2. Principles of Data Mining

Contents of this Chapter

2.1 Learning from examples

2.2 Data mining as search in the hypothesis space

2.3 Inductive bias

2.4 Aspects of uncertainty

2.5 Data mining as optimization problem

2.6 Synopsis of Statistics, Machine Learning and Data Mining

2.1 Learning from Examples

Inductive Learning

- Data are instances (records) from an *instance space X*
  often: $X \subseteq D_1 \times \cdots D_d$ D_i: domain of attribute *i*
- Given a (relatively small) sample of data from *X* (*training data*)
- Given a *target function* specifying the learning goal
- Want to induce *general* hypotheses approximating the target function on the whole instance space from the *specific* training data

2.1 Learning from Examples

Inductive Learning

Fundamental assumption:

Any hypothesis approximating the target function well over the training data will also approximate the target function well over the unobserved instances of *X*.

2.1 Learning from Examples

Concept Learning

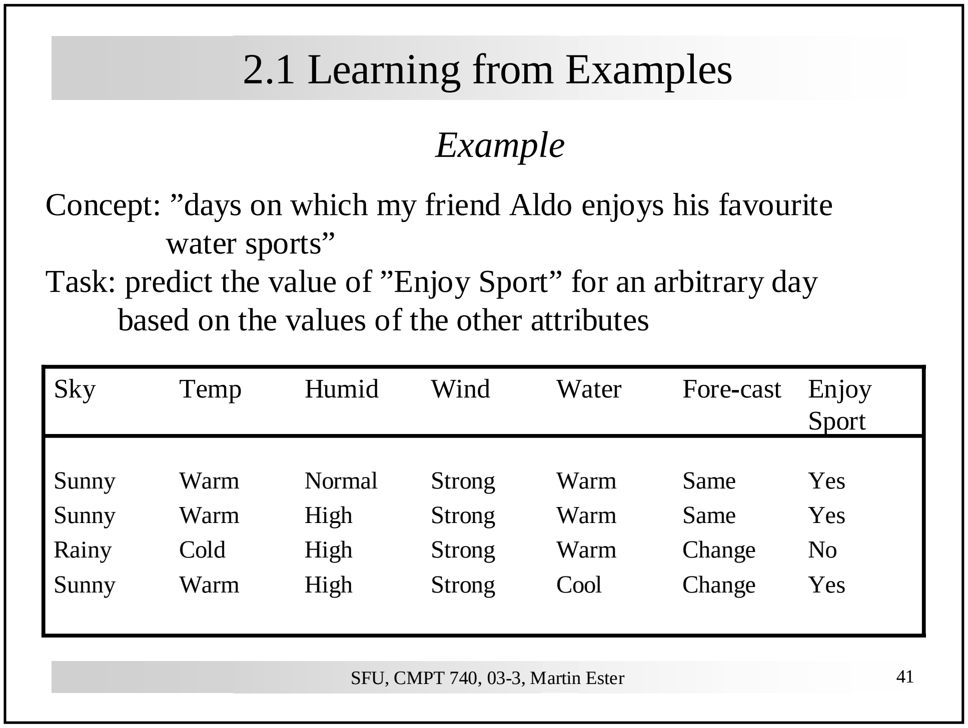
• Concept *C*: subset of *X*

c: $X \rightarrow \{0,1\}$ is the characteristic function of *C*

• Task:

approximate the *target function c* using the attributes of *X* in order to distinguish instances belonging / not belonging to *C*

• training data *D*: positive and negative examples of the concept: $<x_1,c(x_1)>,\ldots, <x_n,c(x_n)>$



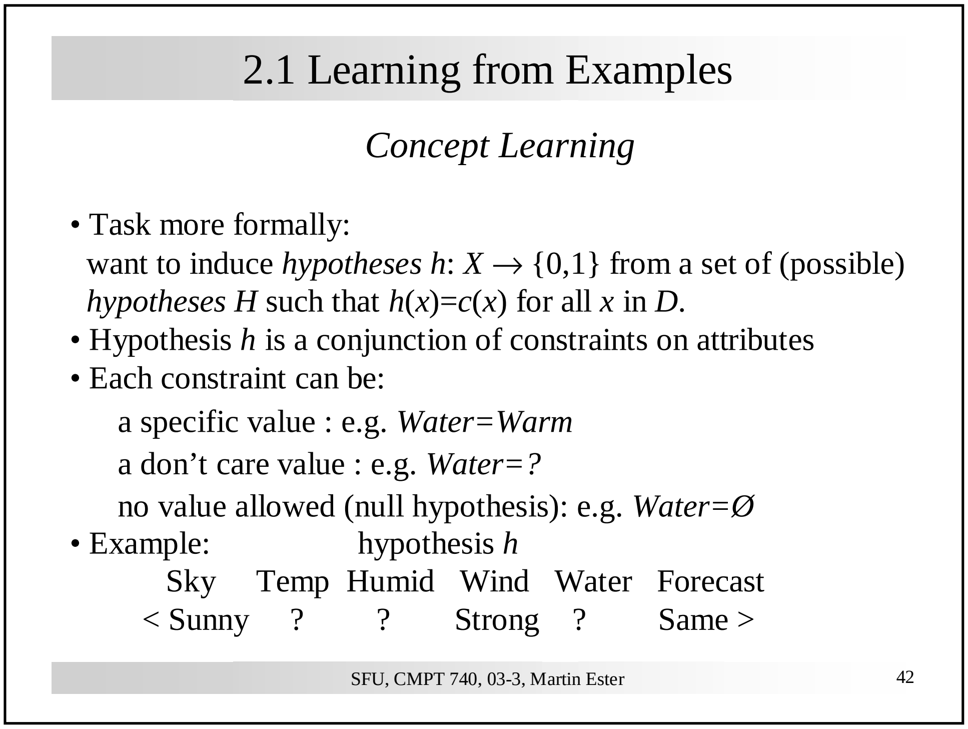
2.1 Learning from Examples

Example

Concept: ”days on which my friend Aldo enjoys his favourite water sports”

Task: predict the value of ”Enjoy Sport” for an arbitrary day based on the values of the other attributes

Sky	Temp	Humid	Wind	Water	Fore-cast	Enjoy Sport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes



2.1 Learning from Examples

Concept Learning

- Task more formally:
  want to induce *hypotheses* h: $X \rightarrow \{0,1\}$ from a set of (possible) *hypotheses* H such that $h(x)=c(x)$ for all x in D.
- Hypothesis h is a conjunction of constraints on attributes
- Each constraint can be:

  a specific value : e.g. *Water=Warm*

  a don’t care value : e.g. *Water=?*

  no value allowed (null hypothesis): e.g. *Water=Ø*
- Example: hypothesis *h*

Sky	Temp	Humid	Wind	Water	Forecast
< Sunny	?	?	Strong	?	Same >

2.2 Data Mining as Search in the Hypothesis Space

Example Hypothesis Space

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

# distinct instances : 3*2*2*2*2*2 = 96

# distinct concepts : 2^{96}

# syntactically distinct hypotheses : 5*4*4*4*4*4 = 5120

# semantically distinct hypotheses : 1+4*3*3*3*3*3 = 973

real life hypothesis spaces much larger!

2.2 Data Mining as Search in the Hypothesis Space

Ordering the Hypothesis Space

- Example:

  h_1 = < Sunny,?,?,Strong,?,?>
  h_2 = < Sunny,?,?,?,?,?>

- Sets of instances covered by h_1 and h_2:

  h_2 imposes fewer constraints than h_1 and therefore classifies more instances x as positive than h_1

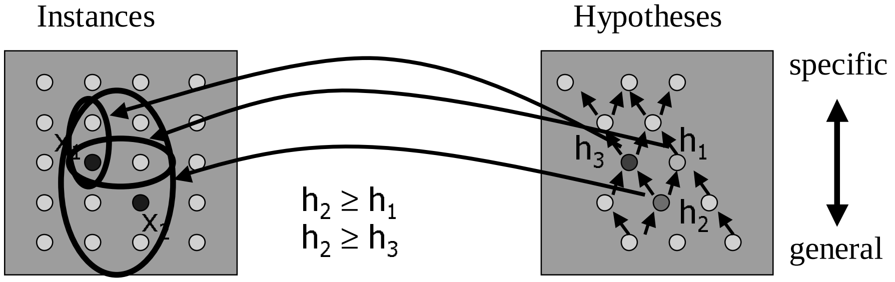
- Let h_j and h_k be hypotheses from H, i.e. boolean-valued functions defined over X.

  Then h_j is *more general than or equal to* h_k ($h_j \geq h_k$) if and only if
  $\forall x \in X : [(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$

- The relation ≥ imposes a partial order over the hypothesis space H (*general-to-specific ordering*).

2.2 Data Mining as Search in the Hypothesis Space

Relationship Instances ←→ Hypotheses



x_1=< Sunny,Warm,High,Strong,Cool,Same>

x_2=< Sunny,Warm,High,Light,Warm,Same>

h_1=< Sunny,?,?,Strong,?,?>

h_2=< Sunny,?,?,?,?,?>

h_3=< Sunny,?,?,?,Cool,?>

2.2 Data Mining as Search in the Hypothesis Space

Searching the Hypothesis Space

- exhaustive search is infeasible in real life applications
- exploit the ordering

top-down:

start with general hypotheses and keep specializing

bottom-up:

start with specialized hypotheses and keep generalizing

- how many hypotheses?

one (which?)

some (which?)

all

2.2 Data Mining as Search in the Hypothesis Space

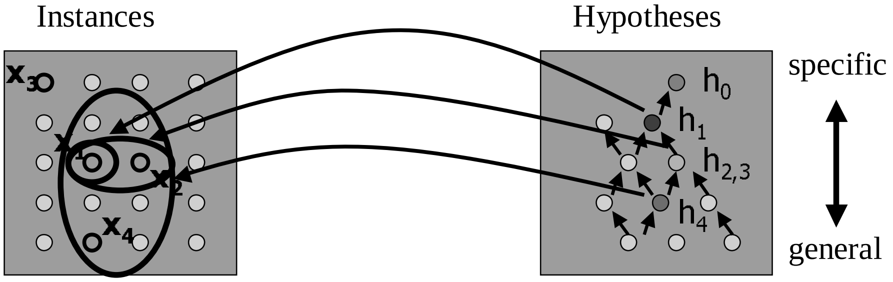
Find-S Algorithm

```
• 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 $a_i$ in $h$ is satisfied by $x$
        then do nothing
        else generalize $a_i$ w.r.t. $\geq$ until $a_i$ is satisfied by $x$
• Output hypothesis $h$
```

finds *one maximally specific* hypothesis

2.2 Data Mining as Search in the Hypothesis Space

Find-S Algorithm



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

x_2=<Sunny,Warm,High,Strong,Warm,Same>+

x_3=<Rainy,Cold,High,Strong,Warm,Change> -

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

h_0=< Ø, Ø, Ø, Ø, Ø, Ø,>

h_1=< Sunny,Warm,Normal, Strong,Warm,Same>

$h_{2,3}$=< Sunny,Warm,?, Strong,Warm,Same>

h_4=< Sunny,Warm,?, Strong,?,?>

2.2 Data Mining as Search in the Hypothesis Space

Find-S Algorithm

- Algorithm is very efficient

    what runtime complexity?

- Ignores negative training examples
- What about the negative examples?

  Under which conditions is *h* consistent with them?

- Why prefer a most specific hypothesis?
- What if there are multiple maximally specific hypotheses?

2.2 Data Mining as Search in the Hypothesis Space

Version Space

- 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 <x,c(x)> in *D*.

    $\text{consistent}(h,D) := \forall \langle x,c(x)\rangle \in D\text{:}\ \ h(x)=c(x)$

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

    $VS_{H,D} = \{h \in H \mid \text{consistent}(h,D)\}$

2.2 Data Mining as Search in the Hypothesis Space

Version Space

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

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

where $x \geq y$ ”x is more general or equal than y”

compact representation of the version space

2.2 Data Mining as Search in the Hypothesis Space

Candidate Elimination Algorithm

```
$G$ ← maximally general hypotheses in $H$
S ← maximally specific hypotheses in $H$
**For each** training example $d = <x,c(x)>$
  **If** $d$ is a positive example
     remove from $G$ any hypothesis that is 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 $S$
```

2.2 Data Mining as Search in the Hypothesis Space

Candidate Elimination Algorithm (contd.)

```
//      For each training example d = <x,c(x)>
  If d is a negative example
     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
                   (1) h consistent with d
                   (2) some member of S is more specific than h
     remove from G any hypothesis that is less general than another
          hypothesis in G
```

2.2 Data Mining as Search in the Hypothesis Space

Example Candidate Elimination

S: ~~{<∅, ∅, ∅, ∅, ∅, ∅ >}~~

G: {<?, ?, ?, ?, ?, ?>}

x_1 = <Sunny Warm Normal Strong Warm Same> +

S: ~~{< Sunny Warm Normal Strong Warm Same >}~~

G: {<?, ?, ?, ?, ?, ?>}

x_2 = <Sunny Warm High Strong Warm Same> +

S: {< Sunny Warm ? Strong Warm Same >}

G: {<?, ?, ?, ?, ?, ?>}

2.2 Data Mining as Search in the Hypothesis Space

Example Candidate Elimination

S: {< Sunny Warm ? Strong Warm Same >}

G: ~~{<?, ?, ?, ?, ?, ?>}~~

x_3 = <Rainy Cold High Strong Warm Change> -

S: ~~{< Sunny Warm ? Strong Warm Same >}~~

G: {<Sunny,?,?,?,?,?>, <?,Warm,?,?,?>, ~~<?,?,?,?,?,Same>~~}

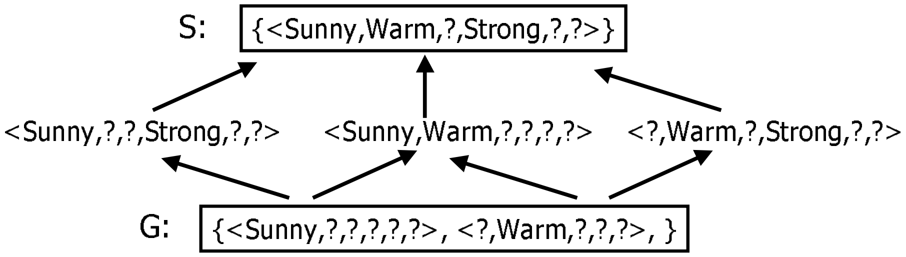
x_4 = <Sunny Warm High Strong Cool Change> +

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

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

2.2 Data Mining as Search in the Hypothesis Space

Classification of new Data



x_5 = <Sunny Warm Normal Strong Cool Change> + 6/0
x_6 = <Rainy Cold Normal Light Warm Same> - 0/6
x_7 = <Sunny Warm Normal Light Warm Same> ? 3/3
x_8 = <Sunny Cold Normal Strong Warm Same> ? 2/4

2.2 Data Mining as Search in the Hypothesis Space

Candidate Elimination Algorithm

- Exploits negative training examples
- Finds all consistent hypotheses from *H*
- Can determine confidence of classification of new data
- Can detect inconsistencies in training data

  How?
- Algorithm is not very efficient

  What runtime complexity?
- What if *H* cannot represent target concept *C*?

2.3 Inductive Bias

Example

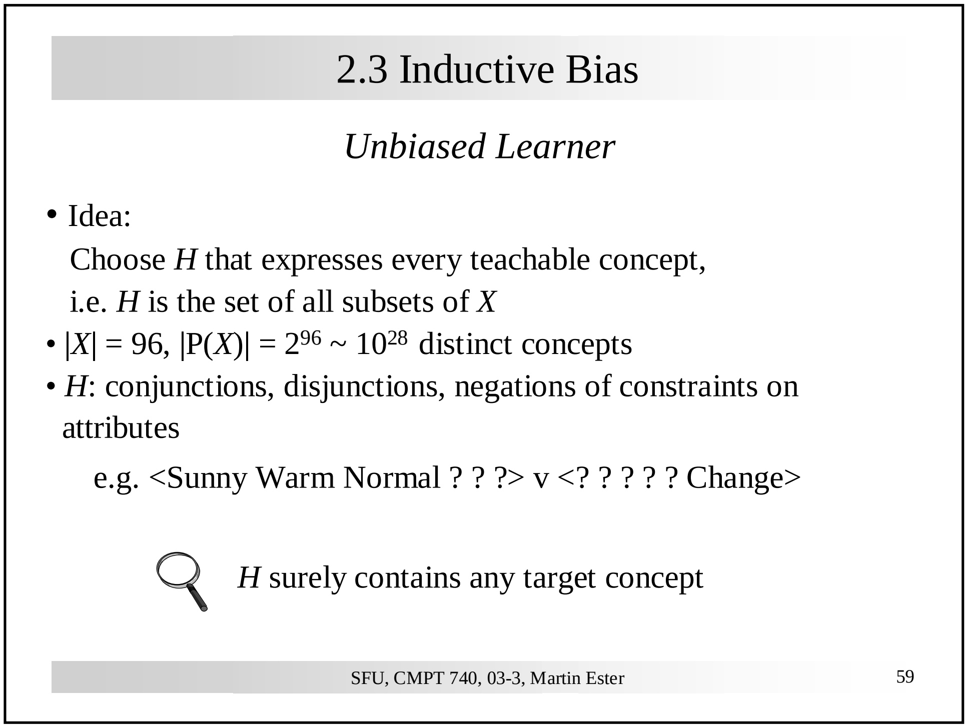
Our hypothesis space is unable to represent a simple disjunctive target concept : (Sky=Sunny) v (Sky=Cloudy)

x_1 = <Sunny Warm Normal Strong Cool Change> +
x_2 = <Cloudy Warm Normal Strong Cool Change> +

S : { <?, Warm, Normal, Strong, Cool, Change> }

x_3 = <Rainy Warm Normal Light Warm Same> -

S : {} // no consistent hypothesis!



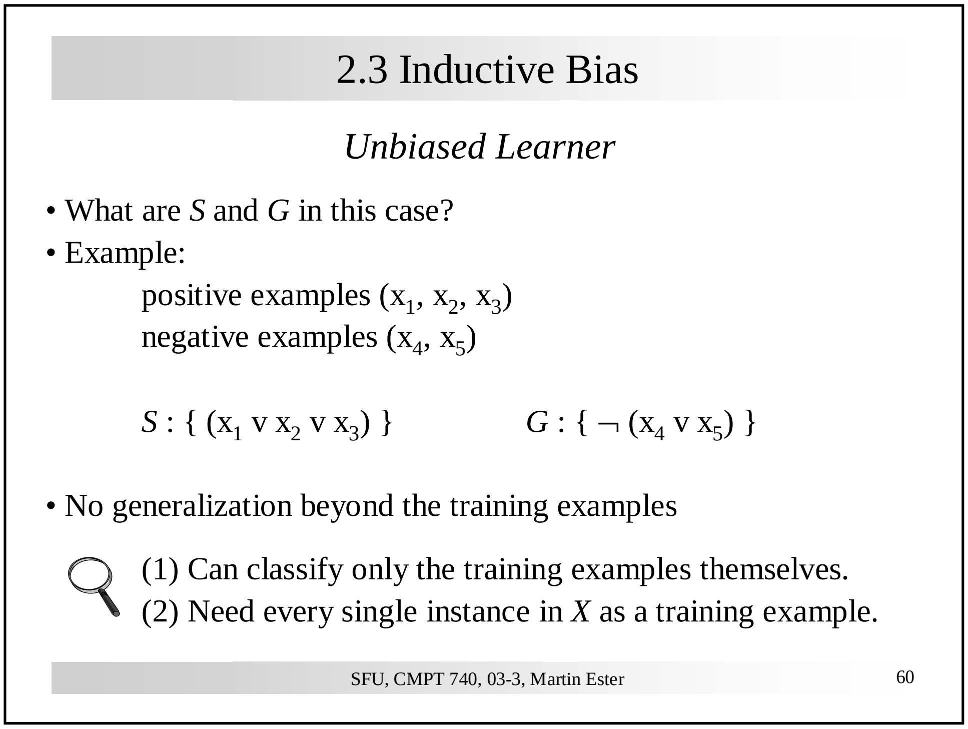
2.3 Inductive Bias

Unbiased Learner

- Idea:
  Choose *H* that expresses every teachable concept,
  i.e. *H* is the set of all subsets of *X*
- $|X| = 96$, $|P(X)| = 2^{96} \sim 10^{28}$ distinct concepts
- *H*: conjunctions, disjunctions, negations of constraints on attributes

  e.g. <Sunny Warm Normal ? ? ?> v <? ? ? ? ? Change>

H surely contains any target concept



2.3 Inductive Bias

Unbiased Learner

- What are *S* and *G* in this case?
- Example:
  positive examples (x_1, x_2, x_3)
  negative examples (x_4, x_5)

  S : { (x_1 v x_2 v x_3) }   *G* : { ¬ (x_4 v x_5) }

- No generalization beyond the training examples

(1) Can classify only the training examples themselves.
(2) Need every single instance in *X* as a training example.

2.3 Inductive Bias

Importance of Inductive Bias

- A learner that makes no prior assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances.
- *Inductive bias*: set of assumptions that justify the inductive inferences as deductive inferences
- Use domain knowledge of KDD application to choose appropriate inductive bias.
- Too vague inductive bias: cannot generalize well
  Too strict inductive bias: no consistent hypothesis.

2.3 Inductive Bias

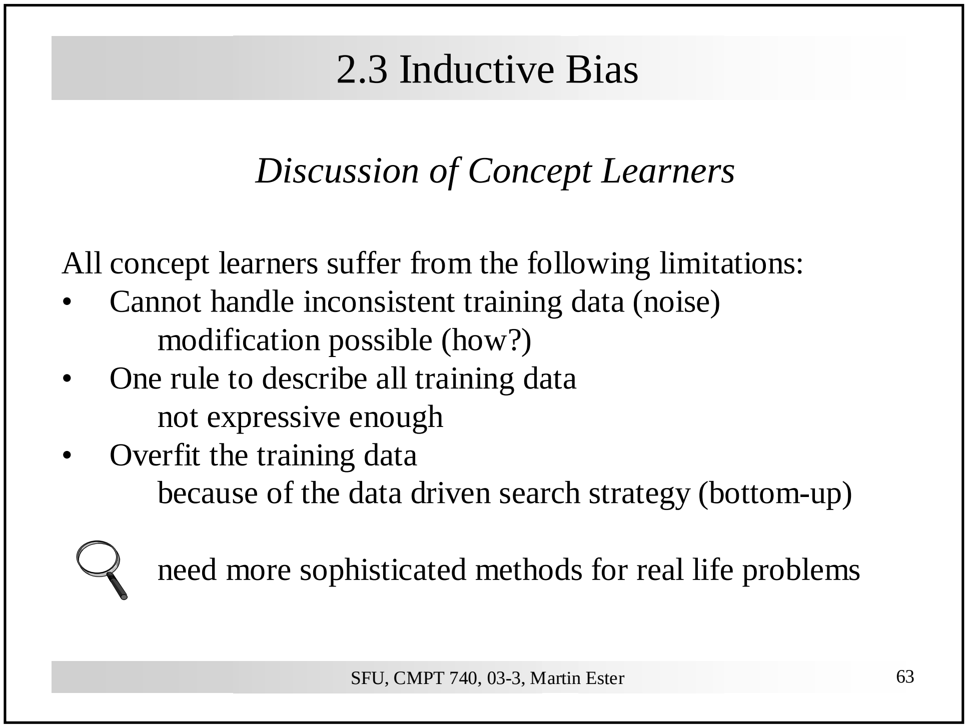
Discussion of Different Learners

Two aspects of inductive bias
(1) Definition of hypothesis space
(2) Treatment of multiple consistent hypotheses

Unbiased learner
(1) No restriction of formulae made from attribute constraints
(2) Unique consistent hypothesis

Candidate elimination algorithm
(1) Target concept can be described as conjunction of attribute constraints
(2) Consider all consistent hypotheses

Find-S algorithm
(1) Same as candidate elimination algorithm
(2) Maximally specific hypotheses are best



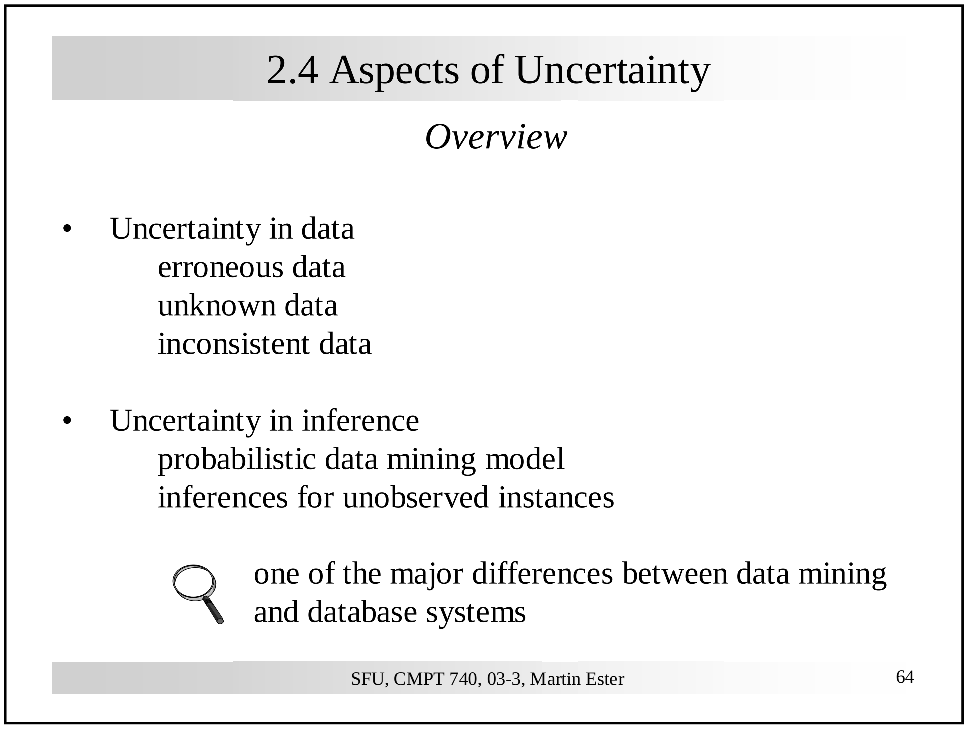
2.3 Inductive Bias

Discussion of Concept Learners

All concept learners suffer from the following limitations:

- Cannot handle inconsistent training data (noise)
  modification possible (how?)
- One rule to describe all training data
  not expressive enough
- Overfit the training data
  because of the data driven search strategy (bottom-up)

need more sophisticated methods for real life problems



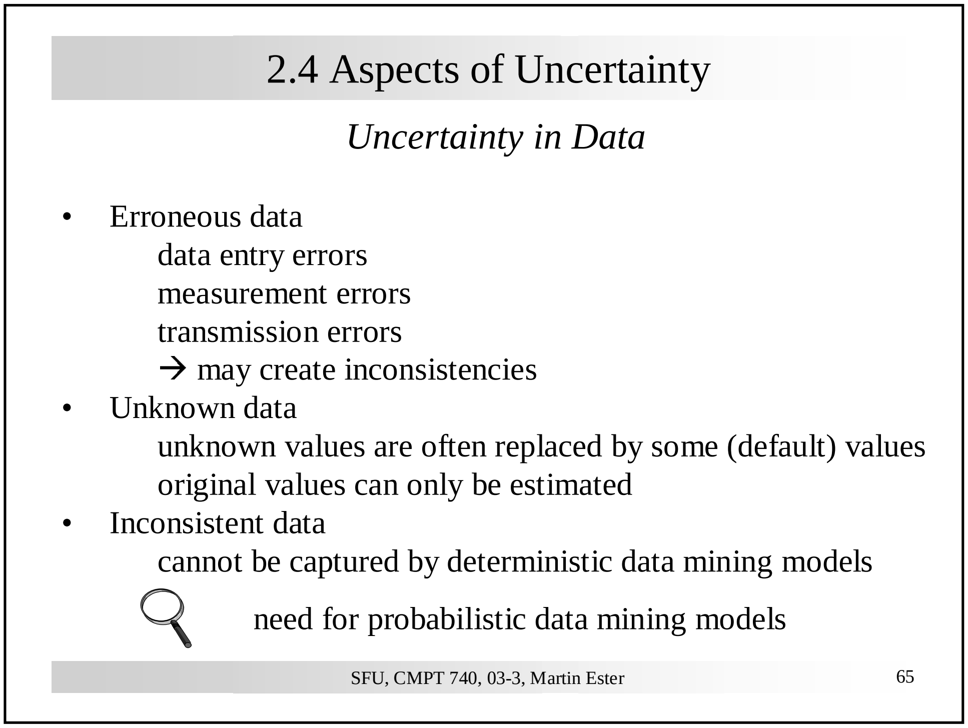
2.4 Aspects of Uncertainty

Overview

- Uncertainty in data
  erroneous data
  unknown data
  inconsistent data

- Uncertainty in inference
  probabilistic data mining model
  inferences for unobserved instances

one of the major differences between data mining and database systems

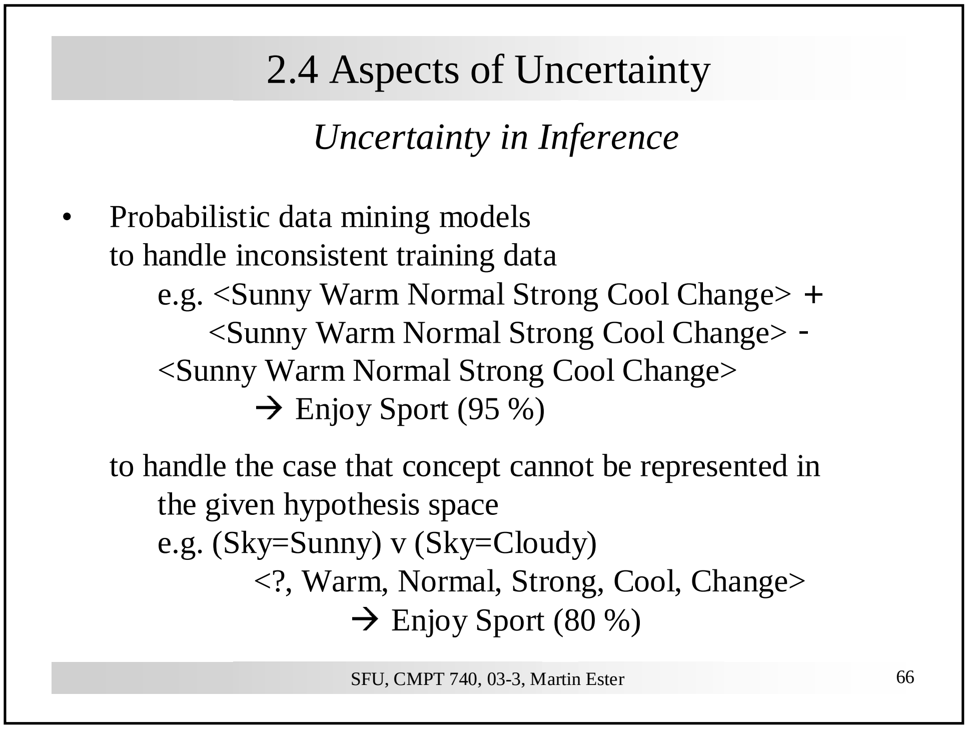


2.4 Aspects of Uncertainty

Uncertainty in Data

- Erroneous data
  - data entry errors
  - measurement errors
  - transmission errors
  - → may create inconsistencies
- Unknown data
  - unknown values are often replaced by some (default) values
  - original values can only be estimated
- Inconsistent data
  - cannot be captured by deterministic data mining models

need for probabilistic data mining models



2.4 Aspects of Uncertainty

Uncertainty in Inference

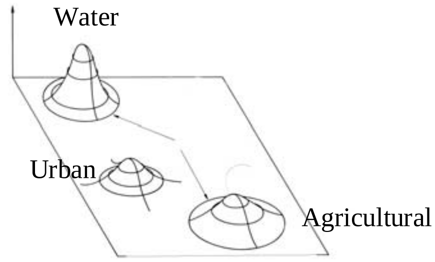
- Probabilistic data mining models
  to handle inconsistent training data
  - e.g. <Sunny Warm Normal Strong Cool Change> +
    <Sunny Warm Normal Strong Cool Change> -
  - <Sunny Warm Normal Strong Cool Change>
    → Enjoy Sport (95 %)

  to handle the case that concept cannot be represented in
  - the given hypothesis space
  - e.g. (Sky=Sunny) v (Sky=Cloudy)
    <?, Warm, Normal, Strong, Cool, Change>
    → Enjoy Sport (80 %)

2.4 Aspects of Uncertainty

Uncertainty in Inference

- Probabilistic data mining models (contd.)
  to handle inherently probabilistic phenomena



Emissions from the earth surface in two different spectral bands

clusters generated from multidimensional Normal distributions

2.4 Aspects of Uncertainty

Uncertainty in Inference

- Inferences for unobserved instances
  have only (relatively small) sample of data from instance space *X*
  Let hypothesis *h* approximate the target function with confidence *c* % over the training data
  ? How well does it approximate the target function over the unobserved instances of *X*?

The larger the training data set, the better an estimate is *c* for the actual confidence over whole *X*

Heuristic rules, e.g. „simpler hypotheses generalize better“

2.5 Data Mining as Optimization Problem

Overview

Goal

find *model*(s) that *best fit* the given training data

Steps

1. Choice of model category (manual)
   depending on type of data and data mining task
2. Definition of score function (manual)
   to measure the fit of model and training data
3. Choice of model structure (semi-automatic)
   within the given model category
4. Search for model parameters (automatic)
   for the given model structure

2.5 Data Mining as Optimization Problem

Optimization Scheme

```
Choose model category and score function;

For each possible model structure in this model category do
    For each possible set of parameter values do
        Determine the score of the model with this parameter setting;
        Keep structure and parameters with optimal score;
```

Comments

- Not efficient
- Sometimes, independent determination of model structure and parameter values (approximation of score)
- Sometimes, manual choice of model structure

2.5 Data Mining as Optimization Problem

Example 1: Concept Learning

1. Model category
   conjunction of attribute constraints
2. Score function
   confidence of hypotheses on training data
3. Model structure
   selection of attributes (features)
4. Model parameters
   actual attribute constraints for each attribute

2.5 Data Mining as Optimization Problem

Example 2: Linear Regression

1. Model category
   linear function
2. Score function
   sum of squared errors
   (deviation of function values from observed values)
3. Model structure
   selection of attributes (variables)
4. Model parameters
   coefficients of the linear function

2.5 Data Mining as Optimization Problem

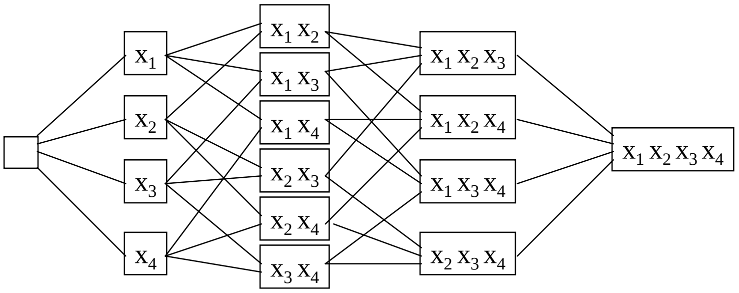
Example 3: Mixture Modelling

1. Model category
   mixture of Normal distributions
2. Score function
   likelihood
   (probability that training data have been generated by this model)
3. Model structure
   selection of attributes (variables)
   choice of number of different Normal distributions
4. Model parameters
   mean vectors and covariance matrices of the Normal distributions

2.5 Data Mining as Optimization Problem

Optimization in Discrete Spaces

Search space: Graph with
nodes = states (e.g. different subsets of attributes)
edges = „legal moves“ (e.g. add/remove one attribute)



2.5 Data Mining as Optimization Problem

A Simple Search Algorithm

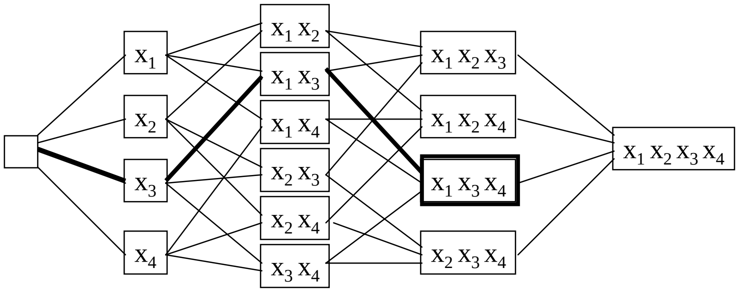
Hill Climbing Algorithm

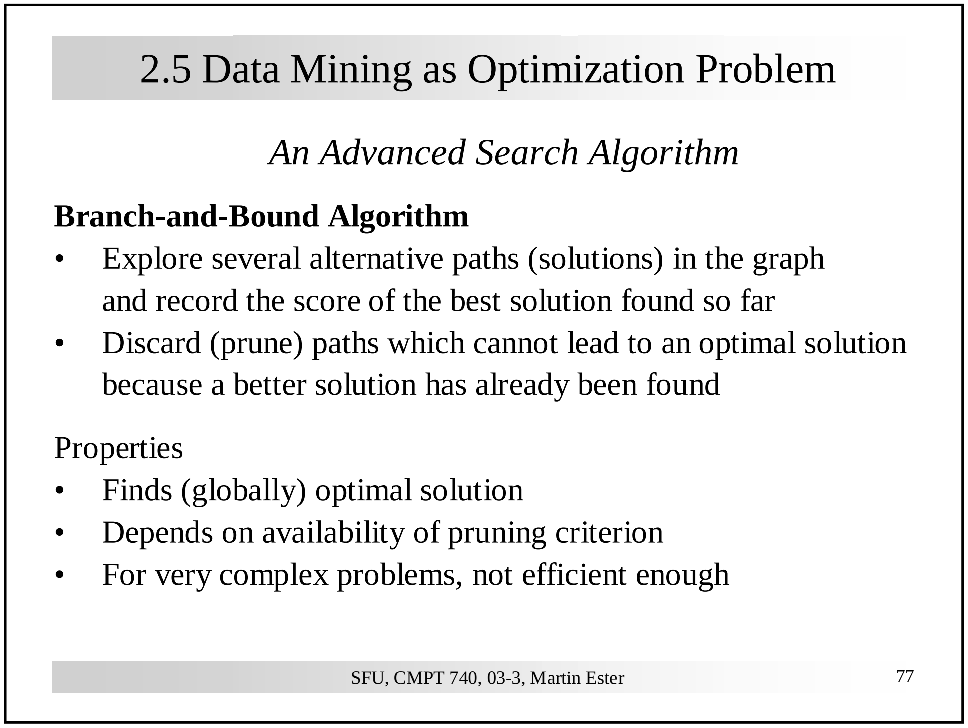
- Initialize
  choose an initial state S_0
- Iterate
  S_i: current state of the i-th iteration
  Evaluate the score function for all adjacent states of S_i
  Choose S_{i+1} as the best adjacent state
- Stop
  when no adjacent state improves score

finds a local optimum of the score function
multiple restarts alleviate these effects

2.5 Data Mining as Optimization Problem

Example Hill Climbing





2.5 Data Mining as Optimization Problem

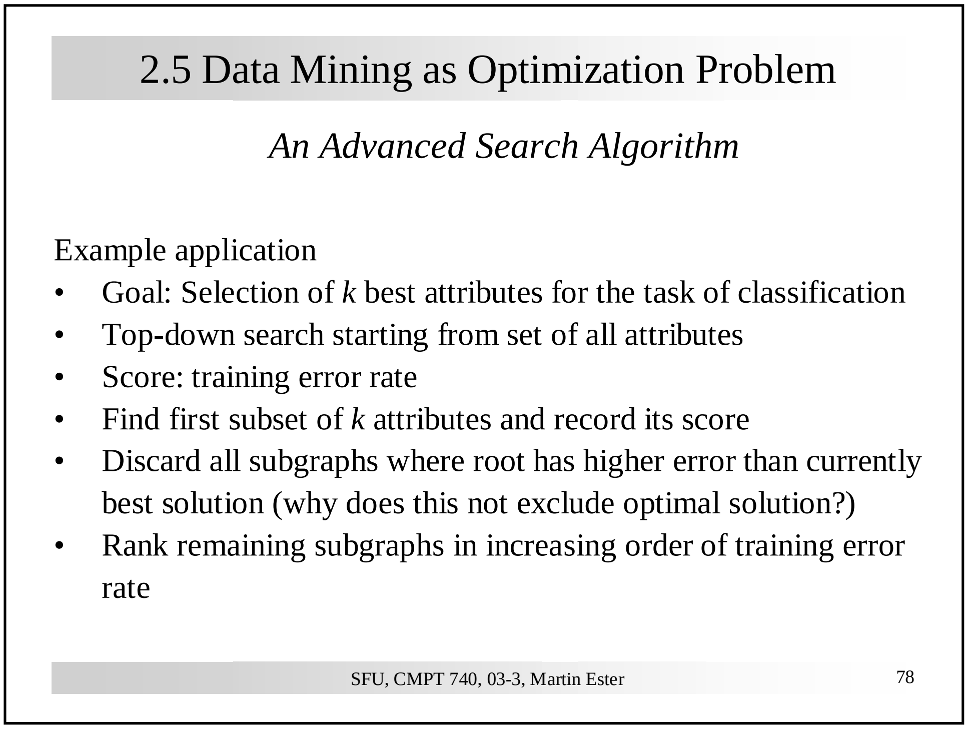
An Advanced Search Algorithm

Branch-and-Bound Algorithm

- Explore several alternative paths (solutions) in the graph and record the score of the best solution found so far
- Discard (prune) paths which cannot lead to an optimal solution because a better solution has already been found

Properties

- Finds (globally) optimal solution
- Depends on availability of pruning criterion
- For very complex problems, not efficient enough



2.5 Data Mining as Optimization Problem

An Advanced Search Algorithm

Example application

- Goal: Selection of *k* best attributes for the task of classification
- Top-down search starting from set of all attributes
- Score: training error rate
- Find first subset of *k* attributes and record its score
- Discard all subgraphs where root has higher error than currently best solution (why does this not exclude optimal solution?)
- Rank remaining subgraphs in increasing order of training error rate

2.5 Data Mining as Optimization Problem

Optimization in Continuous Spaces

- For parameter optimization
- θ: *d*-dimensional vector of parameters

  $S(\theta)$: score function
- Often, $S(\theta) = \sum_{i=1}^{n} e(y(i), \hat{y}_\theta(i))$

  where $y(i)$ denotes the target value of training instance *i*

  $\hat{y}_\theta(i)$ denotes the estimate of the model with parameters θ

  e denotes a function measuring the error

the complexity of *S* depends on the complexity of the model structure and the form of the error function

2.5 Data Mining as Optimization Problem

Optimization in Continuous Spaces

- Gradient function

$$g(\theta) = \nabla_\theta S(\theta) = (\frac{\partial S(\theta)}{\partial \theta_1}, \frac{\partial S(\theta)}{\partial \theta_2}, \cdots, \frac{\partial S(\theta)}{\partial \theta_d})$$

where $\frac{\partial S(\theta)}{\partial \theta_i}$ denote the partial derivatives

- Necessary condition for an optimum

$$g(\theta) = 0$$

2.5 Data Mining as Optimization Problem

Optimization in Continuous Spaces

- Solution in closed form

  e.g. if $S(\theta)$ is quadratic function,

  i.e. $g(\theta)$ is linear function
- $S(\theta)$ smooth non-linear function without solution in closed form

  perform local search on surface of S

  iterative improvement techniques

  based on local information about the curvature

  (such as steepest descent)

2.5 Data Mining as Optimization Problem

A Simple Search Algorithm

Gradient-Based Local Optimization

- Initialize

  choose an initial value θ_0 for the parameter vector (randomly)
- Iterate

  θ_i : current state of the i-th iteration

  Choose $\theta_{i+1} = \theta_i + \lambda_i v_i$

  where v_i is the direction of the next step (steepest descent)

  and λ_i determines the size of the next step
- Stop when a local optimum *appears* to be found

  finds a local optimum of the score function

  multiple restarts to improve th result

2.6 Synopsis of Machine Learning, Statistics, Data Mining

	Statistics	Machine Learning	Data Mining
Components of training data	variables	features	attributes
Result of learning	model	hypothesis	patterns

- Model: global
- Pattern: local
- Combination of these views

model = set of (all) patterns
data = global model (rule) + local patterns (exceptions)