $\neg P$ is true iff P is false
- $P \wedge Q$ is true iff both P and Q are true
- $P \vee Q$ is true iff either P or Q or both are true
- $P \Rightarrow Q$ is true unless P is true and Q is false
- $P \Leftrightarrow Q$ is true iff P and Q have equal truth values; i.e., iff both P and Q are true or iff both P and Q are false

Truth Tables

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Possible Confusion about ⇒

- P⇒Q is true unless P is true and Q is false
- If 5 is odd, then 7 is odd. T or F?
- If 5 is odd, then 10 is odd. T or F?
- If 5 is even, then 7 is even. T or F?
- If 5 is even, then 7 is odd. T or F?
- If 5 is even, then 10 is even. T or F?
- If 5 is even, then Chicago is the capital of US. T or F?
- **The key: Logic does not assume any causation or correlation between P and Q.**

Back to Concept Learning

ENJOYSPORT PROBLEM FORMULATION

- **Given:**
  - Instances X: Possible days, each described by the attributes
    - *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*), and
    - *Forecast* (with values *Same* and *Change*).
  - Hypotheses H: Each hypothesis is described by a conjunction of constraints on the attributes *Sky*, *AirTemp*, *Humidity*, *Wind*, *Water*, and *Forecast*. The constraints may be "?" (any value is acceptable), "$\emptyset$" (no value is acceptable), or a specific value.
  - Target concept c: $EnjoySport : X \rightarrow \{0, 1\}$
  - Training examples D: Positive and negative examples of the target function (see Table 2.1).
- **Determine:**
  - A hypothesis h in H such that $h(x) = c(x)$ for all x in X.

TABLE 2.2
The *EnjoySport* concept learning task.

True Concept

- Let the true concept be $c(x)$
- We do not know what $c(x)$ is
- All we have is a training dataset D that consists of $<x, c(x)>$ pairs
- We define a hypothesis space H, for which we hope $c \in H$, and we search for $h \in H$ such that
  - $h(x) = c(x)\ \forall x \in D$

Most-general and Most-specific

- Most-general hypothesis, i.e., the hypothesis where $h(x) = 1 \; \forall x \in X$
  - <?, ?, ?, ?, ?, ?>
- Most-specific hypothesis, i.e., the hypothesis where $h(x) = 0 \; \forall x \in X$
  - <ϕ, ϕ, ϕ, ϕ, ϕ, ϕ>
- Note that in this formulation, $h(x) = 0 \; \forall x \in X$ when at least one entry is ϕ. For example, $h(x) = 0 \; \forall x \in X$ when h = <?, ?, ϕ, ?, ?, ?>.

'More-General' Relation

- Let h_j and h_k be Boolean-valued functions defined over X. Then h_j is *more general than or equal to* h_k (written has $h_j \geq h_k$) if and only if
  - $(\forall x \in X)\left[h_k(x) = 1 \Rightarrow h_j(x) = 1\right]$
  - That is, whenever h_k says positive, h_j also says positive; h_j might say positive to other instances that h_k says negative

The Instance and Hypothesis Space

- Six attributes:
  - Sky has three possible values, others have two possible values
- Total number of possible instances
  - $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$
- One hypothesis
  - Each attribute is ?, ϕ, or a specific value
- Total number of syntactically-different hypotheses
  - $5 \times 4 \times 4 \times 4 \times 4 \times 4 = 5120$
- Any hypothesis that contains at least one ϕ has the same meaning; i.e., it classifies all instances as negative
- Total number of semantically-different hypotheses
  - $(4 \times 3 \times 3 \times 3 \times 3 \times 3) + 1 = 973$
- How do we search this space efficiently?

The Instance and Hypothesis Space

- For the loan approval example
  - The size of the instance space?
  - The number of syntactically-different hypotheses?
  - The number of semantically-different hypotheses?

Algorithms

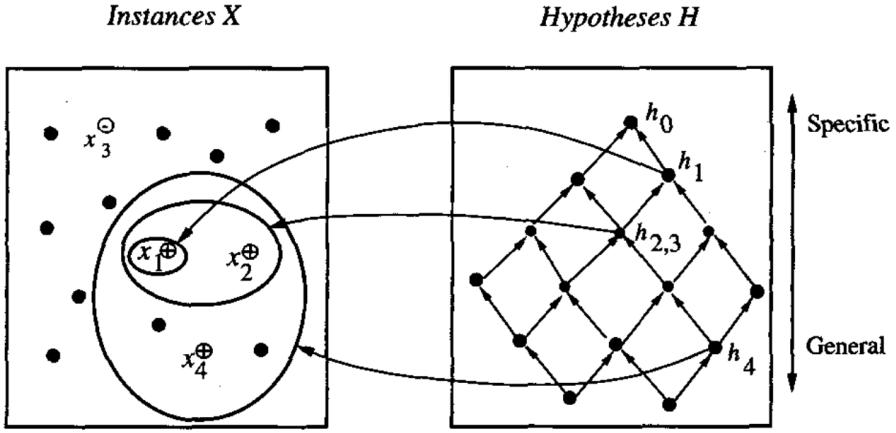
- Find-S
- List-Then-Eliminate
- Candidate-Elimination

FIND-S

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 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

TABLE 2.3
FIND-S Algorithm.

FIND-S TRACE



$h_0 = \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$

$x_1 = \langle Sunny\ Warm\ Normal\ Strong\ Warm\ Same \rangle, +$ $h_1 = \langle Sunny\ Warm\ Normal\ Strong\ Warm\ Same \rangle$

$x_2 = \langle Sunny\ Warm\ High\ Strong\ Warm\ Same \rangle, +$ $h_2 = \langle Sunny\ Warm\ ?\ Strong\ Warm\ Same \rangle$

$x_3 = \langle Rainy\ Cold\ High\ Strong\ Warm\ Change \rangle, -$ $h_3 = \langle Sunny\ Warm\ ?\ Strong\ Warm\ Same \rangle$

$x_4 = \langle Sunny\ Warm\ High\ Strong\ Cool\ Change \rangle, +$ $h_4 = \langle Sunny\ Warm\ ?\ Strong\ ?\ ? \rangle$

FIGURE 2.2
The hypothesis space search performed by FIND-S. The search begins (h_0) with the most specific hypothesis in H, then considers increasingly general hypotheses (h_1 through h_4) as mandated by the training examples. In the instance space diagram, positive training examples are denoted by "+," negative by "–," and instances that have not been presented as training examples are denoted by a solid circle.

Problems with Find-S

- How do we know the $h(.)$ returned by Find-S is the actual $c(.)$?
- If there are more than one hypotheses that are consistent with data, Find-S finds only the most specific one. Why settle for the most specific one? For example, why not the most general one?
- What happens when the training data has errors?
- What if the most-specific hypothesis is not unique?

Version Space

- A hypothesis $h()$ is consistent with a set of training examples D and a target concept $c()$ if and only if $h()$ agrees with $c()$ on each training example in D
  - $Consistent(h, D) \equiv (\forall \langle x, c(x) \rangle \in D) h(x) = c(x)$
- Version space with respect to hypothesis space H and dataset D is the set of all hypotheses in H that are consistent with all examples in D
  - $VS_{H,D} \equiv \{h \in H | Consistent(h, D)\}$

VERSION SPACE

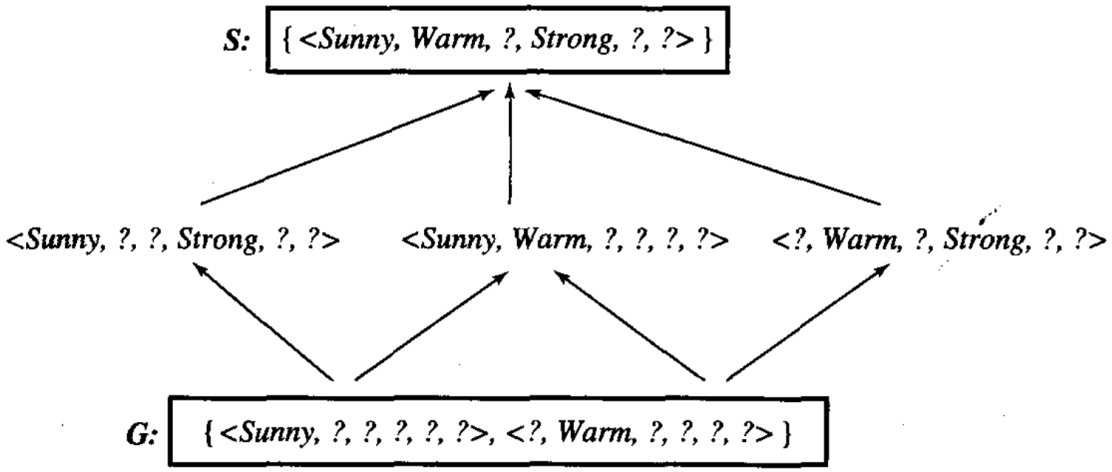


FIGURE 2.3
A version space with its general and specific boundary sets. The version space includes all six hypotheses shown here, but can be represented more simply by S and G. Arrows indicate instances of the *more_general_than* relation. This is the version space for the *EnjoySport* concept learning problem and training examples described in Table 2.1.

List-Then-Eliminate

1. $VS \leftarrow$ a list of containing every hypothesis in H
2. For each training example, $\langle x, c(x) \rangle$
   1. Remove from VS any hypothesis h for which $h(x) \neq c(x)$
3. Output VS

- Advantages
  - Outputs all consistent hypotheses
- Disadvantages
  - Need to list all possible hypotheses
    - Impossible for infinite hypothesis spaces
    - Impractical for large hypothesis spaces

CANDIDATE-ELIMINATION

- **General boundary *G***: the set of maximally-general consistent hypotheses.
  - $G \equiv \{g \in H | Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' > g) \wedge Consistent(g', D)]\}$
- **Specific boundary *S*:** the set of maximally-specific consistent hypotheses.
  - $S \equiv \{s \in H | Consistent(s, D) \wedge (\neg \exists s' \in H)[(s > s') \wedge Consistent(s', D)]\}$
- **Version space representation theorem:** for every consistent hypothesis there is at least one more-general-or-equal-to hypothesis in G and there is at least one more-specific-or-equal-to hypothesis in S
  - $VS_{H,D} = \{h \in H | (\exists s \in S)(\exists g \in G)(g \geq h \geq s)\}$
- **Candidate elimination algorithm**
  - Start with S that has only the most-specific hypothesis and G that has only the most-general hypothesis, and modify S and G with each training data

CANDIDATE ELIMINATION ALGORITHM

Initialize G to the set of maximally general hypotheses in H
Initialize S to the set of 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
      - 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
- 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
      - 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

TABLE 2.5
CANDIDATE-ELIMINATION algorithm using version spaces. Notice the duality in how positive and negative examples influence S and G.

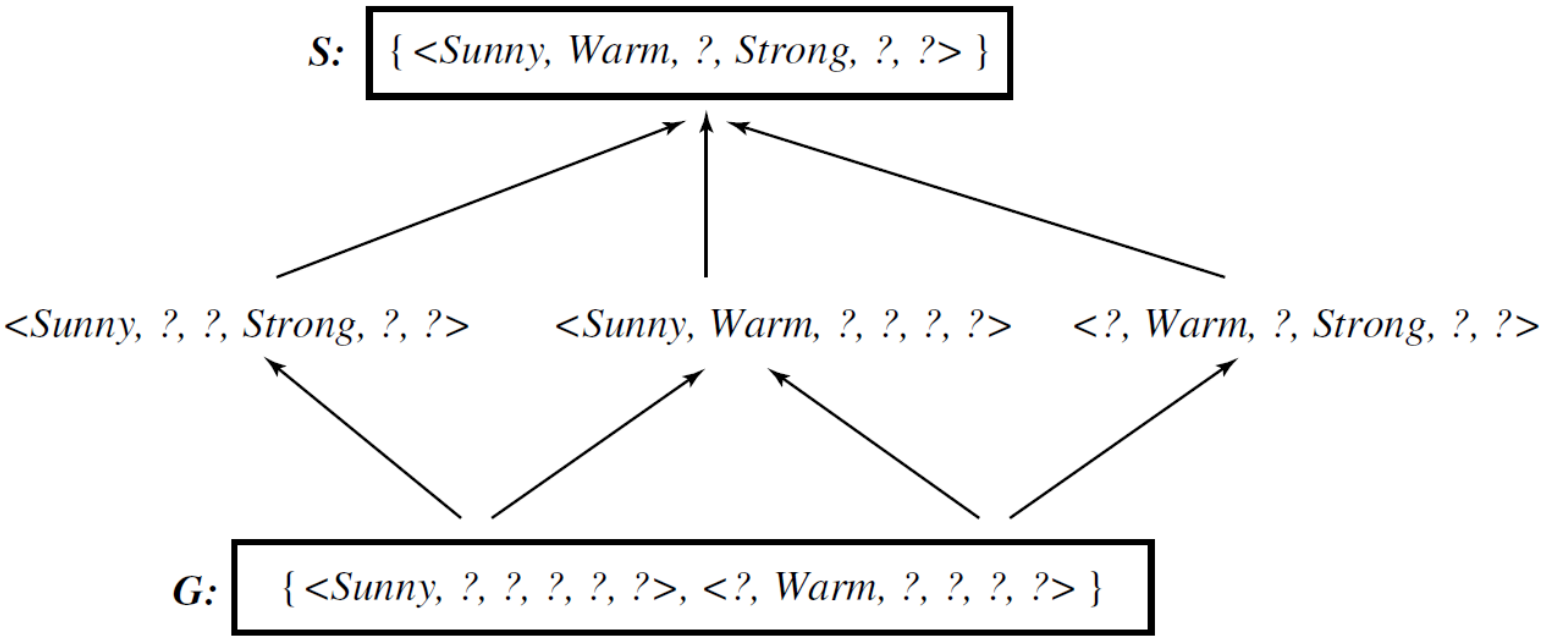
Running Example

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

TABLE 2.1
Positive and negative training examples for the target concept *EnjoySport*.

Trace the Candidate-Elimination algorithm on this dataset

SOLUTION


S:
{ <Sunny, Warm, ?, Strong, ?, ?> }
<Sunny, ?, ?, Strong, ?, ?>
<Sunny, Warm, ?, ?, ?, ?>
<?, Warm, ?, Strong, ?, ?>
G:
{ <Sunny, ?, ?, ?, ?, ?>, <?, Warm, ?, ?, ?, ?> }

Correct Hypothesis?

- If the training data does not contain errors, and, if the correct hypothesis is in H, then the candidate elimination algorithm will converge toward the correct hypothesis with each new training example
- If the training data contains errors, for example, let's say a positive example is annotated incorrectly as negative, then the correct hypothesis will surely be removed from the solution. Given enough data S and G might eventually become empty
- If the correct hypothesis is not in H, for example, if the correct hypothesis contains disjunctions whereas H contains only conjunctive hypotheses, given enough data, S and G might eventually become empty

Exercise

- Run the candidate elimination algorithm on the loan example

Given S and G, how do we classify a new example?

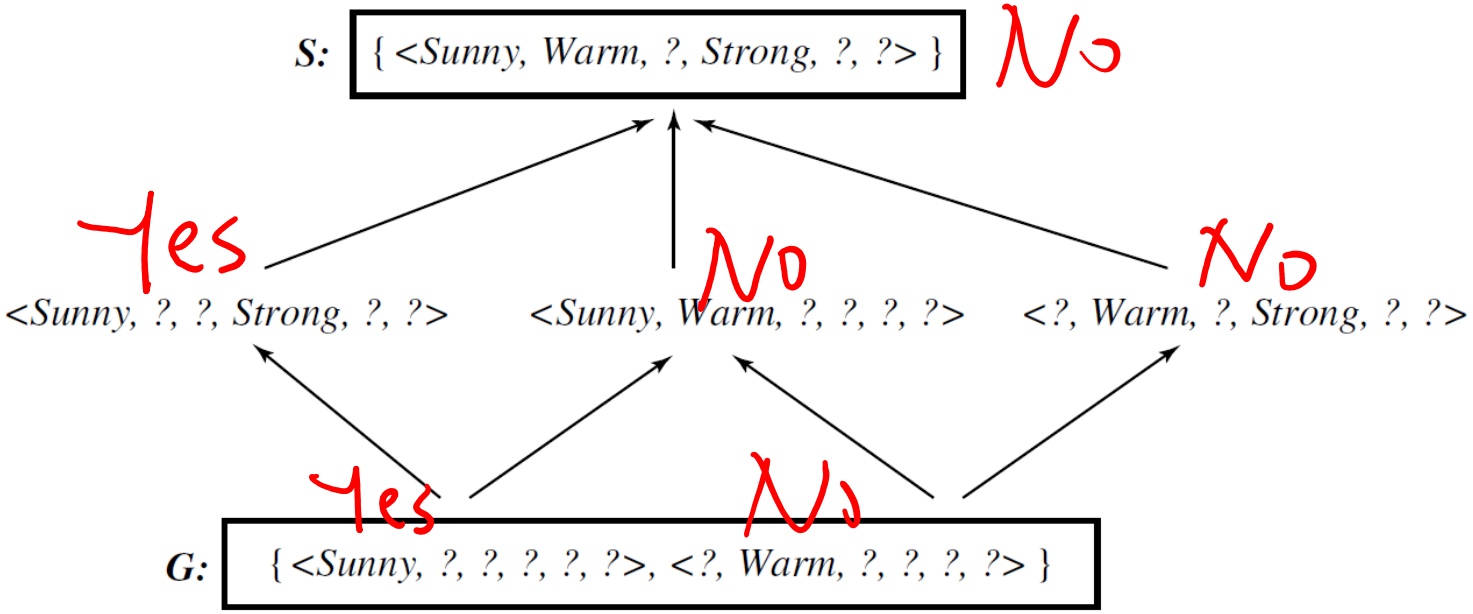
- If the version space contain only one hypothesis, then classification of a new example is straightforward
- What if the version space contains multiple hypotheses, like the one that we just saw?

CLASSIFY THE FOLLOWING

Instance	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
A	Sunny	Warm	Normal	Strong	Cool	Change	?
B	Rainy	Cold	Normal	Light	Warm	Same	?
C	Sunny	Warm	Normal	Light	Warm	Same	?
D	Sunny	Cold	Normal	Strong	Warm	Same	?

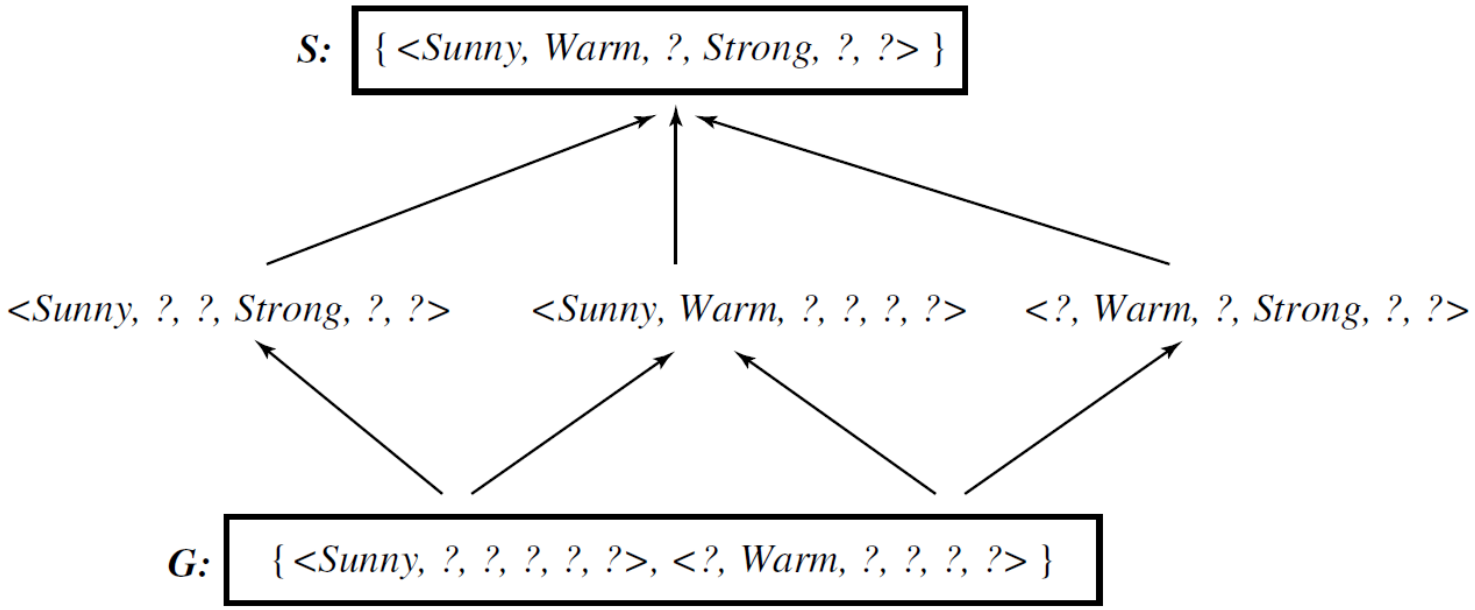
Yes
No
3 No, 3 Yes
4 No, 2 Yes

TABLE 2.6
New instances to be classified.



Active Learning

- Given the following version space, and if we give the algorithm the choice to choose the next example and ask for its label, what example should it ask about?



Inductive Bias

- We assumed that H is the conjunction of attributes. This is our inductive bias.
- What happens when the target concept is not in H?
- Can we avoid these problems by having a hypothesis space that has all possible hypotheses? That is, what if our hypothesis space is unbiased?
- First, how big is such a hypothesis space?
  - Given n Boolean attributes, there are 2^n possible examples
  - Each example can be a positive or negative example
  - Therefore, there are 2^{2^n} possible hypotheses!
- Second, how useful are such hypotheses?

Unbiased Learning

- H ≡ conjunctions, disjunctions, and negations
- Assume x_1, x_2, x_3 are positive and x_4 and x_5 are negative
- S is
  - $S = \{(x_1 \vee x_2 \vee x_3)\}$
- G is
  - $G = \{\neg(x_4 \vee x_5)\}$
- How do you classify a new/unseen example?

Biased vs Unbiased Learning

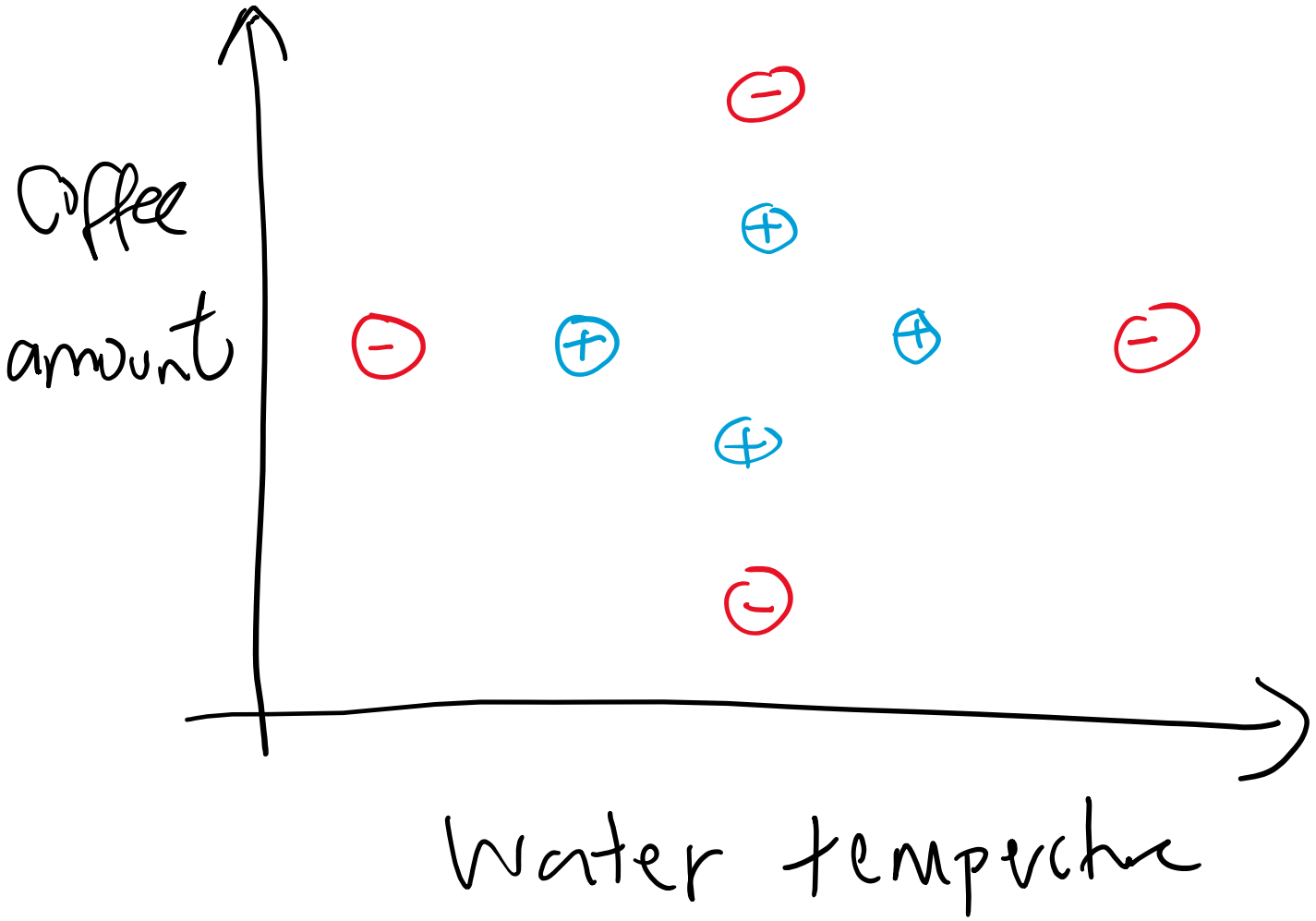
- In biased learning, we make assumptions about the hypothesis space
- In unbiased learning, no assumptions are made about the hypothesis space
- Purpose of concept learning: generalize to unseen data
- Unbiased learning simply memorizes the training data and it has no hope of generalizing to unseen data

Another Example

Is the Coffee Good?

- You grind and brew your own coffee at home
- You can control grind size, the amount of coffee, the water temperature, brewing time, etc.
- For simplicity, let's look at only the coffee amount and the water temperature
- Too little or too much coffee is not good
- Too cold or too hot water is not good
- What's best depends on the person
- You experiment with different coffee amounts and water temperatures and record your experience as positive or negative

EXPERIMENT



Experiment

- Assume the hypothesis space is the space of rectangles
- That is,
  - $h(a,t) = positive$ if $a_1 \leq a \leq a_2$ and $t_1 \leq t \leq t_2$
- What are S, G, and version space based on the training data?

S, G, Version Space, Max Margin