Data Mining and Knowledge Discovery

Part of
“New Media and e-Science” M.Sc. Programme
and “Statistics” M.Sc. Programme

2008 / 2009

Nada Lavrač

Jožef Stefan Institute

Ljubljana, Slovenia

Course participants

I. IPS students

- Aleksovski
- Bole
- Cimperman
- Dali
- Dervišević
- Djuras
- Dovgan
- Kaluža
- Mirčevska
- Piltaver
- Pollak
- Rusu
- Tomašev
- Tomaško
- Vukašinović
- Zenkovič

II. Statistics students

- Breznik
- Golob
- Korošec
- Limbek
- Ostrež
- Suklan

Course Schedule - 2007/08
Data Mining and Knowledge Discovery (DM)

- 21 October 2008 15-19 Lectures (Lavrač)
- 22 October 2008 15-19 Practice (Kralj Novak)
- 11 November 2008 15-19 Lectures (Lavrač)
- 12 November 2008 15-19 Practice (Kralj Novak)
- 1 December 2008 16-17 written exam - theory
- 8 December 2008 15-17 seminar topics presentations
- 14 January 2009 15-19 seminar presentations (exam ?)
- Spare date, if needed:
  (28 January 2009 15-19 seminar presentations ?, exam ?)

http://kt.ijs.si/petra_kralj/IPSKnowledgeDiscovery0809.html

DM - Credits and coursework

"New Media and eScience" / "Statistics"

- 12 credits (30 hours / 36 hours)
- Lectures
- Practice
  – Theory exercises and hands-on (WEKA)
- Seminar – choice:
  – Data analysis of your own data (e.g., using WEKA for questionnaire data analysis)
  – Programming assignment - write your own data mining module, and evaluate it on a (few) domain(s)
- Contacts:
  – Nada Lavrač nada.lavrac@ijs.si
  – Petra Kralj Novak petra.kralj@ijs.si

DM - Credits and coursework

Exam: Written exam (60 minutes) - Theory

Seminar: topic selection + results presentation

- Oral presentations of your seminar topic (DM task or dataset presentation, max. 4 minutes)
- Presentation of your seminar results (10 minutes + discussion)
- Deliver written report + electronic copy (in Information Society paper format, see instructions on the web page),
  - Report on data analysis of own data needs to follow the CRISP-DM methodology
  - Report on DM SW development needs to include SW uploaded on a Web page – format to be announced

http://kt.ijs.si/petra_kralj/IPSKnowledgeDiscovery0809.html

Course Outline

I. Introduction

- Data Mining and KDD process
- DM standards, tools and visualization
- Classification of Data Mining techniques: Predictive and descriptive DM

  (Mladenić et al. Ch. 1 and 11, Kononenko & Kukar Ch. 1)

II. Predictive DM Techniques

- Bayesian classifier (Kononenko Ch. 9.6)
- Decision Tree learning (Mitchell Ch. 3, Kononenko Ch. 9.1)
- Classification rule learning (Berthold book Ch. 7, Kononenko Ch. 9.2)
- Classifier Evaluation (Bramer Ch. 6)

III. Regression

(Kononenko Ch. 9.4)

IV. Descriptive DM

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning (Kononenko Ch. 9.3)
- Hierarchical clustering (Kononenko Ch. 12.3)

– **V. Relational Data Mining**

- RDM and Inductive Logic Programming (Dzeroski & Lavrac Ch. 3, Ch. 4)
- Propositionalization approaches
- Relational subgroup discovery

Part I. Introduction

- Data Mining and the KDD process
- DM standards, tools and visualization
- Classification of Data Mining techniques: Predictive and descriptive DM

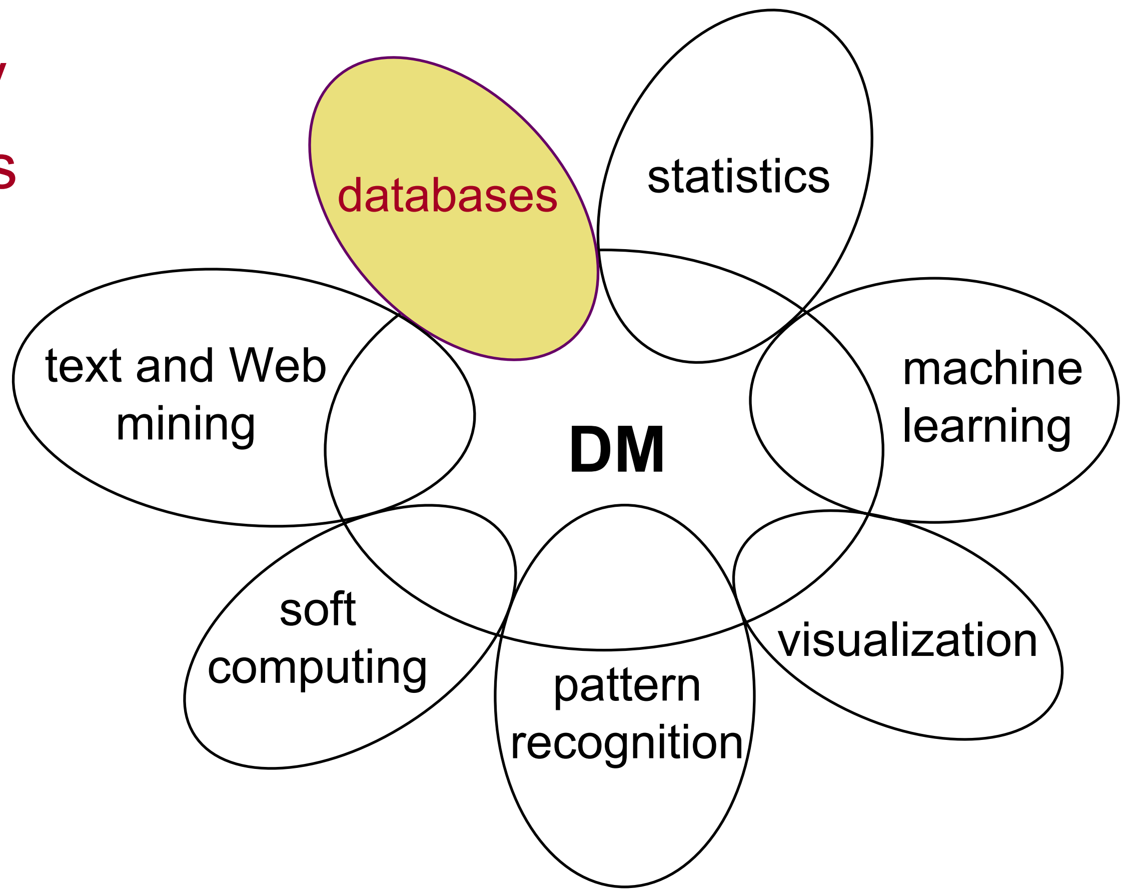
What is DM

- Extraction of useful information from data: discovering relationships that have not previously been known
- The viewpoint in this course: Data Mining is the application of Machine Learning techniques to solve real-life data analysis problems

Related areas

Database technology
and data warehouses

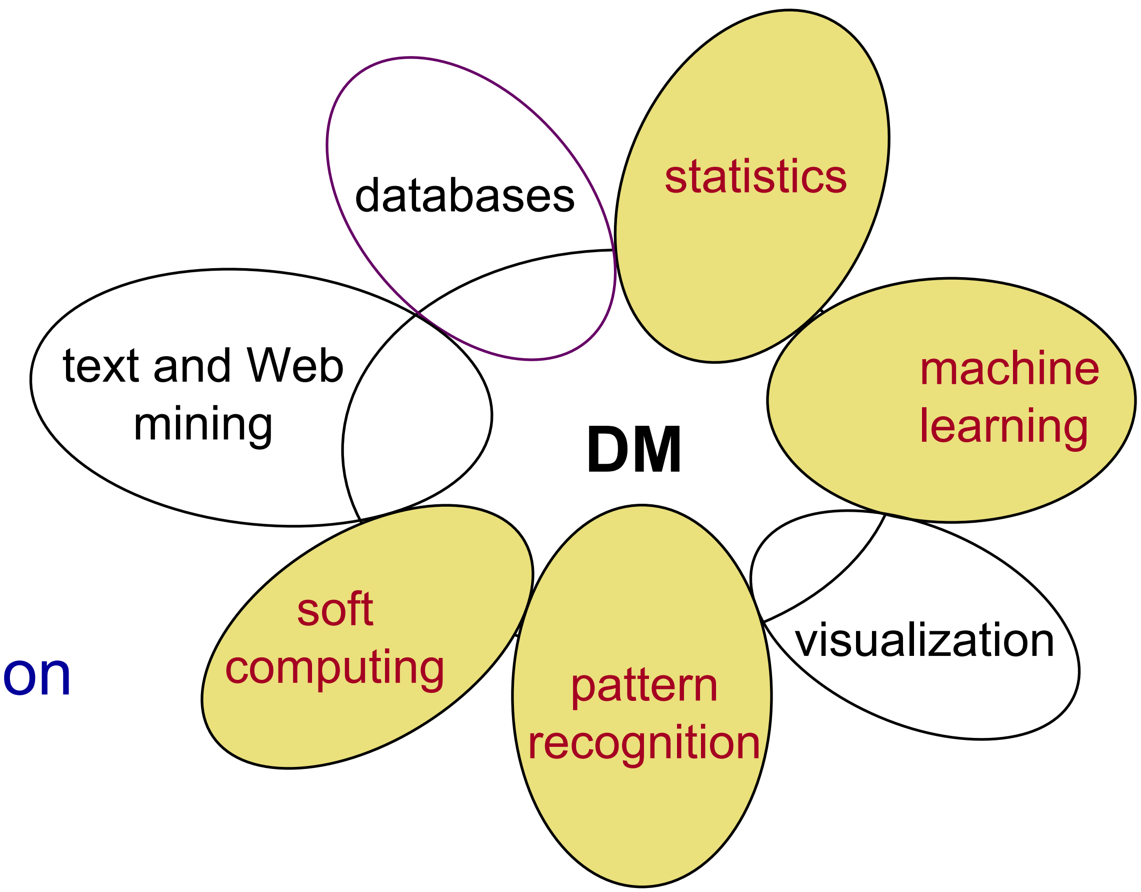
- efficient storage, access and manipulation of data



Related areas

Statistics,
machine learning,
pattern recognition
and soft computing*

- classification techniques and techniques for knowledge extraction from data

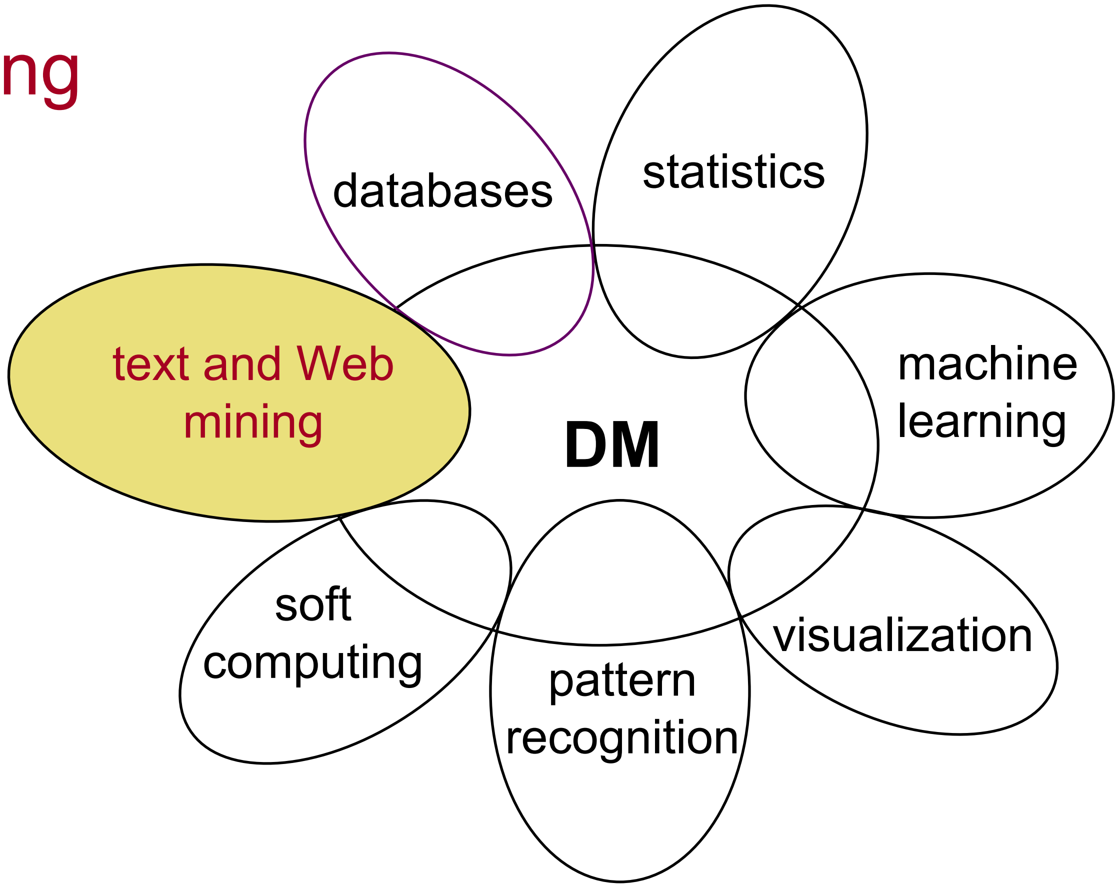


* neural networks, fuzzy logic, genetic algorithms, probabilistic reasoning

Related areas

Text and Web mining

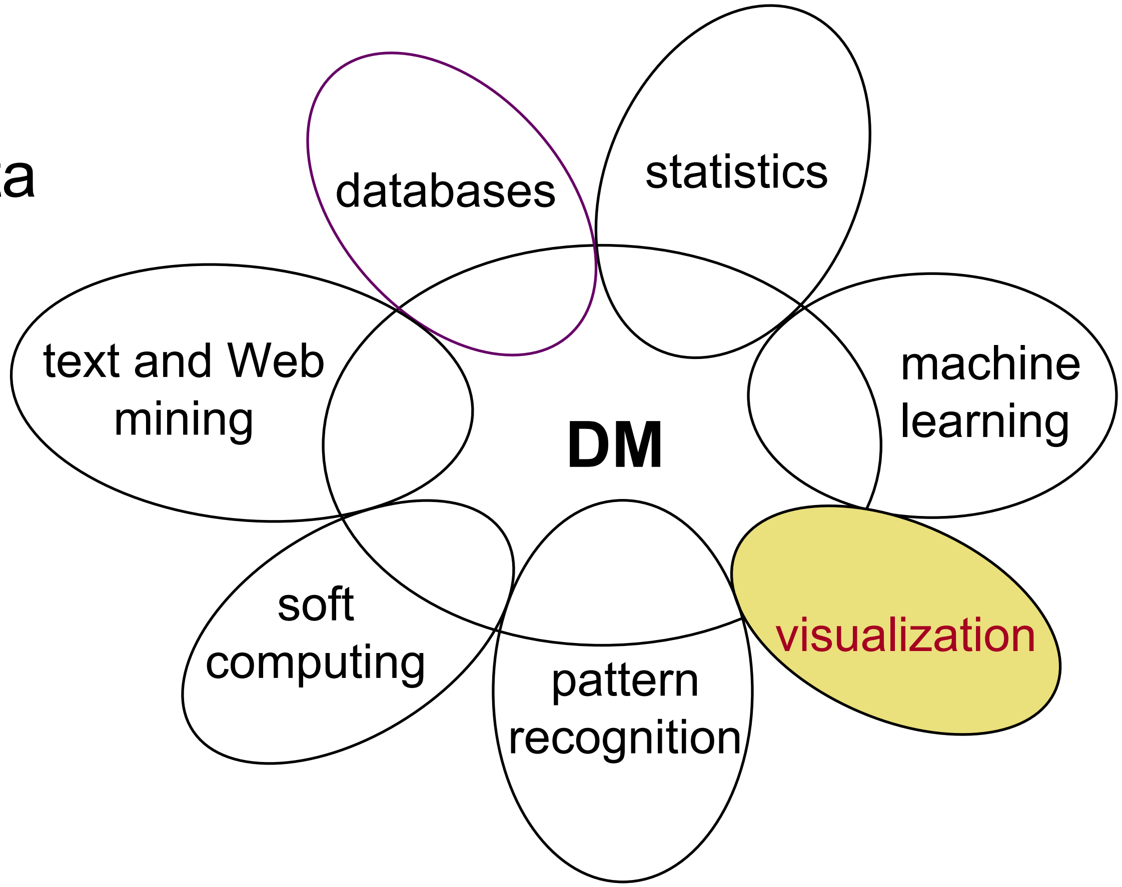
- Web page analysis
- text categorization
- acquisition, filtering and structuring of textual information
- natural language processing



Related areas

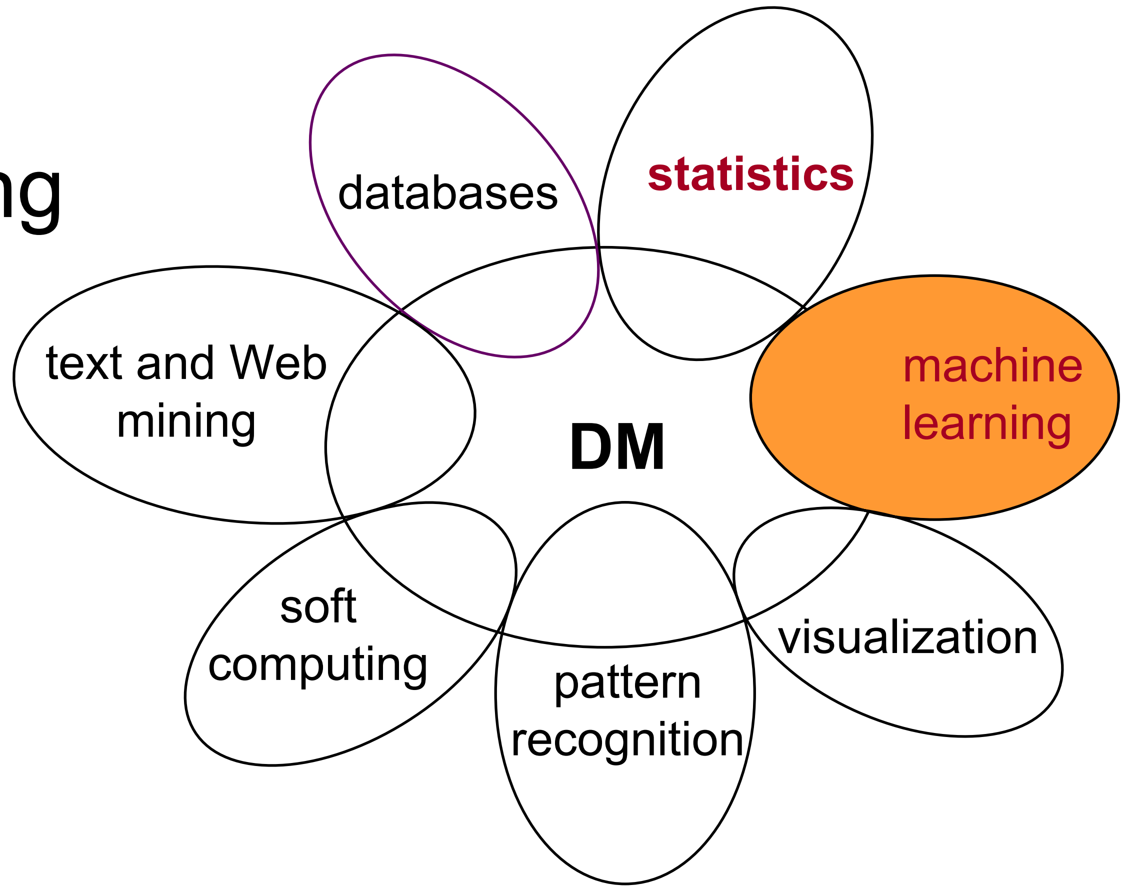
Visualization

- visualization of data and discovered knowledge



Point of view in this tutorial

Knowledge discovery using machine learning methods



Data Mining, ML and Statistics

- All areas have a long tradition of developing <u>inductive techniques</u> for data analysis.
  - reasoning from properties of a data sample to properties of a population
- DM vs. ML - Viewpoint in this course:
  - Data Mining is the application of Machine Learning techniques to hard real-life data analysis problems
- DM vs. Statistics:
  - Statistics
    - Hypothesis testing when certain theoretical expectations about the data distribution, independence, random sampling, sample size, etc. are satisfied
    - Main approach: best fitting all the available data
  - Data mining
    - Automated construction of understandable patterns, and structured models
    - Main approach: structuring the data space, heuristic search for decision trees, rules, … covering (parts of) the data space

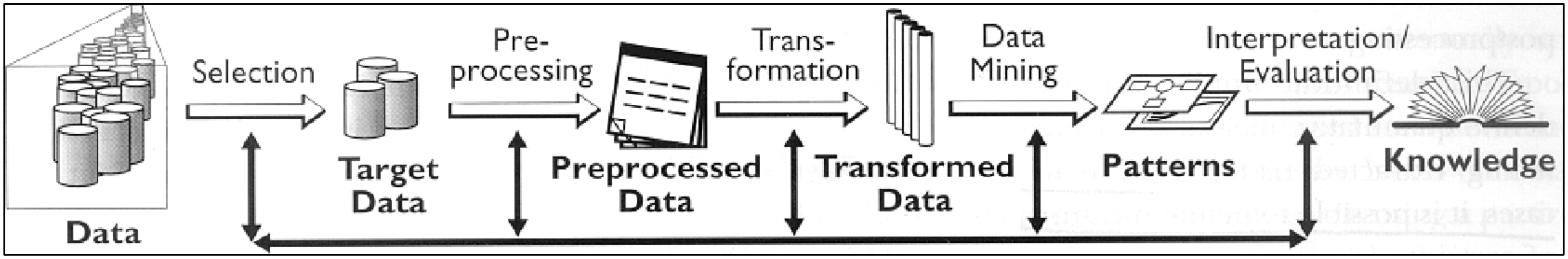
Data Mining and KDD

- KDD is defined as "the process of identifying valid, novel, potentially useful and ultimately understandable models/patterns in data." *
- Data Mining (DM) is the key step in the KDD process, performed by using data mining techniques for extracting models or interesting patterns from the data.

Usama M. Fayyad, Gregory Piatesky-Shapiro, Pedhraic Smyth: The KDD Process for Extracting Useful Knowledge form Volumes of Data. Comm ACM, Nov 96/Vol 39 No 11

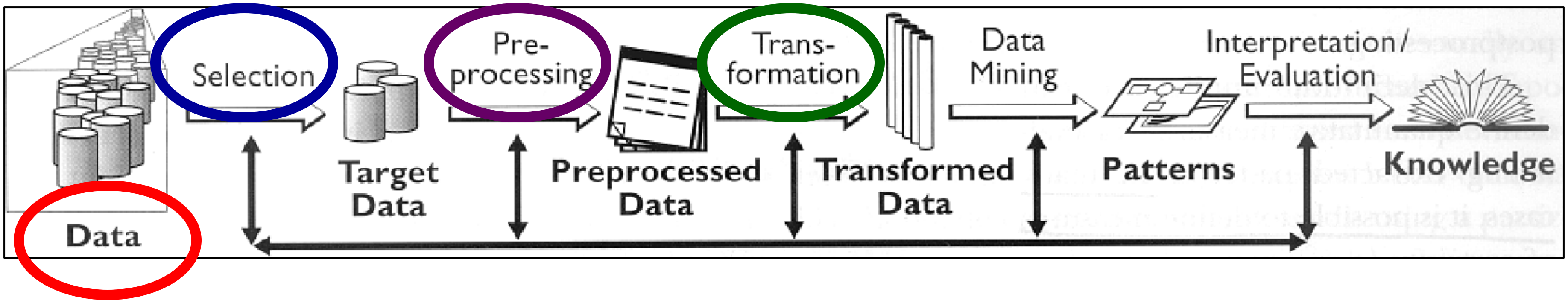
KDD Process

KDD process of discovering useful knowledge from data



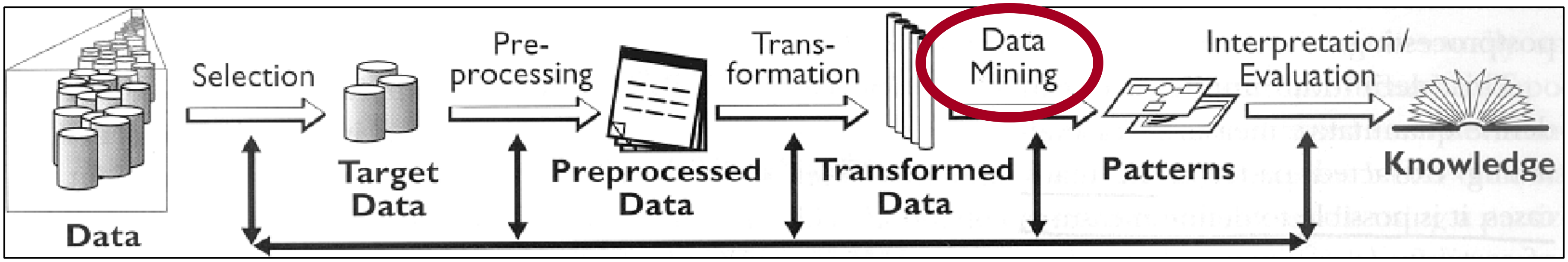
- KDD process involves several phases:
  - data preparation
  - data mining (machine learning, statistics)
  - evaluation and use of discovered patterns
- Data mining is the key step, but represents only 15%-25% of the entire KDD process

MEDIANA – analysis of media research data



- Questionnaires about journal/magazine reading, watching of TV programs and listening of radio programs, since 1992, about 1200 questions. Yearly publication: frequency of reading/listening/watching, distribution w.r.t. Sex, Age, Education, Buying power,..
- Data for 1998, about 8000 questionnaires, covering lifestyle, spare time activities, personal viewpoints, reading/listening/watching of media (yes/no/how much), interest for specific topics in media, social status
- good quality, “clean” data
- table of n-tuples (rows: individuals, columns: attributes, in classification tasks selected class)

MEDIANA – media research pilot study



- Patterns uncovering regularities concerning:
  - Which other journals/magazines are read by readers of a particular journal/magazine ?
  - What are the properties of individuals that are consumers of a particular media offer ?
  - Which properties are distinctive for readers of different journals ?
- Induced models: description (association rules, clusters) and classification (decision trees, classification rules)

Simplified association rules

Finding profiles of readers of the Delo daily newspaper

1. read_Marketing_magazine 116 =>
   read_Delo 95 (0.82)
2. read_Financial_News (Finance) 223 => read_Delo 180 (0.81)
3. read_Views (Razgledi) 201 => read_Delo 157 (0.78)
4. read_Money (Denar) 197 => read_Delo 150 (0.76)
5. read_Vip 181 => read_Delo 134 (0.74)

Interpretation: Most readers of Marketing magazine, Financial News, Views, Money and Vip read also Delo.

Simplified association rules (in Slovene)

1. bere_Sara 332 => bere_Slovenske novice 211 (0.64)
2. bere_Ljubezenske zgodbe 283 =>
   bere_Slovenske novice 174 (0.61)
3. bere_Dolenjski list 520 =>
   bere_Slovenske novice 310 (0.6)
4. bere_Omama 154 => bere_Slovenske novice 90 (0.58)
5. bere_Delavska enotnost 177 =>
   bere_Slovenske novice 102 (0.58)

Večina bralcev Sare, Ljubezenskih zgodb, Dolenjskega lista, Omame in Delavske enotnosti bere tudi Slovenske novice.

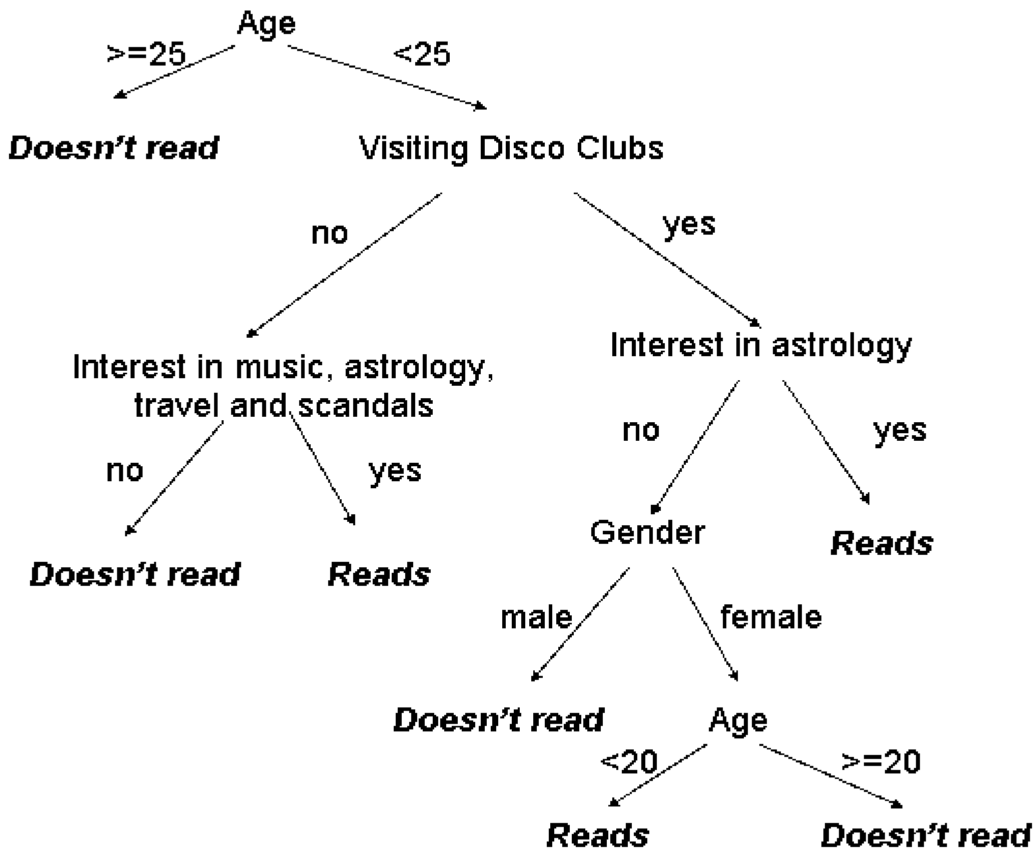
Simplified association rules (in Slovene)

1. bere_Sportske novosti 303 **=>**
   bere_Slovenski delnicar 164 (0.54)
2. bere_Sportske novosti 303 **=>**
   bere_Salomonov oglasnik 155 (0.51)
3. bere_Sportske novosti 303 **=>**
   bere_Lady 152 (0.5)

Več kot pol bralcev Sportskih novosti bere tudi Slovenskega delničarja, Salomonov oglasnik in Lady.

Decision tree

Finding reader profiles: decision tree for classifying people into readers and non-readers of a teenage magazine.

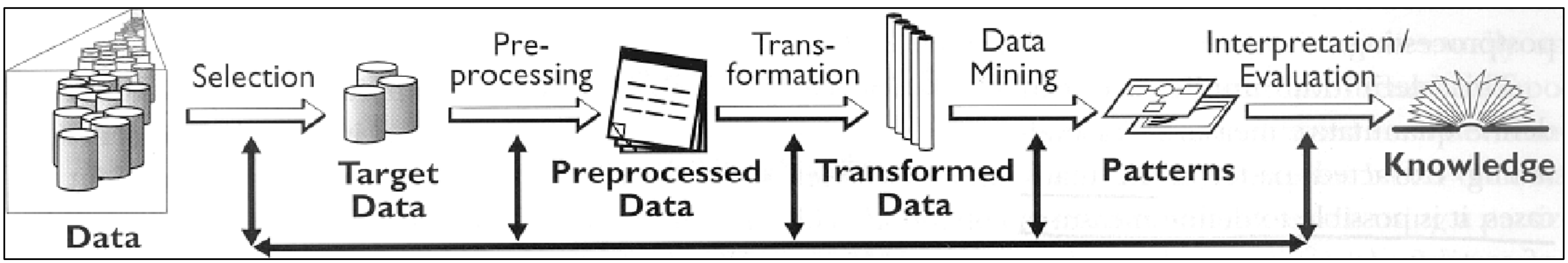


Part I. Introduction

- Data Mining and the KDD process
- DM standards, tools and visualization
- Classification of Data Mining techniques: Predictive and descriptive DM

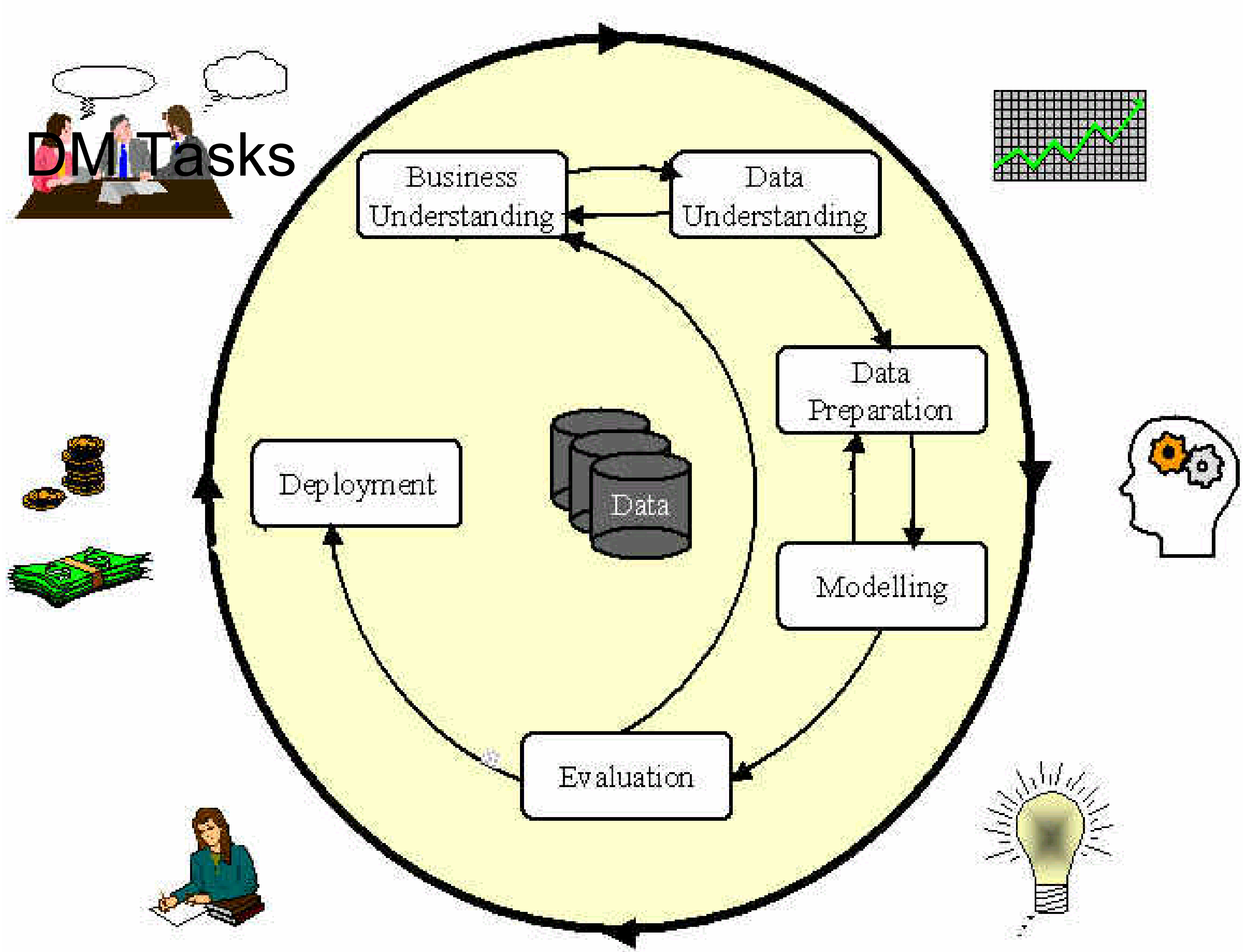
CRISP-DM

- Cross-Industry Standard Process for DM
- A collaborative, 18-months partially EC founded project started in July 1997
- NCR, ISL (Clementine), Daimler-Benz, OHRA (Dutch health insurance companies), and SIG with more than 80 members
- DM from art to engineering
- Views DM more broadly than Fayyad et al. (actually DM is treated as KDD process):

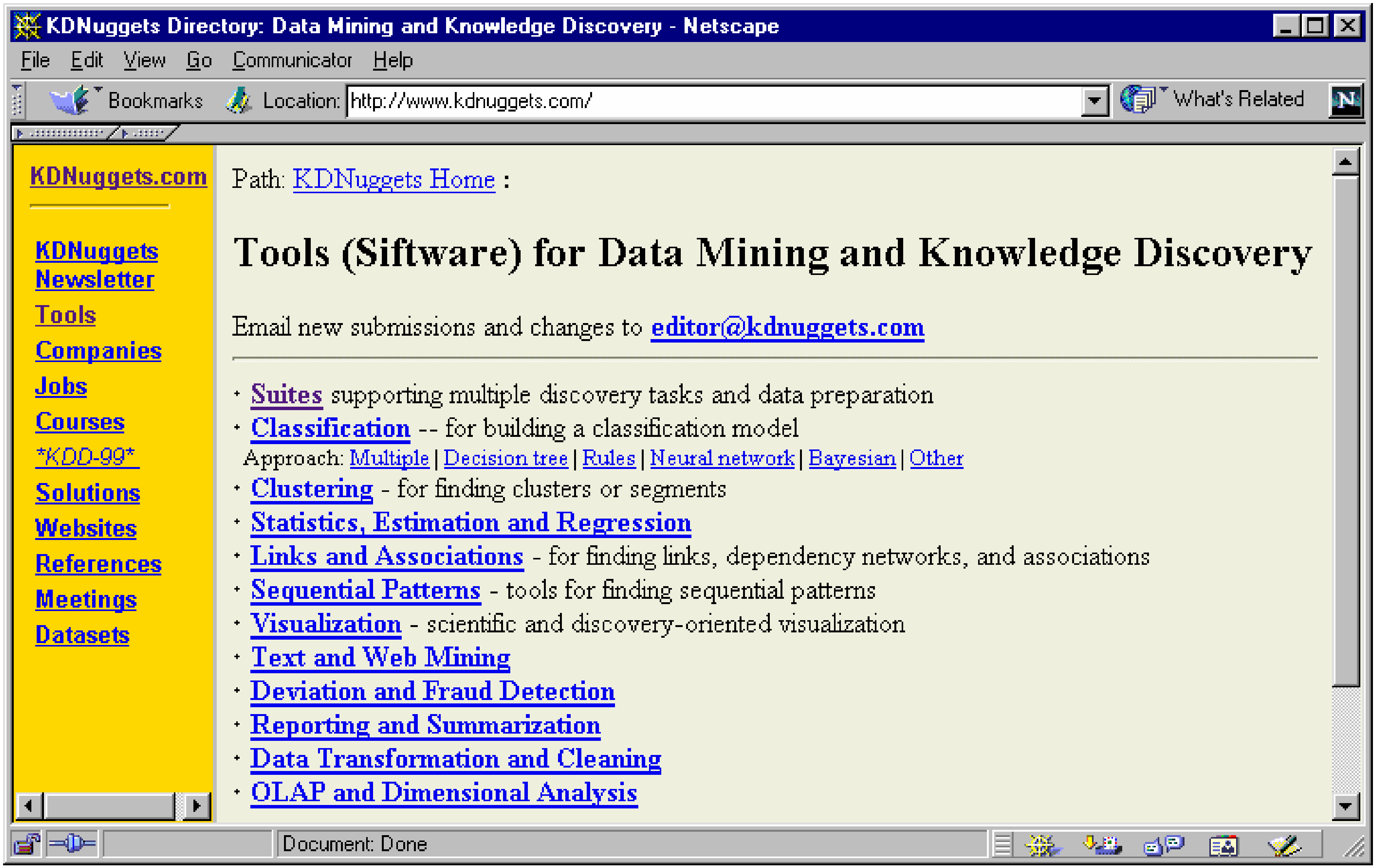


CRISP Data Mining Process

- DM Tasks

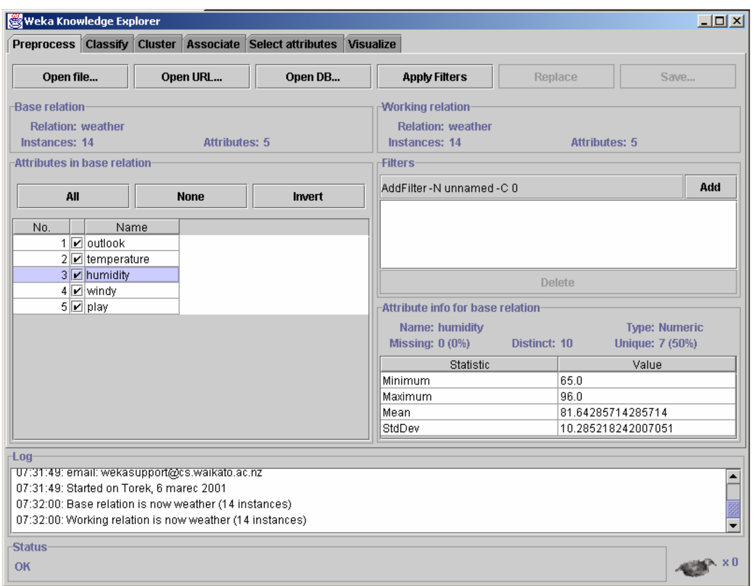


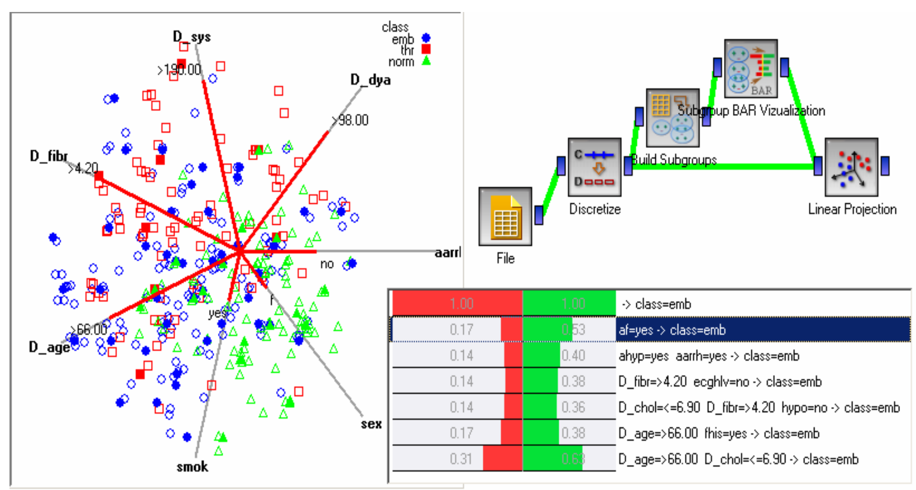
DM tools


KDNuggets Directory: Data Mining and Knowledge Discovery - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://www.kdnuggets.com/
What's Related
KDNuggets.com
KDNuggets Newsletter
Tools
Companies
Jobs
Courses
KDD-99
Solutions
Websites
References
Meetings
Datasets
Path: KDNuggets Home :
Tools (Siftware) for Data Mining and Knowledge Discovery
Email new submissions and changes to editor@kdnuggets.com
Suites supporting multiple discovery tasks and data preparation
Classification -- for building a classification model
Approach: Multiple | Decision tree | Rules | Neural network | Bayesian | Other
Clustering - for finding clusters or segments
Statistics, Estimation and Regression
Links and Associations - for finding links, dependency networks, and associations
Sequential Patterns - tools for finding sequential patterns
Visualization - scientific and discovery-oriented visualization
Text and Web Mining
Deviation and Fraud Detection
Reporting and Summarization
Data Transformation and Cleaning
OLAP and Dimensional Analysis
Document: Done

Public DM tools

- WEKA - Waikato Environment for Knowledge Analysis
- Orange
- KNIME - Konstanz Information Miner
- R – Bioconductor, …

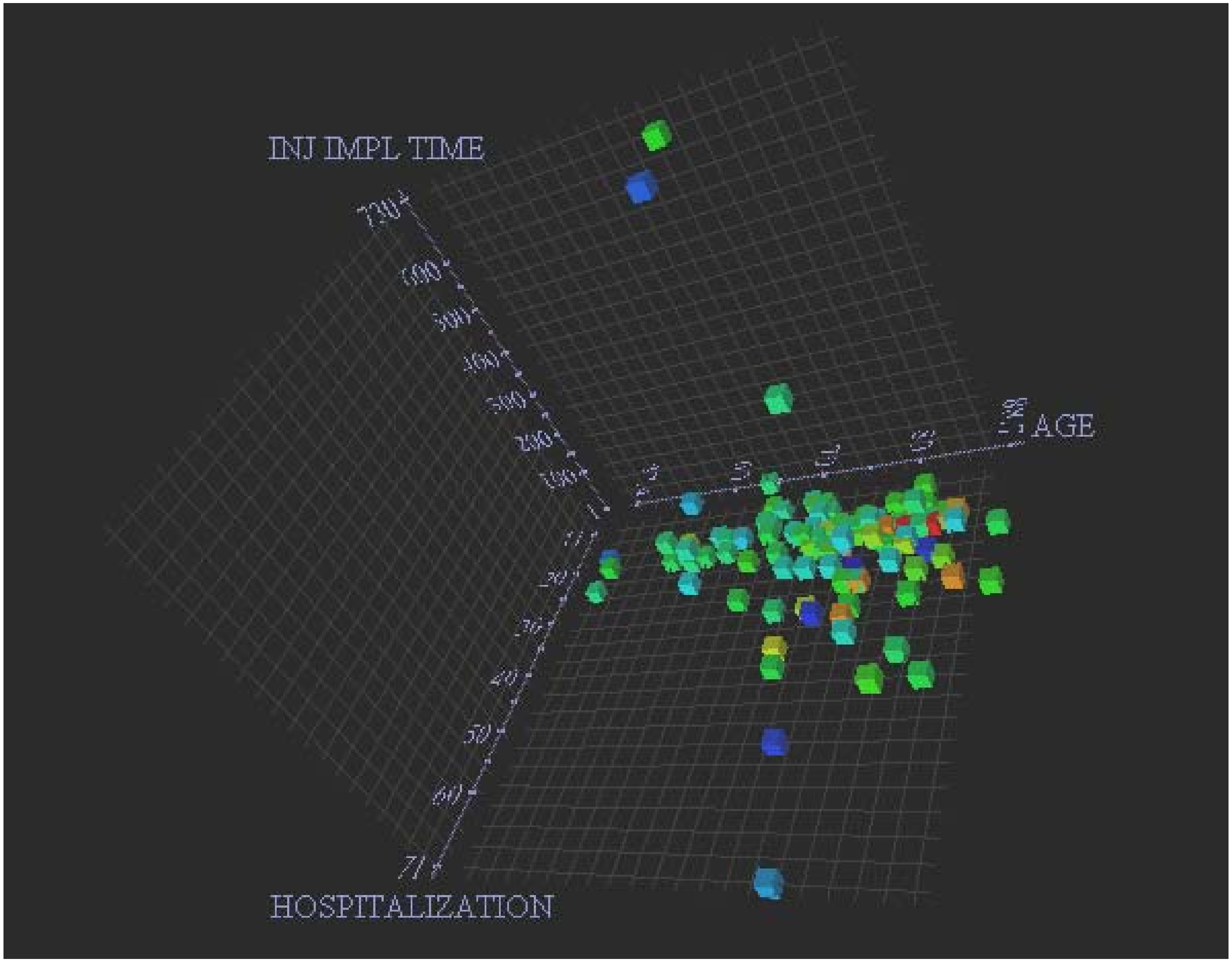




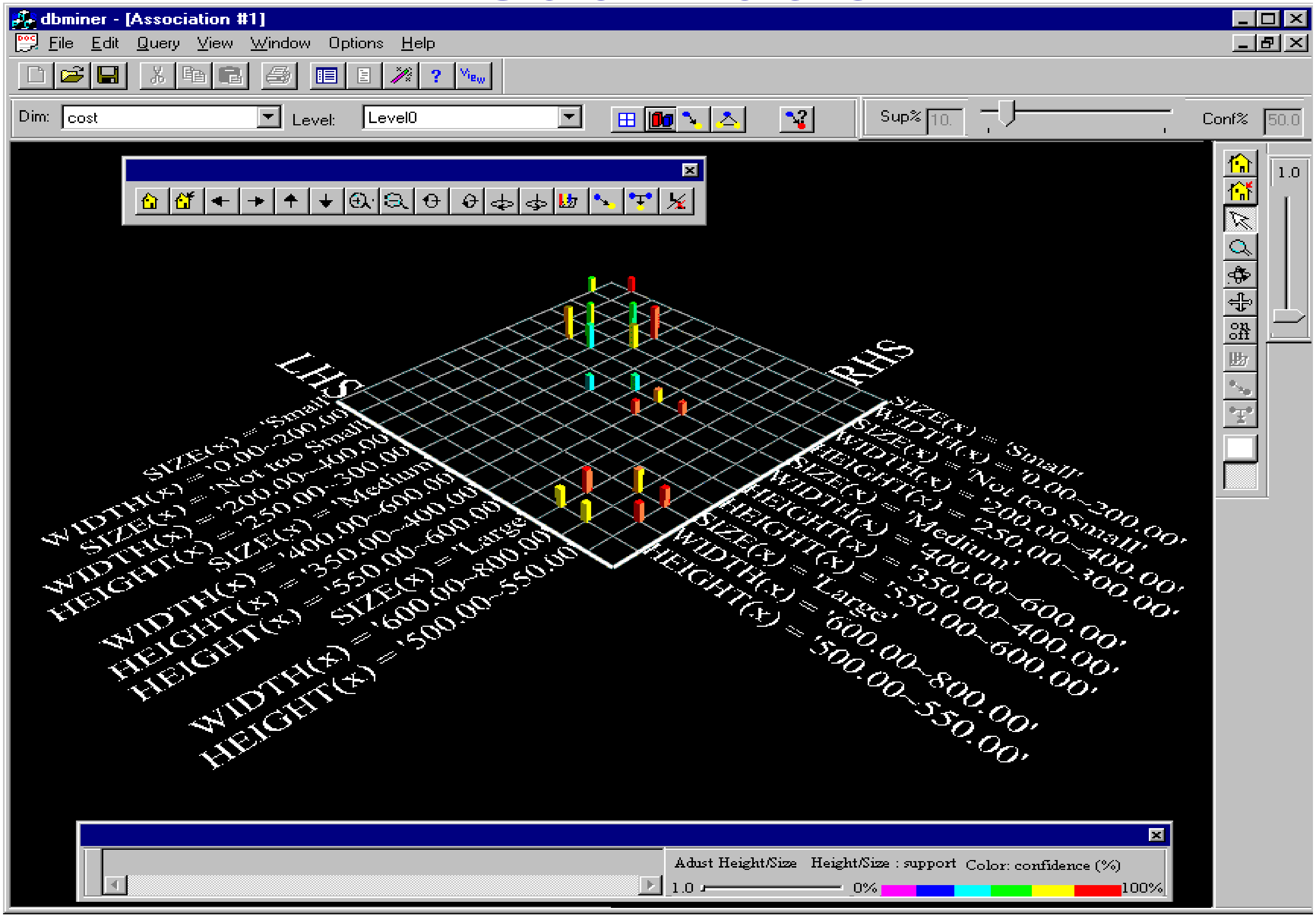
Visualization

- can be used on its own (usually for description and summarization tasks)
- can be used in combination with other DM techniques, for example
  - visualization of decision trees
  - cluster visualization
  - visualization of association rules
  - subgroup visualization

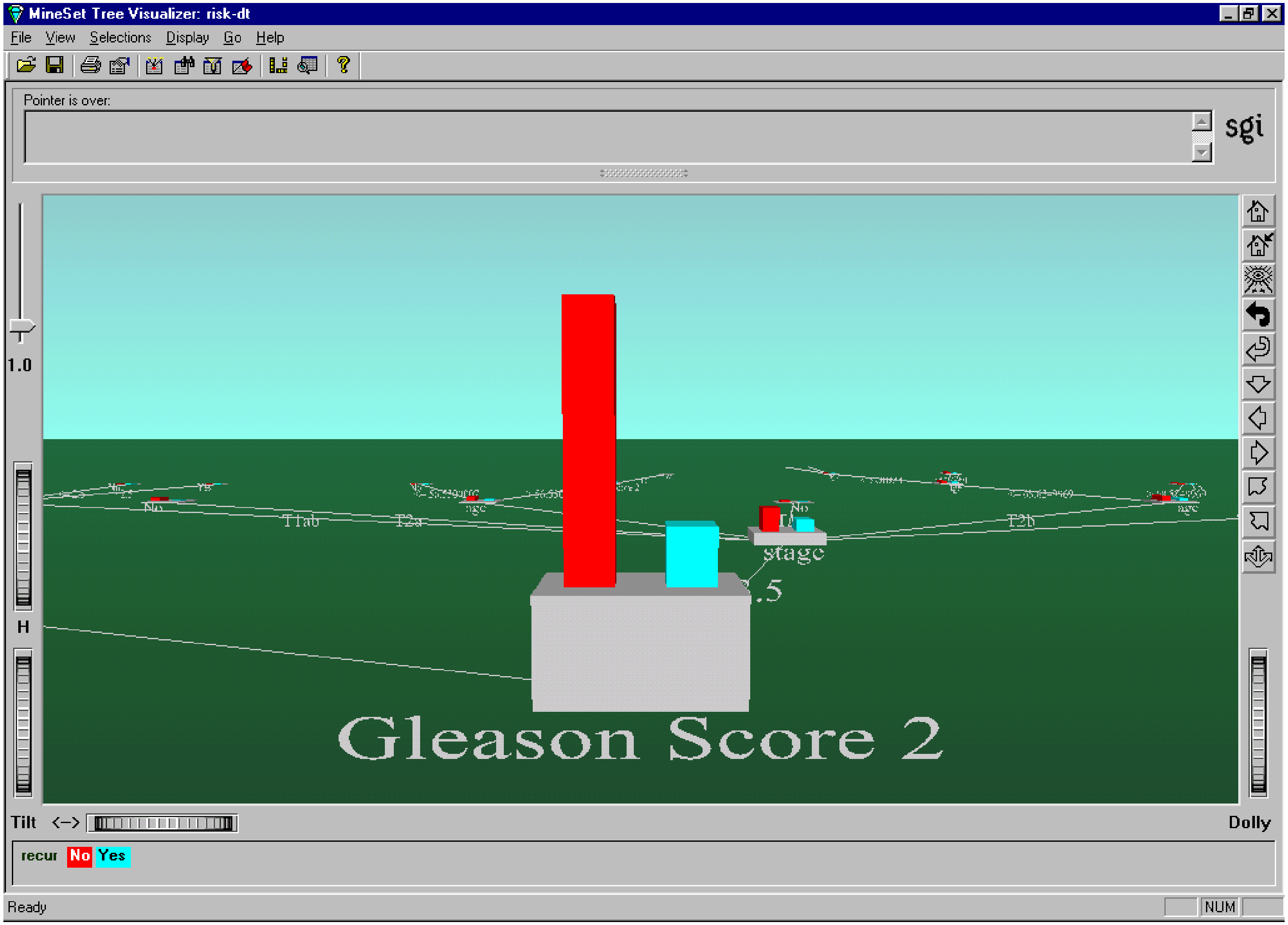
Data visualization: Scatter plot



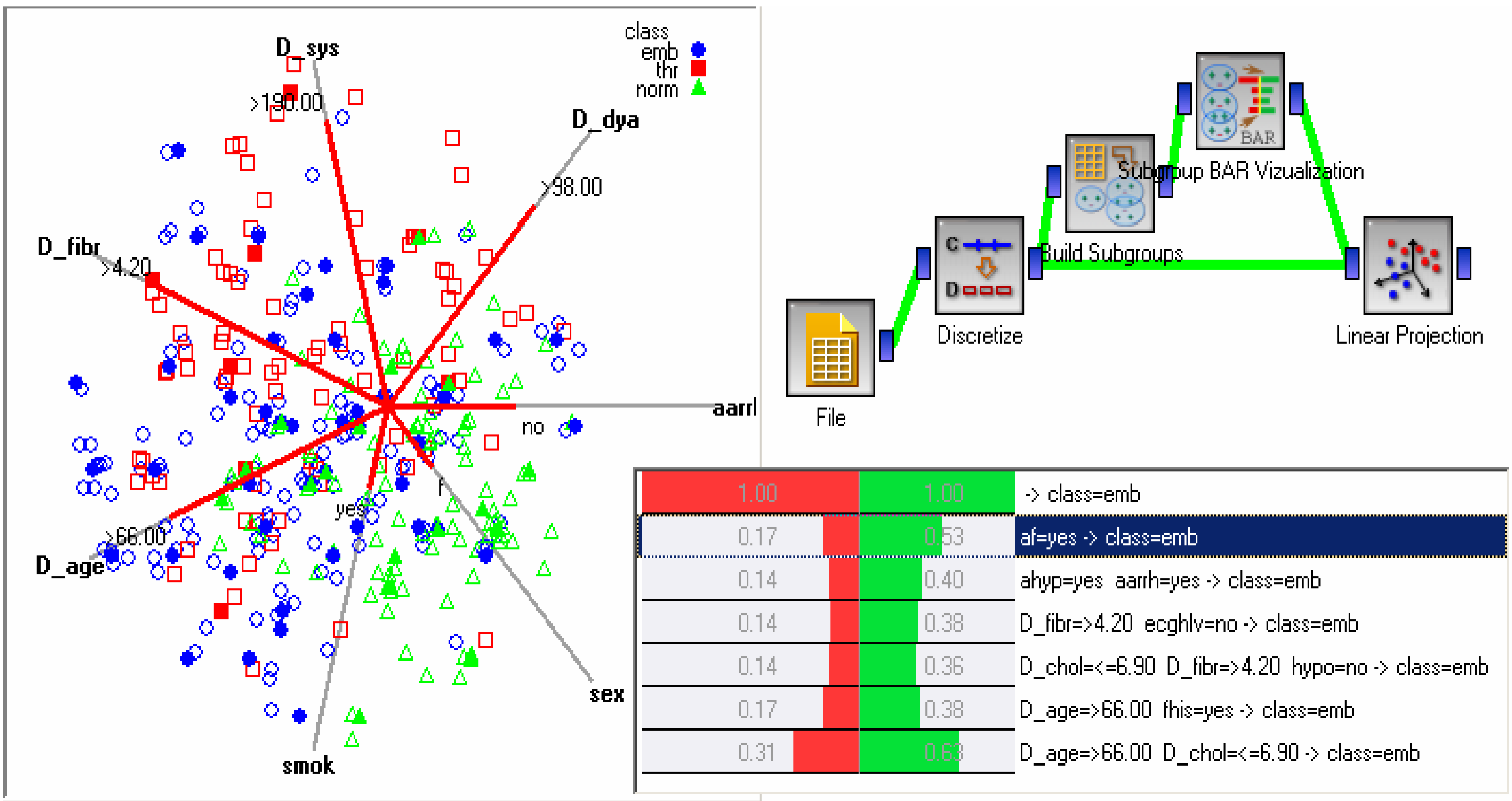
DB Miner: Association rule visualization



MineSet: Decision tree visualization



Orange: Visual programming and subgroup discovery visualization



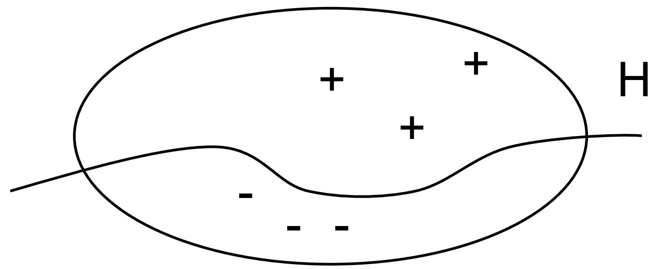
Part I. Introduction

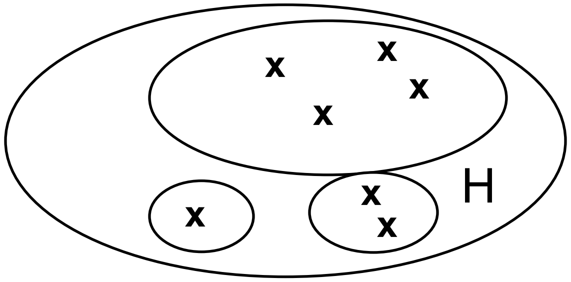
Data Mining and the KDD process

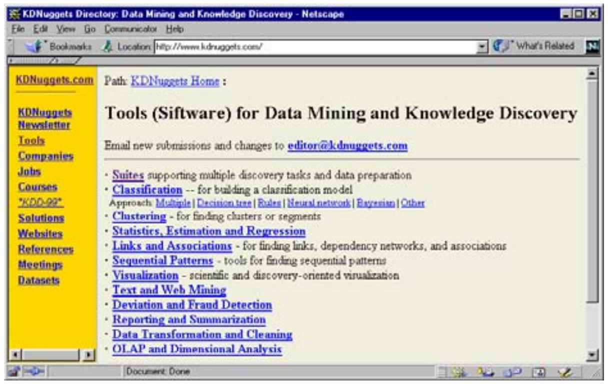
- DM standards, tools and visualization
- Classification of Data Mining techniques: Predictive and descriptive DM

Types of DM tasks

- **Predictive DM:**
  - Classification (learning of rules, decision trees, ...)
  - Prediction and estimation (regression)
  - Predictive relational DM (ILP)
- **Descriptive DM:**
  - description and summarization
  - dependency analysis (association rule learning)
  - discovery of properties and constraints
  - segmentation (clustering)
  - subgroup discovery
- **Text, Web and image analysis**

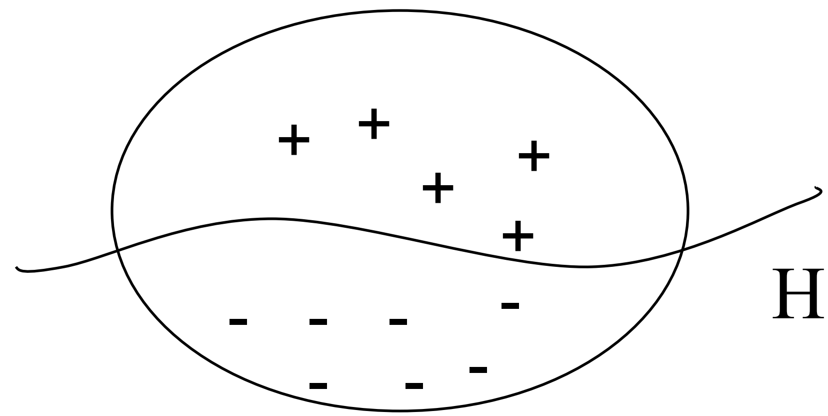




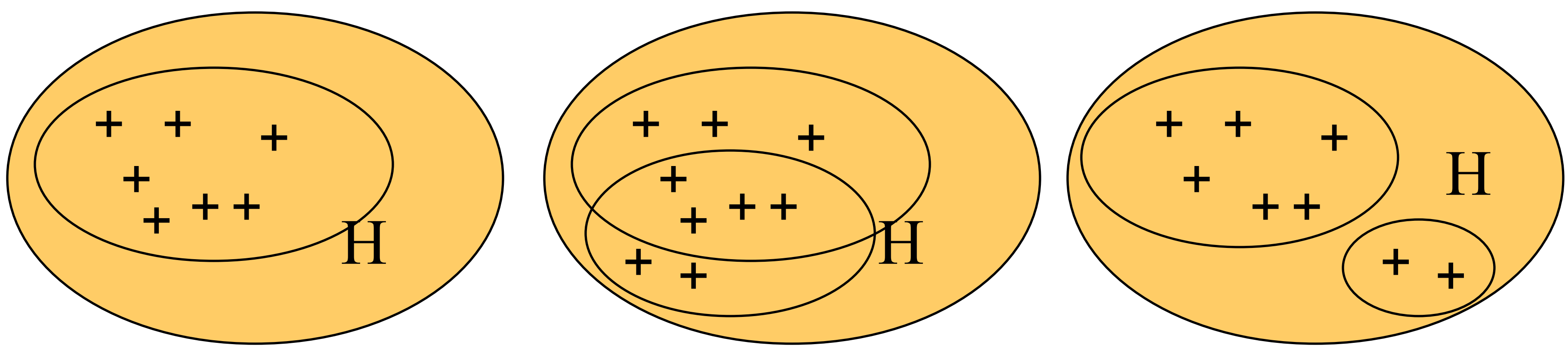


Predictive vs. descriptive induction

Predictive induction



Descriptive induction



Predictive vs. descriptive induction

- **Predictive induction:** Inducing classifiers for solving classification and prediction tasks,
  - Classification rule learning, Decision tree learning, ...
  - Bayesian classifier, ANN, SVM, ...
  - Data analysis through hypothesis generation and testing
- **Descriptive induction:** Discovering interesting regularities in the data, uncovering patterns, ... for solving KDD tasks
  - Symbolic clustering, Association rule learning, Subgroup discovery, ...
  - Exploratory data analysis

Predictive DM formulated as a machine learning task:

- Given a set of labeled **training examples** (n-tuples of attribute values, labeled by class name)

	A1	A2	A3	Class
example1	$v_{1,1}$	$v_{1,2}$	$v_{1,3}$	C_1
example2	$v_{2,1}$	$v_{2,2}$	$v_{2,3}$	C_2

. .

- By performing generalization from examples (induction) find a **hypothesis** (classification rules, decision tree, …) which explains the training examples, e.g. rules of the form:

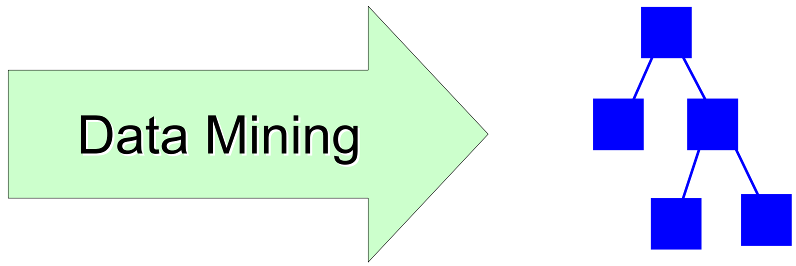
$$(A_i = v_{i,k}) \ \& \ (A_j = v_{j,l}) \ \& \ \ldots \rightarrow \text{Class} = C_n$$

Data Mining in a Nutshell

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

data

knowledge discovery from data



model, patterns, …

Given: transaction data table, relational database, text documents, Web pages

Find: a classification model, a set of interesting patterns

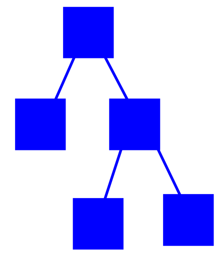
Data Mining in a Nutshell

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyc	hypermetrope	no	normal	SOFT
O15	pre-presbyc	hypermetrope	yes	reduced	NONE
O16	pre-presbyc	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

data

knowledge discovery
from data

Data Mining

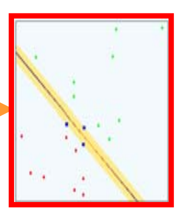


model, patterns, ...

Given: transaction data table, relational database, text documents, Web pages

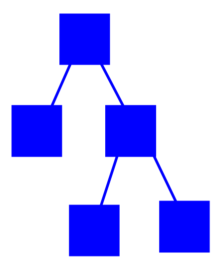
Find: a classification model, a set of interesting patterns

new unclassified instance



classified instance

black box classifier
no explanation



symbolic model
symbolic patterns

explanation

Predictive DM - Classification

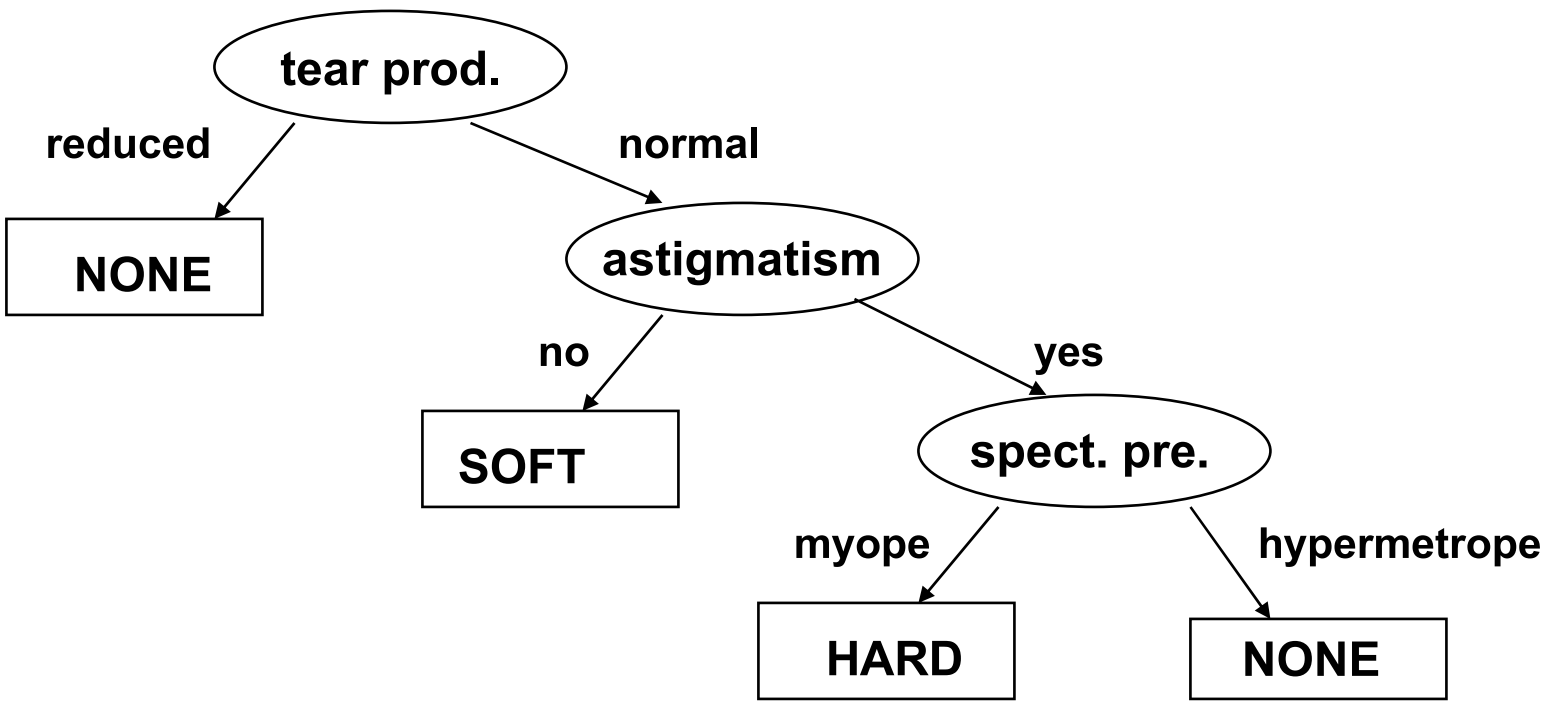
- data are objects, characterized with attributes - they belong to different classes (discrete labels)
- given objects described with attribute values, induce a model to predict different classes
- decision trees, if-then rules, discriminant analysis, ...

Data mining example
Input: Contact lens data

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

Contact lens data: Decision tree

Type of task: prediction and classification
Hypothesis language: decision trees
(nodes: attributes, arcs: values of attributes, leaves: classes)



Contact lens data: Classification rules

Type of task: prediction and classification
Hypothesis language: rules X ➔ C, if X then C
X conjunction of attribute values, C class

tear production=reduced → lenses=NONE
tear production=normal & astigmatism=yes &
spect. pre.=hypermetrope → lenses=NONE
tear production=normal & astigmatism=no →
lenses=SOFT
tear production=normal & astigmatism=yes &
spect. pre.=myope → lenses=HARD
DEFAULT lenses=NONE

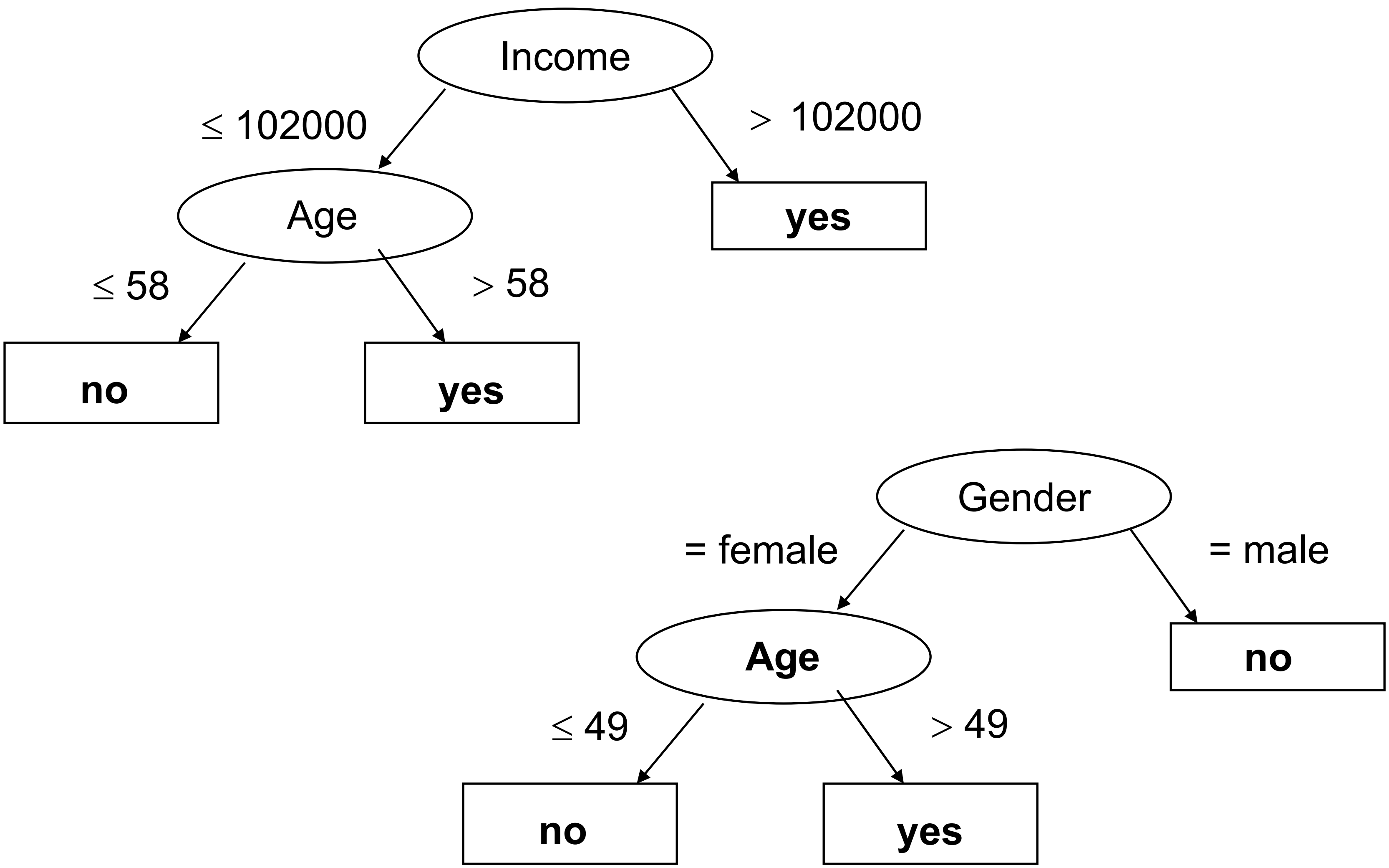
Task reformulation: Concept learning problem
(positive vs. negative examples of Target class)

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NO
O2	young	myope	no	normal	YES
O3	young	myope	yes	reduced	NO
O4	young	myope	yes	normal	YES
O5	young	hypermetrope	no	reduced	NO
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	YES
O15	pre-presbyo	hypermetrope	yes	reduced	NO
O16	pre-presbyo	hypermetrope	yes	normal	NO
O17	presbyopic	myope	no	reduced	NO
O18	presbyopic	myope	no	normal	NO
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NO

Illustrative example: Customer data

Customer	Gender	Age	Income	Spent	BigSpender
c1	male	30	214000	18800	yes
c2	female	19	139000	15100	yes
c3	male	55	50000	12400	no
c4	female	48	26000	8600	no
c5	male	63	191000	28100	yes
O6-O13	...	...	...	...	...
c14	female	61	95000	18100	yes
c15	male	56	44000	12000	no
c16	male	36	102000	13800	no
c17	female	57	215000	29300	yes
c18	male	33	67000	9700	no
c19	female	26	95000	11000	no
c20	female	55	214000	28800	yes

Customer data: Decision trees



Customer data: Association rules

Type of task: description (pattern discovery)

Hypothesis language: rules **X ➔ Y,** if X then Y

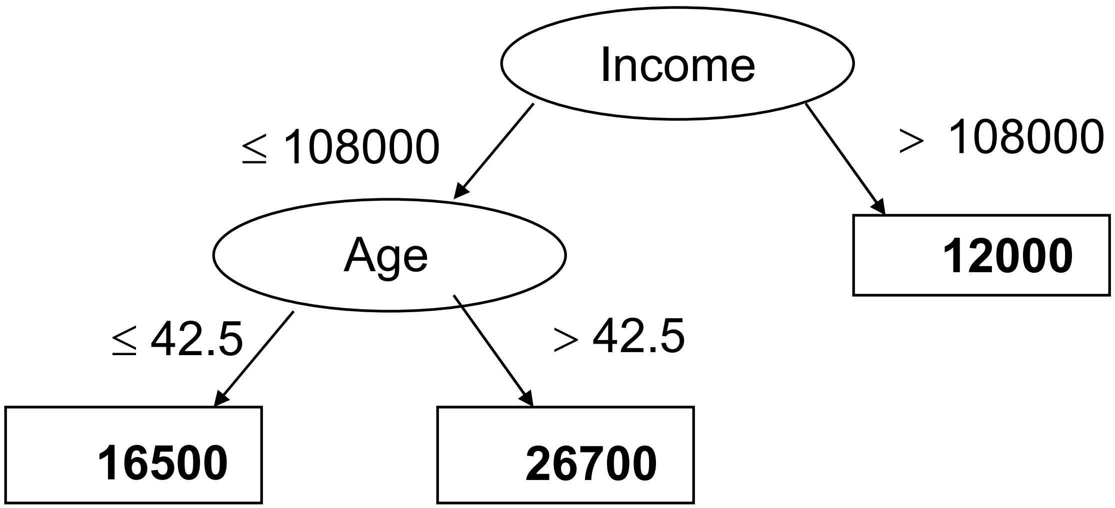
X, Y conjunctions of items (binary-valued attributes)

1. Age > 52 & BigSpender = no ➔ Sex = male
2. Age > 52 & BigSpender = no ➔
   Sex = male & Income ≤ 73250
3. Sex = male & Age > 52 & Income ≤ 73250 ➔
   BigSpender = no

Predictive DM - Estimation

- often referred to as regression
- data are objects, characterized with attributes (discrete or continuous), classes of objects are continuous (numeric)
- given objects described with attribute values, induce a model to predict the numeric class value
- regression trees, linear and logistic regression, ANN, kNN, ...

Customer data: regression tree



In the nodes one usually has Predicted value +- st. deviation

Relational Data Mining (Inductive Logic Programming) in a Nutshell

customer

ID	Zip	Sex	SoSt	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

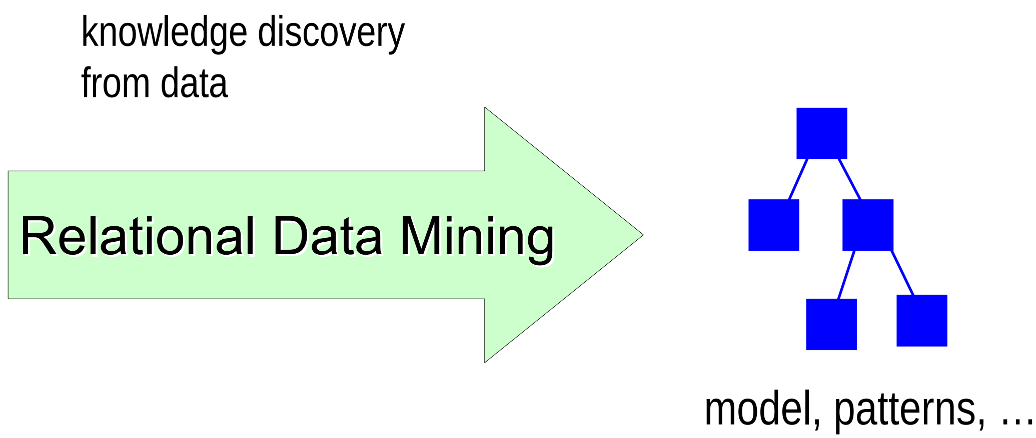
order

Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

store

Store ID	Size	Type	Location
...	...	...	...
12	small	franchise	city
17	large	indep	rural
...	...	...	...

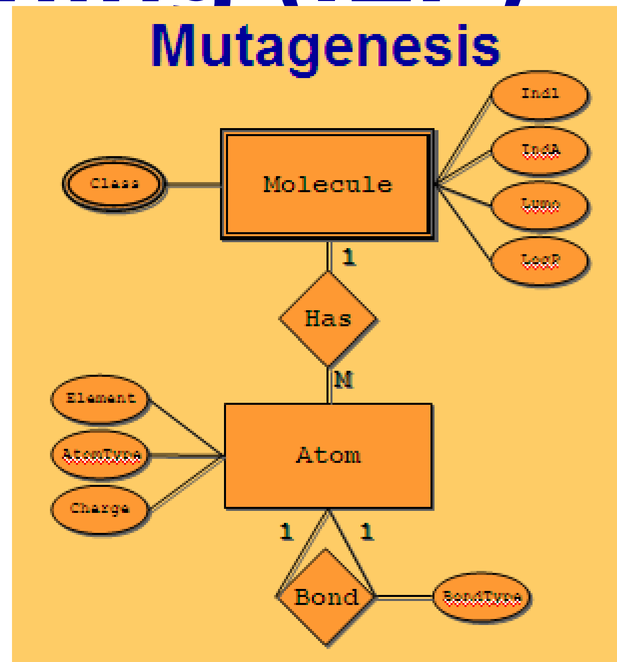
Relational representation of customers, orders and stores.

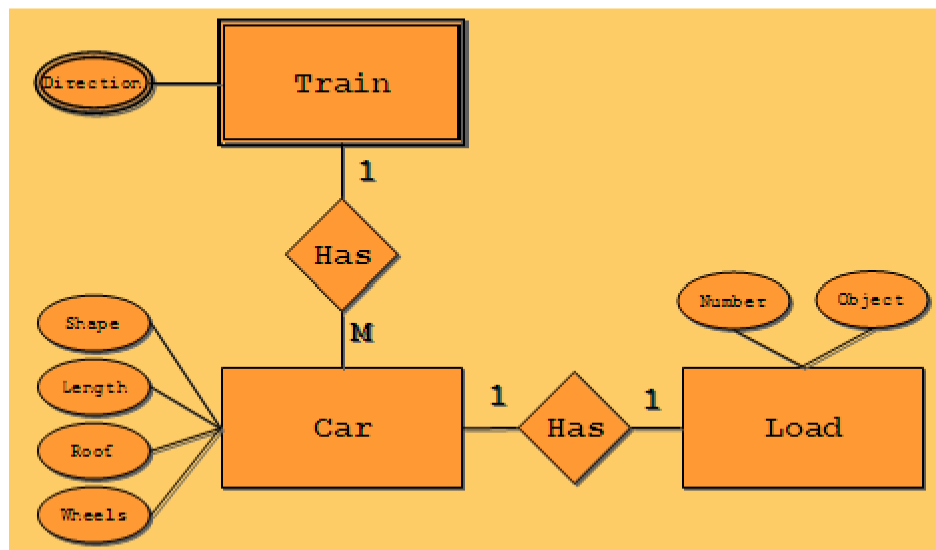


Given: a relational database, a set of tables. sets of logical facts, a graph, …
Find: a classification model, a set of interesting patterns

Relational Data Mining (ILP)

- Learning from multiple tables
- Complex relational problems:
  - temporal data: time series in medicine, trafic control, ...
  - structured data: representation of molecules and their properties in protein engineering, biochemistry, ...
- Illustrative example: structured objects - Trains





Relational Data Mining (Inductive Logic Programming)

customer							
ID	Zip	S ex	So St	In come	A ge	Cl ub	Re sp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

order				
Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

store			
Store ID	Size	Type	Location
...	...	...	...
12	small	franchise	city
17	large	indep	rural
...	...	...	...

Relational representation of customers, orders and stores.

customer							
ID	Zip	Sex	So St	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

order				
Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

store			
Store ID	Size	Type	Location
...	...	...	...
12	small	franchise	city
17	large	indep	rural
...	...	...	...

Relational representation of customers, orders and stores.

ID	Zip	Sex	Soc St	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34667	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

Basic table for analysis

ID	Zip	Sex	Soc St	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34667	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

Data table presented as logical facts (Prolog format)

customer(Id,Zip,Sex,SoSt,In,Age,Club,Re)

Prolog facts describing data in Table 2:

customer(3478,34667,m,si,60-70,32,me,nr).

customer(3479,43666,f,ma,80-90,45,nm,re).

Expressing a property of a relation:

customer(_,_,f,_,_,_,_,_).

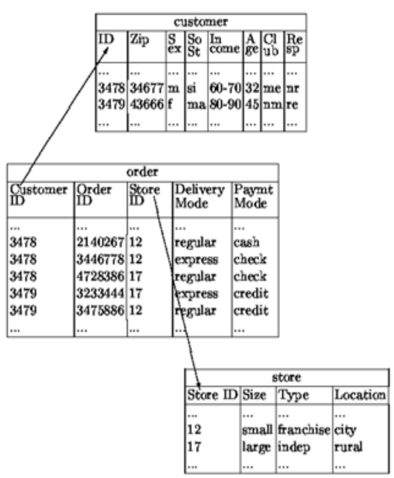
Relational Data Mining (Inductive Logic Programming)

Data bases:

- Name of relation p
- Attribute of p
- n-tuple $< v_1, ..., v_n >$ = row in a relational table
- relation p = set of n-tuples = relational table

Logic programming:

- Predicate symbol p
- Argument of predicate p
- Ground fact $p(v_1, ..., v_n)$
- Definition of predicate p
  - Set of ground facts
  - Prolog clause or a set of Prolog clauses



customer

ID	Zip	Sex	SoSt	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

order

Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

store

Store ID	Size	Type	Location
...	...	...	...
12	small	franchise	city
17	large	indep	rural
...	...	...	...

Relational representation of customers, orders and stores.

Example predicate definition:

good_customer(C) :-
customer(C,_,female,_,_,_,_,_),
order(C,_,_,_,creditcard).

Part I: Summary

- KDD is the overall process of discovering useful knowledge in data
  - many steps including data preparation, cleaning, transformation, pre-processing
- Data Mining is the data analysis phase in KDD
  - DM takes only 15%-25% of the effort of the overall KDD process
  - employing techniques from machine learning and statistics
- Predictive and descriptive induction have different goals: classifier vs. pattern discovery
- Many application areas
- Many powerful tools available

Part II. Predictive DM techniques

- Naive Bayesian classifier
- Decision tree learning
- Classification rule learning
- Classifier evaluation

Bayesian methods

- Bayesian methods – simple but powerful classification methods
  - Based on Bayesian formula

$$p(H \mid D) = \frac{p(D \mid H)}{p(D)} p(H)$$

- Main methods:
  - Naive Bayesian classifier
  - Semi-naïve Bayesian classifier
  - Bayesian networks *

* Out of scope of this course

Naïve Bayesian classifier

- Probability of class, for given attribute values

$$p(c_j \mid v_1 \ldots v_n) = p(c_j) \cdot \frac{p(v_1 \ldots v_n \mid c_j)}{p(v_1 \ldots v_n)}$$

- For all C_j compute probability $p(C_j)$, given values v_i of all attributes describing the example which we want to classify (assumption: conditional independence of attributes, when estimating $p(C_j)$ and $p(C_j \mid v_i)$)

$$p(c_j \mid v_1 \ldots v_n) \approx p(c_j) \cdot \prod_i \frac{p(c_j \mid v_i)}{p(c_j)}$$

- Output C_{MAX} with maximal posterior probability of class:

$$C_{MAX} = \arg\max_{Cj}\, p(c_j \mid v_1 \ldots v_n)$$

Naïve Bayesian classifier

$$p(c_j \mid v_1 \ldots v_n) = \frac{p(c_j \cdot v_1 \ldots v_n)}{p(v_1 \ldots v_n)} = \frac{p(v_1 \ldots v_n \mid c_j) \cdot p(c_j)}{p(v_1 \ldots v_n)} =$$

$$= \frac{\prod_i p(v_i \mid c_j) \cdot p(c_i)}{p(v_1 \ldots v_n)} = \frac{p(c_j)}{p(v_1 \ldots v_n)} \prod_i \frac{p(c_j \mid v_i) \cdot p(v_i)}{p(c_j)} =$$

$$= p(c_j) \cdot \frac{\prod p(v_i)}{p(v_1 \ldots v_n)} \prod_i \frac{p(c_j \mid v_i)}{p(c_j)} \approx p(c_j) \cdot \prod_i \frac{p(c_j \mid v_i)}{p(c_j)}$$

Semi-naïve Bayesian classifier

- Naive Bayesian estimation of probabilities (reliable)

$$\frac{p(c_j \mid v_i)}{p(c_j)} \cdot \frac{p(c_j \mid v_k)}{p(c_j)}$$

- Semi-naïve Bayesian estimation of probabilities (less reliable)

$$\frac{p(c_j \mid v_i, v_k)}{p(c_j)}$$

Probability estimation

- Relative frequency:

$$p(c_j) = \frac{n(c_j)}{N}, \; p(c_j \mid v_i) = \frac{n(c_j, v_i)}{n(v_i)}$$

j = 1. . k, for k classes

- Prior probability: Laplace law

$$p(c_j) = \frac{n(c_j) + 1}{N + k}$$

- m-estimate:

$$p(c_j) = \frac{n(c_j) + m \cdot p_a(c_j)}{N + m}$$

Probability estimation: intuition

- Experiment with N trials, n successful
- Estimate probability of success of next trial
- **Relative frequency: n/N**
  - reliable estimate when number of trials is large
  - Unreliable when number of trials is small, e.g., 1/1=1
- **Laplace: (n+1)/(N+2), (n+1)/(N+k),** k classes
  - Assumes uniform distribution of classes
- **m-estimate: $(n+m.p_{a)}$ /(N+m)**
  - Prior probability of success p_a, parameter m (weight of prior probability, i.e., number of ‘virtual’ examples)

Explanation of Bayesian classifier

- Based on information theory
  – Expected number of bits needed to encode a message = optimal code length -log p for a message, whose probability is p (*)
- Explanation based of the sum of information gains of individual attribute values v_i (Kononenko and Bratko 1991, Kononenko 1993)

$$-\log(p(c_j \mid v_1 \ldots v_n)) =$$

$$= -\log(p(c_j)) - \sum_{i=1}^{n}(-\log p(c_j) + \log(p(c_j \mid v_i))$$

* log p denotes binary logarithm

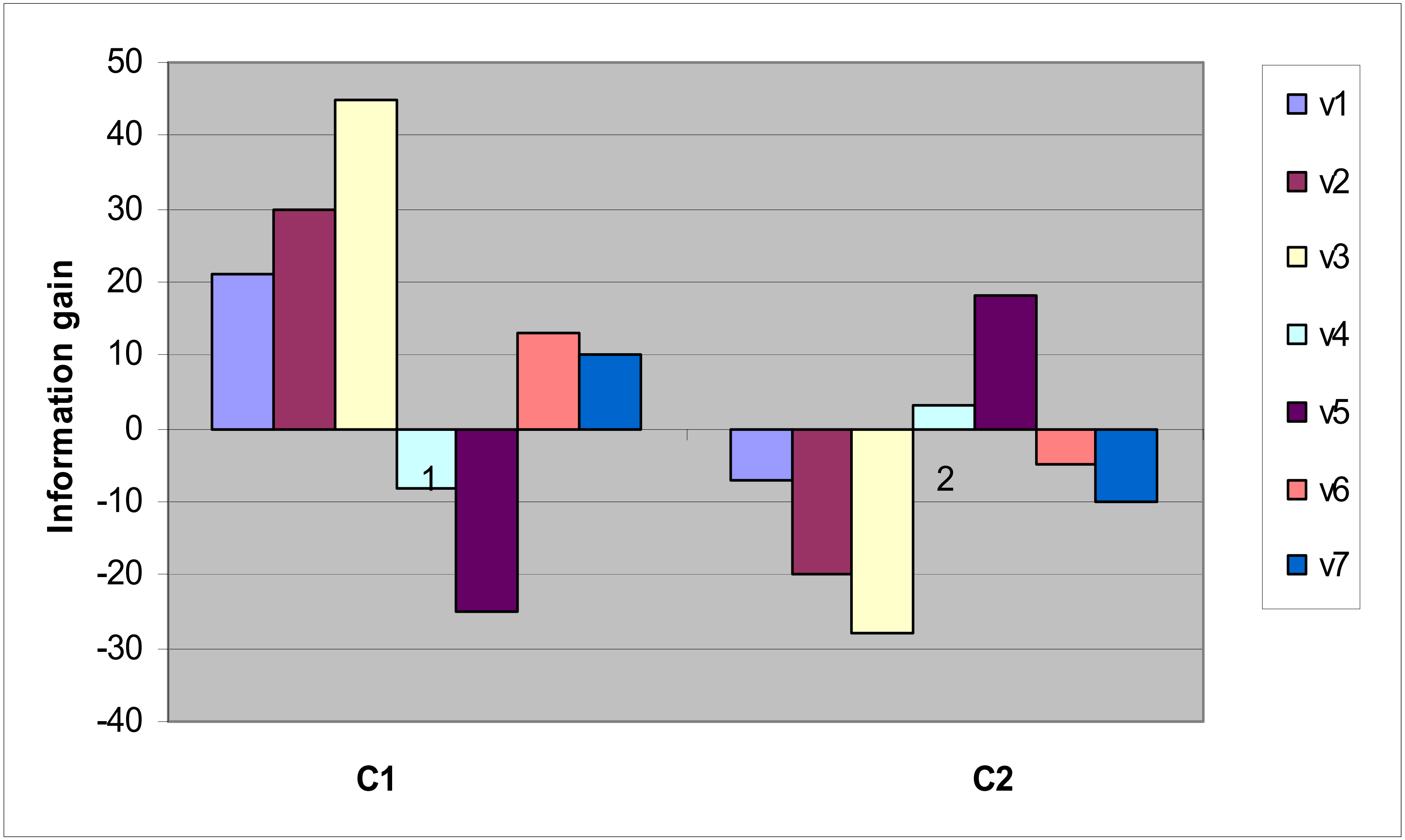
Example of explanation of semi-naïve Bayesian classifier

Hip surgery prognosis

Class = no (“no complications”, most probable class, 2 class problem)

Attribute value	For decision (bit)	Against (bit)
Age = 70-80	0.07	
Sex = Female		-0.19
Mobility before injury = Fully mobile	0.04	
State of health before injury = Other	0.52	
Mechanism of injury = Simple fall		-0.08
Additional injuries = None	0	
Time between injury and operation > 10 days	0.42	
Fracture classification acc. To Garden = Garden III		-0.3
Fracture classification acc. To Pauwels = Pauwels III		-0.14
Transfusion = Yes	0.07	
Antibiotic profilaxies = Yes		-0.32
Hospital rehabilitation = Yes	0.05	
General complications = None		0
Combination:	0.21	
Time between injury and examination < 6 hours		
AND Hospitalization time between 4 and 5 weeks		
Combination:	0.63	
Therapy = Artroplastic AND anticoagulant therapy = Yes		

Visualization of information gains for/against C_i



Naïve Bayesian classifier

- Naïve Bayesian classifier can be used
  – when we have sufficient number of training examples for reliable probability estimation
- It achieves good classification accuracy
  – can be used as 'gold standard' for comparison with other classifiers
- Resistant to noise (errors)
  – Reliable probability estimation
  – Uses all available information
- Successful in many application domains
  – Web page and document classification
  – Medical diagnosis and prognosis, …

Improved classification accuracy due to using m-estimate

	Primary tumor	Breast cancer	thyroid	Rheumatology
#instan	339	288	884	355
#class	22	2	4	6
#attrib	17	10	15	32
#values	2	2.7	9.1	9.1
majority	25%	80%	56%	66%
entropy	3.64	0.72	1.59	1.7

	Relative freq.	m-estimate
Primary tumor	48.20%	52.50%
Breast cancer	77.40%	79.70%
hepatitis	58.40%	90.00%
lymphography	79.70%	87.70%

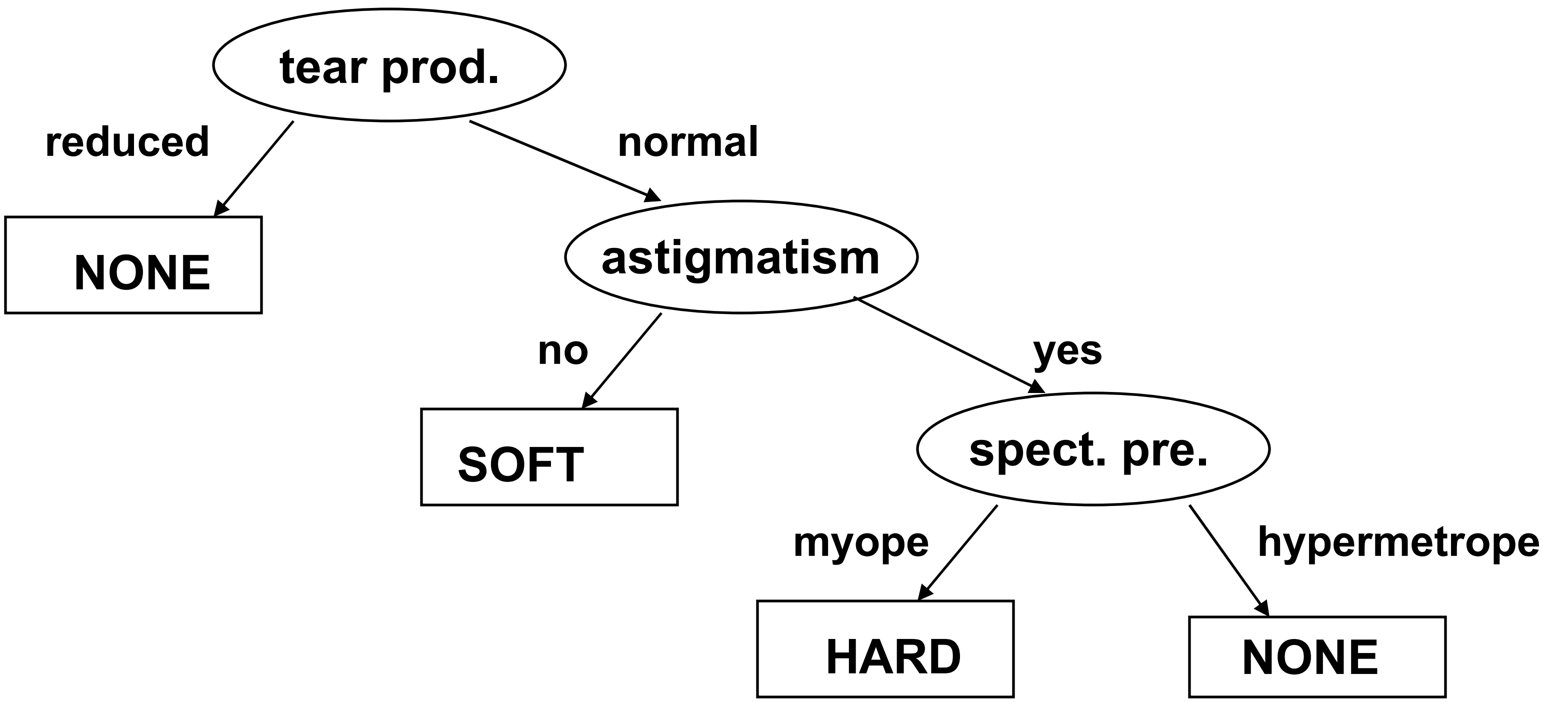
Part II. Predictive DM techniques

- Naïve Bayesian classifier
- Decision tree learning
- Classification rule learning
- Classifier evaluation

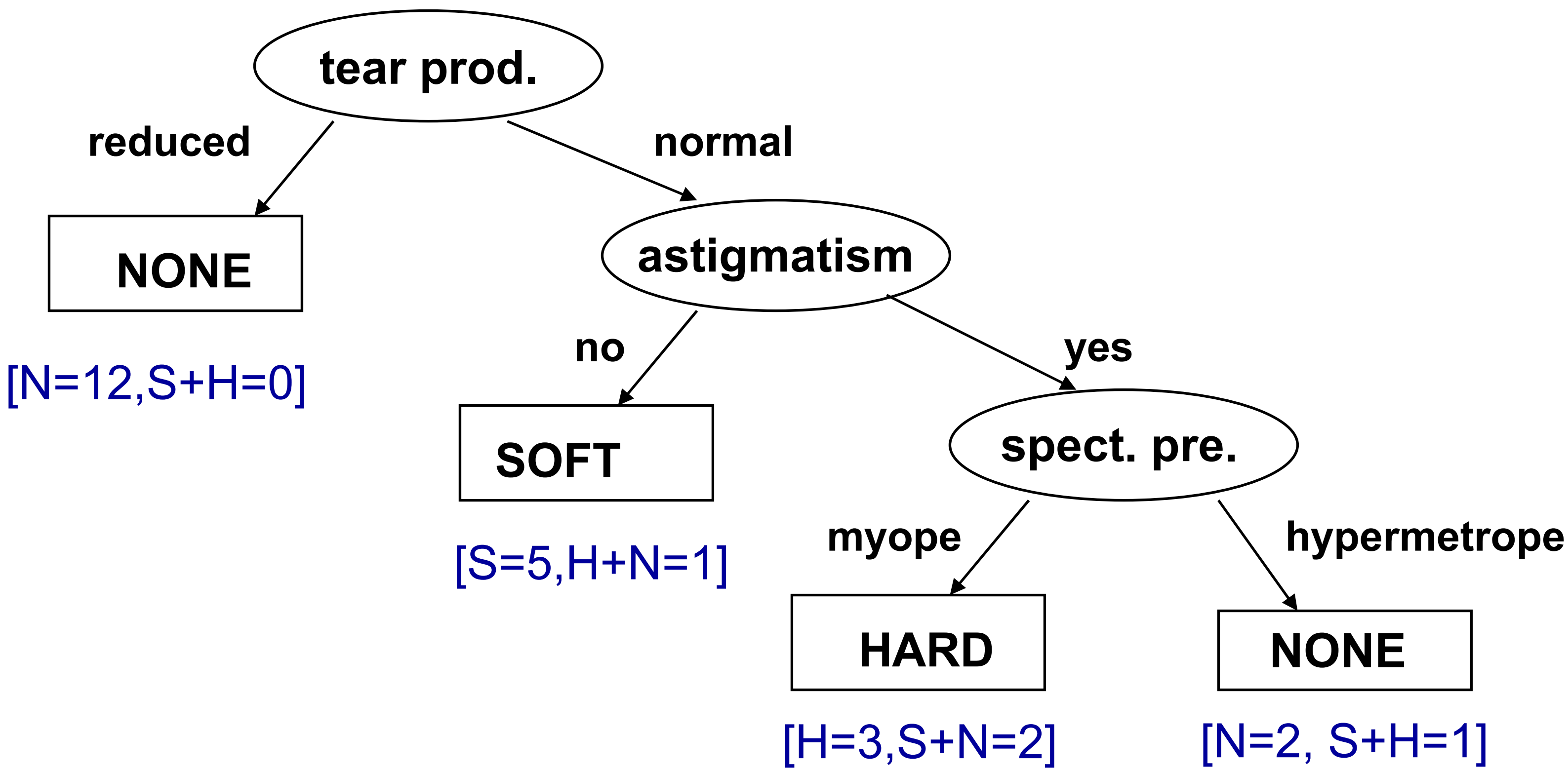
Illustrative example: Contact lenses data

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

Decision tree for contact lenses recommendation



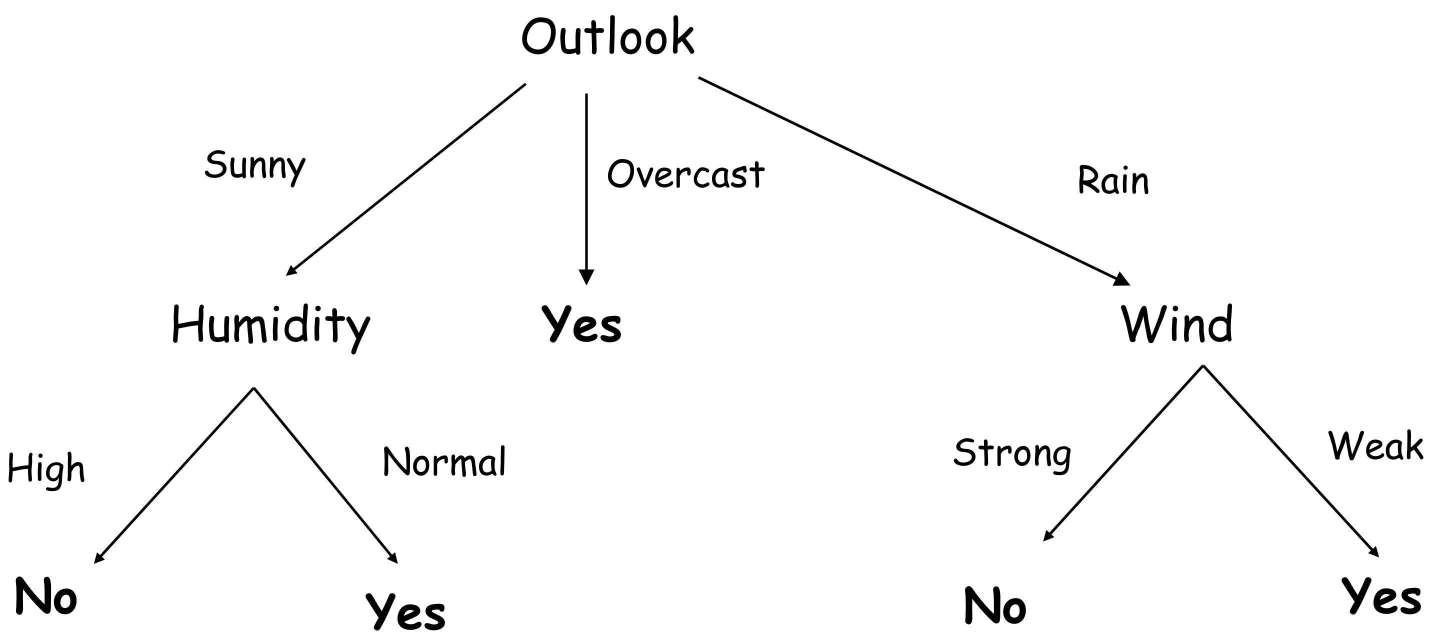
Decision tree for contact lenses recommendation



PlayTennis: Training examples

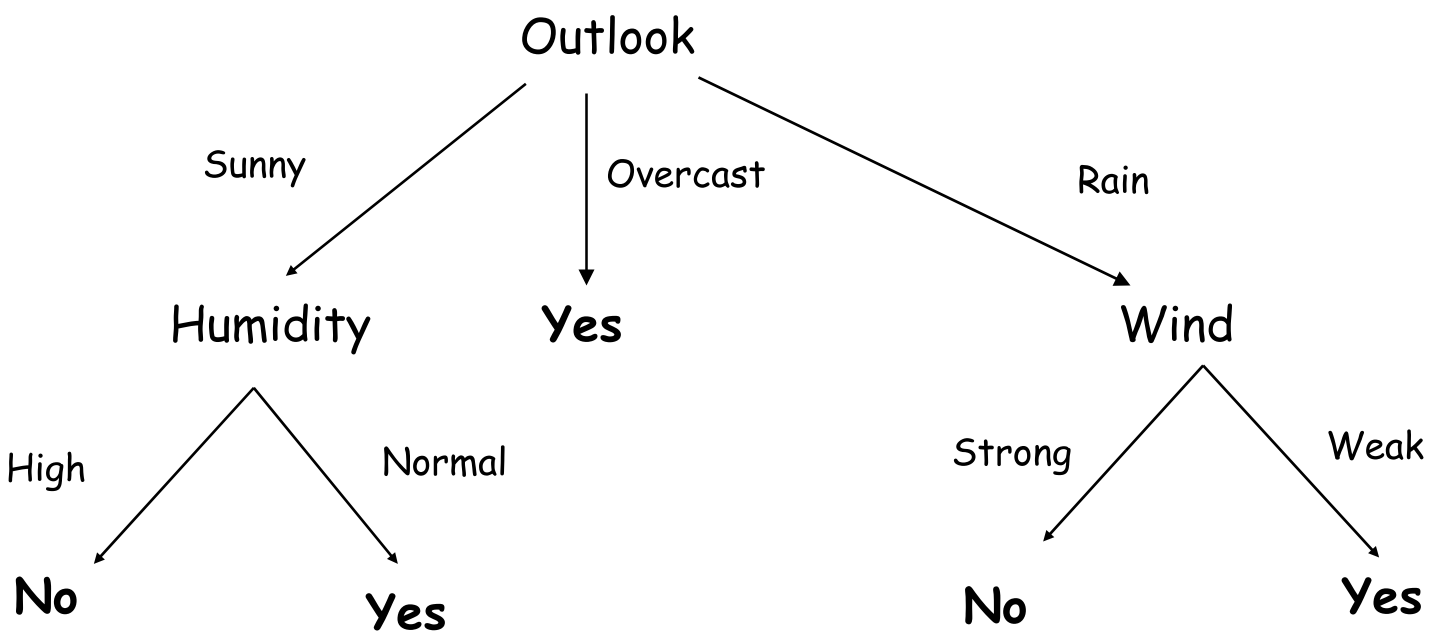
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree representation for PlayTennis



- each internal node is a test of an attribute
- each branch corresponds to an attribute value
- each path is a conjunction of attribute values
- each leaf node assigns a classification

Decision tree representation for PlayTennis



Decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances

(Outlook=Sunny $\wedge$ Humidity=Normal)

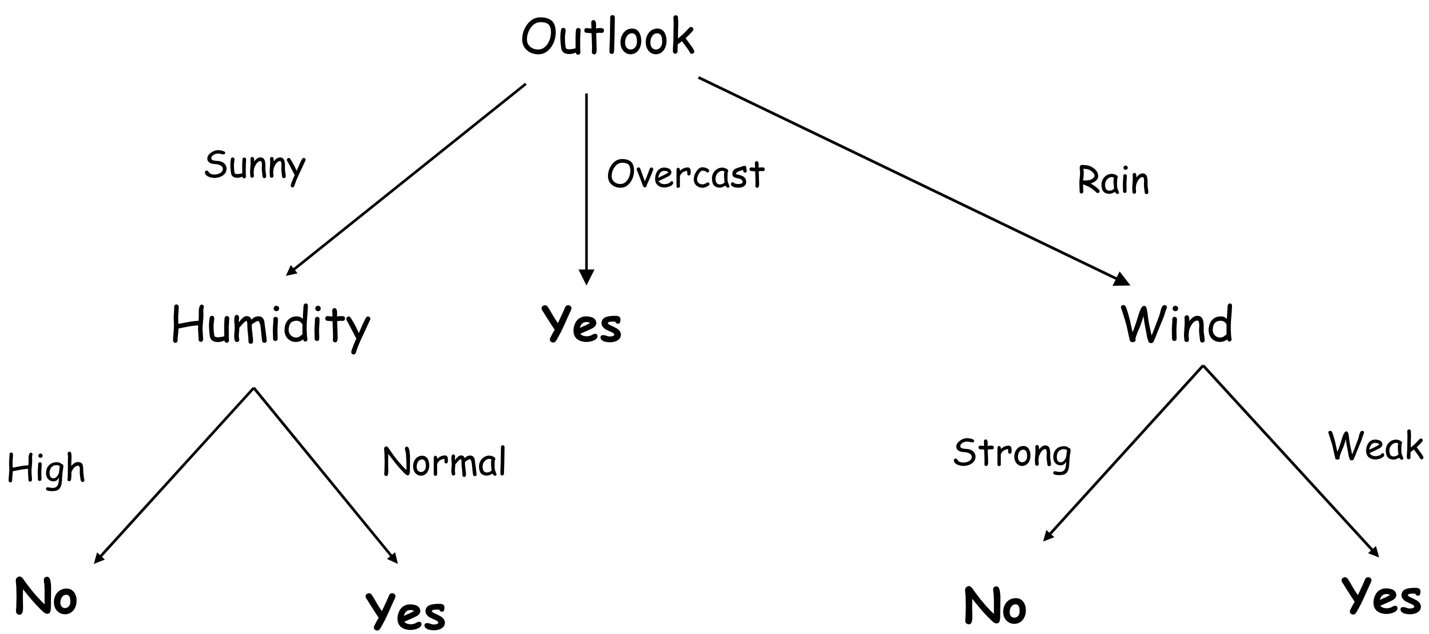
V (Outlook=Overcast)

V (Outlook=Rain $\wedge$ Wind=Weak)

PlayTennis: Other representations

- Logical expression for PlayTennis=Yes:
  - (Outlook=Sunny $\wedge$ Humidity=Normal) $\vee$ (Outlook=Overcast) $\vee$ (Outlook=Rain $\wedge$ Wind=Weak)
- Converting a tree to if-then rules
  - **IF** Outlook=Sunny $\wedge$ Humidity=Normal **THEN** PlayTennis=Yes
  - **IF** Outlook=Overcast **THEN** PlayTennis=Yes
  - **IF** Outlook=Rain $\wedge$ Wind=Weak **THEN** PlayTennis=Yes
  - **IF** Outlook=Sunny $\wedge$ Humidity=High **THEN** PlayTennis=No
  - **IF** Outlook=Rain $\wedge$ Wind=Strong **THEN** PlayTennis=No

PlayTennis: Using a decision tree for classification



Is Saturday morning OK for playing tennis?

Outlook=Sunny, Temperature=Hot, Humidity=High, Wind=Strong

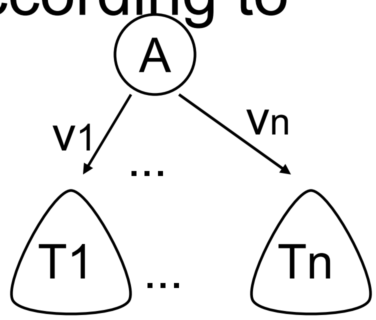
PlayTennis = No, because Outlook=Sunny $\wedge$ Humidity=High

Appropriate problems for decision tree learning

- Classification problems: classify an instance into one of a discrete set of possible categories (medical diagnosis, classifying loan applicants, …)
- Characteristics:
  - instances described by attribute-value pairs
    (discrete or real-valued attributes)
  - target function has discrete output values
    (boolean or multi-valued, if real-valued then regression trees)
  - disjunctive hypothesis may be required
  - training data may be noisy
    (classification errors and/or errors in attribute values)
  - training data may contain missing attribute values

Learning of decision trees

- ID3 (Quinlan 1979), CART (Breiman et al. 1984), C4.5, WEKA, ...
  - create the root node of the tree
  - if all examples from S belong to the same class Cj
    - then label the root with Cj
  - else
    - select the ‘most informative’ attribute **A** with values **v1, v2, … vn**
    - divide training set **S** into **S1,… , Sn** according to values **v1,…,vn**
    - recursively build sub-trees **T1,…,Tn** for **S1,…,Sn**



Search heuristics in ID3

- Central choice in ID3: Which attribute to test at each node in the tree ? The attribute that is most useful for classifying examples.
- Define a statistical property, called **information gain**, measuring how well a given attribute separates the training examples w.r.t their target classification.
- First define a measure commonly used in information theory, called **entropy**, to characterize the (im)purity of an arbitrary collection of examples.

Entropy

- **S** - training set, $\mathbf{C_1,\ldots,C_N}$ - classes
- **Entropy E(S)** – measure of the impurity of training set S

$$E(S) = -\sum_{c=1}^{N} p_c . \log_2 p_c$$

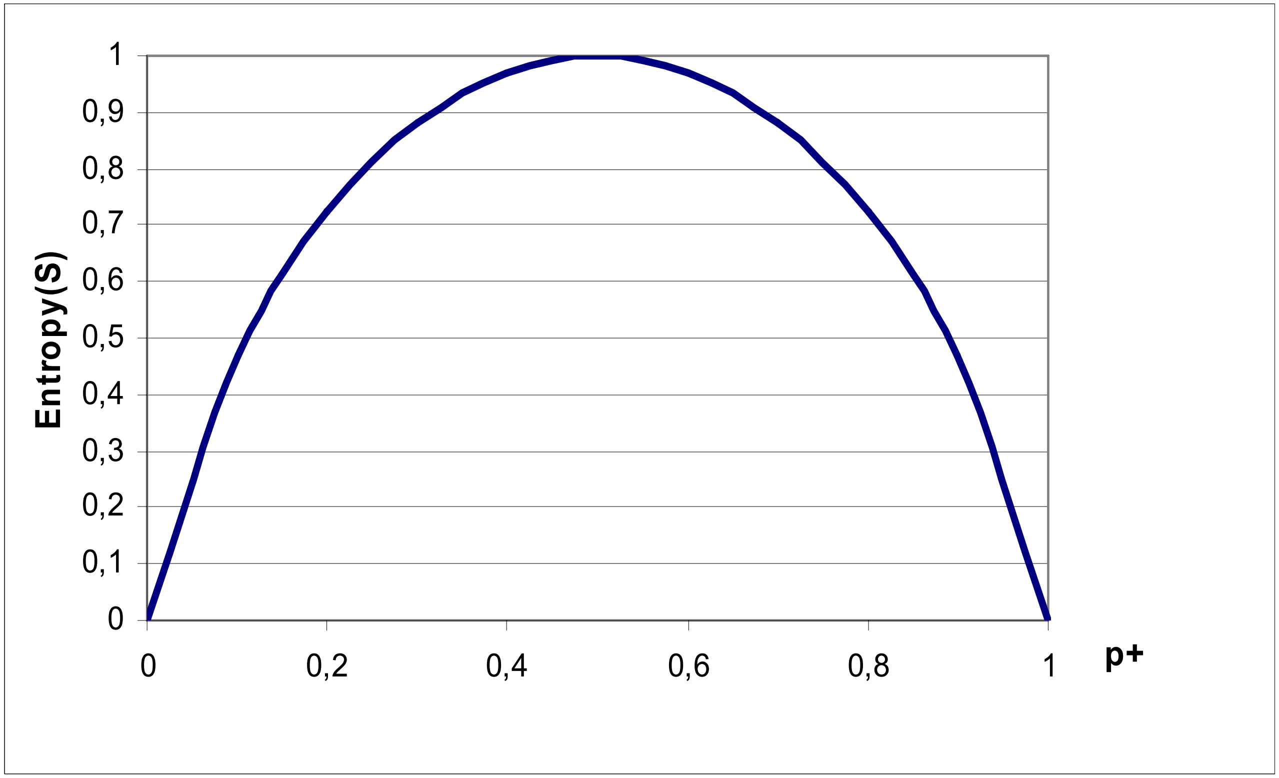
$\mathbf{p_c}$ - prior probability of class $\mathbf{C_c}$
(relative frequency of $\mathbf{C_c}$ in **S**)

- Entropy in binary classification problems

$$\mathbf{E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-}$$

Entropy

- $E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-$
- The entropy function relative to a Boolean classification, as the proportion $\mathbf{p_+}$ of positive examples varies between 0 and 1



Entropy – why ?

- **Entropy E(S) =** expected amount of information (in bits) needed to assign a class to a randomly drawn object in S (under the optimal, shortest-length code)
- Why ?
- Information theory: optimal length code assigns $-\log_2 p$ bits to a message having probability p
- So, in binary classification problems, the expected number of bits to encode + or – of a random member of S is:

$$p_+ (- \log_2 p_+) + p_- (- \log_2 p_-) = - p_+ \log_2 p_+ - p_- \log_2 p_-$$

PlayTennis: Entropy

- Training set S: 14 examples (9 pos., 5 neg.)
- Notation: S = [9+, 5-]
- $E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-$
- Computing entropy, if probability is estimated by relative frequency

$$E(S) = -\left(\frac{|S_+|}{|S|} \cdot \log \frac{|S_+|}{|S|} \right) - \left(\frac{|S_-|}{|S|} \cdot \log \frac{|S_-|}{|S|} \right)$$

- $E([9+,5-]) = - (9/14) \log_2(9/14) - (5/14) \log_2(5/14)$

  $= 0.940$

PlayTennis: Entropy

- $E(S) = - p_+ \log_2 p_+ - p_- \log_2 p_-$
- $E(9+,5-) = -(9/14) \log_2(9/14) - (5/14) \log_2(5/14) = 0.940$

Outlook?
- $\xrightarrow{\text{Sunny}}$ {D1,D2,D8,D9,D11} [2+, 3-] E=0.970
- $\xrightarrow{\text{Overcast}}$ {D3,D7,D12,D13} [4+, 0-] E=0
- $\xrightarrow{\text{Rain}}$ {D4,D5,D6,D10,D14} [3+, 2-] E=0.970

Humidity?
- $\xrightarrow{\text{High}}$ [3+, 4-] E=0.985
- $\xrightarrow{\text{Normal}}$ [6+, 1-] E=0.592

Wind?
- $\xrightarrow{\text{Weak}}$ [6+, 2-] E=0.811
- $\xrightarrow{\text{Strong}}$ [3+, 3-] E=1.00

Information gain search heuristic

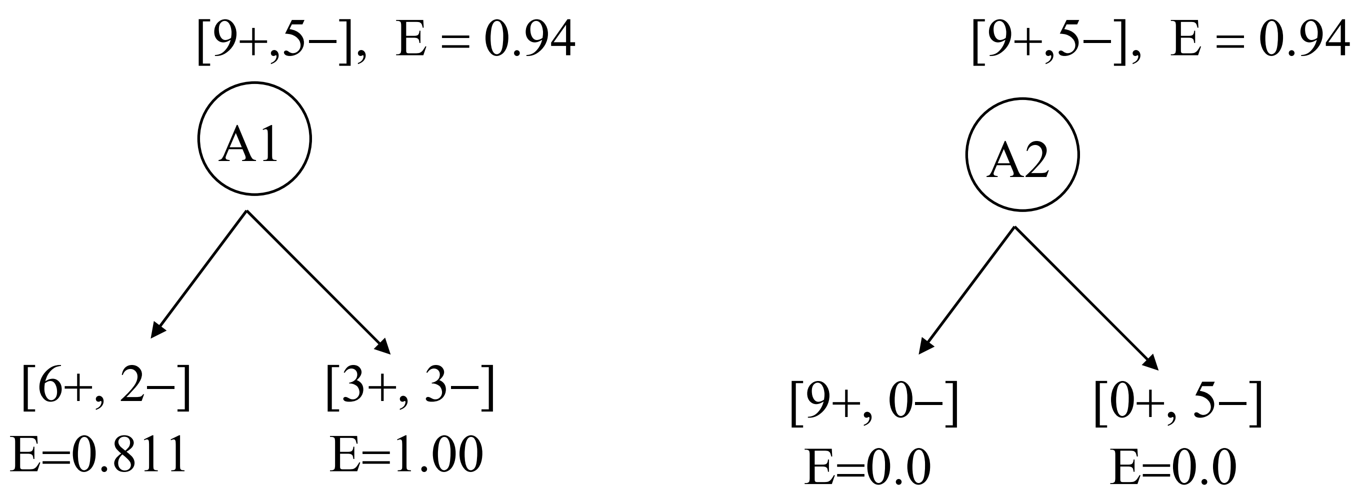
- **Information gain** measure is aimed to minimize the number of tests needed for the classification of a new object
- **Gain(S,A)** – expected reduction in entropy of S due to sorting on A

$$Gain(S,A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

- **Most informative** attribute: **max Gain(S,A)**

Information gain search heuristic

- Which attribute is more informative, A1 or A2 ?



- Gain(S,A1) = 0.94 – (8/14 x 0.811 + 6/14 x 1.00) = 0.048
- Gain(S,A2) = 0.94 – 0 = 0.94 A2 has max Gain

PlayTennis: Information gain

$$Gain(S, A) = E(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \cdot E(S_v)$$

- Values(Wind) = {Weak, Strong}

$$\begin{tikzcd} & {[6+,\ 2-]\quad E=0.811} \\ \text{Wind?} \arrow[ur, "\text{Weak}"] \arrow[dr, "\text{Strong}"'] & \\ & {[3+,\ 3-]\quad E=1.00} \end{tikzcd}$$

  – S = [9+,5-], E(S) = 0.940
  – S_{weak} = [6+,2-], E(S_{weak}) = 0.811
  – S_{strong} = [3+,3-], E(S_{strong}) = 1.0
  – **Gain(S,Wind) =** E(S) - (8/14)E(S_{weak}) - (6/14)E(S_{strong}) = 0.940 - (8/14)x0.811 - (6/14)x1.0=**0.048**

PlayTennis: Information gain

- **Which attribute is the best?**
  - Gain(S,Outlook)=0.246 *MAX !*
  - Gain(S,Humidity)=0.151
  - Gain(S,Wind)=0.048
  - Gain(S,Temperature)=0.029

PlayTennis: Information gain

Outlook?

Rain → {D4,D5,D6,D10,D14} [3+, 2-] E > 0 ***???***

Overcast → {D3,D7,D12,D13} [4+, 0-] E = 0 ***OK - assign class Yes***

Sunny → {D1,D2,D8,D9,D11} [2+, 3-] E > 0 ***???***

- Which attribute should be tested here?
  - Gain(S_{sunny}, Humidity) = 0.97-(3/5)0-(2/5)0 = 0.970 ***MAX !***
  - Gain(S_{sunny},Temperature) = 0.97-(2/5)0-(2/5)1-(1/5)0 = 0.570
  - Gain(S_{sunny},Wind) = 0.97-(2/5)1-(3/5)0.918 = 0.019

Probability estimates

- **Relative frequency** :
  - problems with small samples

$$p(Class \mid Cond) = = \frac{n(Class.Cond)}{n(Cond)}$$

[6+,1-] (7) = 6/7

[2+,0-] (2) = 2/2 = 1

- **Laplace estimate** :
  - assumes uniform prior distribution of k classes

$$= \frac{n(Class.Cond) + 1}{n(Cond) + k} \qquad k = 2$$

[6+,1-] (7) = 6+1 / 7+2 = 7/9

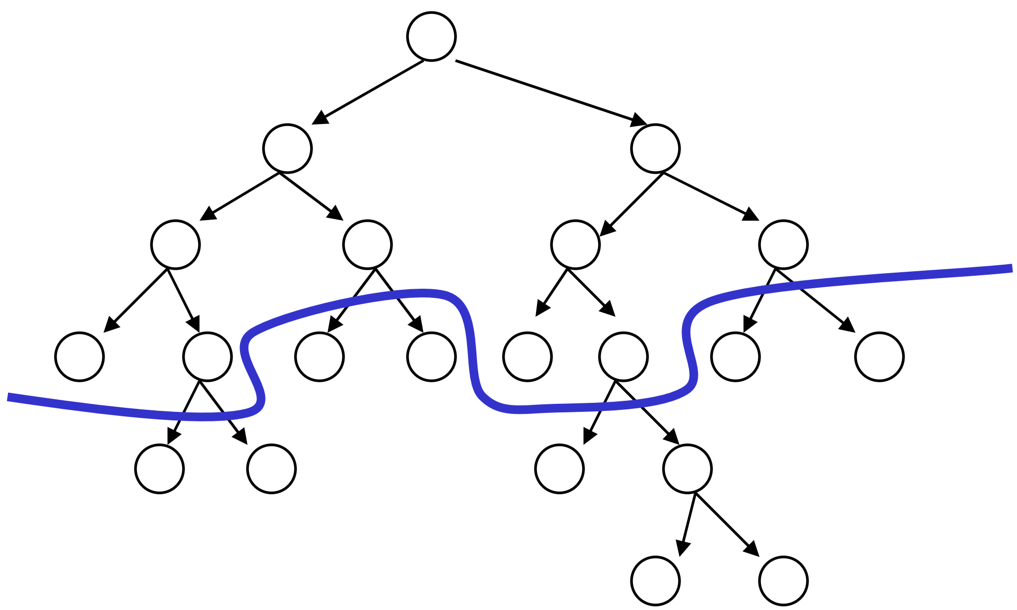
[2+,0-] (2) = 2+1 / 2+2 = 3/4

Heuristic search in ID3

- **Search bias:** Search the space of decision trees from simplest to increasingly complex (greedy search, no backtracking, prefer small trees)
- **Search heuristics:** At a node, select the attribute that is most useful for classifying examples, split the node accordingly
- **Stopping criteria:** A node becomes a leaf
  - if all examples belong to same class C_j, label the leaf with C_j
  - if all attributes were used, label the leaf with the most common value C_k of examples in the node
- **Extension to ID3:** handling noise - tree pruning

Pruning of decision trees

- Avoid overfitting the data by tree pruning
- Pruned trees are
  - less accurate on training data
  - more accurate when classifying unseen data



Handling noise – Tree pruning

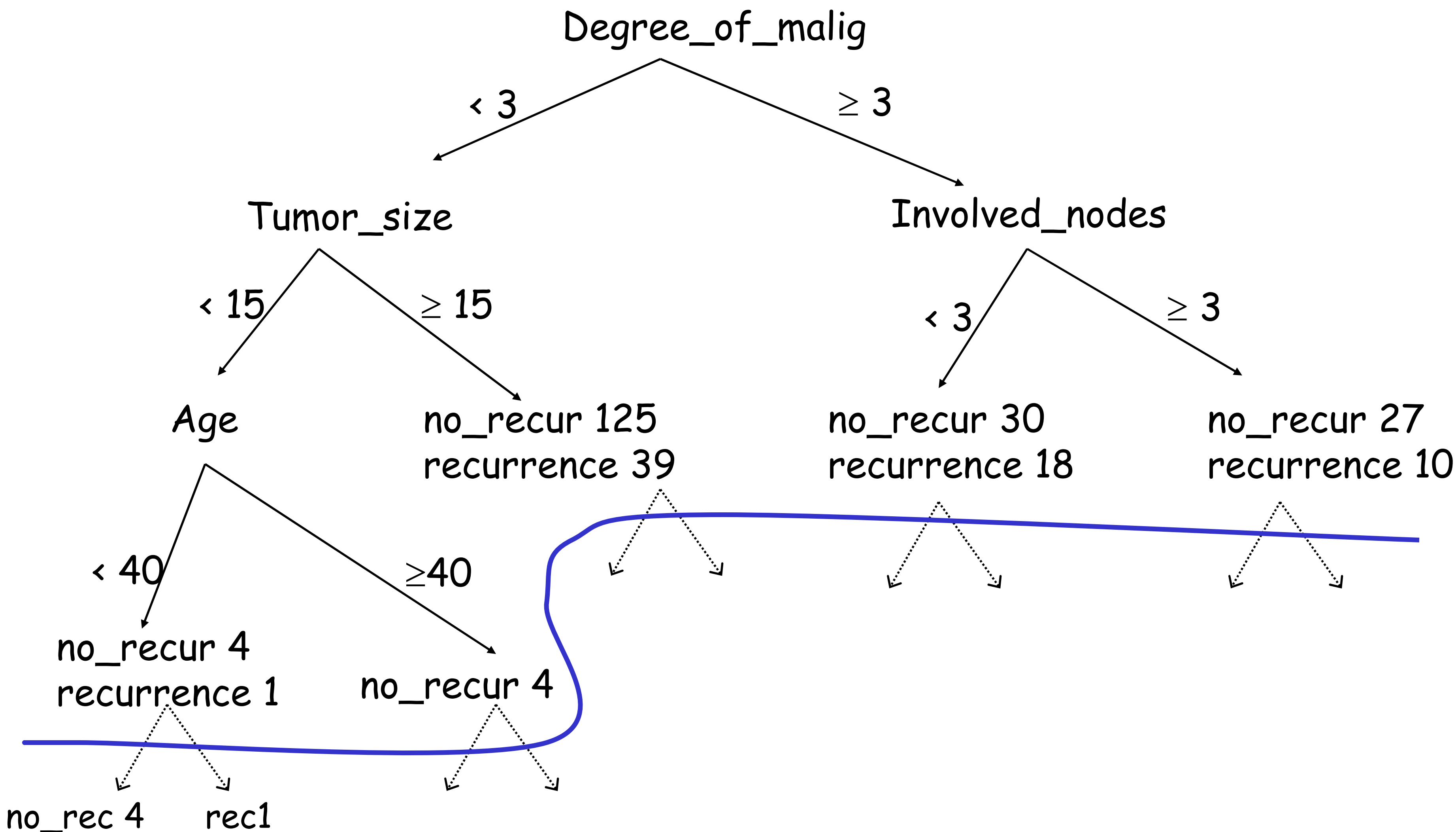
Sources of imperfection

1. Random errors (noise) in training examples
   - erroneous attribute values
   - erroneous classification
2. Too sparse training examples (incompleteness)
3. Inappropriate/insufficient set of attributes (inexactness)
4. Missing attribute values in training examples

Handling noise – Tree pruning

- Handling imperfect data
  - handling imperfections of type 1-3
    - pre-pruning (stopping criteria)
    - post-pruning / rule truncation
  - handling missing values
- Pruning avoids perfectly fitting noisy data: relaxing the completeness (fitting all +) and consistency (fitting all -) criteria in ID3

Prediction of breast cancer recurrence: Tree pruning

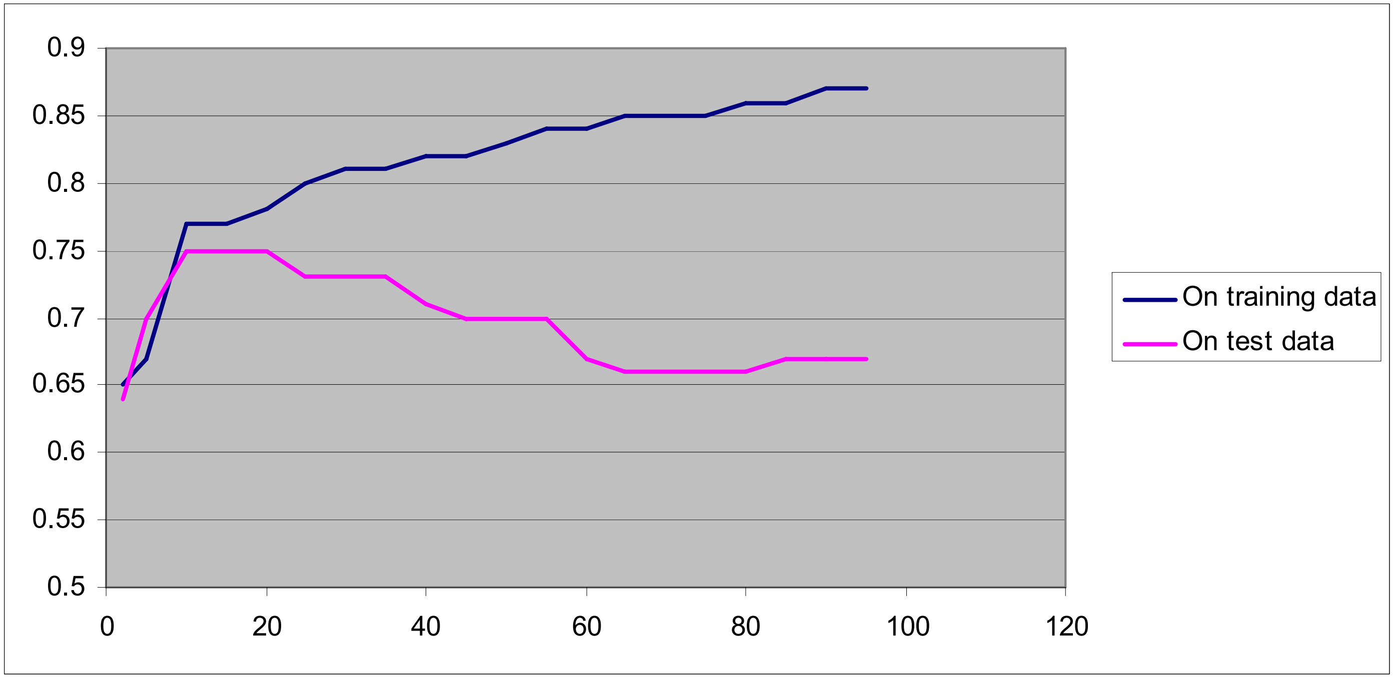


Accuracy and error

- Accuracy: percentage of correct classifications
  - on the training set
  - on unseen instances
- How accurate is a decision tree when classifying unseen instances
  - An estimate of accuracy on unseen instances can be computed, e.g., by averaging over 4 runs:
    - split the example set into training set (e.g. 70%) and test set (e.g. 30%)
    - induce a decision tree from training set, compute its accuracy on test set
- Error = 1 - Accuracy
- High error may indicate data overfitting

Overfitting and accuracy

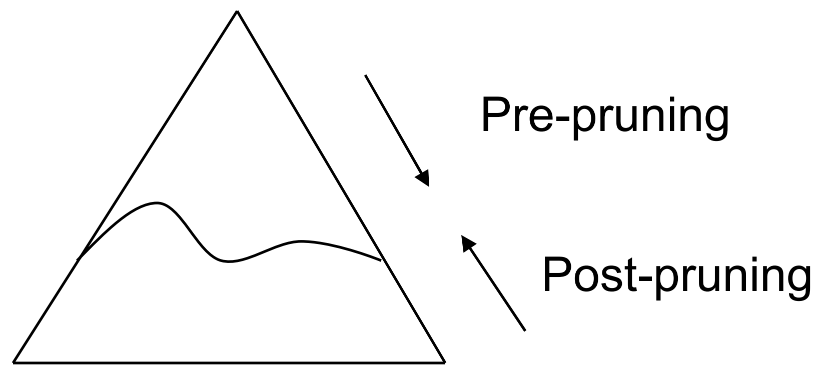
- Typical relation between tree size and accuracy



- Question: how to prune optimally?

Avoiding overfitting

- How can we avoid overfitting?
  - Pre-pruning (forward pruning): stop growing the tree e.g., when data split not statistically significant or too few examples are in a split
  - Post-pruning: grow full tree, then post-prune



- forward pruning considered inferior (myopic)
- post pruning makes use of sub trees

How to select the “best” tree

- Measure performance over training data (e.g., pessimistic post-pruning, Quinlan 1993)
- Measure performance over separate validation data set (e.g., reduced error pruning, Quinlan 1987)
  - until further pruning is harmful DO:
    - for each node evaluate the impact of replacing a subtree by a leaf, assigning the majority class of examples in the leaf, if the pruned tree performs no worse than the original over the validation set
    - greedily select the node whose removal most improves tree accuracy over the validation set
- MDL: minimize size(tree)+size(misclassifications(tree))

Selected decision/regression tree learners

- Decision tree learners
  - ID3 (Quinlan 1979)
  - CART (Breiman et al. 1984)
  - Assistant (Cestnik et al. 1987)
  - C4.5 (Quinlan 1993), C5 (See5, Quinlan)
  - J48 (available in WEKA)
- Regression tree learners, model tree learners
  - M5, M5P (implemented in WEKA)

Features of C4.5

- Implemented as part of the WEKA data mining workbench
- Handling noisy data: post-pruning
- Handling incompletely specified training instances: ‘unknown’ values (**?**)
  – in learning assign conditional probability of value v: $p(v|C) = p(vC) / p(C)$
  – in classification: follow all branches, weighted by prior prob. of missing attribute values

Other features of C4.5

- Binarization of attribute values
  - for continuous values select a boundary value maximally increasing the informativity of the attribute: sort the values and try every possible split (done automaticaly)
  - for discrete values try grouping the values until two groups remain *
- 'Majority' classification in NULL leaf (with no corresponding training example)
  - if an example 'falls' into a NULL leaf during classification, the class assigned to this example is the majority class of the parent of the NULL leaf

* the basic C4.5 doesn't support binarisation of discrete attributes, it supports grouping

Part II. Predictive DM techniques

- Naïve Bayesian classifier
- Decision tree learning
- Classification rule learning
- Classifier evaluation

Rule Learning in a Nutshell

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

data

knowledge discovery
from data

Rule learning

Model: a set of rules

Patterns: individual rules

Given: transaction data table, relational database (a set of objects, described by attribute values)

Find: a classification model in the form of a set of rules;
or a set of interesting patterns in the form of individual rules

Rule set representation

- Rule base is a disjunctive set of conjunctive rules
- Standard form of rules:

  IF Condition THEN Class

  Class IF Conditions

  Class $\leftarrow$ Conditions

IF Outlook=Sunny $\wedge$ Humidity=Normal **THEN** PlayTennis=Yes
IF Outlook=Overcast **THEN** PlayTennis=Yes
IF Outlook=Rain $\wedge$ Wind=Weak **THEN** PlayTennis=Yes

- Form of CN2 rules:

  IF Conditions THEN MajClass [ClassDistr]

- Rule base: {R1, R2, R3, …, DefaultRule}

Data mining example
Input: Contact lens data

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NONE
O2	young	myope	no	normal	SOFT
O3	young	myope	yes	reduced	NONE
O4	young	myope	yes	normal	HARD
O5	young	hypermetrope	no	reduced	NONE
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	SOFT
O15	pre-presbyo	hypermetrope	yes	reduced	NONE
O16	pre-presbyo	hypermetrope	yes	normal	NONE
O17	presbyopic	myope	no	reduced	NONE
O18	presbyopic	myope	no	normal	NONE
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NONE

Contact lens data: Classification rules

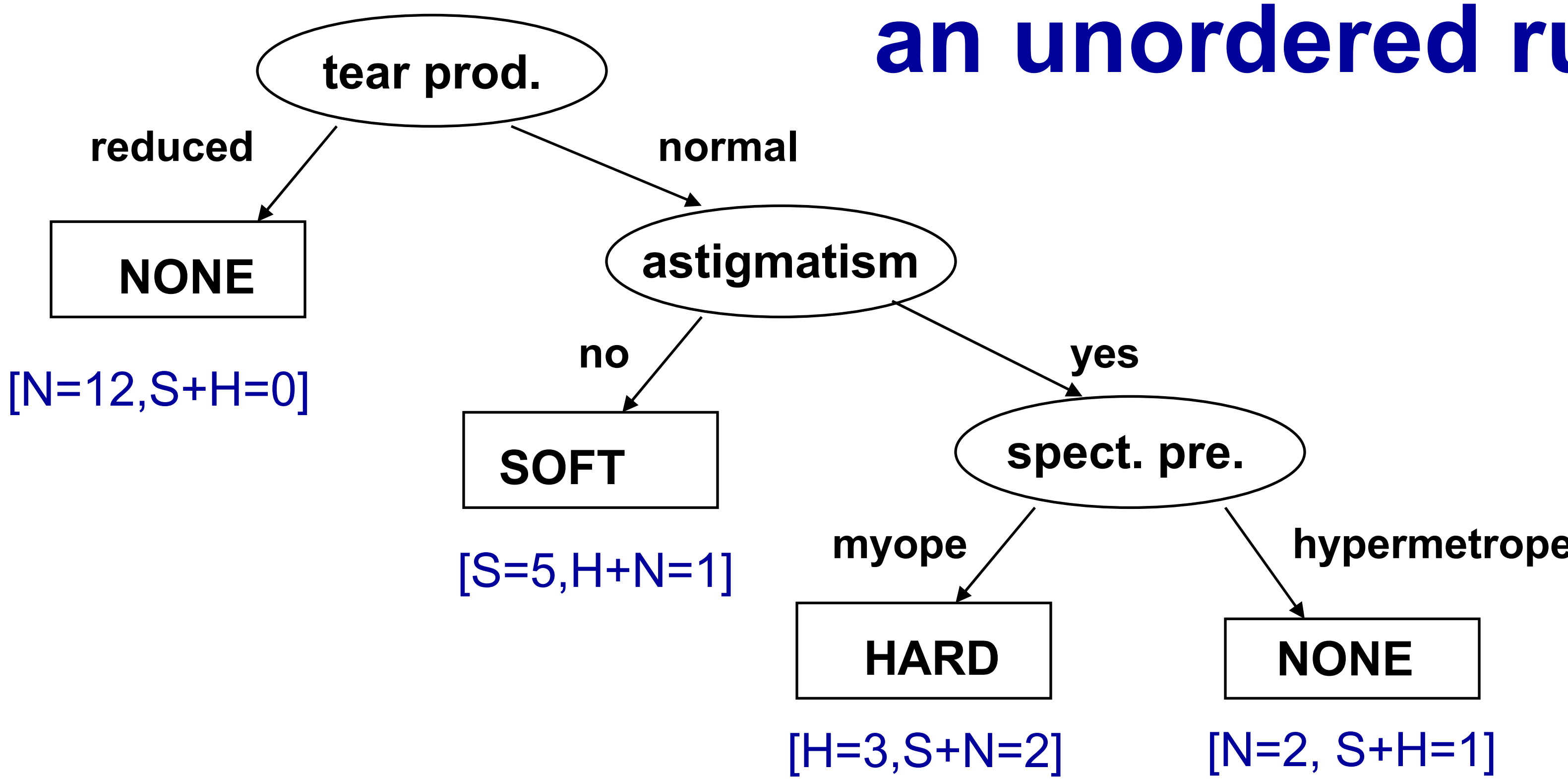
Type of task: prediction and classification
Hypothesis language: rules X ➔ C, if X then C
X conjunction of attribute values, C class

tear production=reduced → lenses=NONE
tear production=normal & astigmatism=yes & spect. pre.=hypermetrope → lenses=NONE
tear production=normal & astigmatism=no → lenses=SOFT
tear production=normal & astigmatism=yes & spect. pre.=myope → lenses=HARD
DEFAULT lenses=NONE

Rule learning

- Two rule learning approaches:
  – Learn decision tree, convert to rules
  – Learn set/list of rules
    - Learning an unordered set of rules
    - Learning an ordered list of rules
- Heuristics, overfitting, pruning

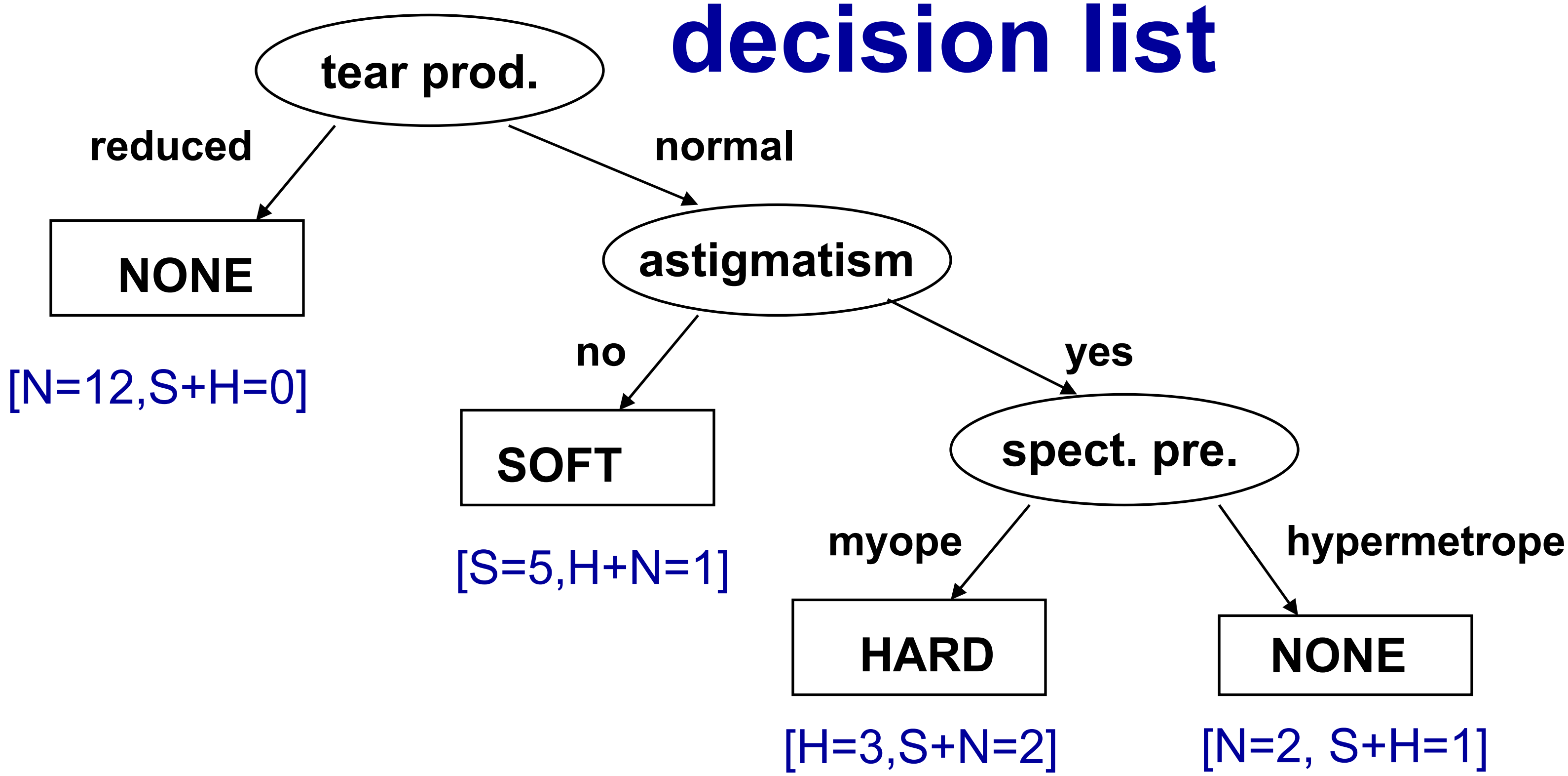
Contact lenses: convert decision tree to an unordered rule set



tear production=reduced **=>** lenses=NONE [S=0,H=0,N=12]
tear production=normal & astigmatism=yes & spect. pre.=hypermetrope **=>** lenses=NONE [S=0,H=1,N=2]
tear production=normal & astigmatism=no **=>** lenses=SOFT [S=5,H=0,N=1]
tear production=normal & astigmatism=yes & spect. pre.=myope **=>** lenses=HARD [S=0,H=3,N=2]
DEFAULT lenses=NONE

Order independent rule set (may overlap)

Contact lenses: convert decision tree to decision list



IF tear production=reduced THEN lenses=NONE
 ELSE /*tear production=normal*/
   IF astigmatism=no THEN lenses=SOFT
   ELSE /*astigmatism=yes*/
     IF spect. pre.=myope THEN lenses=HARD
      ELSE /* spect.pre.=hypermetrope*/
       lenses=NONE

Ordered (order dependent) rule list

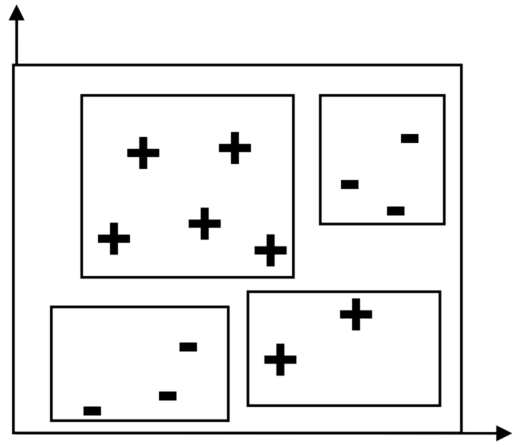
Converting decision tree to rules, and rule post-pruning (Quinlan 1993)

- Very frequently used method, e.g., in C4.5 and J48
- Procedure:
  - grow a full tree (allowing overfitting)
  - convert the tree to an equivalent set of rules
  - prune each rule independently of others
  - sort final rules into a desired sequence for use

Concept learning: Task reformulation for rule learning: (pos. vs. neg. examples of Target class)

Person	Age	Spect. presc.	Astigm.	Tear prod.	Lenses
O1	young	myope	no	reduced	NO
O2	young	myope	no	normal	YES
O3	young	myope	yes	reduced	NO
O4	young	myope	yes	normal	YES
O5	young	hypermetrope	no	reduced	NO
O6-O13	...	...	...	...	...
O14	pre-presbyo	hypermetrope	no	normal	YES
O15	pre-presbyo	hypermetrope	yes	reduced	NO
O16	pre-presbyo	hypermetrope	yes	normal	NO
O17	presbyopic	myope	no	reduced	NO
O18	presbyopic	myope	no	normal	NO
O19-O23	...	...	...	...	...
O24	presbyopic	hypermetrope	yes	normal	NO

Original covering algorithm (AQ, Michalski 1969,86)

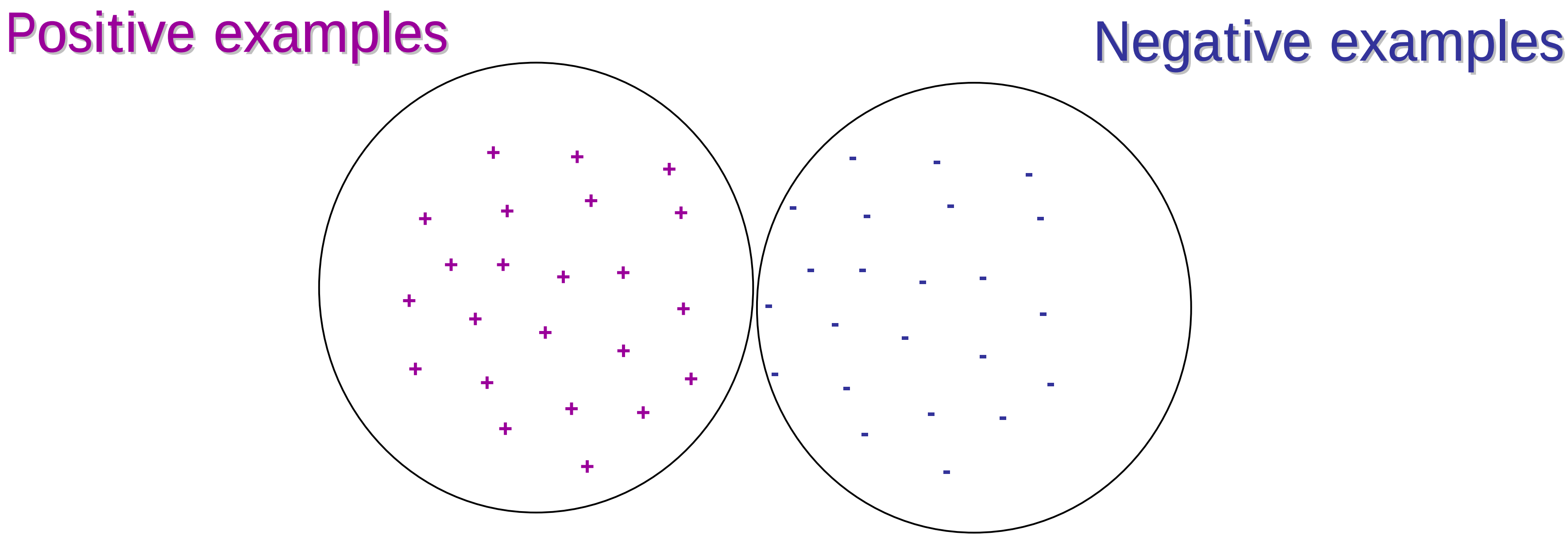


Given examples of N classes $C_1, \ldots, C_N$

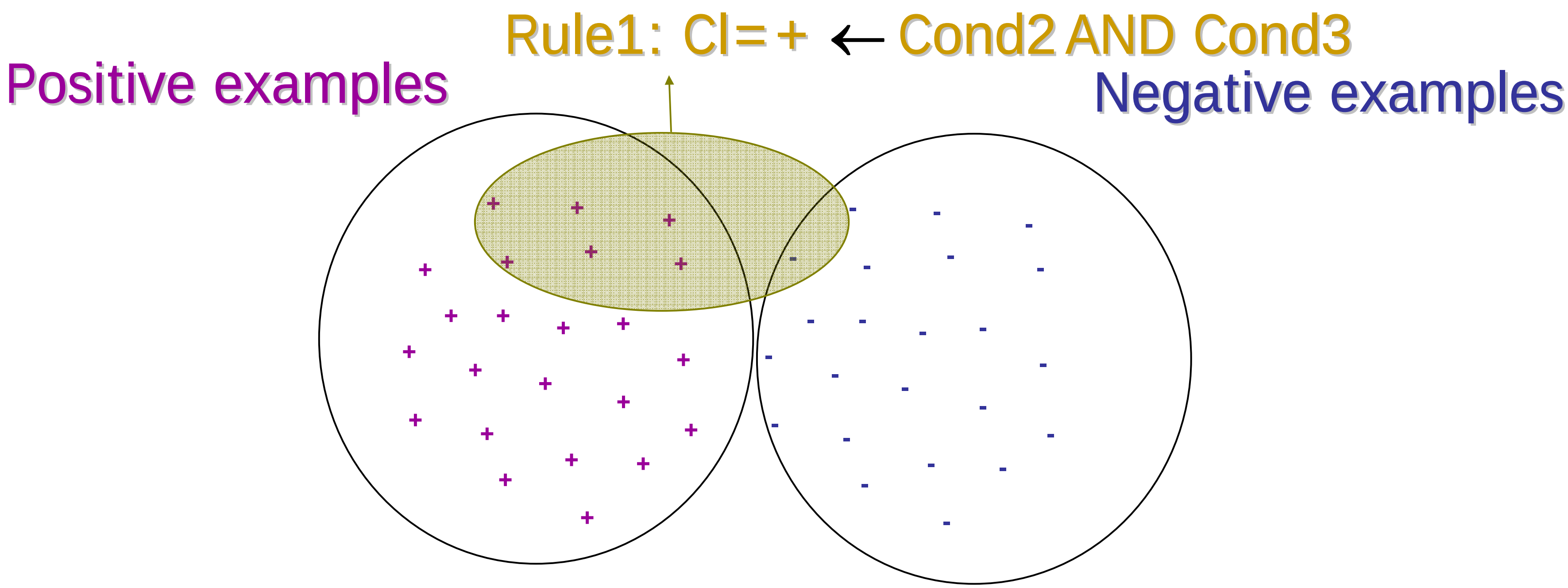
for each class Ci **do**

- Ei := Pi U Ni (Pi pos., Ni neg.)
- RuleBase(Ci) := empty
- **repeat {learn-set-of-rules}**
  - **learn-one-rule** R covering some positive examples and no negatives
  - add R to RuleBase(Ci)
  - delete from Pi all pos. ex. covered by R
- **until** Pi = empty

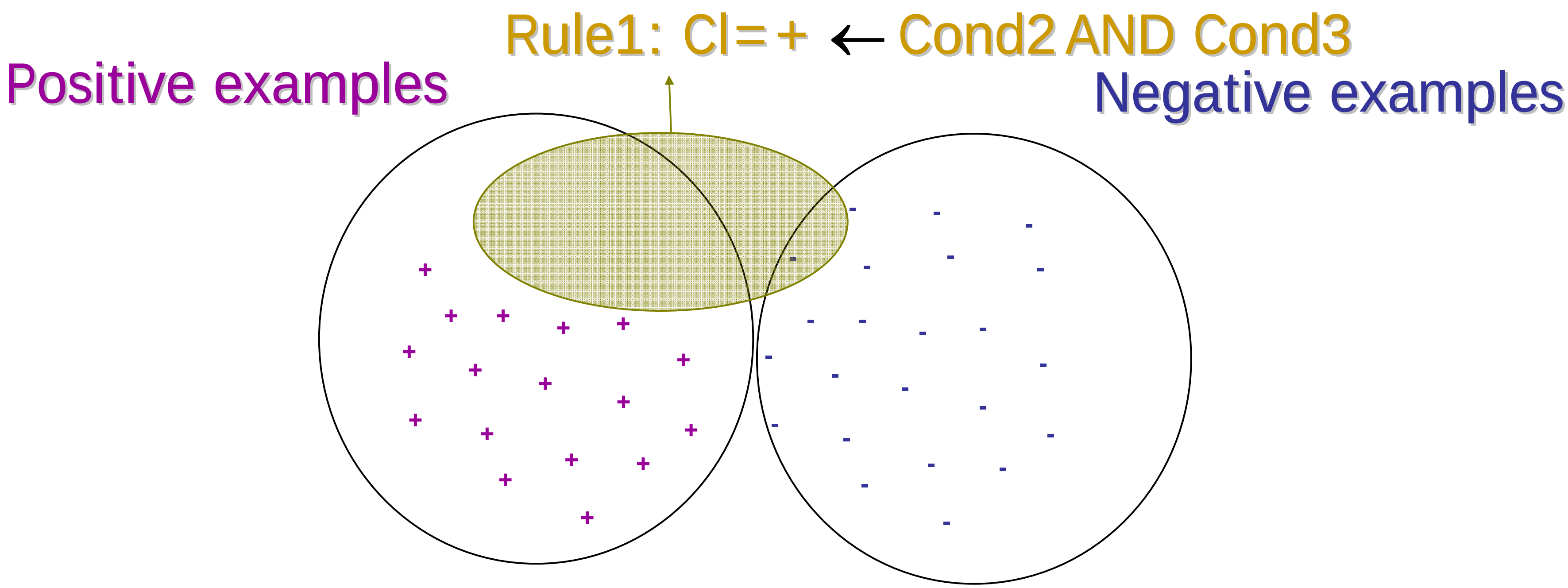
Covering algorithm



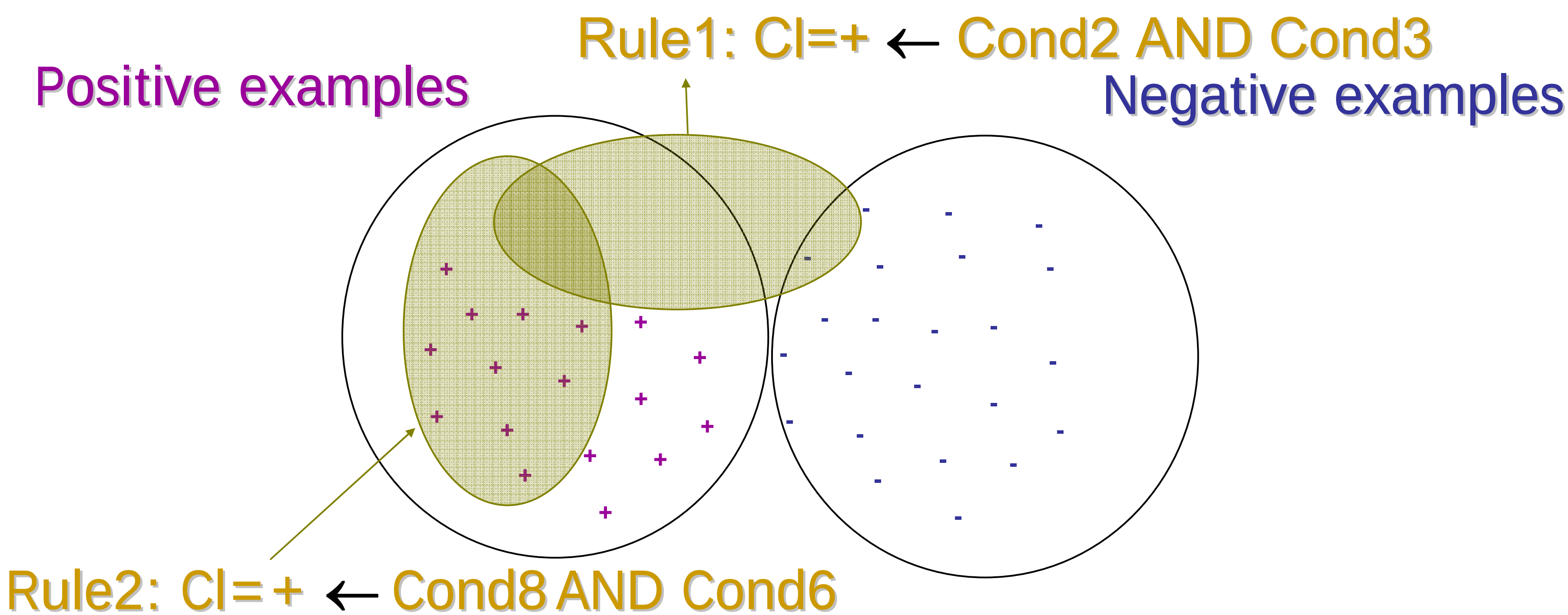
Covering algorithm



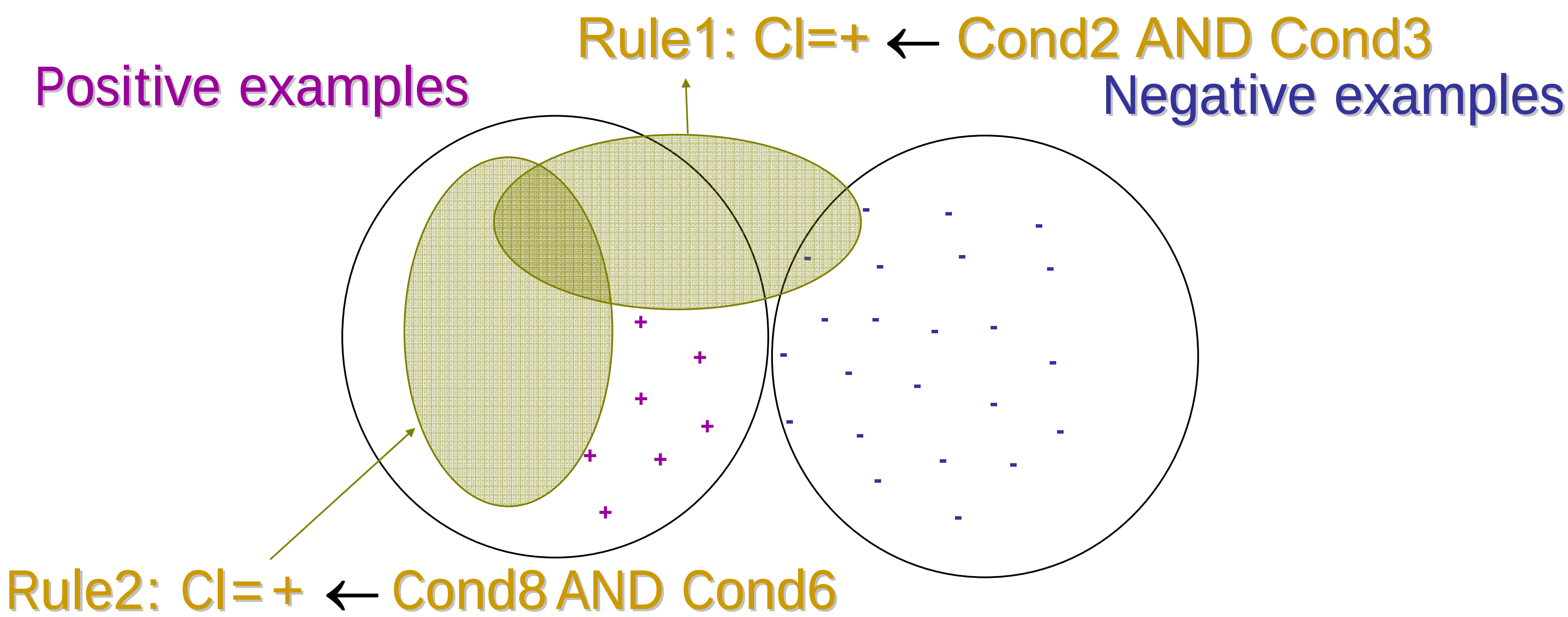
Covering algorithm



Covering algorithm



Covering algorithm



PlayTennis: Training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Weak	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Heuristics for learn-one-rule: PlayTennis example

PlayTennis = yes [9+,5-] (14)

PlayTennis = yes ← Wind=weak [6+,2-] (8)
← Wind=strong [3+,3-] (6)
← Humidity=normal [6+,1-] (7)
← …

PlayTennis = yes ← Humidity=normal
Outlook=sunny [2+,0-] (2)
← …

Estimating **rule accuracy (rule precision)** with the **probability** that a covered example is positive

A(Class ← Cond) = p(Class| Cond)

Estimating the **probability** with the **relative frequency** of covered pos. ex. / all covered ex.

[6+,1-] (7) = 6/7, [2+,0-] (2) = 2/2 = 1

Probability estimates

- **Relative frequency** :
  - problems with small samples

$$p(Class \mid Cond) =$$

$$= \frac{n(Class.Cond)}{n(Cond)}$$

[6+,1-] (7) = 6/7

[2+,0-] (2) = 2/2 = 1

- **Laplace estimate** :
  - assumes uniform prior distribution of k classes

$$= \frac{n(Class.Cond) + 1}{n(Cond) + k} \qquad k = 2$$

[6+,1-] (7) = 6+1 / 7+2 = 7/9

[2+,0-] (2) = 2+1 / 2+2 = 3/4

Learn-one-rule: search heuristics

- Assume a two-class problem
- Two classes (+,-), learn rules for + class (Cl).
- Search for specializations R' of a rule R = Cl $\leftarrow$ Cond from the RuleBase.
- Specializarion R' of rule R = Cl $\leftarrow$ Cond

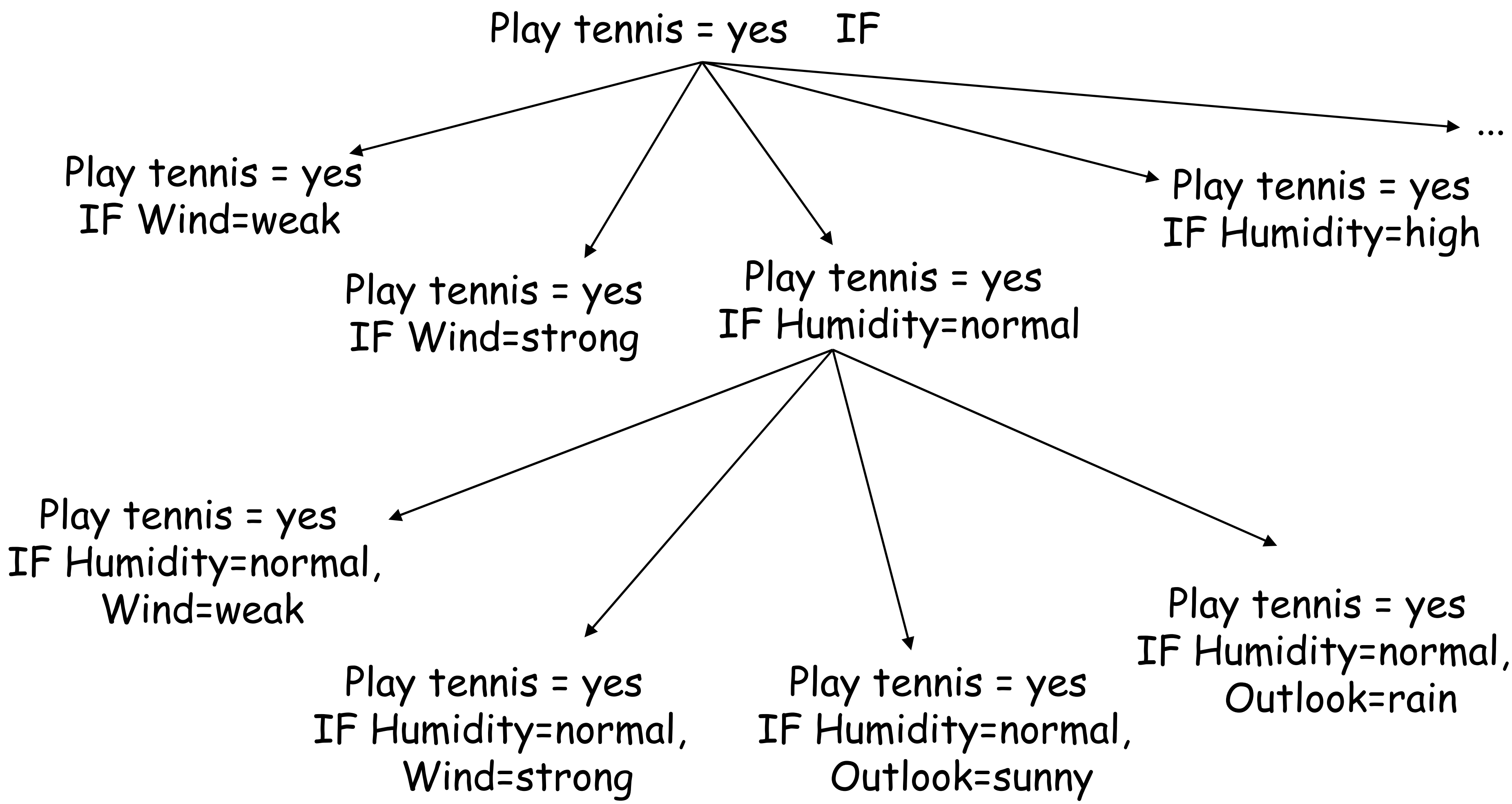
  has the form R' = Cl $\leftarrow$ Cond & Cond'
- Heuristic search for rules: find the 'best' Cond' to be added to the current rule R, such that rule accuracy is improved, e.g., such that Acc(R') > Acc(R)
  - where the expected **classification accuracy** can be estimated as A(R) = p(Cl|Cond)

Learn-one-rule: Greedy vs. beam search

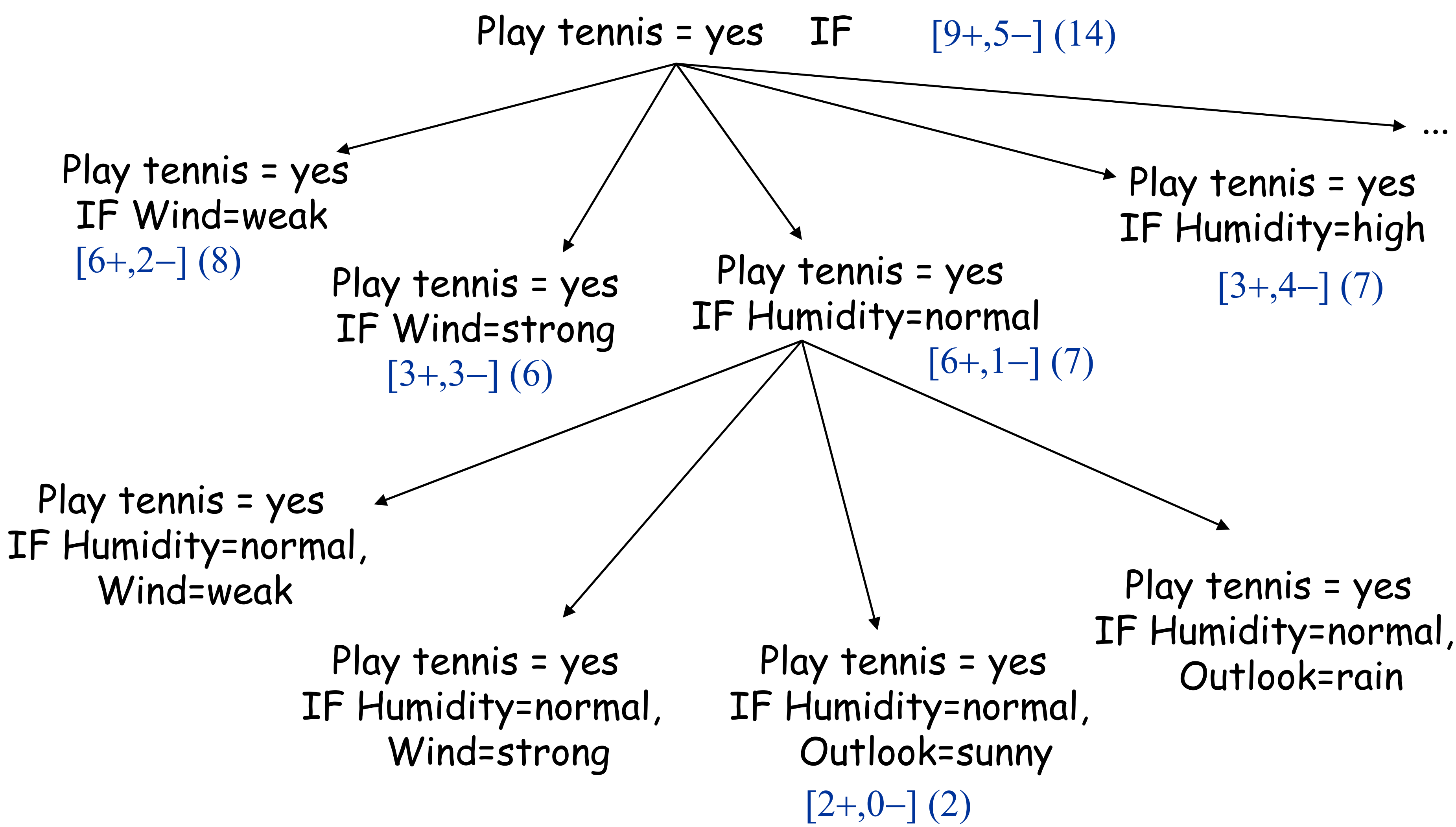
- learn-one-rule by greedy general-to-specific search, at each step selecting the `best' descendant, no backtracking
  - e.g., the best descendant of the initial rule

    PlayTennis = yes $\leftarrow$
  - is rule PlayTennis = yes $\leftarrow$ Humidity=normal
- beam search: maintain a list of k best candidates at each step; descendants (specializations) of each of these k candidates are generated, and the resulting set is again reduced to k best candidates

Learn-one-rule as search: PlayTennis example



Learn-one-rule as heuristic search: PlayTennis example



What is “high” rule accuracy (rule precision) ?

- Rule evaluation measures:
  - aimed at maximizing classification accuracy
  - minimizing Error = 1 - Accuracy
  - avoiding overfitting
- BUT: Rule accuracy/precision should be traded off against the “default” accuracy/precision of the rule **Cl ←true**
  - 68% accuracy is OK if there are 20% examples of that class in the training set, but bad if there are 80%
- **Relative accuracy**
  - $RAcc(Cl \leftarrow Cond) = p(Cl \mid Cond) - p(Cl)$

Weighted relative accuracy

- If a rule covers a single example, its accuracy/precision is either 0% or 100%
  - maximising relative accuracy tends to produce many overly specific rules

- **Weighted relative accuracy**

  WRAcc(Cl←Cond) = p(Cond) . [p(Cl | Cond) – p(Cl)]

- WRAcc is a fundamental rule evaluation measure:
  - WRAcc can be used if you want to assess both accuracy and significance
  - WRAcc can be used if you want to compare rules with different heads **and** bodies

Learn-one-rule: search heuristics

- Assume two classes (+,-), learn rules for + class (Cl). Search for specializations of one rule $R = Cl \leftarrow Cond$ from RuleBase.
- Expected **classification accuracy**: $A(R) = p(Cl|Cond)$
- **Informativity** (info needed to specify that example covered by Cond belongs to Cl): $I(R) = -\log_2 p(Cl|Cond)$
- **Accuracy gain** (increase in expected accuracy):

  $AG(R',R) = p(Cl|Cond') - p(Cl|Cond)$
- **Information gain** (decrease in the information needed):

  $IG(R',R) = \log_2 p(Cl|Cond') - \log_2 p(Cl|Cond)$
- **Weighted** measures favoring more general rules: WAG, WIG

  $WAG(R',R) =$

  $p(Cond')/p(Cond) \,.\, (p(Cl|Cond') - p(Cl|Cond))$
- **Weighted relative accuracy** trades off coverage and relative accuracy $WRAcc(R) = p(Cond).(p(Cl|Cond) - p(Cl))$

Ordered set of rules: if-then-else rules

- rule Class IF Conditions is learned by first determining Conditions and then Class
- **Notice:** mixed sequence of classes C1, …, Cn in RuleBase
- **But: ordered** execution when classifying a new instance: rules are sequentially tried and the first rule that `fires' (covers the example) is used for classification
- **Decision list {R1, R2, R3, …, D}:** rules Ri are interpreted as **if-then-else** rules
- If no rule fires, then DefaultClass (majority class in E_{cur})

Sequential covering algorithm (similar as in Mitchell's book)

- RuleBase := empty
- E_{cur}:= E
- **repeat**
  - learn-one-rule R
  - RuleBase := RuleBase U R
  - E_{cur} := E_{cur} - {examples covered and correctly classified by R} **(DELETE ONLY POS. EX.!)**
  - **until** performance(R, E_{cur}) < ThresholdR
- RuleBase := sort RuleBase by performance(R,E)
- return RuleBase

Learn ordered set of rules (CN2, Clark and Niblett 1989)

- RuleBase := empty
- E_{cur}:= E
- **repeat**
  - learn-one-rule R
  - RuleBase := RuleBase U R
  - E_{cur} := E_{cur} - {all examples covered by R} **(NOT ONLY POS. EX.!)**
- **until** performance(R, E_{cur}) < ThresholdR
- RuleBase := sort RuleBase by performance(R,E)
- RuleBase := RuleBase U DefaultRule(E_{cur})

Learn-one-rule: Beam search in CN2

- Beam search in CN2 learn-one-rule algo.:
  - construct BeamSize of best rule bodies (conjunctive conditions) that are statistically significant
  - BestBody - min. entropy of examples covered by Body
  - construct best rule R := Head $\leftarrow$ BestBody by adding majority class of examples covered by BestBody in rule Head
- performance (R, E_{cur}) : - Entropy(E_{cur})
  - performance(R, E_{cur}) < ThresholdR (neg. num.)
  - Why? Ent. > t is bad, Perf. = -Ent < -t is bad

Variations

- Sequential vs. simultaneous covering of data (as in TDIDT): choosing between attribute-values vs. choosing attributes
- Learning rules vs. learning decision trees and converting them to rules
- Pre-pruning vs. post-pruning of rules
- What statistical evaluation functions to use
- Probabilistic classification

Probabilistic classification

- In the ordered case of standard CN2 rules are interpreted in an `IF-THEN-ELSE` fashion, and the first fired rule assigns the class.
- In the unordered case all rules are tried and all rules which fire are collected. If a clash occurs, a probabilistic method is used to resolve the clash.
- A simplified example:
  1. tear production=reduced **=>** lenses=NONE [S=0,H=0,N=12]
  2. tear production=normal & astigmatism=yes & spect. pre.=hypermetrope **=>** lenses=NONE [S=0,H=1,N=2]
  3. tear production=normal & astigmatism=no **=>** lenses=SOFT [S=5,H=0,N=1]
  4. tear production=normal & astigmatism=yes & spect. pre.=myope **=>** lenses=HARD [S=0,H=3,N=2]
  5. DEFAULT lenses=NONE

Suppose we want to classify a person with normal tear production and astigmatism. Two rules fire: rule 2 with coverage [S=0,H=1,N=2] and rule 4 with coverage [S=0,H=3,N=2]. The classifier computes total coverage as [S=0,H=4,N=4], resulting in probabilistic classification into class H with probability 0.5 and N with probability 0.5. In this case, the clash can not be resolved, as both probabilities are equal.

Part II. Predictive DM techniques

- Naïve Bayesian classifier
- Decision tree learning
- Classification rule learning
- Classifier evaluation

Classifier evaluation

- Accuracy and Error
- n-fold cross-validation
- Confusion matrix
- ROC

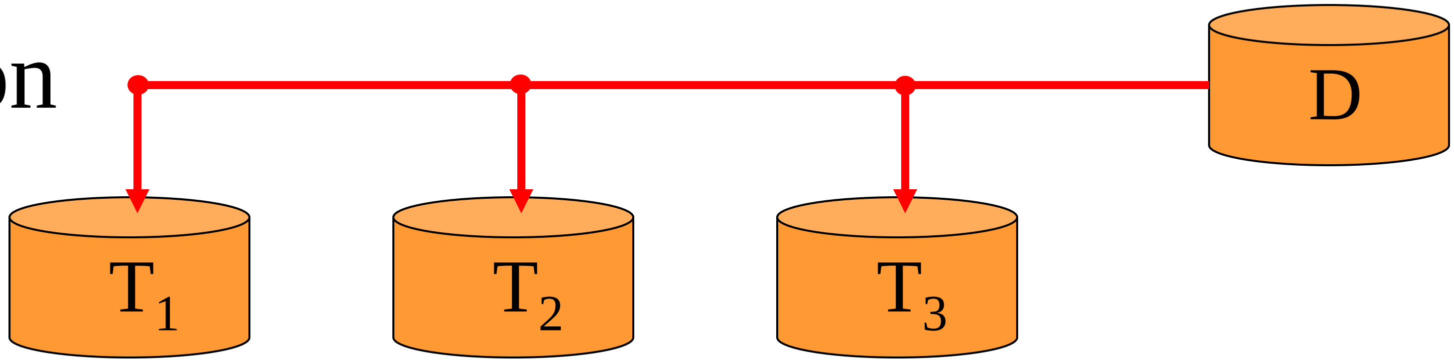
Evaluating hypotheses

- **Use of induced hypotheses**
  - discovery of new patterns, new knowledge
  - classification of new objects
- **Evaluating the quality of induced hypotheses**
  - Accuracy, Error = 1 - Accuracy
  - classification accuracy on testing examples = percentage of correctly classified instances
    - split the example set into training set (e.g. 70%) to induce a concept, and test set (e.g. 30%) to test its accuracy
    - more elaborate strategies: 10-fold cross validation, leave-one-out, ...
  - comprehensibility (compactness)
  - information contents (information score), significance

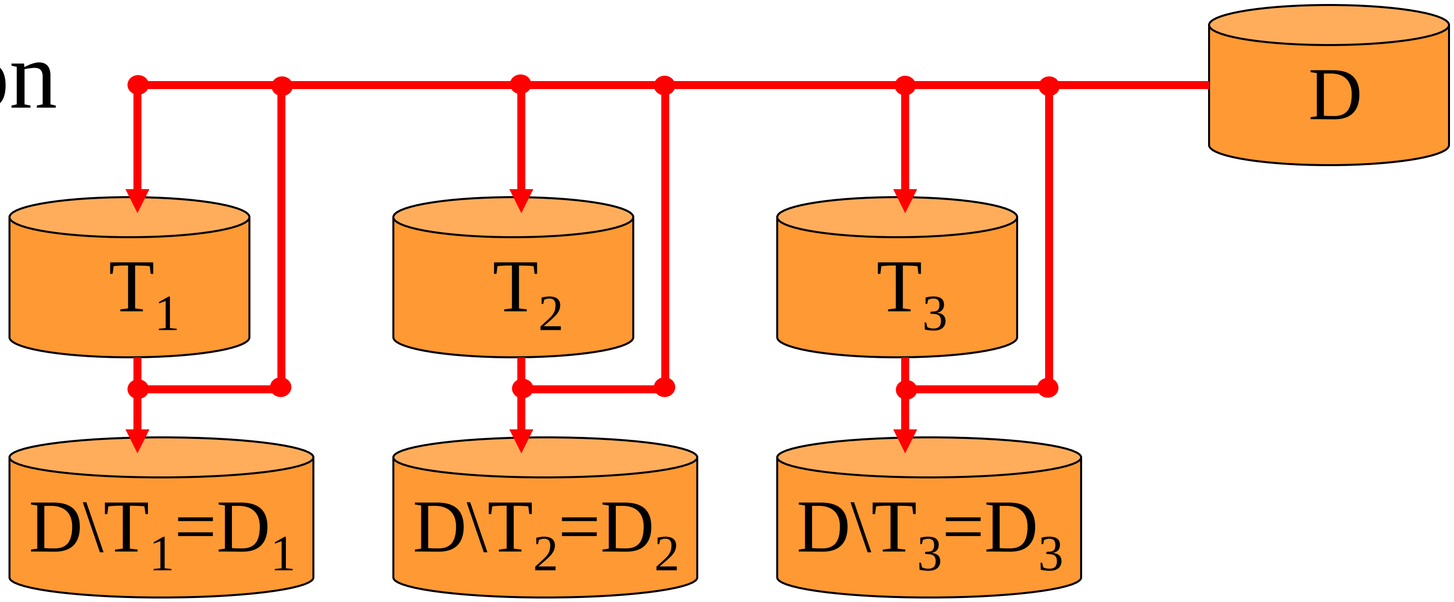
n-fold cross validation

- A method for accuracy estimation of classifiers
- Partition set D into n disjoint, almost equally-sized folds T_i where $U_i\, T_i = D$
- **for** i = 1, ..., n **do**
  - form a training set out of n-1 folds: $Di = D \backslash T_i$
  - induce classifier H_i from examples in Di
  - use fold T_i for testing the accuracy of H_i
- Estimate the accuracy of the classifier by averaging accuracies over 10 folds T_i

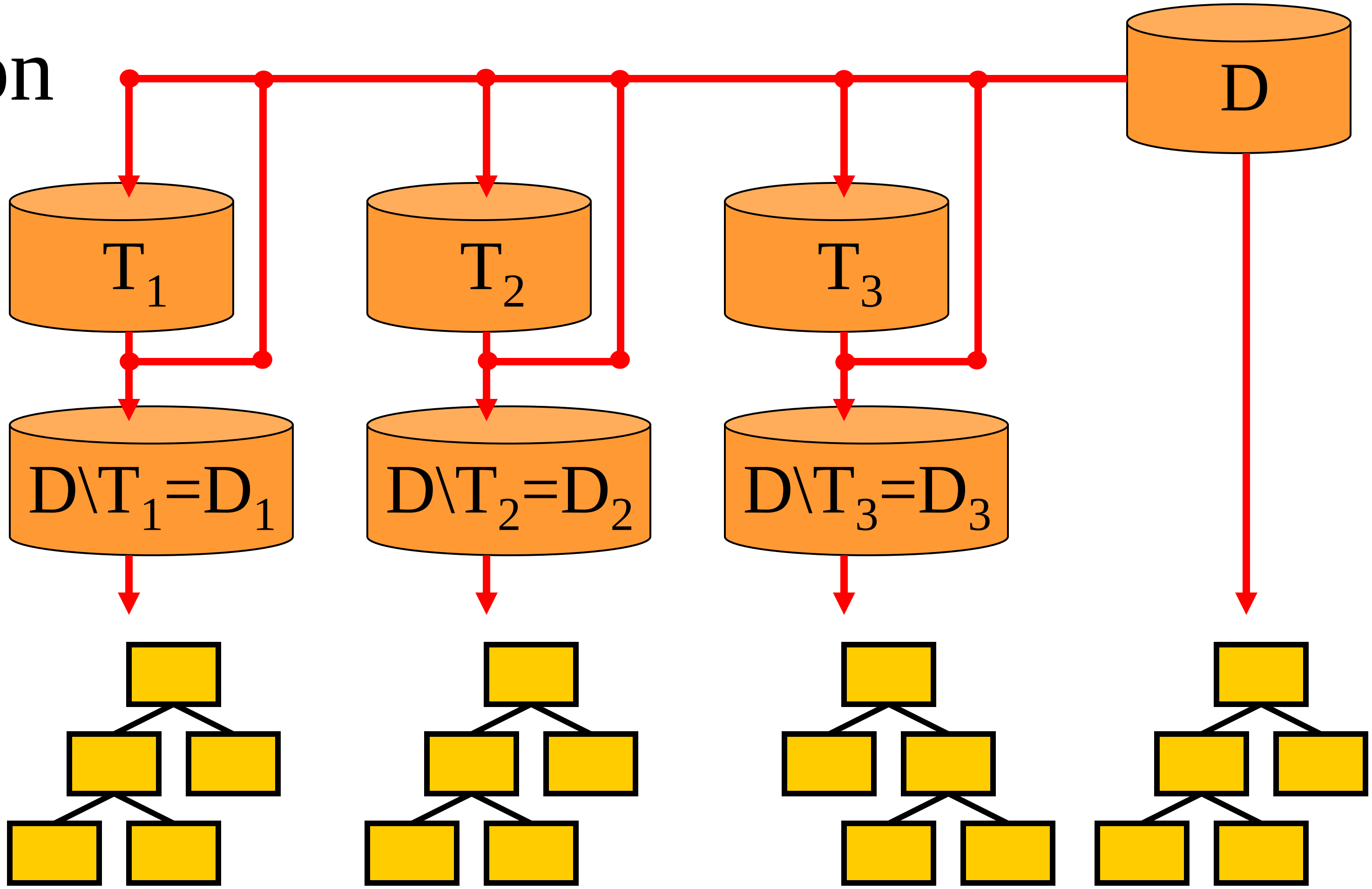
- Partition



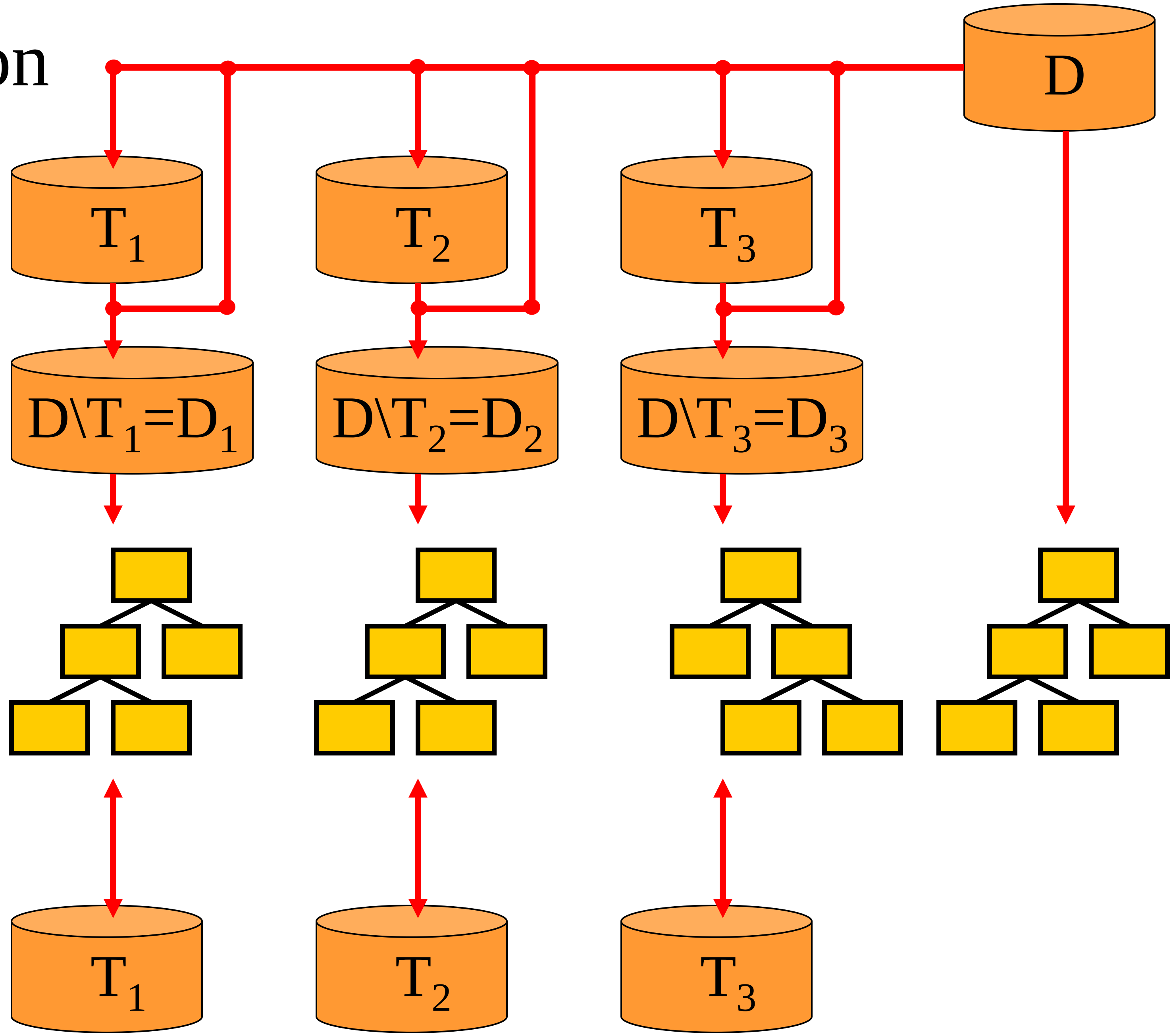
- Partition
- Train



- Partition
- Train



- Partition
- Train
- Test



Confusion matrix and rule (in)accuracy

- Accuracy of a classifier is measured as TP+TN / N.
- Suppose two rules are both 80% accurate on an evaluation dataset, are they always equally good?
  - e.g., Rule 1 correctly classifies 40 out of 50 positives and 40 out of 50 negatives; Rule 2 correctly classifies 30 out of 50 positives and 50 out of 50 negatives
  - on a test set which has more negatives than positives, Rule 2 is preferable;
  - on a test set which has more positives than negatives, Rule 1 is preferable; unless…
  - …the proportion of positives becomes so high that the ‘always positive’ predictor becomes superior!
- Conclusion: classification accuracy is not always an appropriate rule quality measure

Confusion matrix

	Predicted positive	Predicted negative	
Positive examples	**True positives**	**False negatives**	
Negative examples	**False positives**	**True negatives**	

- also called ***contingency table***

Classifier 1

	Predicted positive	Predicted negative	
Positive examples	**40**	**10**	50
Negative examples	**10**	**40**	50
	50	50	100

Classifier 2

	Predicted positive	Predicted negative	
Positive examples	**30**	**20**	50
Negative examples	**0**	**50**	50
	30	70	100

ROC space

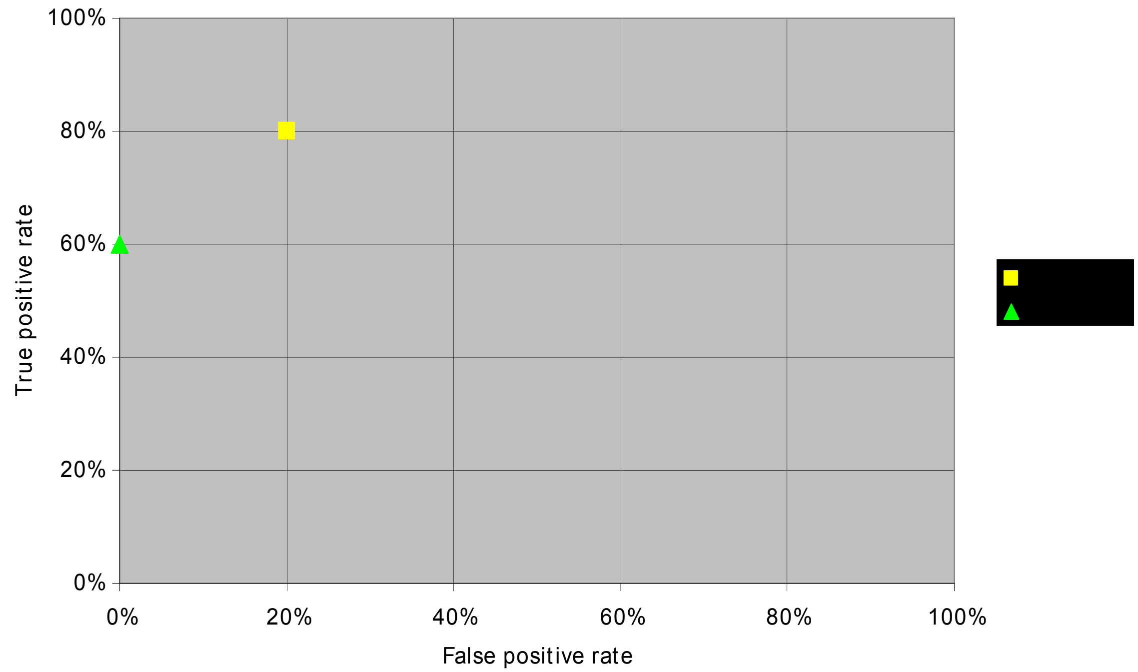
- ***True positive rate*** = #true pos. / #pos.
  - TPr_1 = 40/50 = 80%
  - TPr_2 = 30/50 = 60%
- ***False positive rate*** = #false pos. / #neg.
  - FPr_1 = 10/50 = 20%
  - FPr_2 = 0/50 = 0%
- ***ROC space*** has
  - FPr on X axis
  - TPr on Y axis

Classifier 1

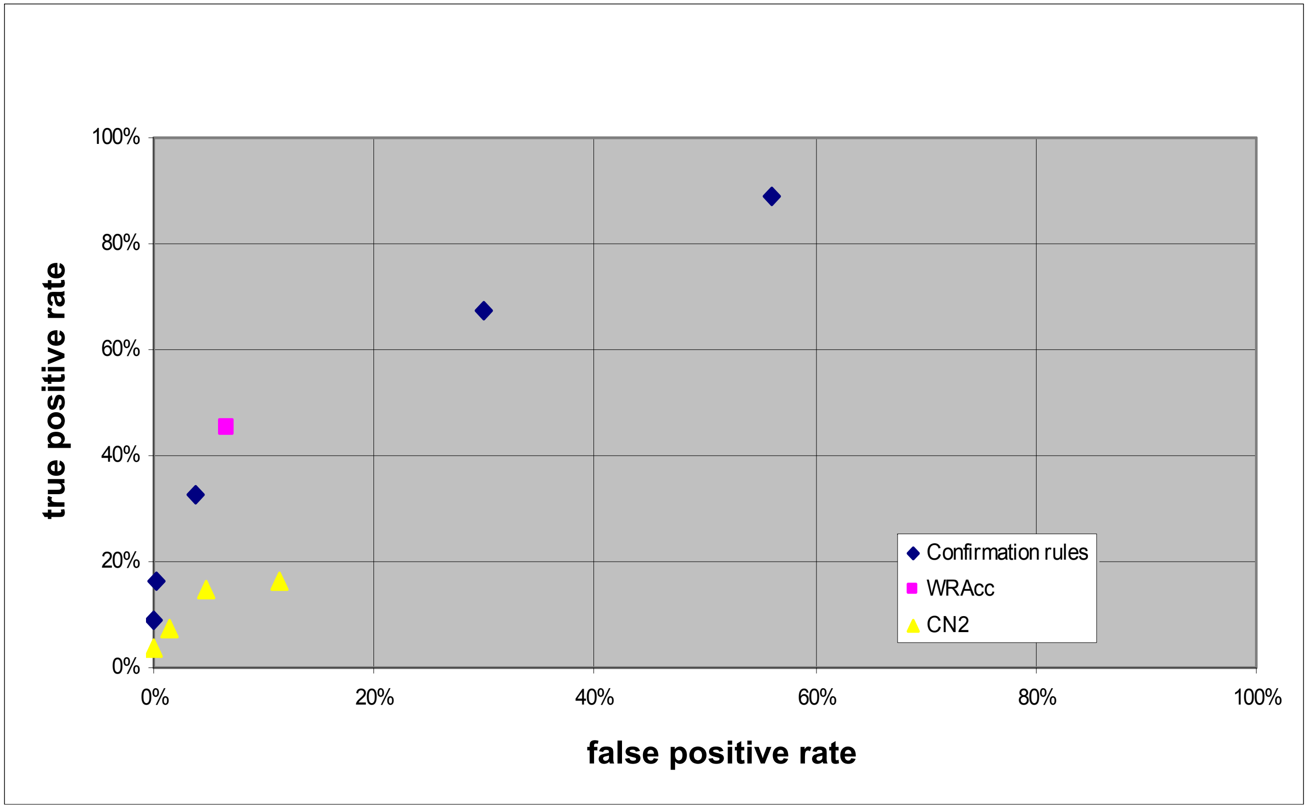
	Predicted positive	Predicted negative	
Positive examples	**40**	**10**	50
Negative examples	**10**	**40**	50
	50	50	100

Classifier 2

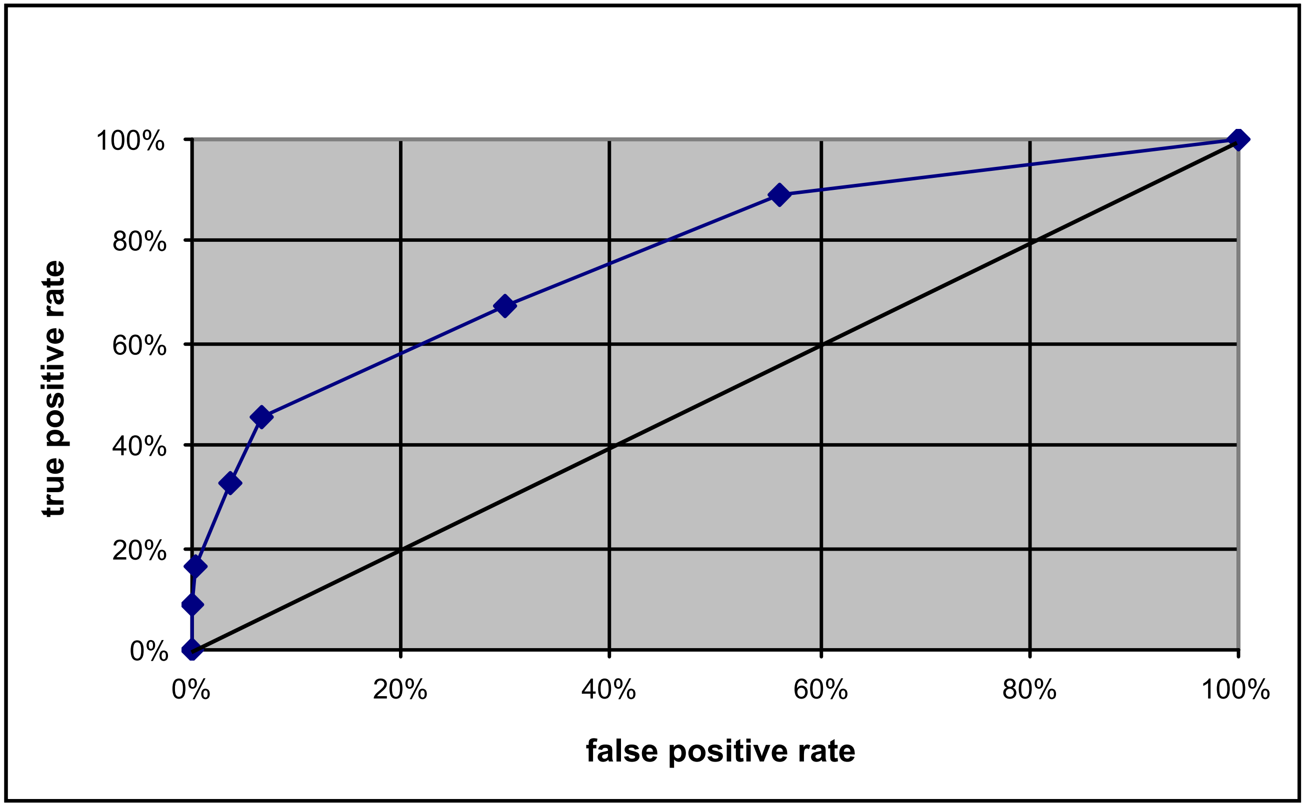
	Predicted positive	Predicted negative	
Positive examples	**30**	**20**	50
Negative examples	**0**	**50**	50
	30	70	100



The ROC space



The ROC convex hull



Summary of evaluation

- 10-fold cross-validation is a standard classifier evaluation method used in machine learning
- ROC analysis is very natural for rule learning and subgroup discovery
  - can take costs into account
  - here used for evaluation
  - also possible to use as search heuristic

Part III. Numeric prediction

- Baseline
- Linear Regression
- Regression tree
- Model Tree
- kNN

Regression	Classification
Data: attribute-value description	
Target variable: Continuous	**Target variable**: Categorical (nominal)
Evaluation: cross validation, separate test set, …	
Error: MSE, MAE, RMSE, …	**Error**: 1-accuracy
Algorithms: Linear regression, regression trees,…	**Algorithms**: Decision trees, Naïve Bayes, …
Baseline predictor: Mean of the target variable	**Baseline predictor**: Majority class

Example

- data about 80 people: Age and Height

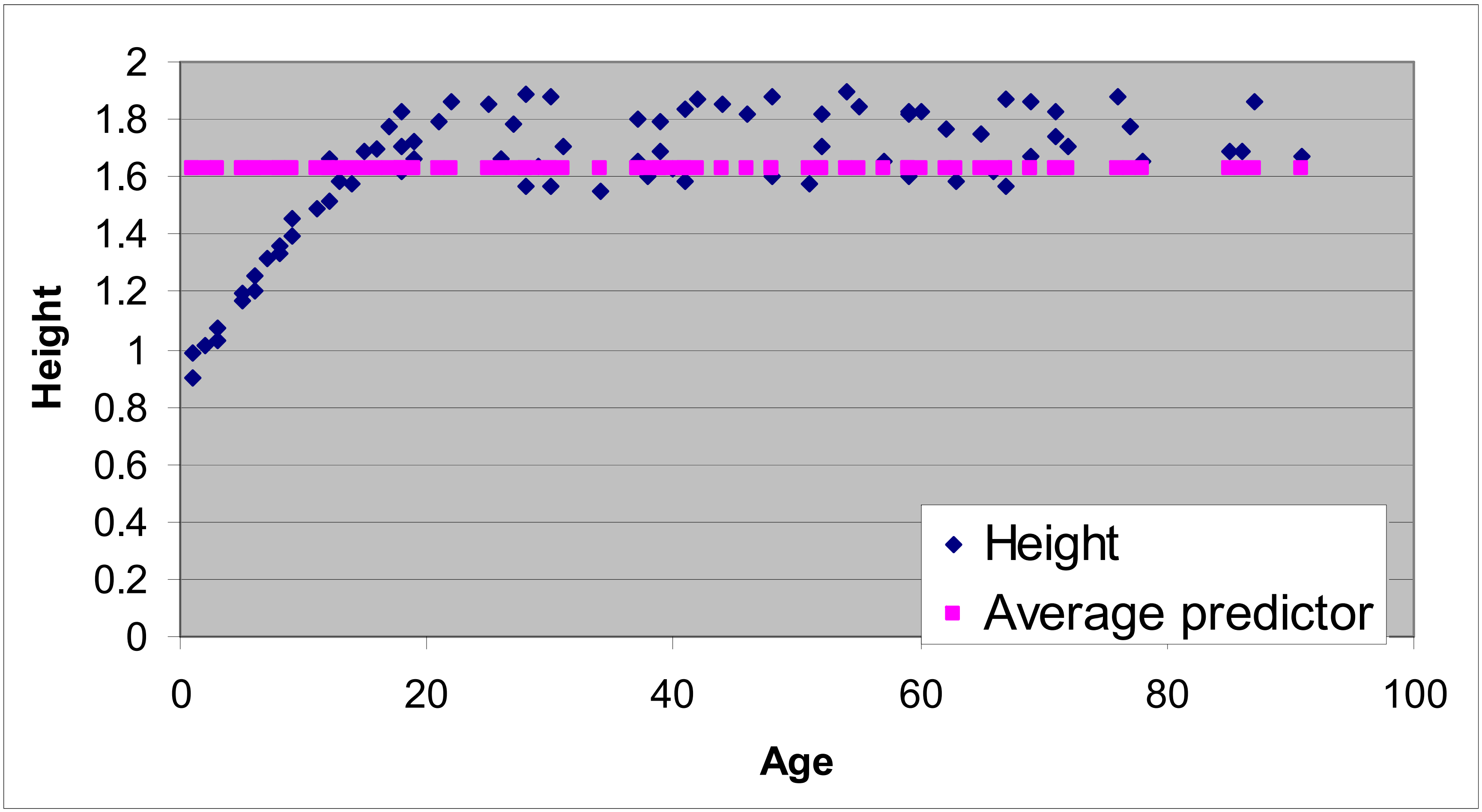
Age	Height
3	1.03
5	1.19
6	1.26
9	1.39
15	1.69
19	1.67
22	1.86
25	1.85
41	1.59
48	1.60
54	1.90
71	1.82
...	...

Test set

Age	Height
2	0.85
10	1.4
35	1.7
70	1.6

Baseline numeric predictor

- Average of the target variable



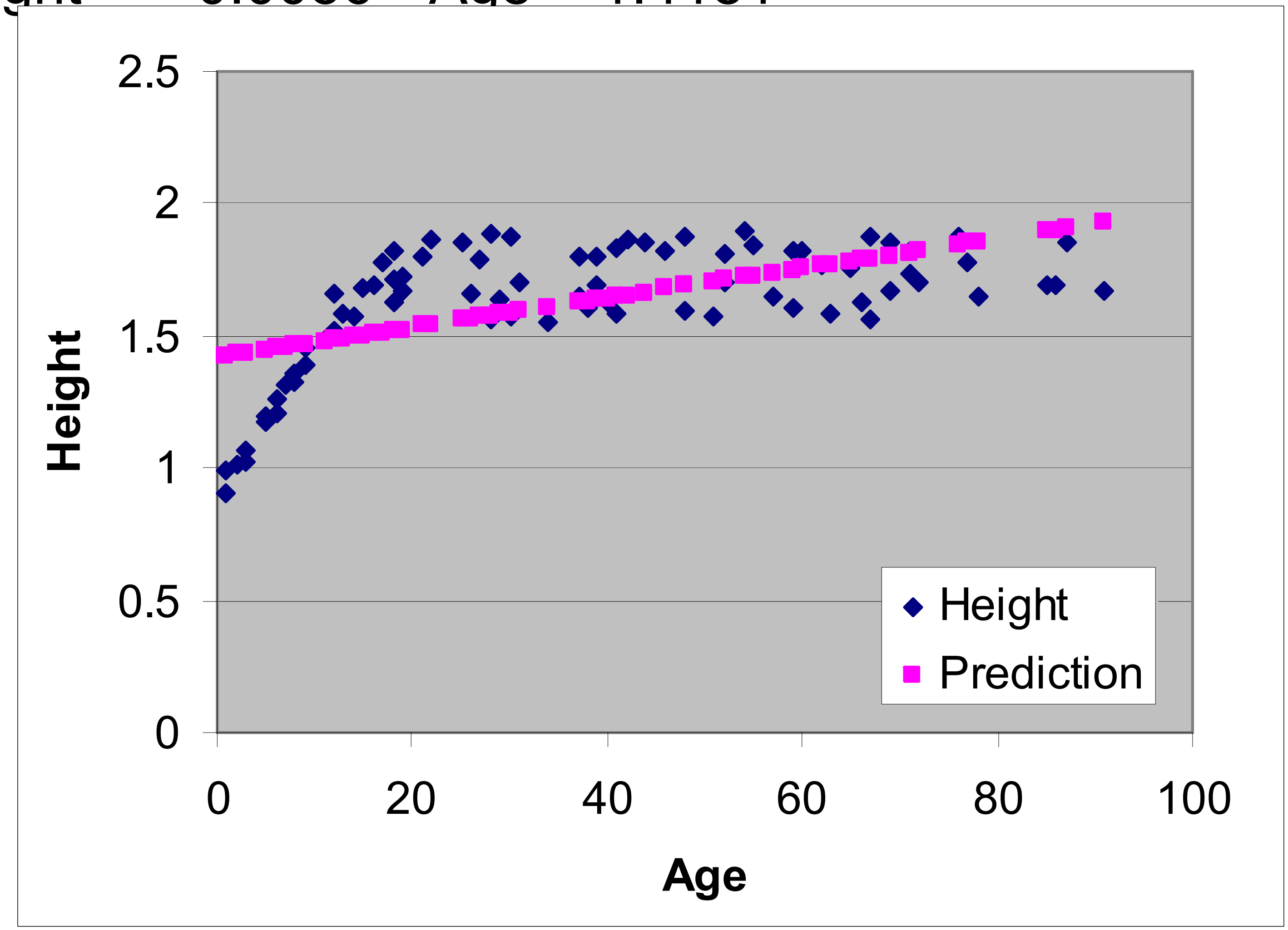
Baseline predictor: prediction

Average of the target variable is 1.63

Age	Height	Baseline
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Linear Regression Model

Height = 0.0056 * Age + 1.4181

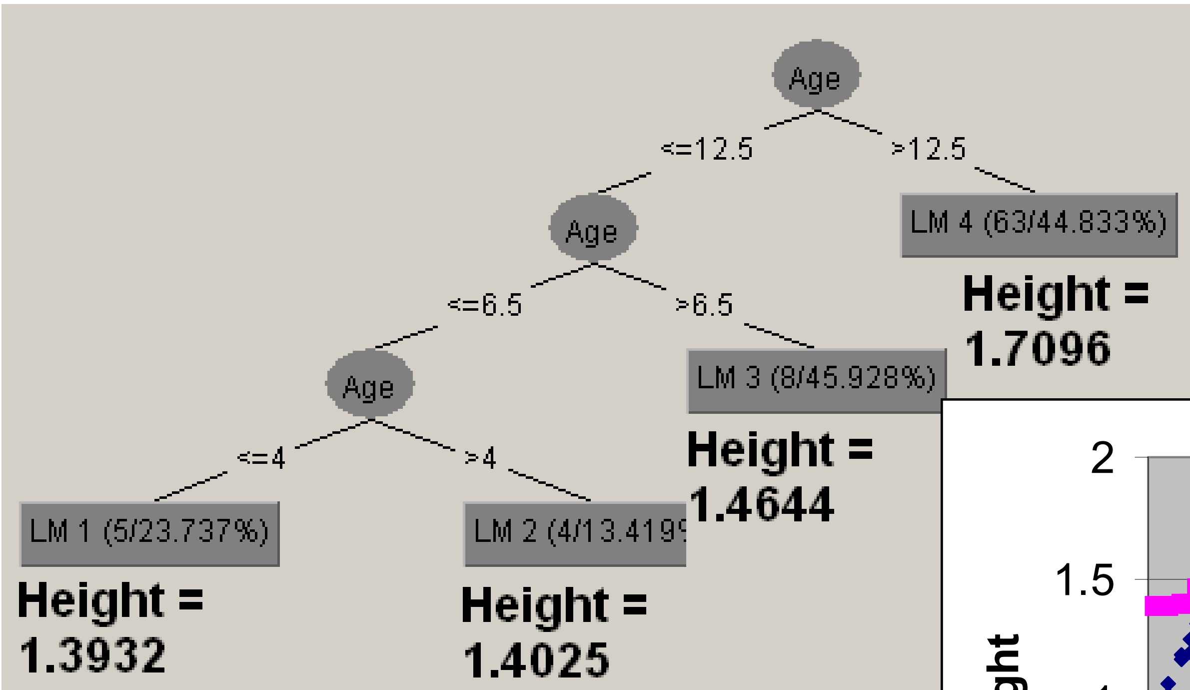


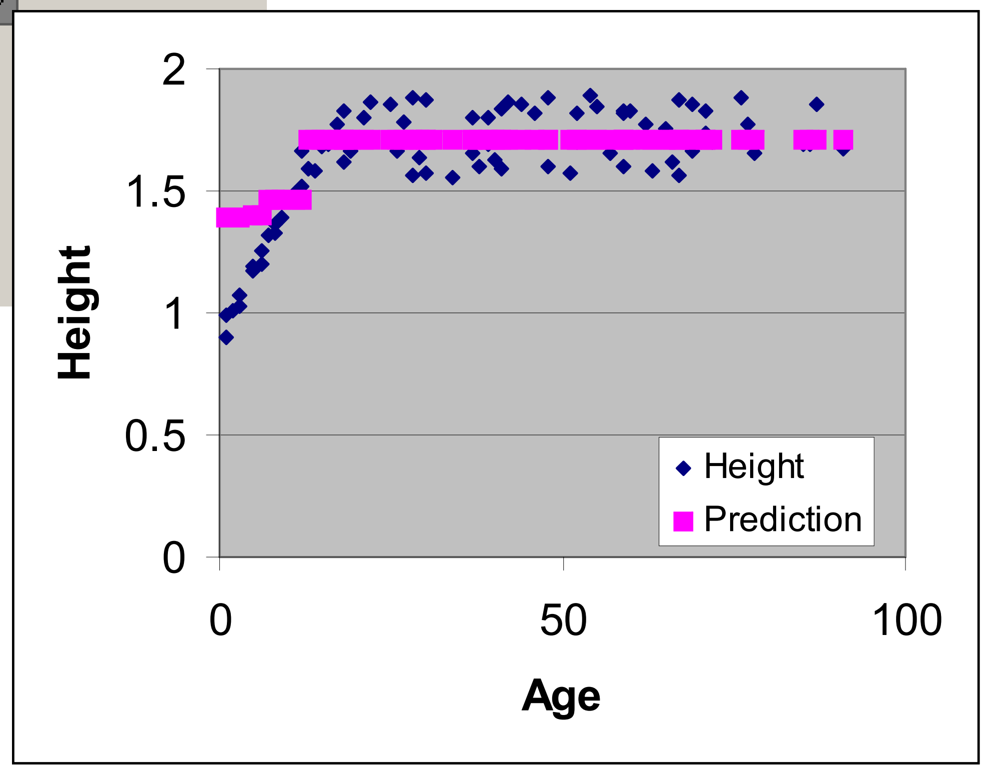
Linear Regression: prediction

Height = 0.0056 * Age + 1.4181

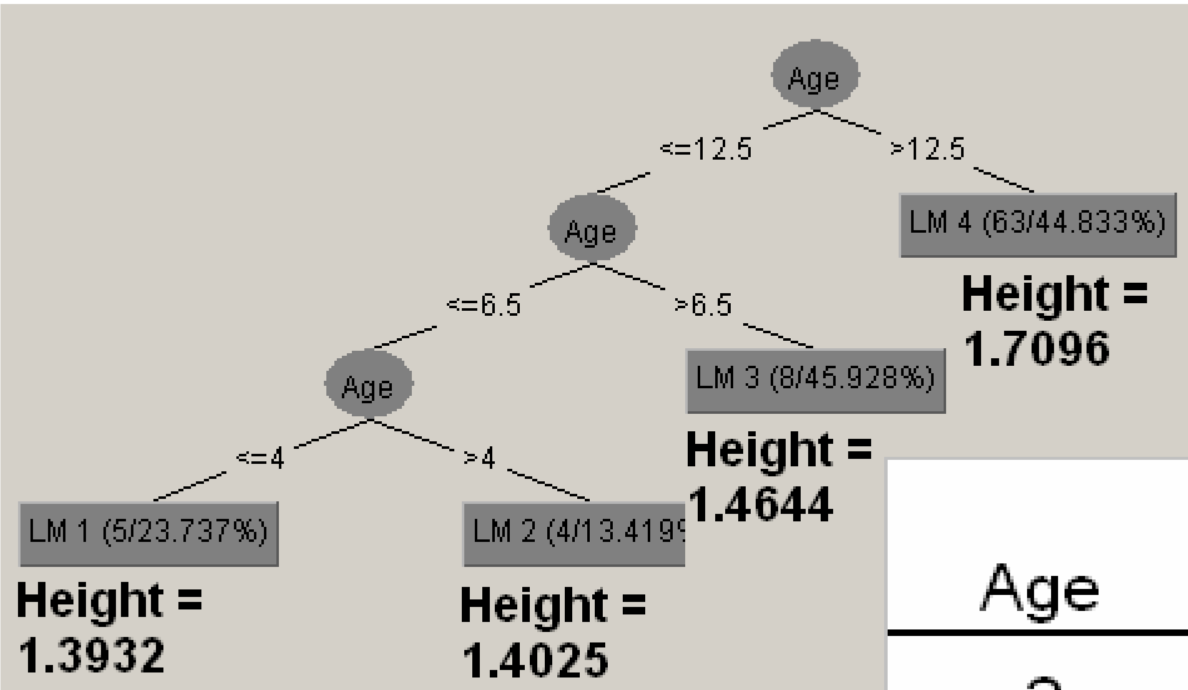
Age	Height	Linear regression
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Regression tree



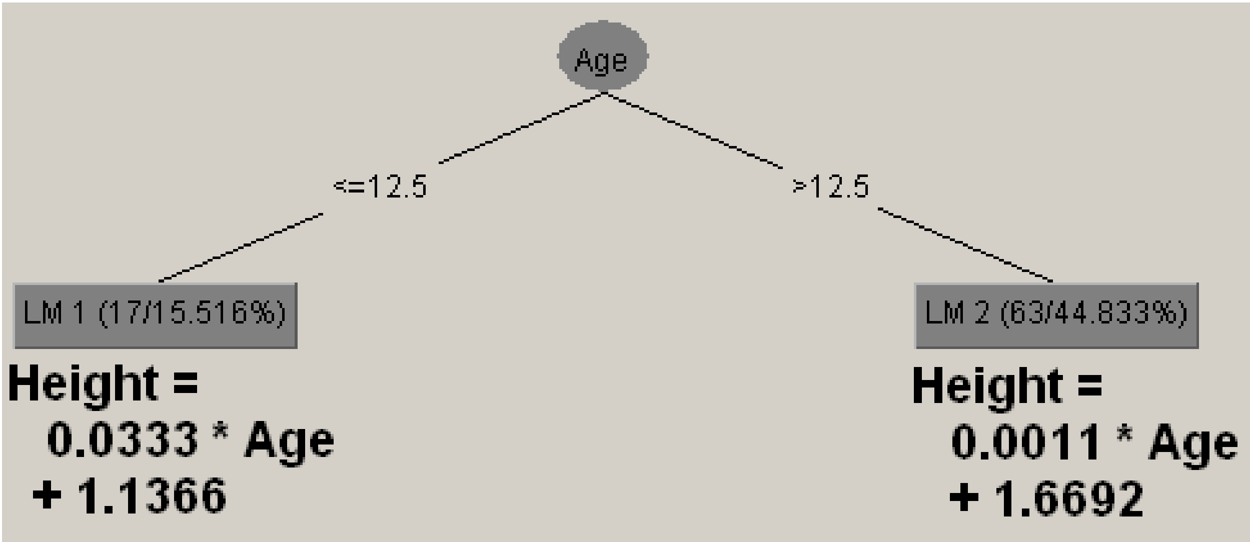


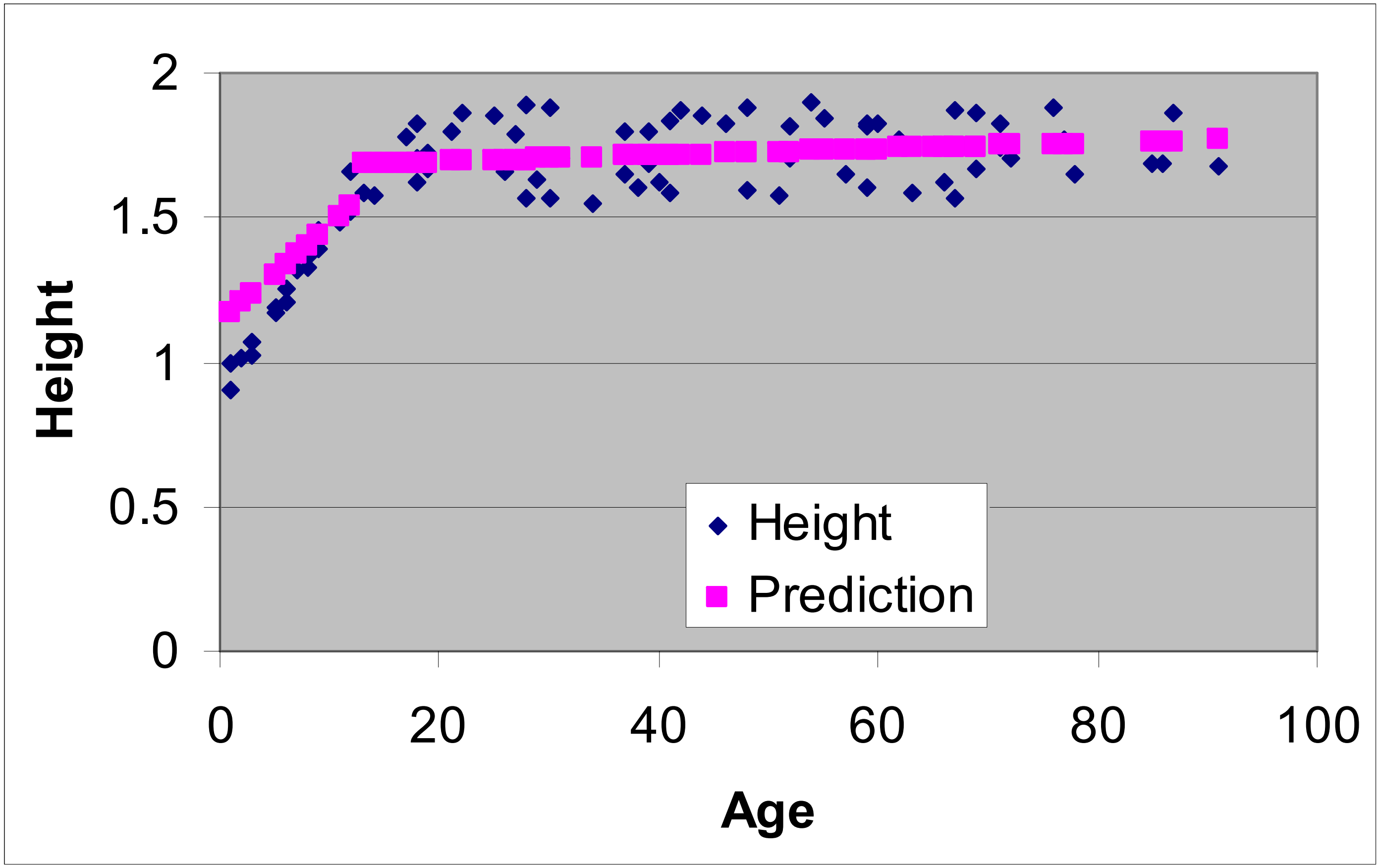
Regression tree: prediction



Age	Height	Regression tree
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Model tree


Age
<=12.5
>12.5
LM 1 (17/15.516%)
LM 2 (63/44.833%)
Height = 0.0333 * Age + 1.1366
Height = 0.0011 * Age + 1.6692

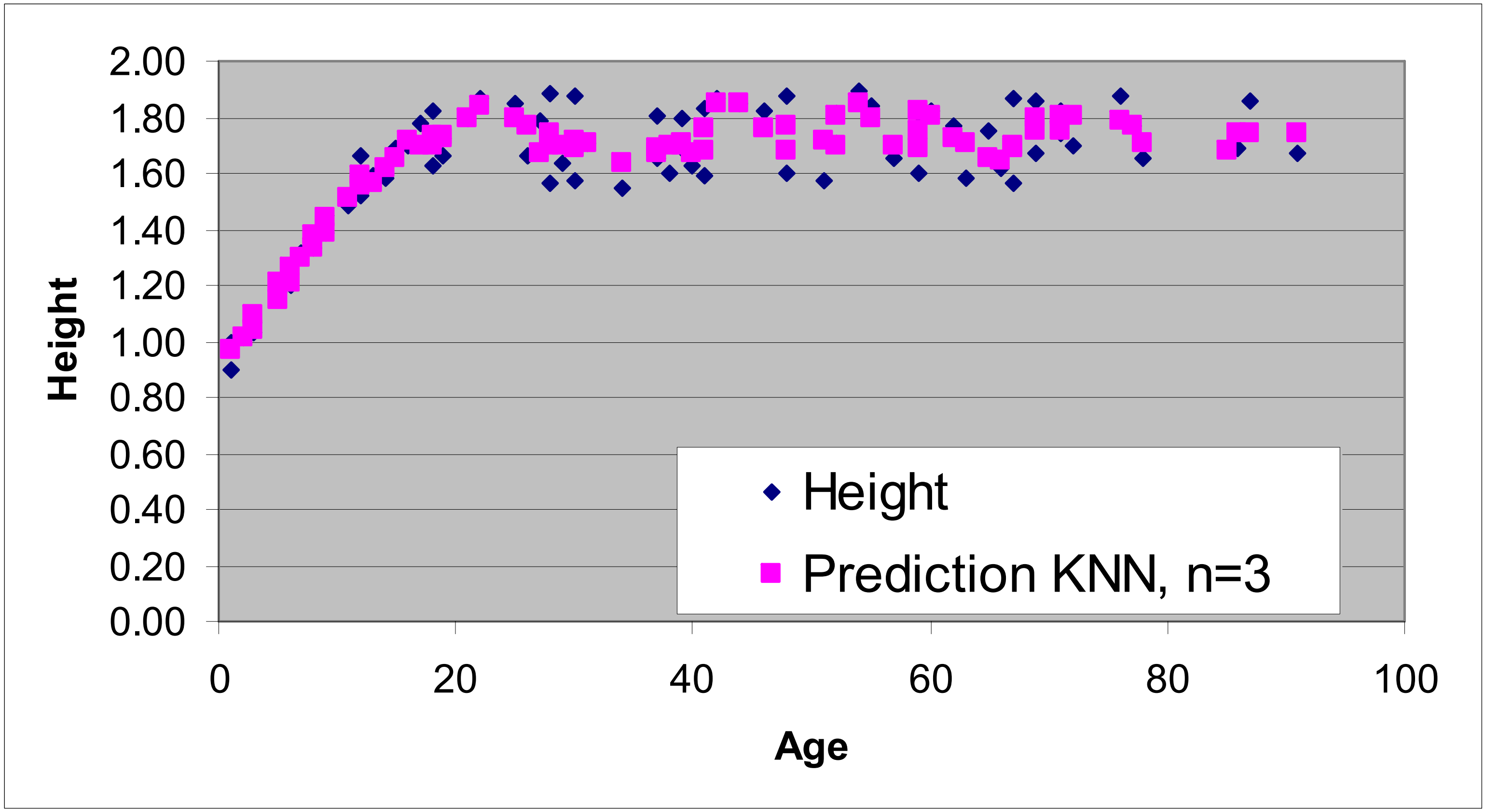

Height
Age
0
0.5
1
1.5
2
0
20
40
60
80
100
Height
Prediction

Model tree: prediction

Age	Height	Model tree
2	0.85	
10	1.4	
35	1.7	
70	1.6	

kNN – K nearest neighbors

- Looks at K closest examples (by age) and predicts the average of their target variable
- K=3



kNN prediction

Age	Height
1	0.90
1	0.99
2	1.01
3	1.03
3	1.07
5	1.19
5	1.17

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

kNN prediction

Age	Height
8	1.36
8	1.33
9	1.45
9	1.39
11	1.49
12	1.66
12	1.52
13	1.59
14	1.58

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

kNN prediction

Age	Height
30	1.57
30	1.88
31	1.71
34	1.55
37	1.65
37	1.80
38	1.60
39	1.69
39	1.80

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

kNN prediction

Age	Height
67	1.56
67	1.87
69	1.67
69	1.86
71	1.74
71	1.82
72	1.70
76	1.88

Age	Height	kNN
2	0.85	
10	1.4	
35	1.7	
70	1.6	

Which predictor is the best?

Age	Height	Baseline	Linear regression	Regression tree	Model tree	kNN
2	0.85	1.63	1.43	1.39	1.20	1.01
10	1.4	1.63	1.47	1.46	1.47	1.51
35	1.7	1.63	1.61	1.71	1.71	1.67
70	1.6	1.63	1.81	1.71	1.75	1.81

Evaluating numeric prediction

Performance measure	Formula
mean-squared error	$\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{n}$
root mean-squared error	$\sqrt{\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{n}}$
mean absolute error	$\frac{\lvert p_1-a_1\rvert+\ldots+\lvert p_n-a_n\rvert}{n}$
relative squared error	$\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{(a_1-\bar{a})^2+\ldots+(a_n-\bar{a})^2}$, where $\bar{a}=\frac{1}{n}\sum_i a_i$
root relative squared error	$\sqrt{\frac{(p_1-a_1)^2+\ldots+(p_n-a_n)^2}{(a_1-\bar{a})^2+\ldots+(a_n-\bar{a})^2}}$
relative absolute error	$\frac{\lvert p_1-a_1\rvert+\ldots+\lvert p_n-a_n\rvert}{\lvert a_1-\bar{a}\rvert+\ldots+\lvert a_n-\bar{a}\rvert}$
correlation coefficient	$\frac{S_{PA}}{\sqrt{S_P S_A}}$, where $S_{PA}=\frac{\sum_i(p_i-\bar{p})(a_i-\bar{a})}{n-1}$, $S_p=\frac{\sum_i(p_i-\bar{p})^2}{n-1}$, and $S_A=\frac{\sum_i(a_i-\bar{a})^2}{n-1}$

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Predictive vs. descriptive induction

- **Predictive induction:** Inducing classifiers for solving classification and prediction tasks,
  - Classification rule learning, Decision tree learning, ...
  - Bayesian classifier, ANN, SVM, ...
  - Data analysis through hypothesis generation and testing
- **Descriptive induction:** Discovering interesting regularities in the data, uncovering patterns, ... for solving KDD tasks
  - Symbolic clustering, Association rule learning, Subgroup discovery, ...
  - Exploratory data analysis

Descriptive DM

- Often used for preliminary explanatory data analysis
- User gets feel for the data and its structure
- Aims at deriving descriptions of characteristics of the data
- Visualization and descriptive statistical techniques can be used

Descriptive DM

- **Description**
  - Data description and summarization: describe elementary and aggregated data characteristics (statistics, …)
  - Dependency analysis:
    - describe associations, dependencies, …
    - discovery of properties and constraints
- **Segmentation**
  - Clustering: separate objects into subsets according to distance and/or similarity (clustering, SOM, visualization, ...)
  - Subgroup discovery: find unusual subgroups that are significantly different from the majority (deviation detection w.r.t. overall class distribution)

Predictive vs. descriptive induction: A rule learning perspective

- **Predictive induction:** Induces **rulesets** acting as classifiers for solving classification and prediction tasks
- **Descriptive induction:** Discovers **individual rules** describing interesting regularities in the data

- **Therefore:** Different goals, different heuristics, different evaluation criteria

Supervised vs. unsupervised learning: A rule learning perspective

- **Supervised learning:** Rules are induced from labeled instances (training examples with class assignment) - usually used in **predictive induction**
- **Unsupervised learning:** Rules are induced from unlabeled instances (training examples with no class assignment) - usually used in **descriptive induction**
- **Exception: Subgroup discovery**

  Discovers **individual rules** describing interesting regularities in the data from **labeled** examples

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Subgroup Discovery

Given: a population of individuals and a target class label (the property of individuals we are interested in)

Find: population subgroups that are statistically most `interesting', e.g., are as large as possible and have most unusual statistical (distributional) characteristics w.r.t. the target class (property of interest)

Subgroup interestingness

Interestingness criteria:

– As large as possible
– Class distribution as different as possible from the distribution in the entire data set
– Significant
– Surprising to the user
– Non-redundant
– Simple
– Useful - actionable

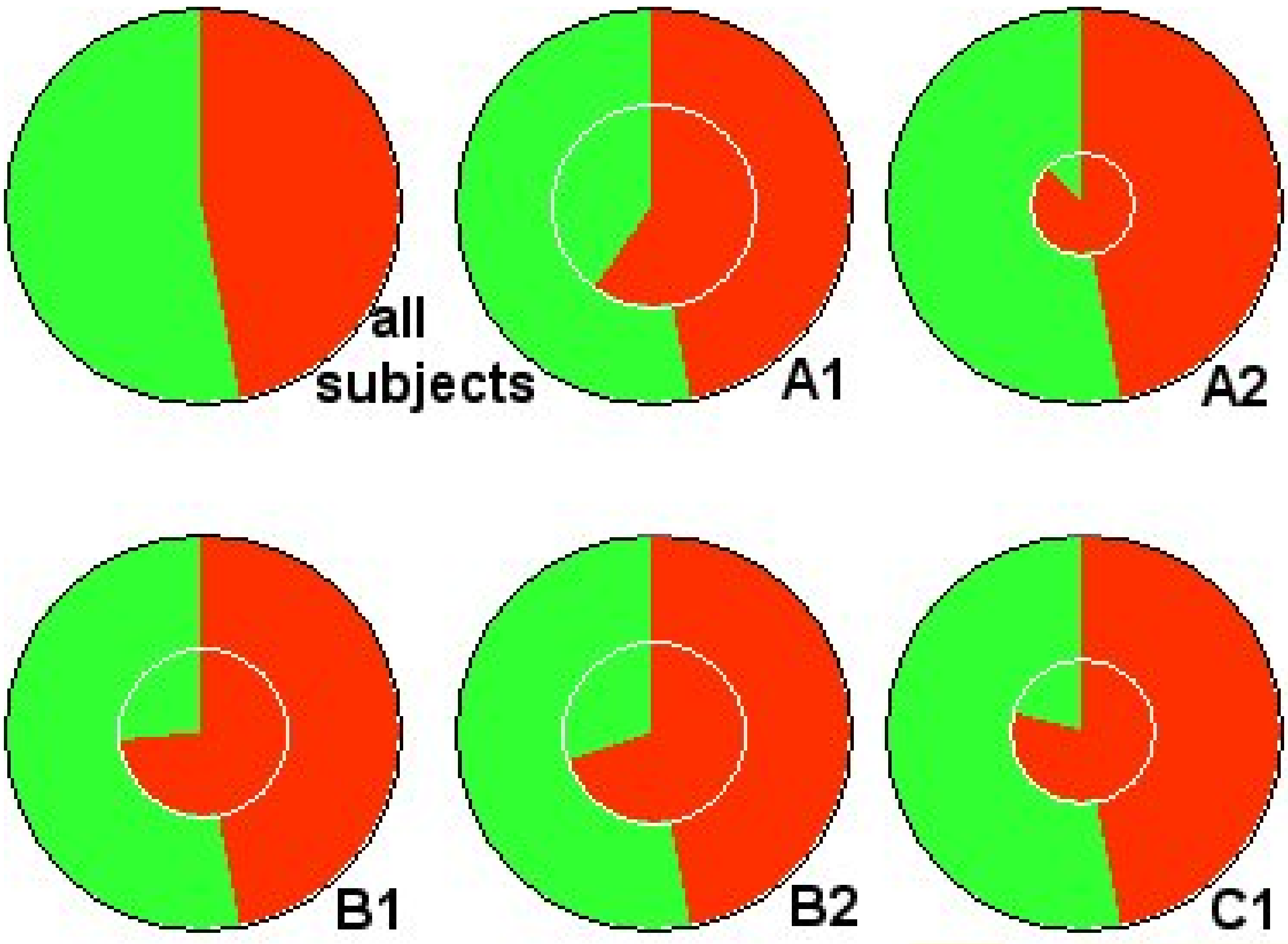
Subgroup Discovery: Medical Case Study

- **Find and characterize population subgroups with high risk for coronary heart disease (CHD)** (Gamberger, Lavrač, Krstačić)
- **A1** for males: **principal risk factors**

  CHD ← pos. fam. history & age > 46
- **A2** for females: **principal risk factors**

  CHD ← bodyMassIndex > 25 & age >63
- **A1, A2** (anamnestic info only), **B1, B2** (an. and physical examination), **C1** (an., phy. and ECG)
- **A1: supporting factors** (found by statistical analysis): psychosocial stress, as well as cigarette smoking, hypertension and overweight

Subgroup visualization

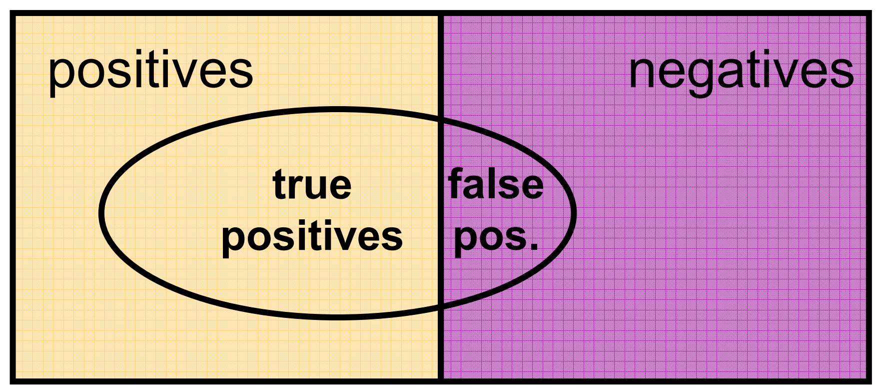


Subgroups of patients with CHD risk

[Gamberger, Lavrač & Wettschereck, IDAMAP2002]

Subgroups vs. classifiers

- Classifiers:
  - Classification rules aim at pure subgroups
  - A set of rules forms a domain model
- Subgroups:
  - Rules describing subgroups aim at significantly higher proportion of positives
  - Each rule is an independent chunk of knowledge
- Link
  - SD can be viewed as cost-sensitive classification
  - Instead of *FNcost* we aim at increased *TPprofit*



Classification Rule Learning for Subgroup Discovery: Deficiencies

- Only first few rules induced by the covering algorithm have sufficient support (coverage)
- Subsequent rules are induced from smaller and strongly biased example subsets (pos. examples not covered by previously induced rules), which hinders their ability to detect population subgroups
- ‘Ordered’ rules are induced and interpreted sequentially as a **if-then-else** decision list

CN2-SD: Adapting CN2 Rule Learning to Subgroup Discovery

- Weighted covering algorithm
- Weighted relative accuracy (WRAcc) search heuristics, with added example weights
- Probabilistic classification
- Evaluation with different interestingness measures

CN2-SD: CN2 Adaptations

- General-to-specific search (beam search) for best rules
- Rule quality measure:
  - CN2: Laplace: Acc(Class $\leftarrow$ Cond) =

    = p(Class|Cond) = $(n_c+1)/(n_{rule}+k)$
  - CN2-SD: Weighted Relative Accuracy

    WRAcc(Class $\leftarrow$ Cond) =

    p(Cond) (p(Class|Cond) - p(Class))
- Weighted covering approach (example weights)
- Significance testing (likelihood ratio statistics)
- Output: Unordered rule sets (probabilistic classification)

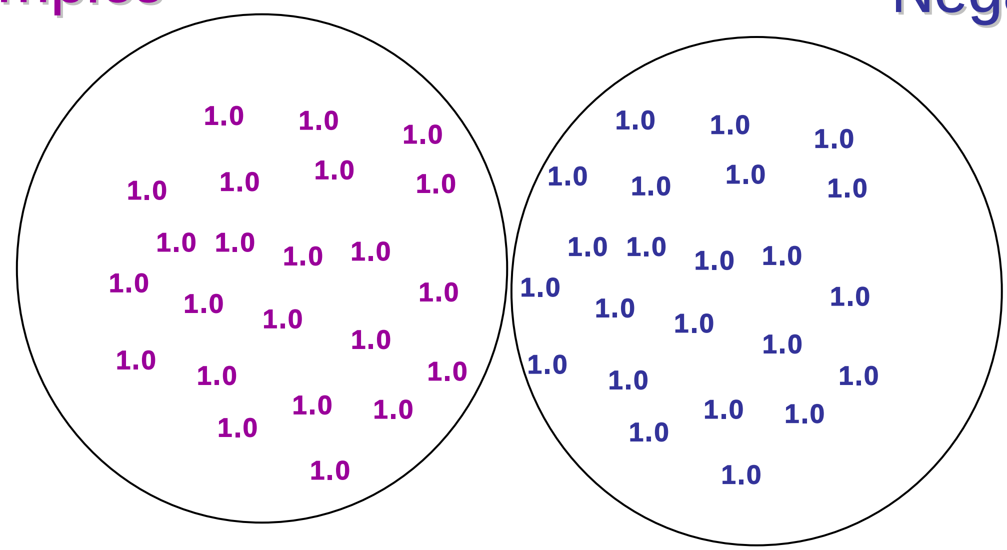
CN2-SD: Weighted Covering

- Standard covering approach:
  covered examples are deleted from current training set
- Weighted covering approach:
  - weights assigned to examples
  - covered pos. examples are re-weighted:
    in all covering loop iterations, store
    count i how many times (with how many
    rules induced so far) a pos. example has
    been covered: w(e,i), w(e,0)=1
    - **Additive weights:** **`w(e,i) = 1/(i+1)`**
      `w(e,i)` – pos. example e being covered `i` times

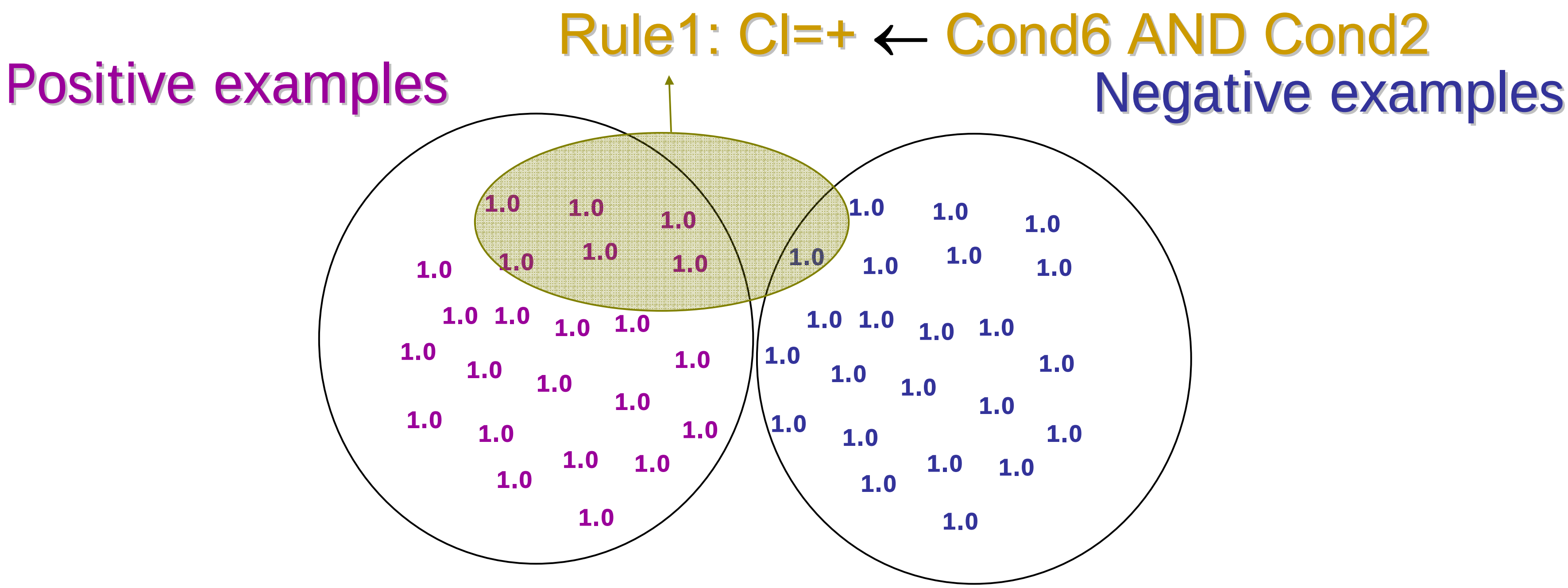
Subgroup Discovery

Positive examples

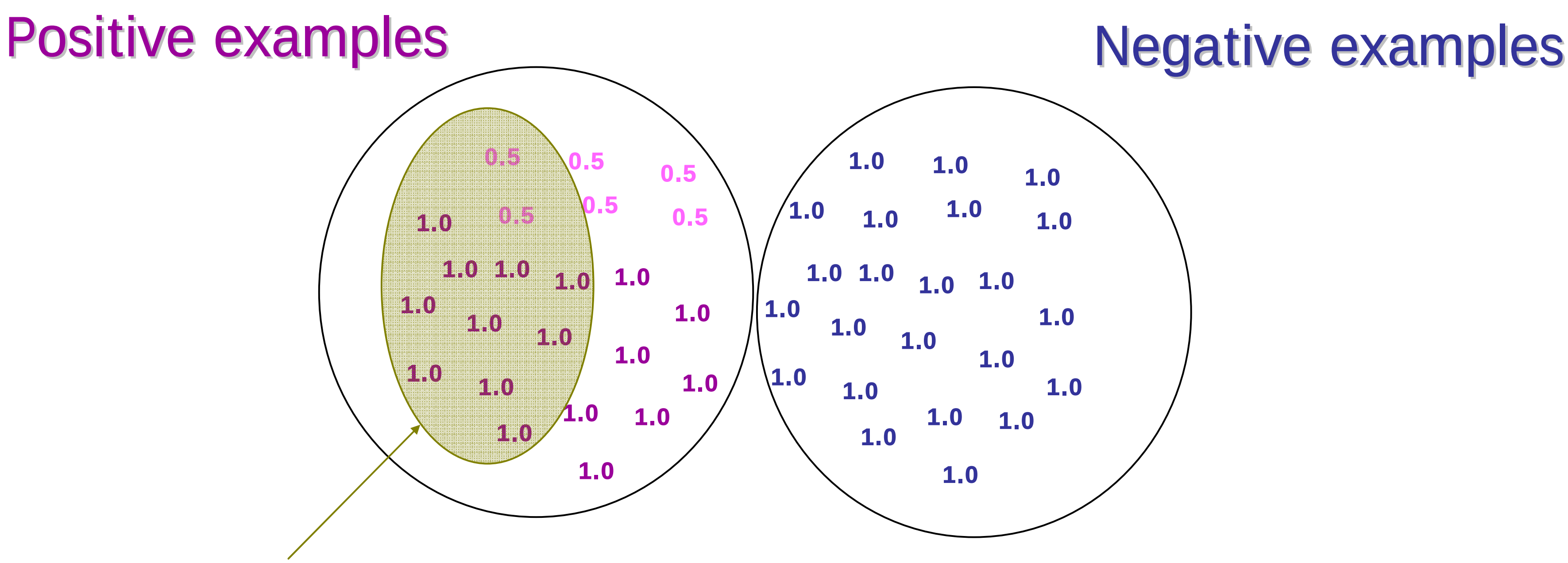
Negative examples



Subgroup Discovery



Subgroup Discovery

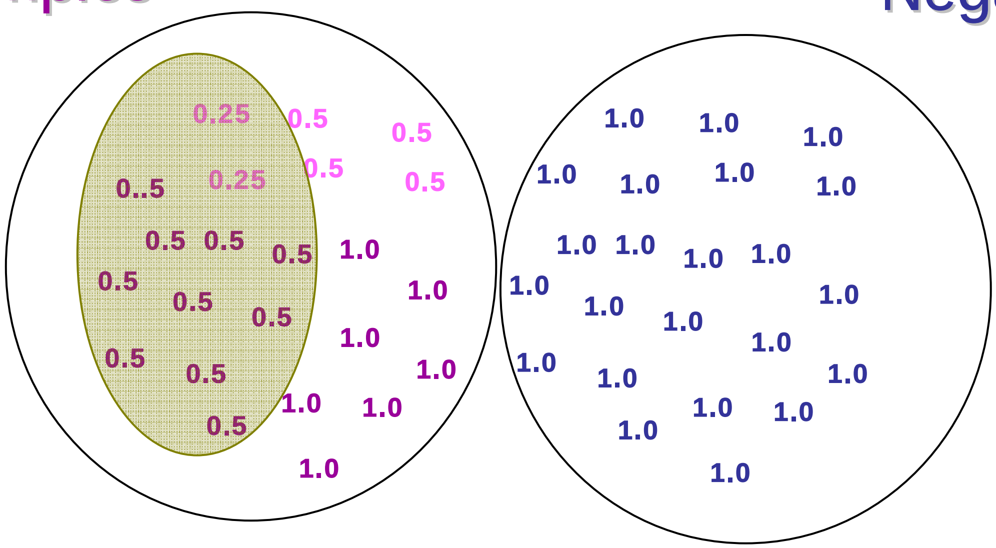


Rule2: Cl=+ ← Cond3 AND Cond4

Subgroup Discovery

Positive examples

Negative examples



CN2-SD: Weighted WRAcc Search Heuristic

- **Weighted relative accuracy (WRAcc) search heuristics, with added example weights**

  WRAcc(Cl $\leftarrow$ Cond) = p(Cond) (p(Cl|Cond) - p(Cl))

  increased coverage, decreased # of rules, approx. equal accuracy (PKDD-2000)

- In WRAcc computation, probabilities are estimated with relative frequencies, adapt:

  WRAcc(Cl $\leftarrow$ Cond) = p(Cond) (p(Cl|Cond) - p(Cl)) =

  n'(Cond)/N' (n'(Cl.Cond)/n'(Cond) - n'(Cl)/N')

  – N' : sum of weights of examples
  – n'(Cond) : sum of weights of all covered examples
  – n'(Cl.Cond) : sum of weights of all correctly covered examples

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Association Rule Learning

Rules: X =>Y, if X then Y

X and Y are itemsets (records, conjunction of items), where items/features are binary-valued attributes)

Given: Transactions

itemsets (records)

	i1	i2	....	i50
t1	1	1		0
t2	0	1		0
	…	…	....	…

Find: A set of association rules in the form X =>Y

Example: Market basket analysis

beer & coke => peanuts & chips (0.05, 0.65)

- Support: Sup(X,Y) = #XY/#D = p(XY)
- Confidence: Conf(X,Y) = #XY/#X = Sup(X,Y)/Sup(X) = = p(XY)/p(X) = p(Y|X)

Association Rule Learning: Examples

- Market basket analysis
  - beer & coke ⇒ peanuts & chips (5%, 65%)
    (IF beer AND coke THEN peanuts AND chips)
  - Support 5%: 5% of all customers buy all four items
  - Confidence 65%: 65% of customers that buy beer and coke also buy peanuts and chips
- Insurance
  - mortgage & loans & savings ⇒ insurance (2%, 62%)
  - Support 2%: 2% of all customers have all four
  - Confidence 62%: 62% of all customers that have mortgage, loan and savings also have insurance

Association rule learning

- **X ⇒ Y** . . . IF X THEN Y, where X and Y are itemsets
- intuitive meaning: transactions that contain X tend to contain Y
- **Items** - binary attributes (features) m,f,headache, muscle pain, arthrotic, arthritic, spondylotic, spondylitic, stiff_less_1_hour
- **Example transactions** – itemsets formed of patient records

	i1	i2 ……	… i50
t1	1	0	0
t2	0	1	0
…	…	…	...

- **Association rules**

  spondylitic ⇒ arthritic & stiff_gt_1_hour [5%, 70%]

  arthrotic & spondylotic ⇒ stiff_less_1_hour [20%, 90%]

Association Rule Learning

Given: a set of transactions D

Find: all association rules that hold on the set of transactions that have

- user defined minimum support, i.e., support > MinSup, and
- user defined minimum confidence, i.e., confidence > MinConf

It is a form of exploratory data analysis, rather than hypothesis verification

Searching for the associations

- Find all large itemsets
- Use the large itemsets to generate association rules
- If XY is a large itemset, compute

  r =support(XY) / support(X)
- If r > MinConf, then $X \Rightarrow Y$ holds

  (support > MinSup, as XY is large)

Large itemsets

- Large itemsets are itemsets that appear in at least MinSup transaction
- All subsets of a large itemset are large itemsets (e.g., if A,B appears in at least MinSup transactions, so do A and B)
- This observation is the basis for very efficient algorithms for association rules discovery (linear in the number of transactions)

Association vs. Classification rules

Association rules

- Exploration of dependencies
- Different combinations of dependent and independent attributes
- Complete search (all rules found)

Classification rules

- Focused prediction
- Predict one attribute (class) from the others
- Heuristic search (subset of rules found)

Part IV. Descriptive DM techniques

- Predictive vs. descriptive induction
- Subgroup discovery
- Association rule learning
- Hierarchical clustering

Hierarchical clustering

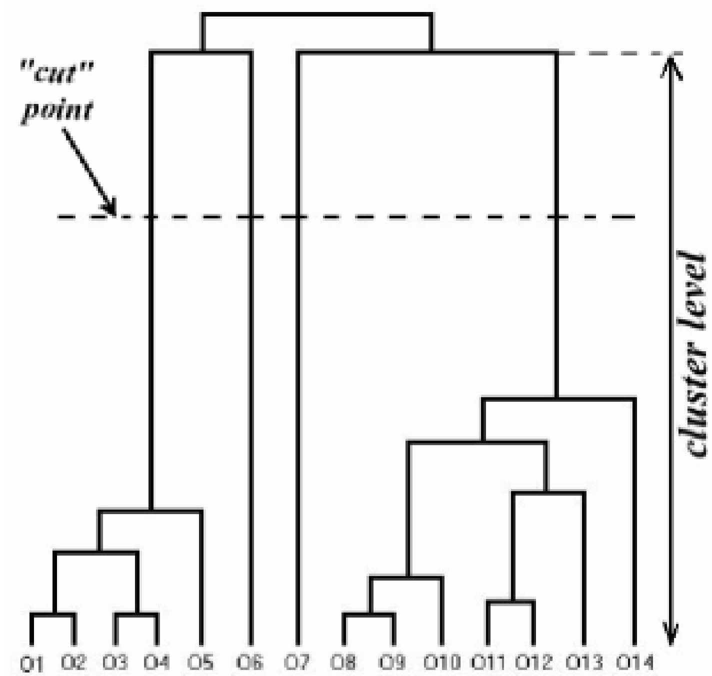
- Algorithm (agglomerative hierarchical clustering):

Each instance is a cluster;

repeat
  find *nearest* pair C_i in C_j;
  ***fuse* C_i in C_j in a new cluster**
    $C_r = C_i \cup C_j$;
  determine *dissimilarities* between
    C_r and other clusters;

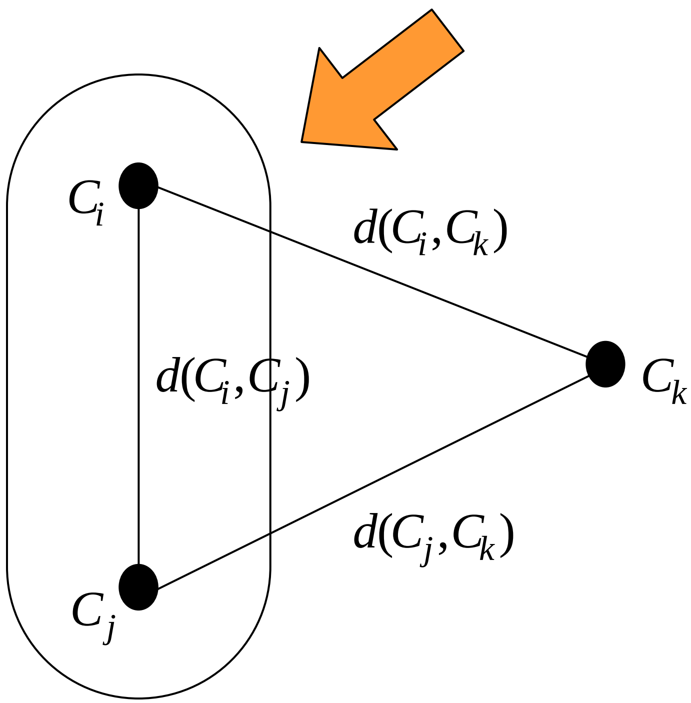
until one cluster left;

- Dendogram:



Hierarchical clustering

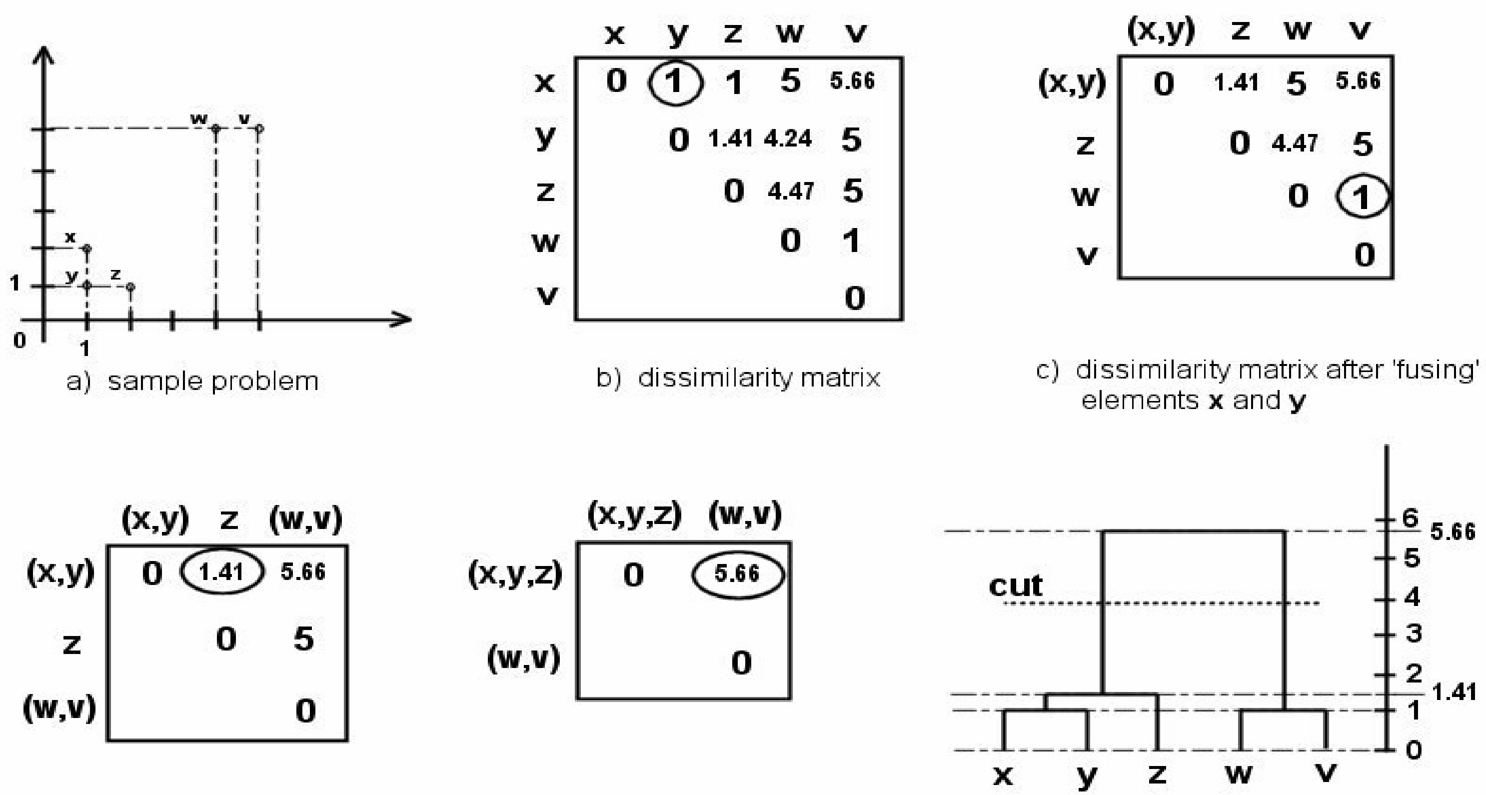
- Fusing the nearest pair of clusters



- Minimizing intra-cluster similarity
- Maximizing inter-cluster similarity

- Computing the dissimilarities from the "new" cluster

Hierarchical clustering: example



a) sample problem

b) dissimilarity matrix

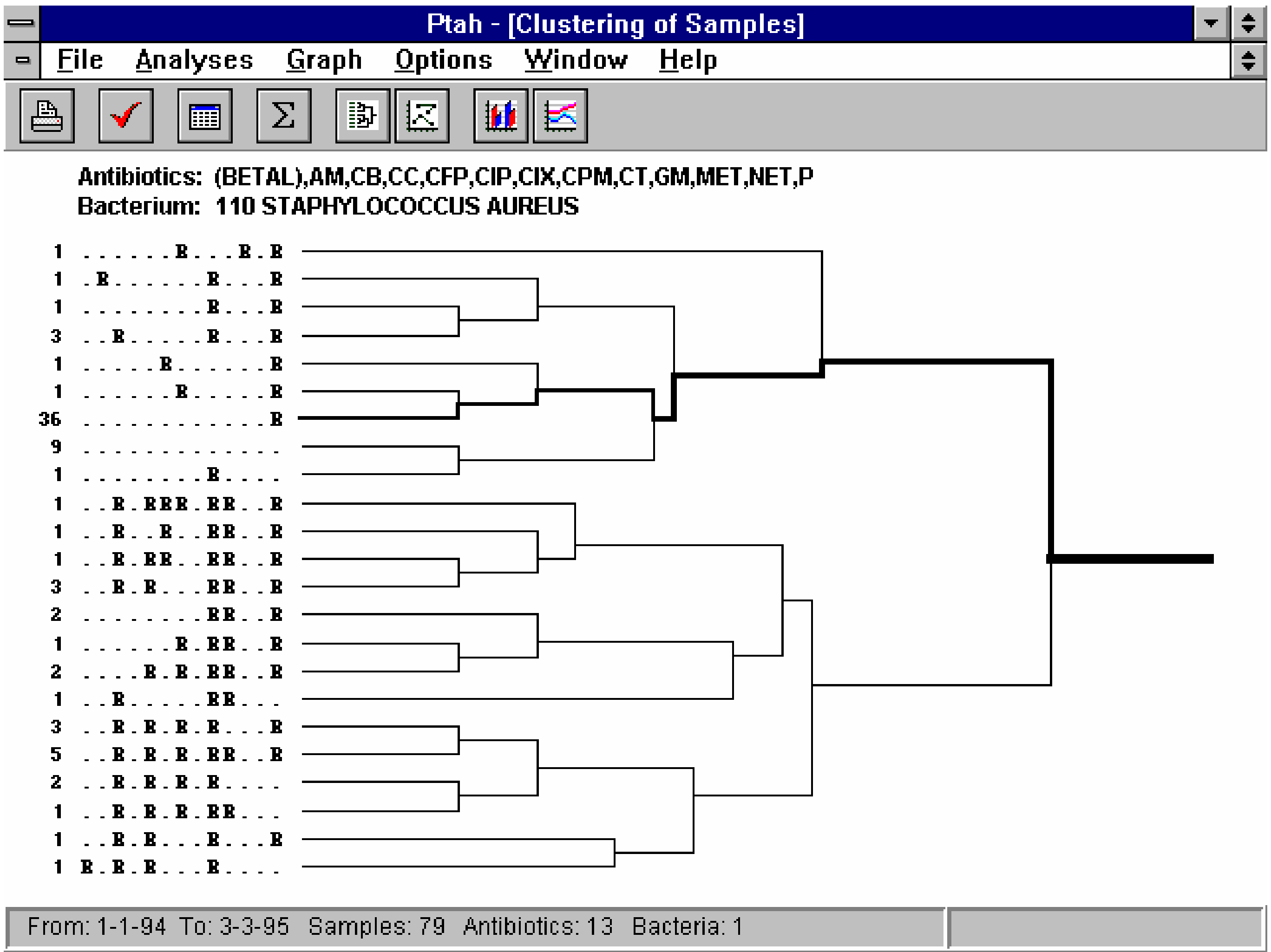
c) dissimilarity matrix after 'fusing' elements **x** and **y**

d) dissimilarity matrix after 'fusing' elements **w** and **v**

e) dissimilarity matrix after 'fusing' cluster **(x,y)** and element **z**

f) dendrogram

Results of clustering



A dendogram of resistance vectors

[Bohanec et al., "PTAH: A system for supporting nosocomial infection therapy", IDAMAP book, 1997]

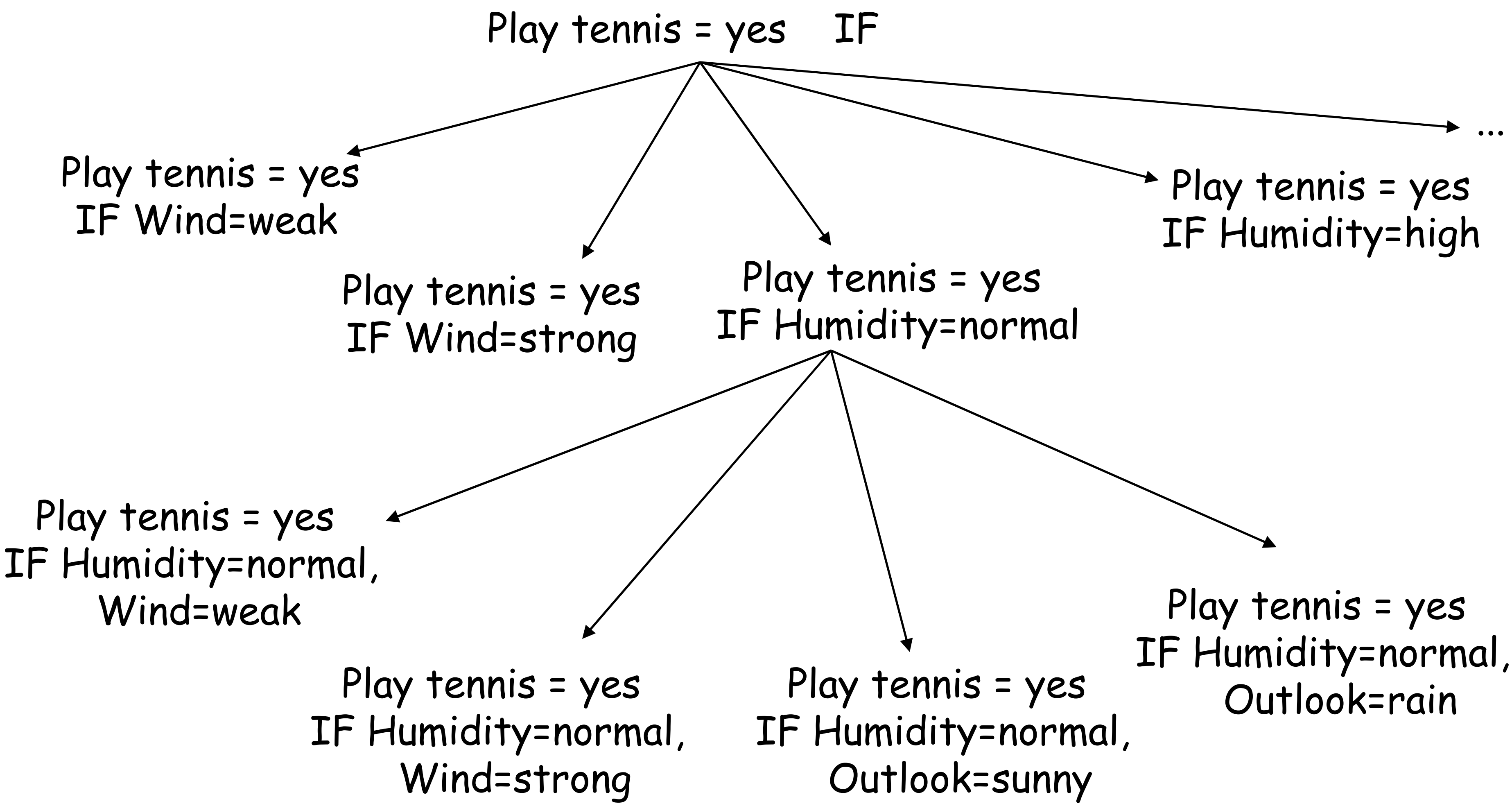
Part V: Relational Data Mining

- Learning as search
- What is RDM?
- Propositionalization techniques
- Inductive Logic Programming

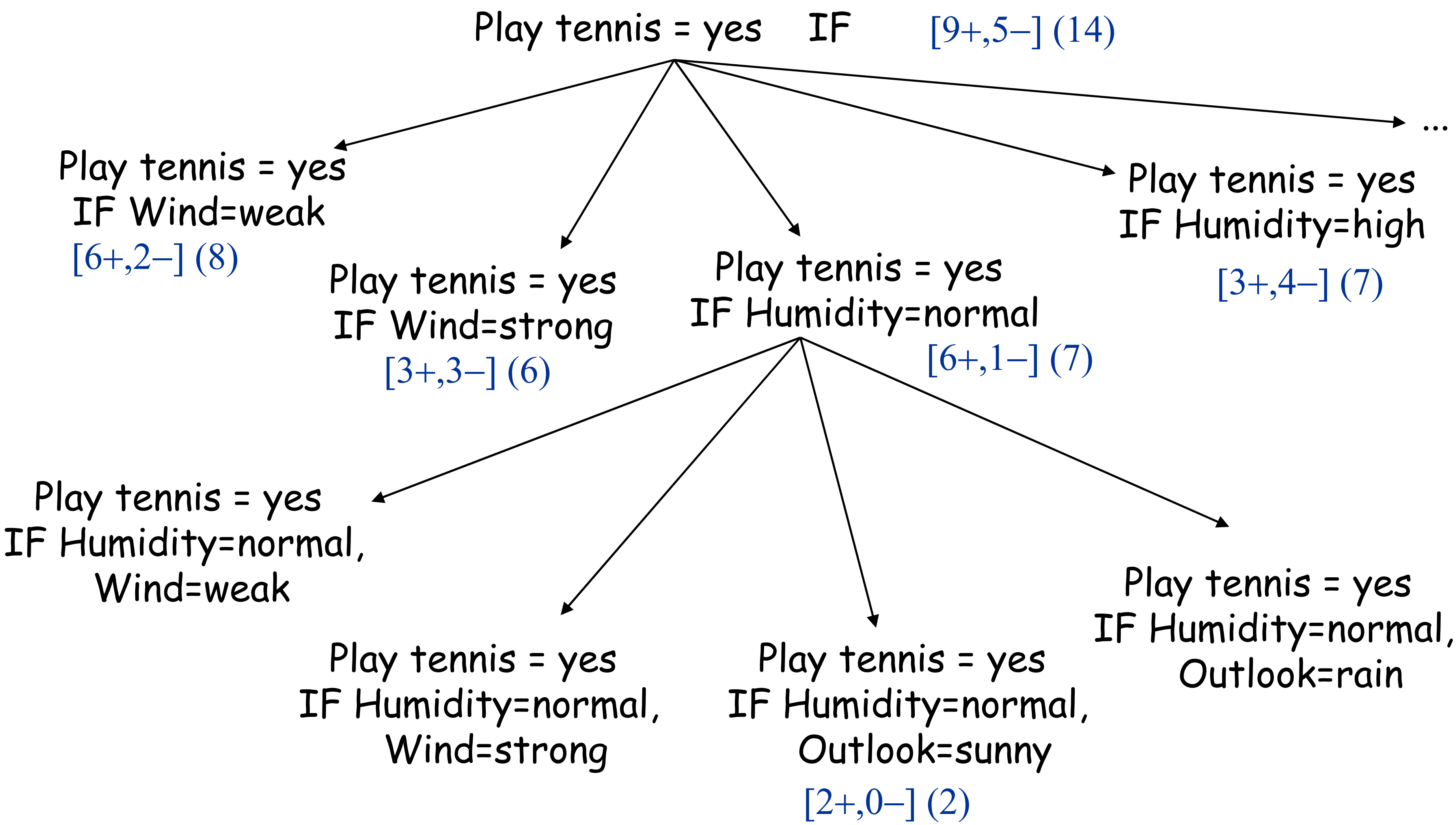
Learning as search

- **Structuring the state space**: Representing a partial order of hypotheses (e.g. rules) as a graph
  - nodes: concept descriptions (hypotheses/rules)
  - arcs defined by specialization/generalization operators : an arc from parent to child exists if-and-only-if parent is a proper most specific generalization of child
- **Specialization operators**: e.g., adding conditions: s(A=a2 & B=b1) = {A=a2 & B=b1 & D=d1, A=a2 & B=b1 & D=d2}
- **Generalization operators**: e.g., dropping conditions: g(A=a2 & B=b1) = {A=a2, B=b1}
- **Partial order of hypotheses defines a lattice (called a refinement graph**)

Learn-one-rule as search - Structuring the hypothesis space: PlayTennis example

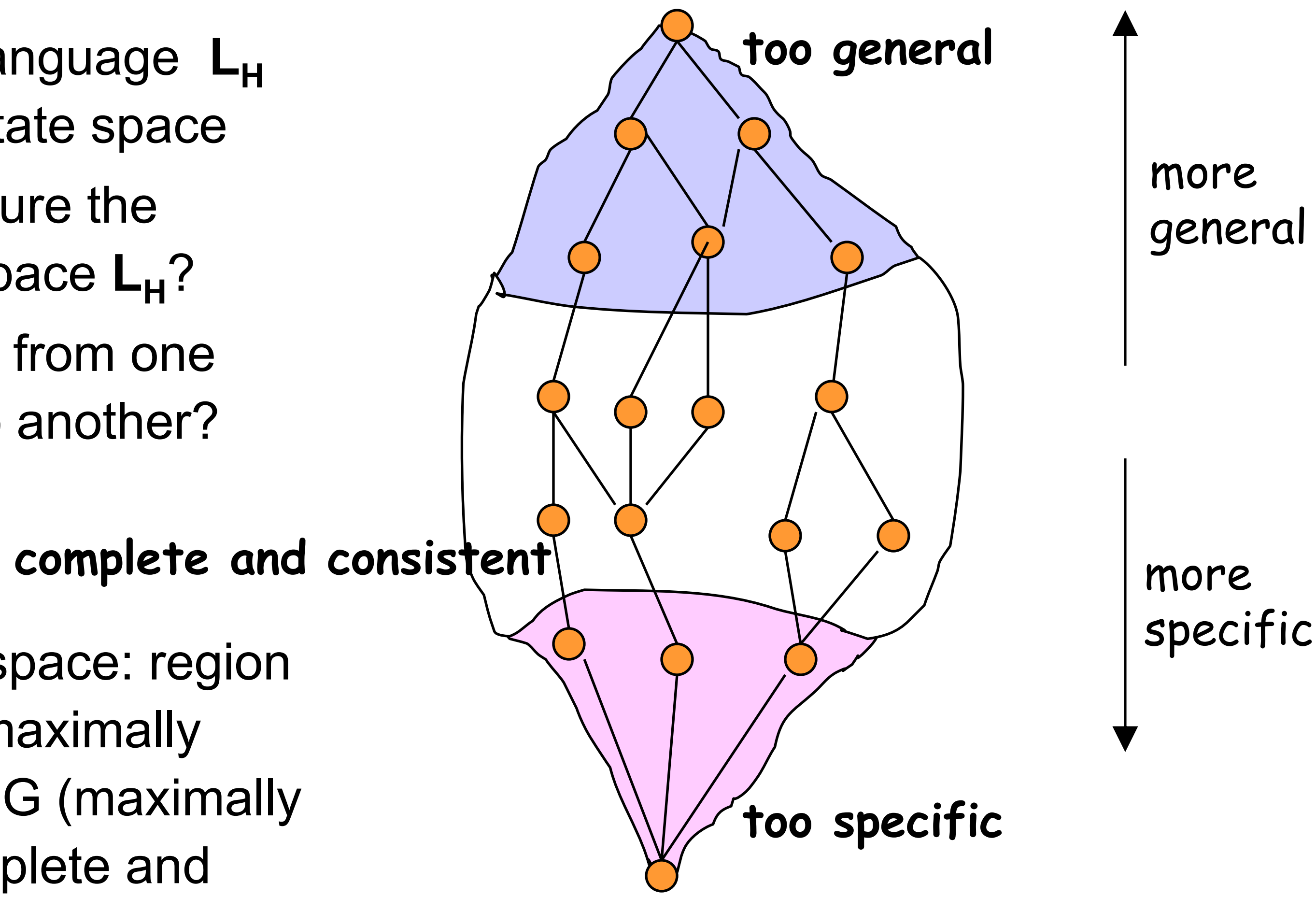


Learn-one-rule as heuristic search: PlayTennis example



Learning as search (Mitchell's version space model)

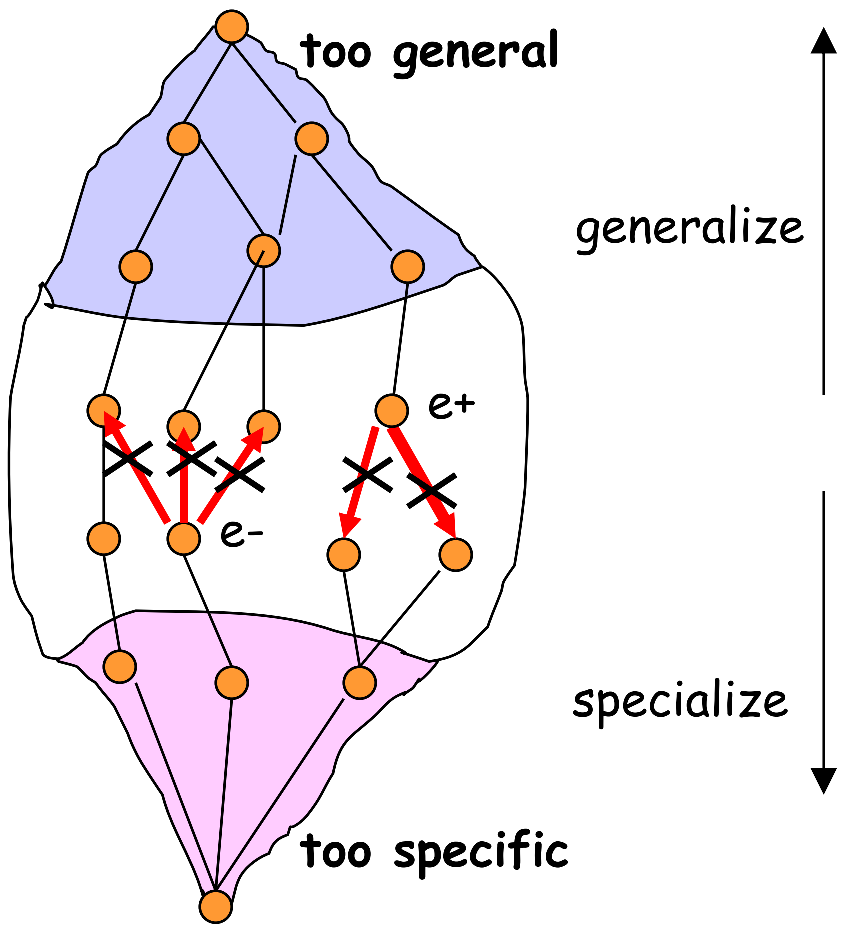
- Hypothesis language **L_H** defines the state space
- How to structure the hypothesis space **L_H**?
- How to move from one hypothesis to another?



- The version space: region between S (maximally specific) and G (maximally general) complete and consistent concept descriptions

Learning as search

- Search/move by applying generalization and specialization
- **Prune generalizations**:
  - if H covers example e then all generalizations of H will also cover e (prune using neg. ex.)
- **Prune specializations:**
  - if H does not cover example e, no specialization will cover e (prune using if H pos. ex.)



Learning as search: Learner’s ingredients

- structure of the search space (specialization and generalization operators)
- search strategy
  - depth-first
  - breath-first
  - heuristic search (best first, hill-climbing, beam search)
- search heuristics
  - measure of attribute ‘informativity’
  - measure of ‘expected classification accuracy’ (relative frequency, Laplace estimate, m-estimate), ...
- stopping criteria (consistency, completeness, statistical significance, …)

Learn-one-rule: search heuristics

- Assume a two-class problem
- Two classes (+,-), learn rules for + class (Cl).
- Search for specializations R' of a rule R = Cl $\leftarrow$ Cond from the RuleBase.
- Specializarion R' of rule R = Cl $\leftarrow$ Cond

  has the form R' = Cl $\leftarrow$ Cond & Cond'
- Heuristic search for rules: find the 'best' Cond' to be added to the current rule R, such that rule accuracy is improved, e.g., such that Acc(R') > Acc(R)
  - where the expected **classification accuracy** can be estimated as A(R) = p(Cl|Cond)

Learn-one-rule – Search strategy: Greedy vs. beam search

- learn-one-rule by greedy general-to-specific search, at each step selecting the `best' descendant, no backtracking
  - e.g., the best descendant of the initial rule

    PlayTennis = yes $\leftarrow$
  - is rule PlayTennis = yes $\leftarrow$ Humidity=normal
- beam search: maintain a list of k best candidates at each step; descendants (specializations) of each of these k candidates are generated, and the resulting set is again reduced to k best candidates

Part V: Relational Data Mining

- Learning as search
- What is RDM?
- Propositionalization techniques
- Inductive Logic Programming

Predictive relational DM

- Data stored in relational databases
- Single relation - propositional DM
  - example is a tuple of values of a fixed number of attributes (one attribute is a class)
  - example set is a table (simple field values)
- Multiple relations - relational DM (ILP)
  - example is a tuple or a set of tuples (logical fact or set of logical facts)
  - example set is a set of tables (simple or complex structured objects as field values)

Data for propositional DM

Sample single relation data table

ID	Name	First Name	Street	City	Zip	Sex	Social Status	In-come	Age	Club Status	Res-ponse
...	...	...	...	...	...	...	...	...	...	...	...
3478	Smith	John	38, Lake Dr	Sam pleton	34677	male	single	i60-70k	32	mem ber	no_res ponse
3479	Doe	Jane	45, Sea Ct	Inven-tion	43666	female	mar-ried	i80-90k	45	non-mem-ber	res-ponse
...	...	...	...	...	...	...	...	...	...	...	...

Basic customer table.

ID	Zip	S ex	So St	In come	A ge	Cl ub	Re sp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

Customer table for analysis.

ID	Zip	S ex	So St	In come	A ge	Cl ub	Re sp	Delivery Mode	Paymt Mode	Store Size	Store Type	Store Locatn
...	...	...	...	...	...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr	regular	cash	small	franchise	city
3479	43666	f	ma	80-90	45	nm	re	express	credit	large	indep	rural
...	...	...	...	...	...	...	...	...	...	...	...	...

Customer table including order and store information.

Multi-relational data made propositional

- Sample relation table

ID	Zip	Sex	SoSt	Income	Age	Club	Resp	Delivery Mode	Paymt Mode	Store Size	Store Type	Store Locatn
...	...	...	...	...	...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr	regular	cash	small	franchise	city
3478	34677	m	si	60-70	32	me	nr	express	check	small	franchise	city
3478	34677	m	si	60-70	32	me	nr	regular	check	large	indep	rural
3479	43666	f	ma	80-90	45	nm	re	express	credit	large	indep	rural
3479	43666	f	ma	80-90	45	nm	re	regular	credit	small	franchise	city
...	...	...	...	...	...	...	...	...	...	...	...	...

Customer table with multiple orders.

- Making data using summary

ID	Zip	Sex	SoSt	Income	Age	Club	Resp	No. of Orders	No. of Stores
...	...	...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr	3	2
3479	43666	f	ma	80-90	45	nm	re	2	2
...	...	...	...	...	...	...	...	...	...

Customer table using summary attributes.

Relational Data Mining (ILP)

- Learning from multiple tables
- Complex relational problems:
  - temporal data: time series in medicine, trafic control, ...
  - structured data: representation of molecules and their properties in protein engineering, biochemistry, ...

customer							
ID	Zip	Sex	SoSt	Income	Age	Club	Resp
...	...	...	...	...	...	...	...
3478	34677	m	si	60-70	32	me	nr
3479	43666	f	ma	80-90	45	nm	re
...	...	...	...	...	...	...	...

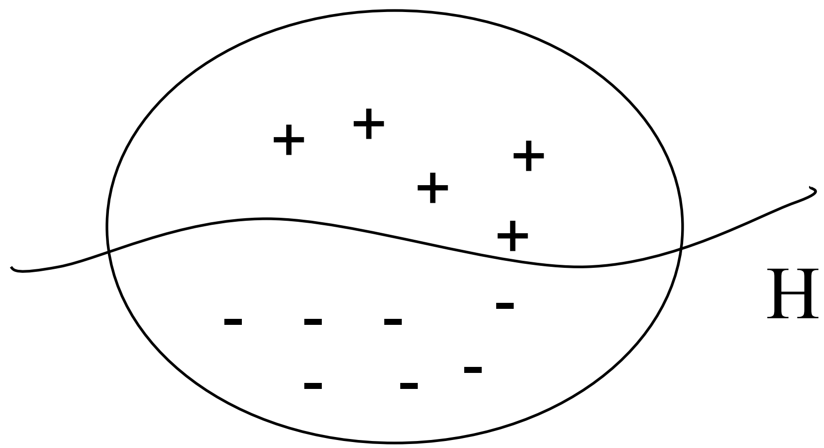
order				
Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

store			
Store ID	Size	Type	Location
...	...	...	...
12	small	franchise	city
17	large	indep	rural
...	...	...	...

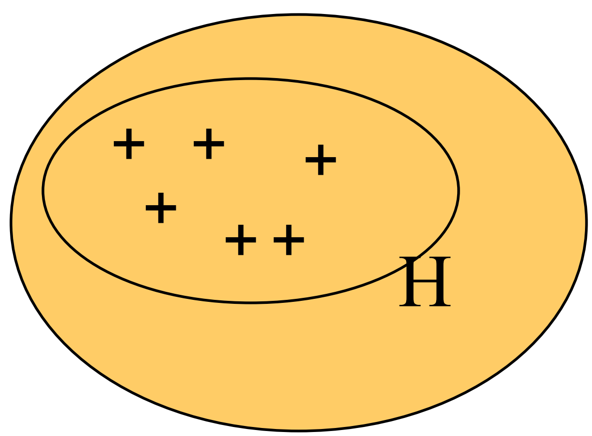
Relational representation of customers, orders and stores.

Basic Relational Data Mining tasks

Predictive RDM

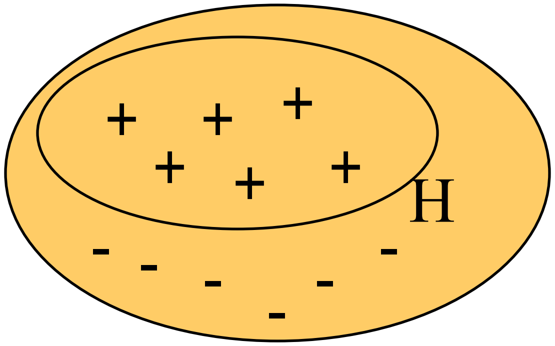


Descriptive RDM



Predictive ILP

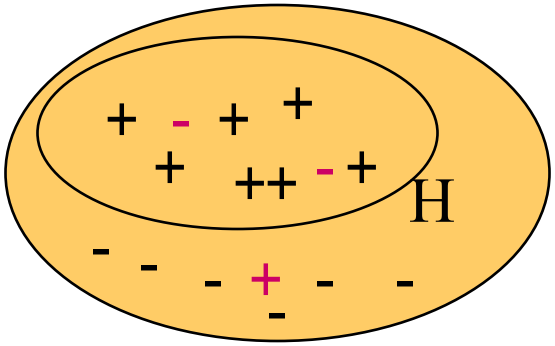
- **Given:**
  - A set of observations
    - positive examples E^+
    - negative examples E^-
  - background knowledge B
  - hypothesis language L_H
  - covers relation
- **Find:**
  A hypothesis $H \in L_H$, such that (given B) H covers all positive and no negative examples



- In logic, **find** H such that
  - $\forall e \in E^+ : \mathrm{B} \wedge \mathrm{H} \models e$ (H is complete)
  - $\forall e \in E^- : \mathrm{B} \wedge \mathrm{H} \not\models e$ (H is consistent)
- In ILP, E are ground facts, B and H are (sets of) definite clauses

Predictive ILP

- **Given:**
  - A set of observations
    - positive examples E^+
    - negative examples E^-
  - background knowledge *B*
  - hypothesis language L_H
  - covers relation
  - quality criterion



- **Find:**

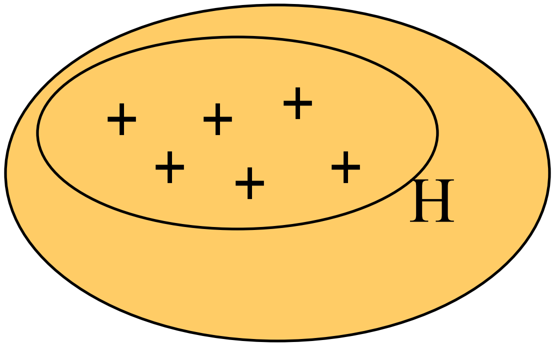
  A hypothesis $H \in L_H$, such that (given *B*) *H* is optimal w.r.t. some quality criterion, e.g., max. predictive accuracy A(H)

(instead of finding a hypothesis $H \in L_H$, such that (given *B*) *H* covers all positive and no negative examples)

Descriptive ILP

- **Given:**
  - A set of observations
    (positive examples E^+)
  - background knowledge *B*
  - hypothesis language L_H
  - covers relation

- **Find:**
  Maximally specific hypothesis $H \in L_H$, such that (given *B*) *H* covers all positive examples



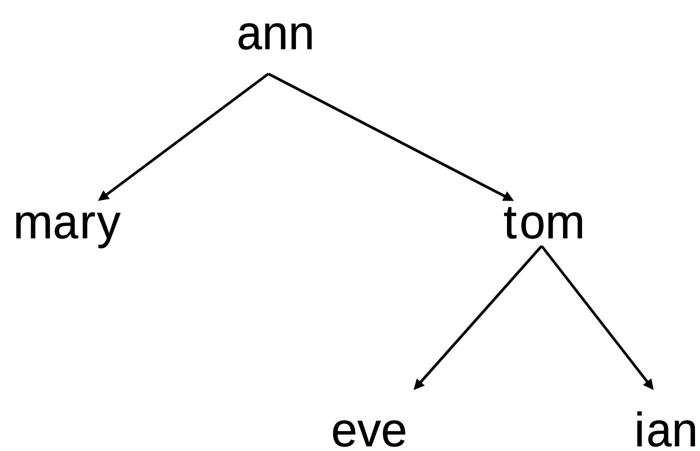
- In logic, **find** *H* such that $\forall c \in H$, *c* is true in some preferred model of $B \cup E$ (e.g., least Herbrand model $M(B \cup E)$)

- **In ILP, *E* are ground facts, *B* are (sets of) general clauses**

Sample problem
Knowledge discovery

E^{+} = {daughter(mary,ann),daughter(eve,tom)}
E^{-} = {daughter(tom,ann),daughter(eve,ann)}

B = {mother(ann,mary), mother(ann,tom), father(tom,eve), father(tom,ian), female(ann), female(mary), female(eve), male(pat),male(tom), parent(X,Y) ← mother(X,Y), parent(X,Y) ← father(X,Y)}



Sample problem
Knowledge discovery

- E^+ = `{daughter(mary,ann),daughter(eve,tom)}`
  E^- = `{daughter(tom,ann),daughter(eve,ann)}`
- B = `{mother(ann,mary),mother(ann,tom),father(tom,eve), father(tom,ian),female(ann),female(mary),female(eve), male(pat),male(tom),parent(X,Y)←mother(X,Y), parent(X,Y)←father(X,Y)}`

- **Predictive ILP -** Induce a definite clause

```
daughter(X,Y) ← female(X), parent(Y,X).
```

or a set of definite clauses

```
daughter(X,Y) ← female(X), mother(Y,X).
daughter(X,Y) ← female(X), father(Y,X).
```

- **Descriptive ILP -** Induce a set of (general) clauses

```
← daughter(X,Y), mother(X,Y).
female(X)← daughter(X,Y).
mother(X,Y); father(X,Y) ← parent(X,Y).
```

Sample problem
Logic programming

E^+ = {`sort([2,1,3],[1,2,3])`}
E^- = {`sort([2,1],[1]),sort([3,1,2],[2,1,3])`}

B : definitions of `permutation/2` and `sorted/1`

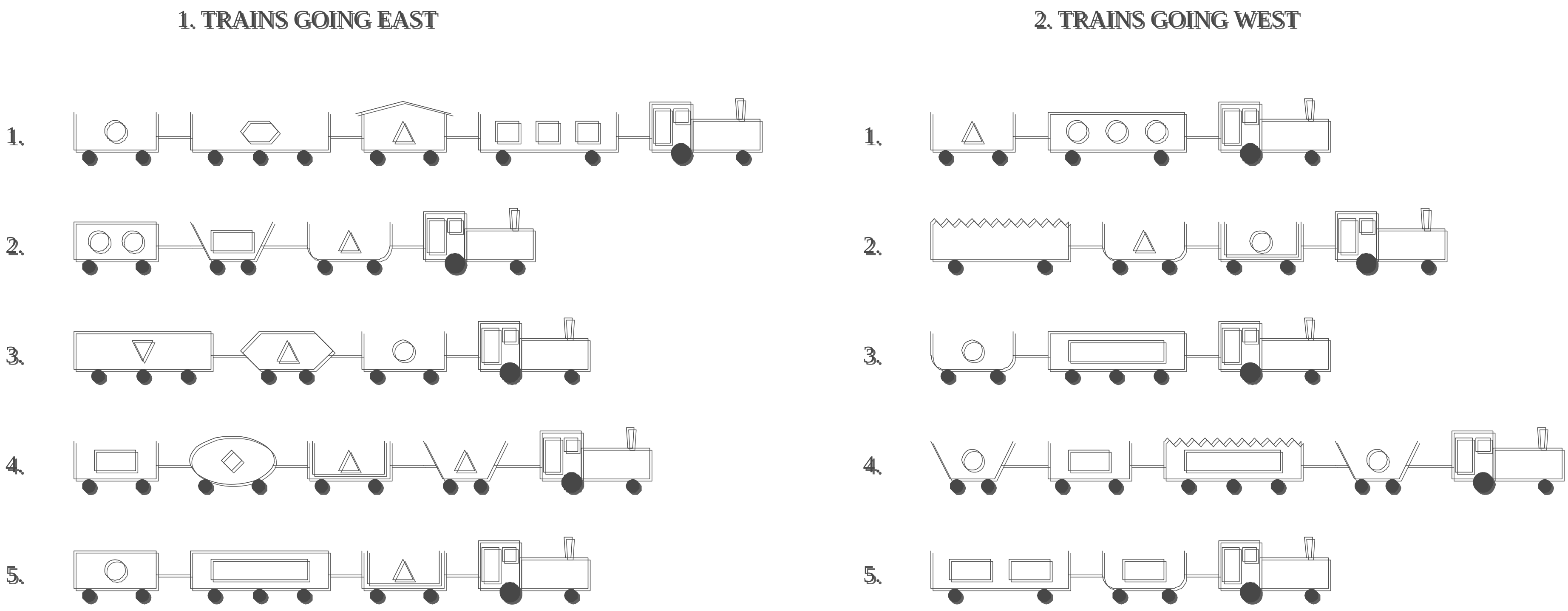
- **Predictive ILP**

```
sort(X,Y) ← permutation(X,Y), sorted(Y).
```

- **Descriptive ILP**

```
sorted(Y) ← sort(X,Y).
permutation(X,Y) ← sort(X,Y)
sorted(X) ← sort(X,X)
```

Sample problem: East-West trains



RDM knowledge representation (database)

LOAD_TABLE

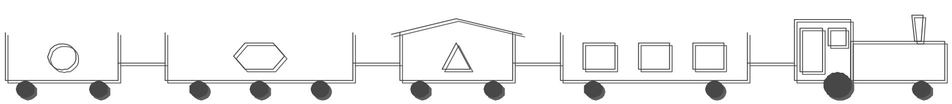
LOAD	CAR	OBJECT	NUMBER
l1	c1	circle	1
l2	c2	hexagon	1
l3	c3	triangle	1
l4	c4	rectangle	3
…	…	…	

TRAIN_TABLE

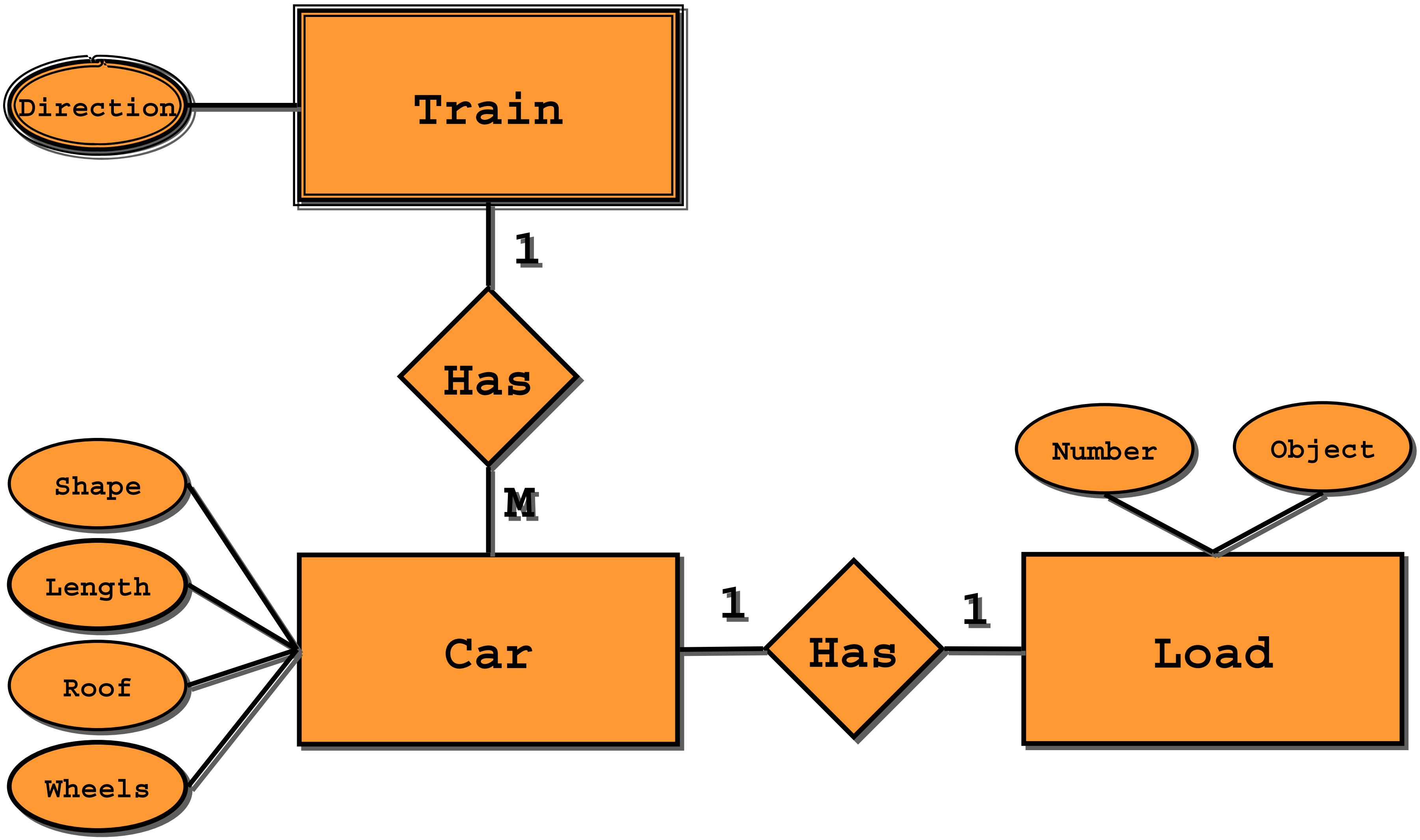
TRAIN	EASTBOUND
t1	TRUE
t2	TRUE
…	…
t6	FALSE
…	…

CAR_TABLE

CAR	TRAIN	SHAPE	LENGTH	ROOF	WHEELS
c1	t1	rectangle	short	none	2
c2	t1	rectangle	long	none	3
c3	t1	rectangle	short	peaked	2
c4	t1	rectangle	long	none	2
…	…	…			…

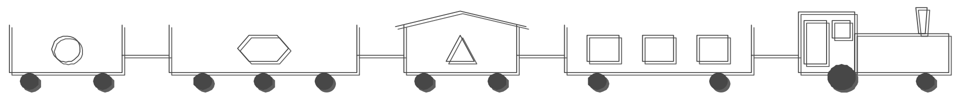


ER diagram for East-West trains



ILP representation: Datalog ground facts

- Example:
  eastbound(t1).



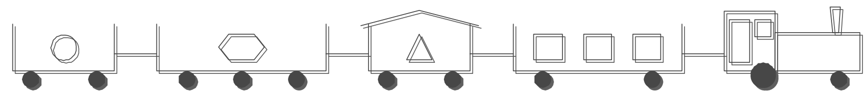
- Background theory:

car(t1,c1).	car(t1,c2).	car(t1,c3).	car(t1,c4).
rectangle(c1).	rectangle(c2).	rectangle(c3).	rectangle(c4).
short(c1).	long(c2).	short(c3).	long(c4).
none(c1).	none(c2).	peaked(c3).	none(c4).
two_wheels(c1).	three_wheels(c2).	two_wheels(c3).	two_wheels(c4).
load(c1,l1).	load(c2,l2).	load(c3,l3).	load(c4,l4).
circle(l1).	hexagon(l2).	triangle(l3).	rectangle(l4).
one_load(l1).	one_load(l2).	one_load(l3).	three_loads(l4).

- Hypothesis (predictive ILP):

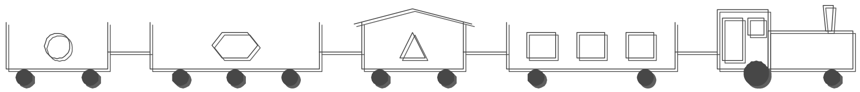
eastbound(T) :- car(T,C),short(C),not none(C).

ILP representation: Datalog ground clauses



- Example:
  eastbound(t1):-
  car(t1,c1),rectangle(c1),short(c1),none(c1),two_wheels(c1),
  load(c1,l1),circle(l1),one_load(l1),
  car(t1,c2),rectangle(c2),long(c2),none(c2),three_wheels(c2),
  load(c2,l2),hexagon(l2),one_load(l2),
  car(t1,c3),rectangle(c3),short(c3),peaked(c3),two_wheels(c3),
  load(c3,l3),triangle(l3),one_load(l3),
  car(t1,c4),rectangle(c4),long(c4),none(c4),two_wheels(c4),
  load(c4,l4),rectangle(l4),three_load(l4).
- Background theory: empty
- Hypothesis:
  eastbound(T):-car(T,C),short(C),not none(C).

ILP representation: Prolog terms



- Example:

  eastbound([c(rectangle,short,none,2,l(circle,1)),
  c(rectangle,long,none,3,l(hexagon,1)),
  c(rectangle,short,peaked,2,l(triangle,1)),
  c(rectangle,long,none,2,l(rectangle,3))]).

- Background theory: member/2, arg/3
- Hypothesis:

  eastbound(T):-member(C,T),arg(2,C,short), not arg(3,C,none).

First-order representations

- **Propositional** representations:
  - datacase is ***fixed-size vector of values***
  - features are those given in the dataset
- **First-order** representations:
  - datacase is ***flexible-size, structured object***
    - sequence, set, graph
    - hierarchical: e.g. set of sequences
  - features need to be **selected** from potentially infinite set

Complexity of RDM problems

- Simplest case: single table with primary key
  - example corresponds to tuple of constants
  - ***attribute-value*** or ***propositional*** learning
- Next: single table without primary key
  - example corresponds to set of tuples of constants
  - ***multiple-instance*** problem
- Complexity resides in many-to-one foreign keys
  - lists, sets, multisets
  - ***non-determinate*** variables

Part V:
Relational Data Mining

- Learning as search
- What is RDM?
- Propositionalization techniques
- Inductive Logic Programming

Rule learning: The standard view

- **Hypothesis construction**: find a set of *n* rules
  - usually simplified by *n* separate rule constructions
    - exception: HYPER
- **Rule construction**: find a pair (Head, Body)
  - e.g. select head (class) and construct body by searching the VersionSpace
    - exceptions: CN2, APRIORI
- **Body construction**: find a set of *m* literals
  - usually simplified by adding one literal at a time
    - problem (ILP): literals introducing new variables

Rule learning revisited

- **Hypothesis construction**: find a set of *n* rules
- **Rule construction**: find a pair (Head, Body)
- **Body construction**: find a set of *m* features
  - Features can be either defined by background knowledge or constructed through constructive induction
  - In propositional learning features may increase expressiveness through negation
  - Every ILP system does constructive induction
- **Feature construction**: find a set of *k* literals
  - finding interesting features is discovery task rather than classification task e.g. interesting subgroups, frequent itemsets
  - excellent results achieved also by feature construction through predictive propositional learning and ILP (Srinivasan)

First-order feature construction

- All the expressiveness of ILP is in the features
- Given a way to construct (or choose) first-order features, body construction in ILP becomes propositional
  - idea: learn non-determinate clauses with LINUS by saturating background knowledge (performing systematic feature construction in a given language bias)

Standard LINUS

- **Example: learning family relationships**

Training examples		Background knowledge	
daughter(sue,eve).	(+)	parent(eve,sue).	female(ann).
daughter(ann,pat).	(+)	parent(ann,tom).	female(sue).
daughter(tom,ann).	(-)	parent(pat,ann).	female(eve).
daughter(eve,ann).	(-)	parent(tom,sue).	

- **Transformation to propositional form:**

Class	Variables		Propositional features						
	X	Y	f(X)	f(Y)	p(X,X)	p(X,Y)	p(Y,X)	p(Y,Y)	X=Y
⊕	sue	eve	true	true	false	false	true	false	false
⊕	ann	pat	true	false	false	false	true	false	false
⊖	tom	ann	false	true	false	false	true	false	false
⊖	eve	ann	true	true	false	false	false	false	false

- **Result of propositional rule learning:**
  Class = ⊕ if (female(X) = true) $\wedge$ (parent(Y,X) = true
- **Transformation to program clause form:**
  daughter(X,Y) $\leftarrow$ female(X),parent(Y,X)

Representation issues (1)

- In the database and Datalog ground fact representations individual examples are not easily separable
- Term and Datalog ground clause representations enable the separation of individuals
- Term representation collects all information about an individual in one structured term

Representation issues (2)

- Term representation provides strong language bias
- Term representation can be flattened to be described by ground facts, using
  - structural predicates (e.g. car(t1,c1), load(c1,l1)) to introduce substructures
  - utility predicates, to define properties of invididuals (e.g. long(t1)) or their parts (e.g., long(c1), circle(l1)).
- This observation can be used as a language bias to construct new features

Declarative bias for first-order feature construction

- In ILP, features involve interactions of local variables
- Features should define properties of individuals (e.g. trains, molecules) or their parts (e.g., cars, atoms)
- Feature construction in LINUS, using the following language bias:
  - one free global variable (denoting an individual, e.g. train)
  - one or more structural predicates: (e.g., has_car(T,C)) ,each introducing a new existential local variable (e.g. car, atom), using either the global variable (train, molecule) or a local variable introduced by other structural predicates (car, load)
  - one or more utility predicates defining properties of individuals or their parts: no new variables, just using variables
  - all variables should be used
  - parameter: max. number of predicates forming a feature

Sample first-order features

- The following rule has two features ‘has a short car’ and ‘has a closed car’:

  eastbound(T):-hasCar(T,C1),clength(C1,short),
     hasCar(T,C2),not croof(C2,none).

- The following rule has one feature ‘has a short closed car’:

  eastbound(T):-hasCar(T,C),clength(C,short),
     not croof(C,none).

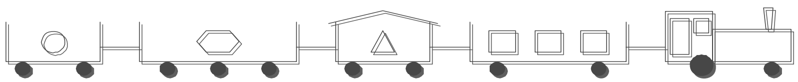
- Equivalent representation:

  eastbound(T):-hasShortCar(T),hasClosedCar(T).

  hasShortCar(T):-hasCar(T,C),clength(C,short).

  hasClosedCar(T):-hasCar(T,C),not croof(C,none).

Propositionalization in a nutshell



Propositionalization task

LOAD	CAR	OBJECT	NUMBER
l1	c1	circle	1
l2	c2	hexagon	1
l3	c3	triangle	1
l4	c4	rectangle	3
…	…	…	

TRAIN_TABLE

TRAIN	EASTBOUND
t1	TRUE
t2	TRUE
…	…
t6	FALSE
…	…

CAR	TRAIN	SHAPE	LENGTH	ROOF	WHEELS
c1	t1	rectangle	short	none	2
c2	t1	rectangle	long	none	3
c3	t1	rectangle	short	peaked	2
c4	t1	rectangle	long	none	2
…	…	…			…

Transform a multi-relational (**multiple-table**) representation to a propositional representation (**single table**)

PROPOSITIONAL TRAIN_TABLE

train(T)	f1(T)	f2(T)	f3(T)	f4(T)	f5(T)
t1	t	t	f	t	t
t2	t	t	t	t	t
t3	f	f	t	f	f
t4	t	f	t	f	f
…	…	…			…

Proposed in ILP systems LINUS (1991), 1BC (1999), ...

Propositionalization in a nutshell

Main propositionalization step: first-order feature construction

f1(T):-hasCar(T,C),clength(C,short).

f2(T):-hasCar(T,C), hasLoad(C,L),
loadShape(L,circle)

f3(T) :- ....

Propositional learning:

t(T) $\leftarrow$ f1(T), f4(T)

Relational interpretation:

eastbound(T) $\leftarrow$
hasShortCar(T),hasClosedCar(T).

LOAD	CAR	OBJECT	NUMBER
l1	c1	circle	1
l2	c2	hexagon	1
l3	c3	triangle	1
l4	c4	rectangle	3
…	…	…	

TRAIN_TABLE

TRAIN	EASTBOUND
t1	TRUE
t2	TRUE
…	…
t6	FALSE
…	…

CAR	TRAIN	SHAPE	LENGTH	ROOF	WHEELS
c1	t1	rectangle	short	none	2
c2	t1	rectangle	long	none	3
c3	t1	rectangle	short	peaked	2
c4	t1	rectangle	long	none	2
…	…	…			…

PROPOSITIONAL TRAIN_TABLE

train(T)	f1(T)	f2(T)	f3(T)	f4(T)	f5(T)
t1	t	t	f	t	t
t2	t	t	t	t	t
t3	f	f	t	f	f
t4	t	f	t	f	f
…	…	…			…

LINUS revisited

- Standard LINUS:
  - transforming an ILP problem to a propositional problem
  - apply background knowledge predicates
- Revisited LINUS:
  - Systematic first-order feature construction in a given language bias
- Too many features?
  - use a relevancy filter (Gamberger and Lavrac)

LINUS revisited: Example: East-West trains

Rules induced by CN2, using 190 first-order features with up to two utility predicates:

eastbound(T):-
hasCarHasLoadSingleTriangle(T),
not hasCarLongJagged(T),
not hasCarLongHasLoadCircle(T).

westbound(T):-
not hasCarEllipse(T),
not hasCarShortFlat(T),
not hasCarPeakedTwo(T).

Meaning:

eastbound(T):-
hasCar(T,C1),hasLoad(C1,L1),lshape(L1,tria),lnumber(L1,1),
not (hasCar(T,C2),clength(C2,long),croof(C2,jagged)),
not (hasCar(T,C3),hasLoad(C3,L3),clength(C3,long),lshape(L3,circ)).

westbound(T):-
not (hasCar(T,C1),cshape(C1,ellipse)),
not (hasCar(T,C2),clength(C2,short),croof(C2,flat)),
not (hasCar(T,C3),croof(C3,peak),cwheels(C3,2)).

Part V: Relational Data Mining

- Learning as search
- What is RDM?
- Propositionalization techniques
- Inductive Logic Programming

ILP as search of program clauses

- An ILP learner can be described by
  - the **structure of the space of clauses**
    - based on the generality relation
    - Let C and D be two clauses.
      C is more general than D ($C \models D$) iff
      $covers(D) \subseteq covers(C)$
    - Example: $p(X,Y) \leftarrow r(Y,X)$ is more general than
      $p(X,Y) \leftarrow r(Y,X), q(X)$
  - its **search strategy**
    - uninformed search (depth-first, breadth-first, iterative deepening)
    - heuristic search (best-first, hill-climbing, beam search)
  - its **heuristics**
    - for directing search
    - for stopping search (quality criterion)

ILP as search of program clauses

- **Semantic generality**
  Hypothesis H_1 is semantically more general than H_2 w.r.t. background theory B if and only if $B \cup H_1 \models H_2$
- **Syntactic generality or θ-subsumption**
  (most popular in ILP)
  - Clause c_1 θ-subsumes c_2 ($c_1 \geq_\theta c_2$)
    if and only if $\exists \theta : c_1\theta \subseteq c_2$
  - Hypothesis $H_1 \geq_\theta H_2$
    if and only if $\forall c_2 \in H_2$ exists $c_1 \in H_1$ such that $c_1 \geq_\theta c_2$
- **Example**

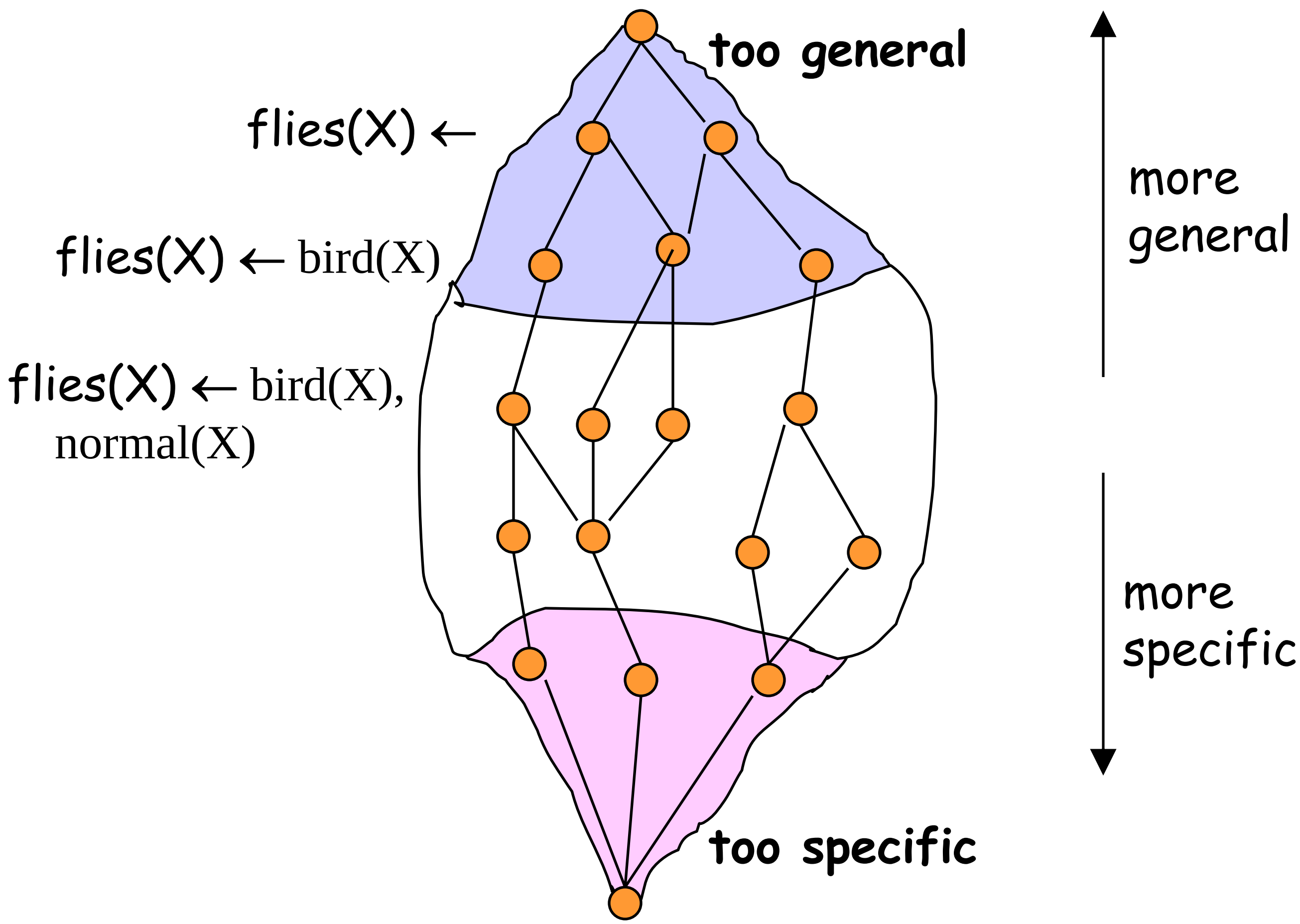
  c1 = daughter(X,Y) ← parent(Y,X)
  c2 = daughter(mary,ann) ← female(mary),
  parent(ann,mary),
  parent(ann,tom).

  c1 θ-subsumes c_2 under θ = {X/mary,Y/ann}

The role of subsumption in ILP

- Generality ordering for hypotheses
- Pruning of the search space:
  - generalization
    - if C covers a neg. example then its generalizations need not be considered
  - specialization
    - if C doesn’t cover a pos. example then its specializations need not be considered
- Top-down search of refinement graphs
- Bottom-up search of the hypo. space by
  - building least general generalizations, and
  - inverting resolutions

Structuring the hypothesis space

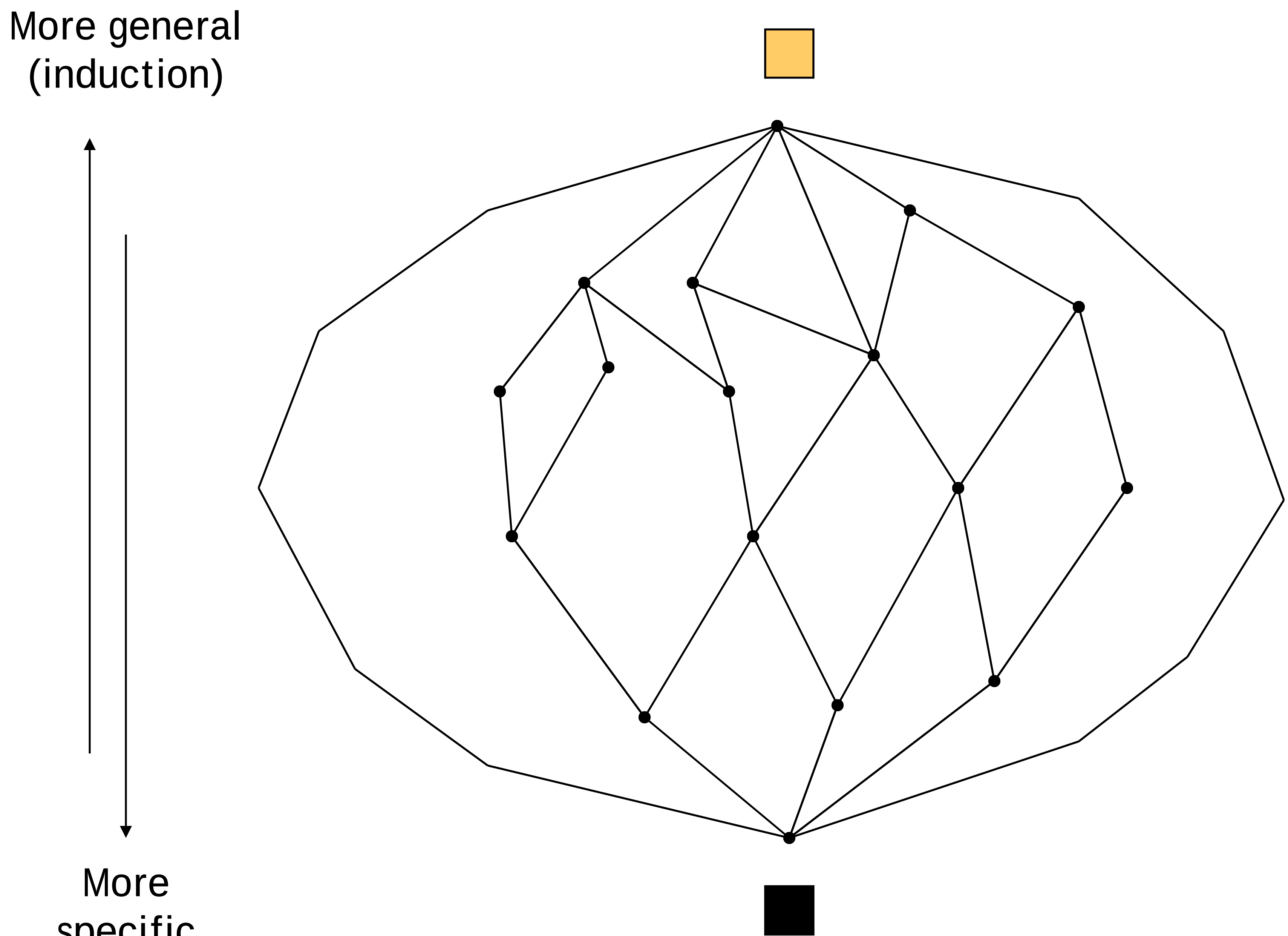


Two strategies for learning

- General-to-specific
  - if Θ-subsumption is used then refinement operators
- Specific-to-general search
  - if Θ-subsumption is used then lgg-operator or generalization operator

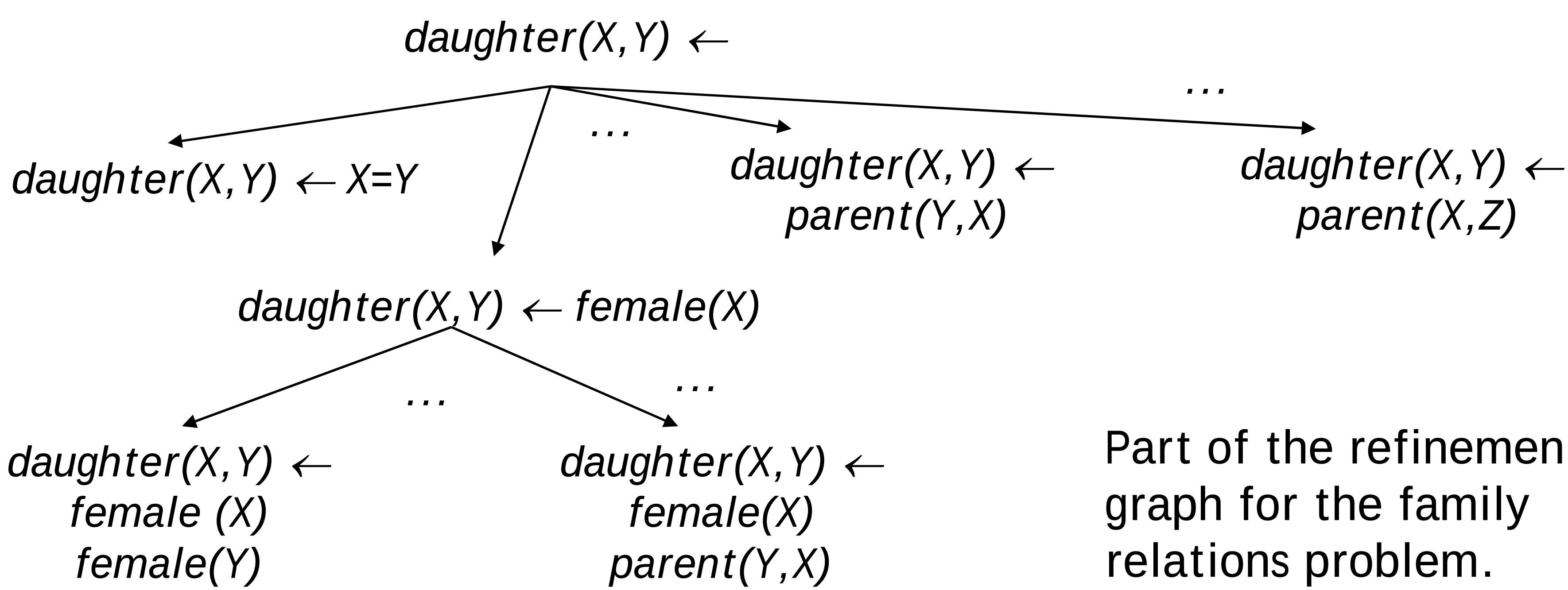
ILP as search of program clauses

- Two strategies for learning
  - Top-down search of refinement graphs
  - Bottom-up search
    - building least general generalizations
    - inverting resolution (CIGOL)
    - inverting entailment (PROGOL)


More general
(induction)
More
specific

Generality ordering of clauses

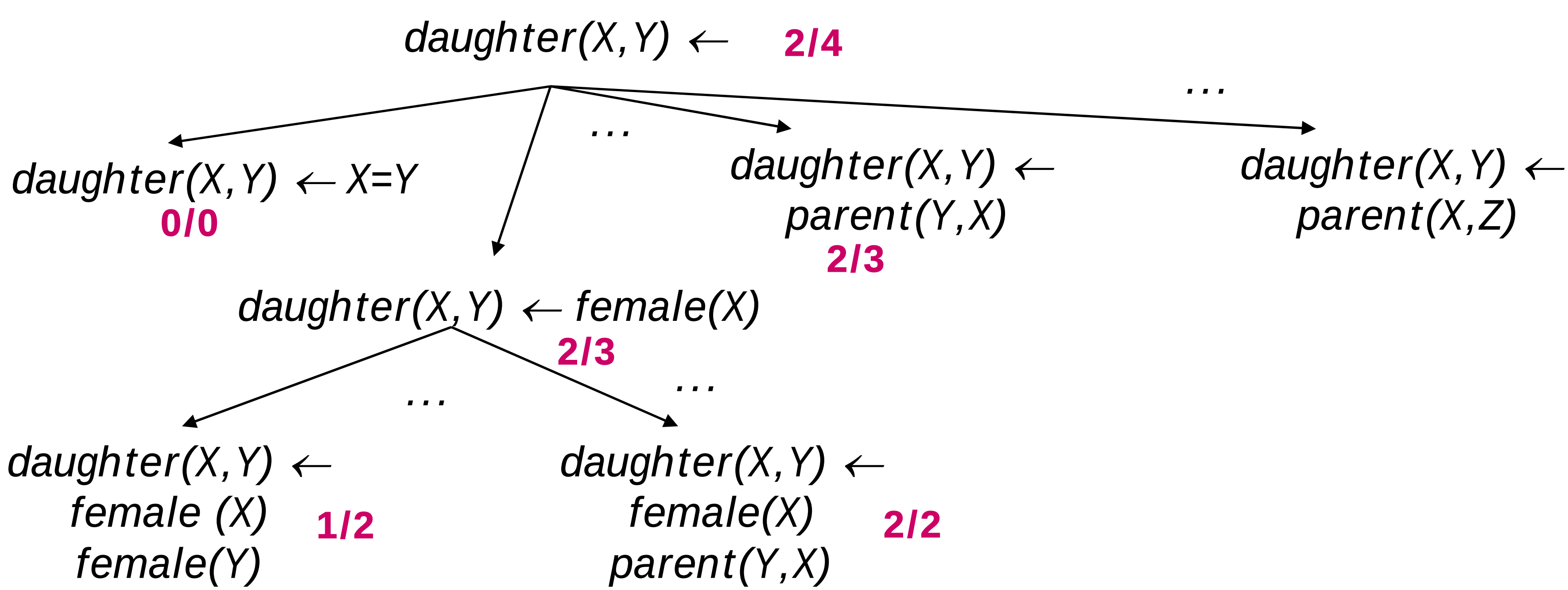
Training examples		Background knowledge	
daughter(mary,ann).	⊕	parent(ann,mary).	female(ann.).
daughter(eve,tom).	⊕	parent(ann,tom).	female(mary).
daughter(tom,ann).	⊖	parent(tom,eve).	female(eve).
daughter(eve,ann).	⊖	parent(tom,ian).	



Part of the refinement graph for the family relations problem.

Greedy search of the best clause

Training examples		Background knowledge	
daughter(mary,ann).	⊕	parent(ann,mary).	female(ann.).
daughter(eve,tom).	⊕	parent(ann,tom).	female(mary).
daughter(tom,ann).	⊖	parent(tom,eve).	female(eve).
daughter(eve,ann).	⊖	parent(tom,ian).	



FOIL

- Language: function-free normal programs recursion, negation, new variables in the body, no functors, no constants (original)
- Algorithm: covering
- Search heuristics: weighted info gain
- Search strategy: hill climbing
- Stopping criterion: encoding length restriction
- Search space reduction: types, in/out modes determinate literals
- Ground background knowledge, extensional coverage
- Implemented in C

Part V: Summary

- RDM extends DM by allowing multiple tables describing structured data
- Complexity of representation and therefore of learning is determined by one-to-many links
- Many RDM problems are individual-centred and therefore allow strong declarative bias