Data Mining and Knowledge Discovery

Part of
"New Media and eScience" MSc Programme
and "Statistics" MSc Programme

2006-2007

Nada Lavrač

Jožef Stefan Institute

Ljubljana, Slovenia

Thanks to Blaz Zupan, Saso Dzeroski and Peter Flach for contributing some slides to this course material

Course participants

I. IPS students

- Fabjan David Aleksander
- Mihajlov Martin
- Fortuna Blaž
- Sergeja Sabo
- Brečko Andraž
- Gašperin Matej
- Raubar Edvin
- Koncilija Jure
- Fortuna Carolina
- Pelko Miha
- Stojanova Danijela
- Taškova Katerina

II. Statistics students

- Miran Juretič
- Andrej Kastrin

III. Other participants

- Ingrid Petrič
- …

IPS Courses - 2006/07
A. Data Mining and Knowledge Discovery
B. Knowledge Management

8 Nov. 06	Data Mining and Knowledge Discovery	Nada Lavrač
15 Nov. 06	Practical work with WEKA	Petra Kralj, Branko Kavšek
29 Nov. 06	15-17 your data presentations 17-19 Know. Management	students, Nada, Petra, Branko
21 Feb. 07	Exam: Presentation of seminar work by students	

Credits and coursework

“New Media and eScience” MSc Programme

- 12 credits (30 hours)
  - lectures
  - hands-on (WEKA)
  - seminar – data analysis using you own data (e.g., using WEKA for survey data analysis)
- contacts:
  - Nada Lavrač nada.lavrac@ijs.si
  - Petra Kralj (MPS student) petra.kralj@gmail.com
  - Branko Kavšek: branko.kavsek@ijs.si

“Statistics” MSc Programme

- 12 credits (36 hours)
- Individual workload
  - same as for MPS students
- contacts:
  - same as for MPS students

Exam

- 29.11.06 Preliminary presentation of your problem/dataset (max. 6 slides)
- 21.2.07 data analysis results (max. 12 slides, report, presentation and report following the CRISP-DM methodology)

Course Outline

I. Introduction

- Data Mining and KDD process
- Examples of discovered patterns and applications
- Data mining tools and visualization

(Ch. 1,2,11,12,13 of DM&DS book)

II. DM Techniques

- Classification of DM tasks and techniques
- Predictive DM
  - Decision Tree induction (Ch. 3 of Mitchell's book)
  - Learning sets of rules (Ch. 7 of IDA book, Ch. 10 of Mitchell's book)
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

III. Evaluation

- Evaluation methodology
- Evaluation measures

IV. Relational Data Mining

- What is RDM?
- Propositionalization
- Inductive Logic Programming

(Ch. 3,4,11 of RDM book)

V. Conclusions and literature

Part I. Introduction

- Data Mining and the KDD process
- Examples of discovered patterns and applications
- Data mining tools and visualization

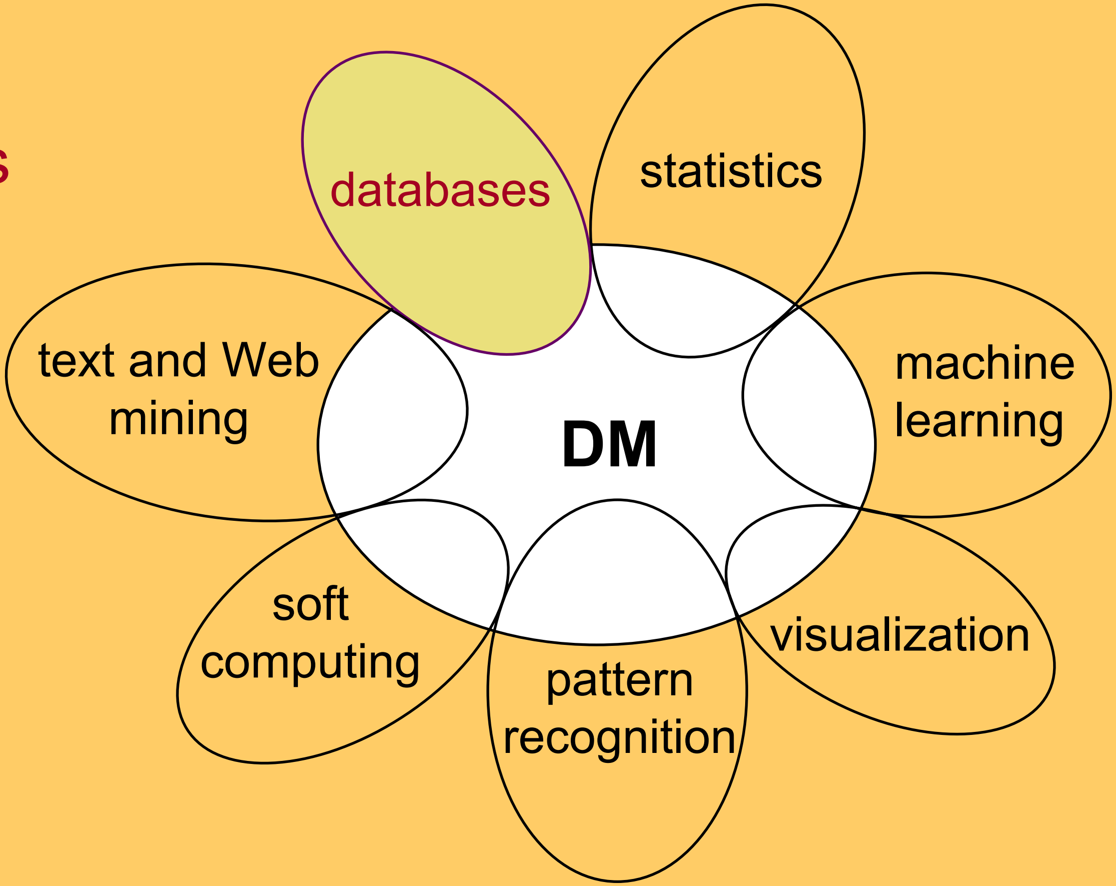
What is DM

- 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 “hard” real-life problems

Related areas

Database technology
and data warehouses

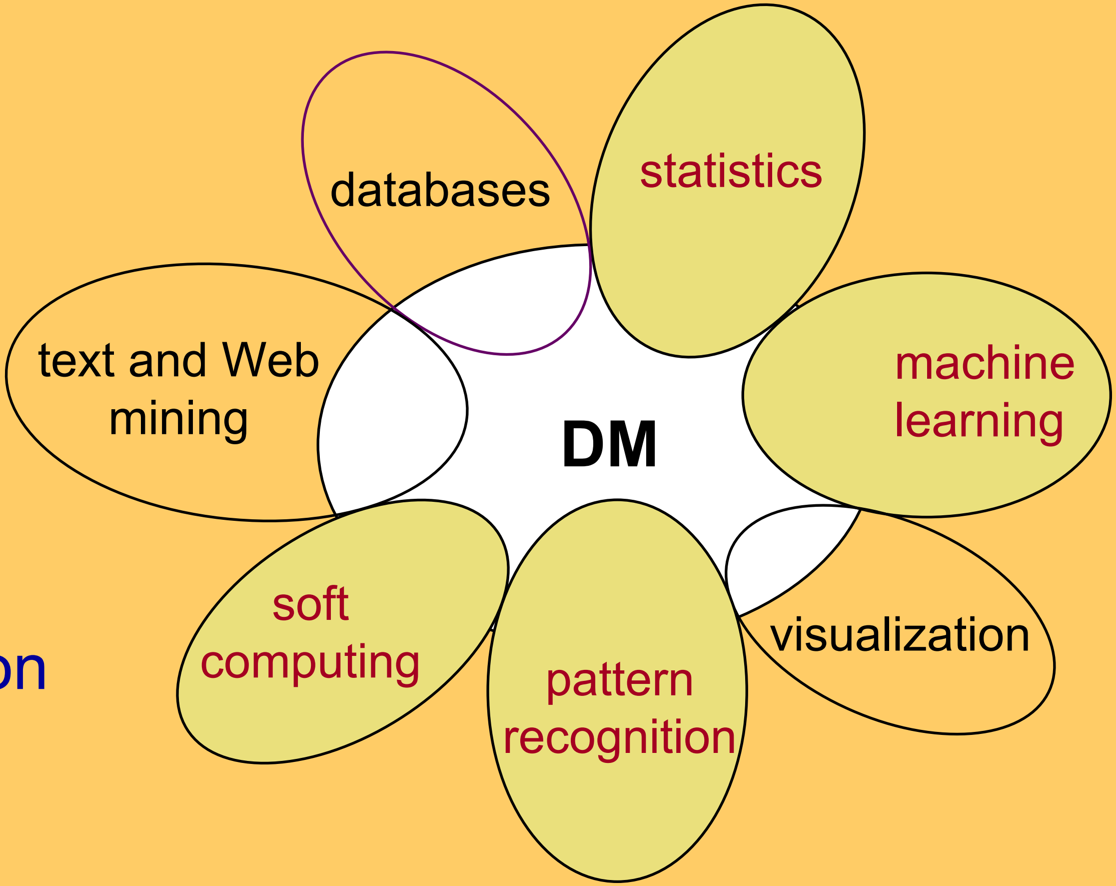
- efficient storage, access and manipulation of data



Related areas

Statistics,
machine learning,
pattern recognition
and soft computing*

- classification techniques and techniques for knowledge extraction from data

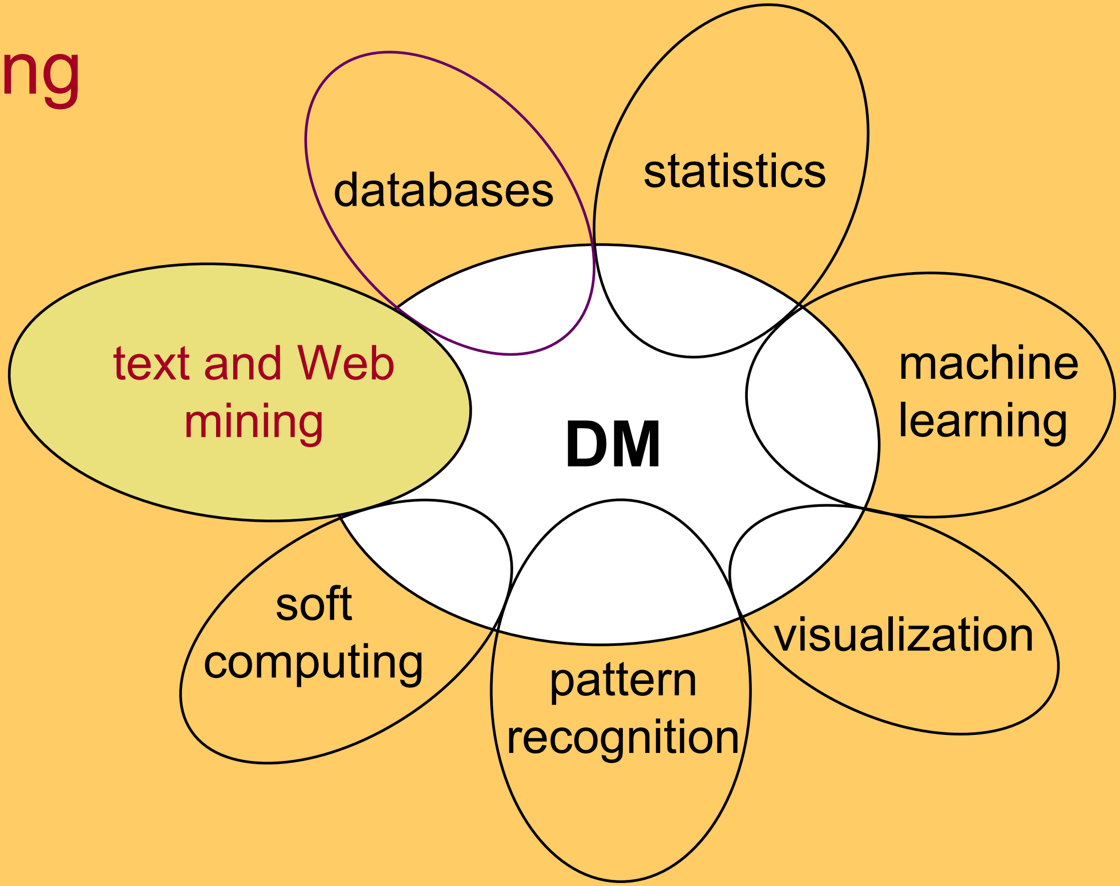


* neural networks, fuzzy logic, genetic algorithms, probabilistic reasoning

Related areas

Text and Web mining

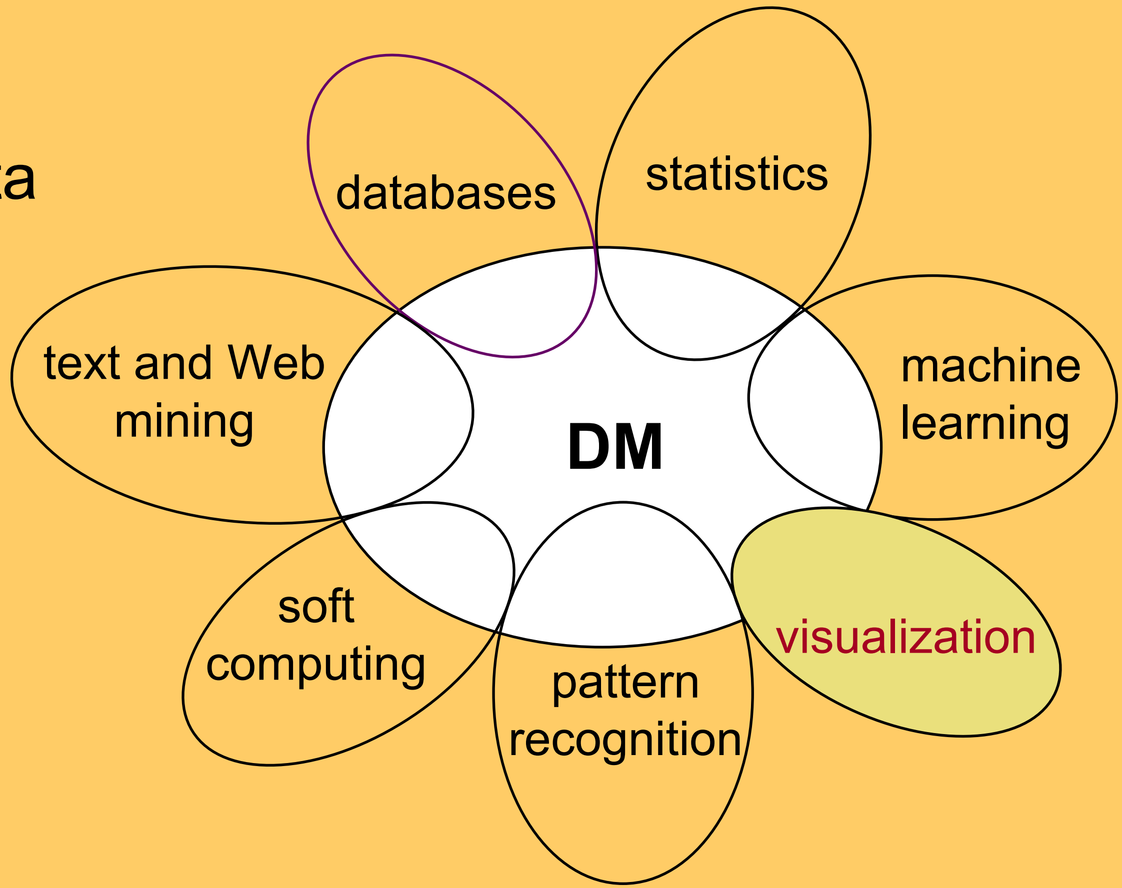
- Web page analysis
- text categorization
- acquisition, filtering and structuring of textual information
- natural language processing



Related areas

Visualization

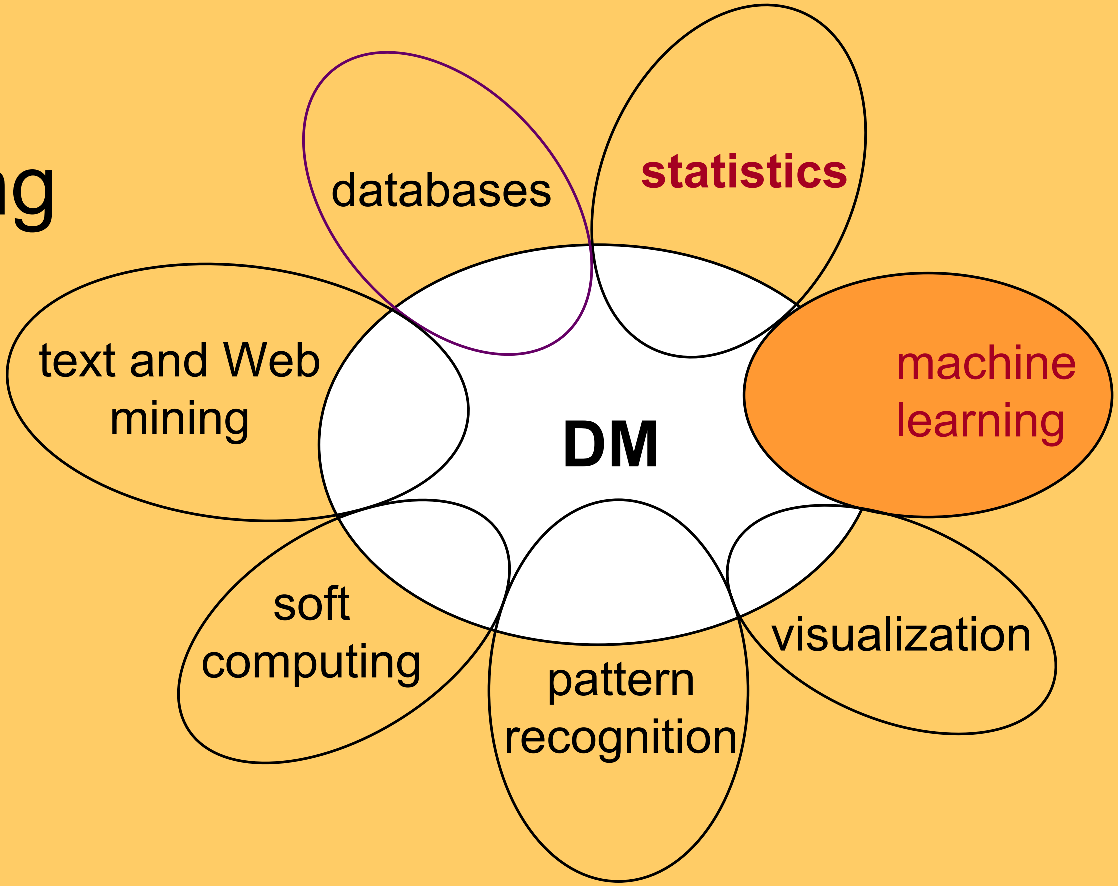
- visualization of data and discovered knowledge



Point of view in this tutorial

Knowledge discovery using machine learning methods

Relation with statistics



Machine Learning and Statistics

- Both areas have a long tradition of developing inductive techniques for data analysis.
  - reasoning from properties of a data sample to properties of a population
- KDD = statistics + marketing ? No !
- KDD = statistics + ... + machine learning
- Statistics is particularly appropriate for hypothesis testing and data analysis when certain theoretical expectations about the data distribution, independence, random sampling, sample size, etc. are satisfied
- ML is particularly appropriate when requiring generalizations that consist of easily understandable patterns, induced both from small and large data samples

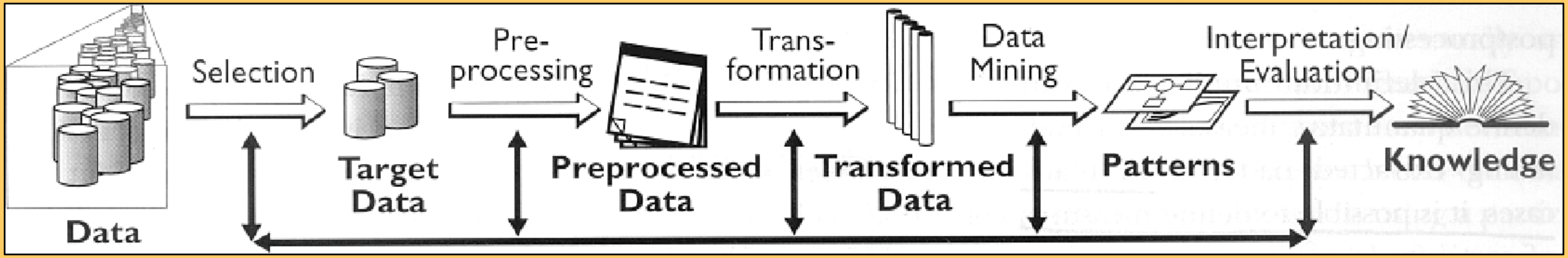
Data Mining and KDD

- Data Mining (DM) is a way of doing data analysis, aimed at finding patterns, revealing hidden regularities and relationships in the data.
- Knowledge Discovery in Databases (KDD) provides a broader view: providing tools to automate the entire process of data analysis, including statistician's art of hypothesis selection
- DM is the key element in this much more elaborate KDD process
- KDD is defined as "the process of identifying valid, novel, potentially useful and ultimately understandable patterns in 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: overall process of discovering useful knowledge from data



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

Part I. Introduction

- Data Mining and the KDD process
- Examples of discovered patterns and applications
- Data mining tools and visualization

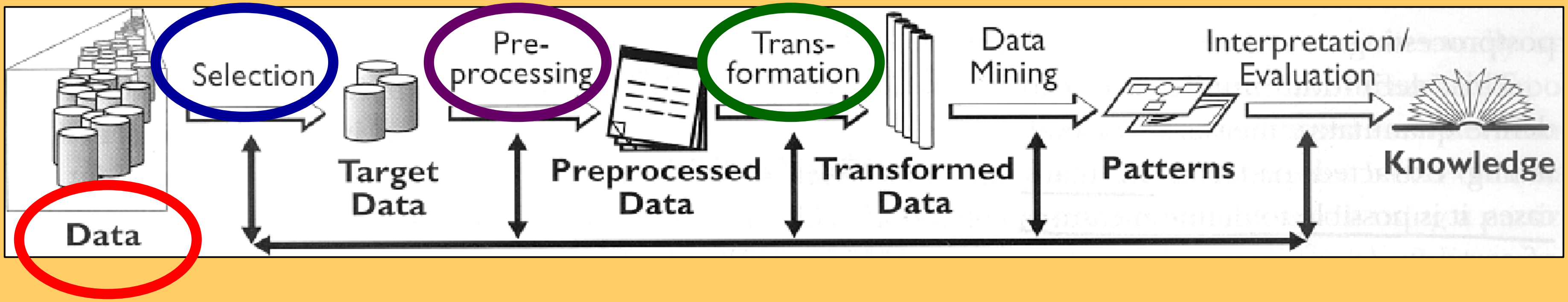
The SolEuNet Project

- European 5FP project "Data Mining and Decision Support for Business Competitiveness: A European Virtual Enterprise", 2000-2003
- Scientific coordinator IJS, administrative FhG
- 3 MEuro, 12 partners (8 academic and 4 business) from 7 countries
- main project objectives:
  - development of prototype solutions for end-users
  - foundation of a virtual enterprise for marketing DM and DS expertise, involving business and academia

Developed Data Mining application prototypes

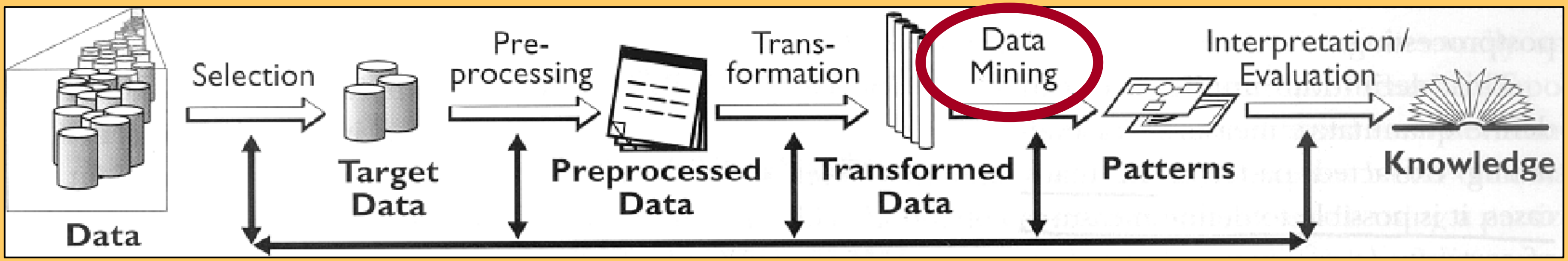
- Mediana – analysis of media research data
- Kline & Kline – improved brand name recognition
- Australian financial house – customer quality evaluation, stock market prediction
- Czech health farm – predict the use of resources
- UK County Council - analysis of traffic accident data
- INE Port. statistical bureau – Web page access analysis for better INE Web page organization
- Coronary heart disease risk group detection
- Online Dating – understanding email dating promiscuity
- EC Harris - analysis of building construction projects
- European Commission - analysis of 5th Fr. IST projects: better understanding of large amounts of text documents, and “clique” identification

MEDIANA - KDD process



- 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)

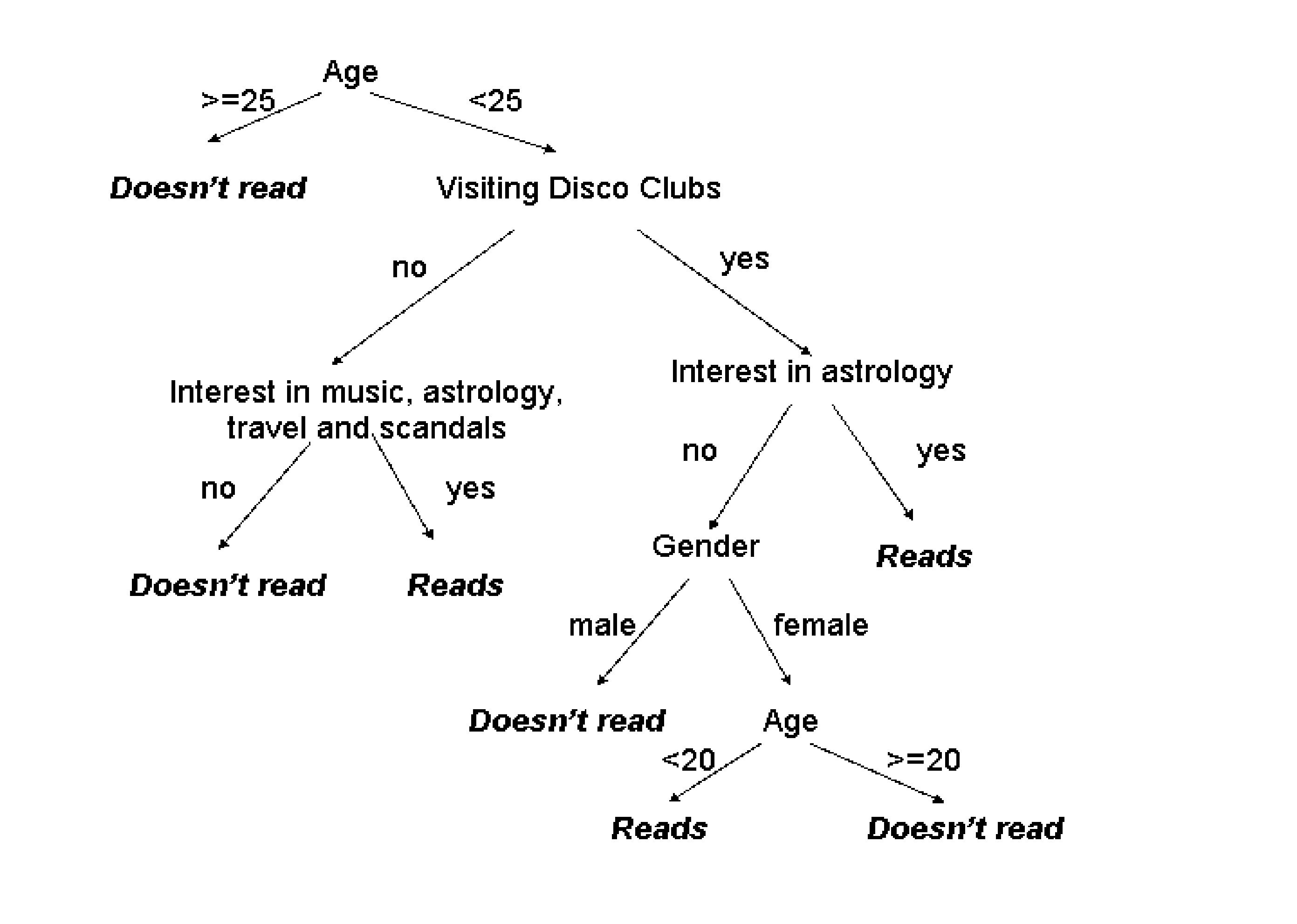
MEDIANA - Pilot study



- 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)

Decision trees

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



Classification rules

Set of Rules: **if Cond then Class**

Interpretation: **if-then** ruleset, or

if-then-else decision list

Class: Reading of daily newspaper EN (Evening News)

if a person does not read MM (Maribor Magazine) and rarely reads the weekly magazine “7Days”

then the person does not read EN (Evening News)

else if a person rarely reads MM and does not read the weekly magazine SN (Sunday News)

then the person reads EN

else if a person rarely reads MM

then the person does not read EN

else the person reads EN.

Association rules

Rules X => Y, X, Y conjunction of bin. attributes

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

Task: Find all association rules that satisfy minimum support and minimum confidence constraints.

Example association rule about readers of yellow press daily newspaper SloN (Slovenian News):

read_Love_Stories_Magazine => read_SloN

sup = 3.5% (3.5% of the whole dataset population reads both LSM and SloN)

conf = 61% (61% of those reading LSM also read SloN)

Association rules

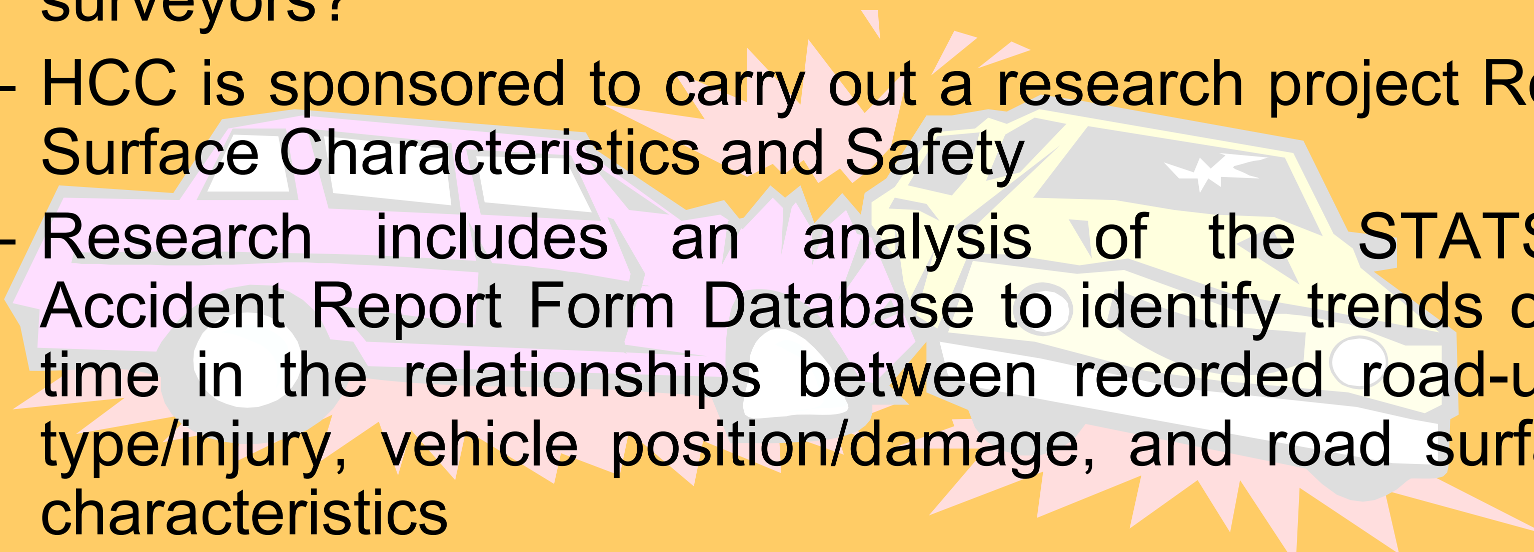
Finding profiles of readers of the Delo daily newspaper

1. read_Marketing magazine 116 **=>** read_Delo 95 (0.82)
2. read_Financial_News 223 **=>** read_Delo 180 (0.81)
3. read_Views 201 **=>** read_Delo 157 (0.78)
4. read_Money 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.

Analysis of UK traffic accidents

- End-user: Hampshire County Council (HCC, UK)
  - Can records of road traffic accidents be analysed to produce road safety information valuable to county surveyors?
  - HCC is sponsored to carry out a research project Road Surface Characteristics and Safety
  - Research includes an analysis of the STATS19 Accident Report Form Database to identify trends over time in the relationships between recorded road-user type/injury, vehicle position/damage, and road surface characteristics



STATS19 Data Base

- Over 5 million accidents recorded in 1979-1999
- 3 data tables

Vehicle VEH7999
(~9 mil. Vehicles,
24 variables)

Which vehicles ? What movement ? Which consequences ?

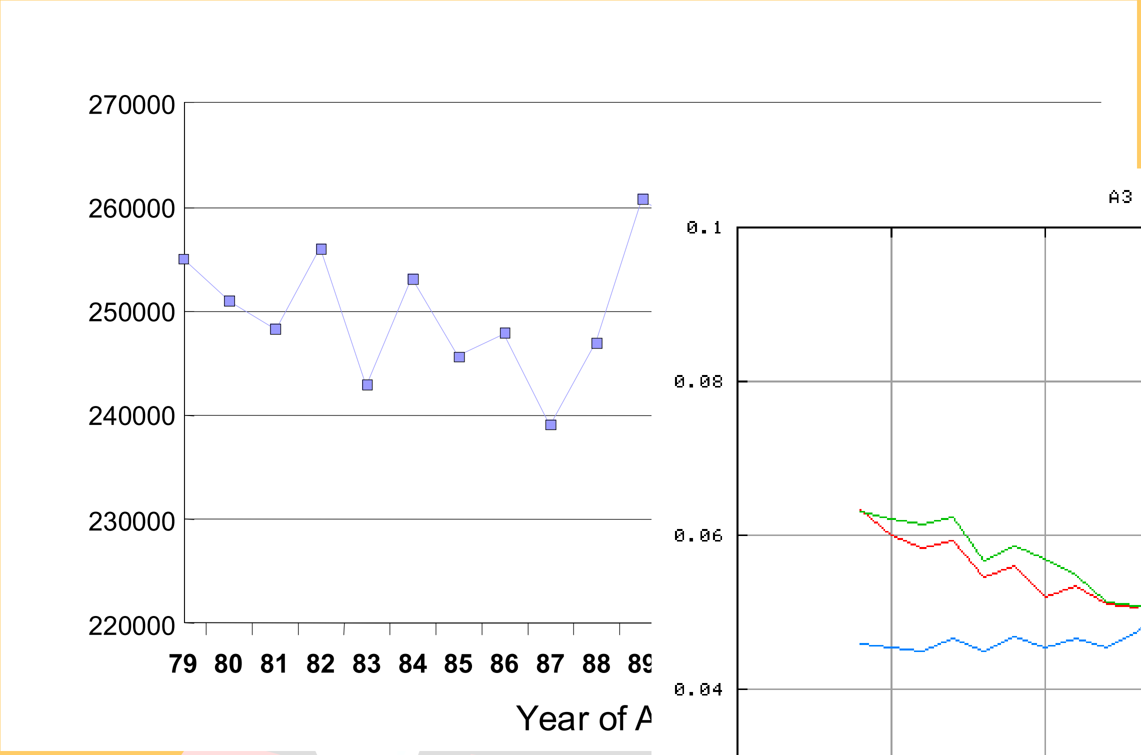
Accident ACC7999
(~5 mil. Accidents,
30 variables)

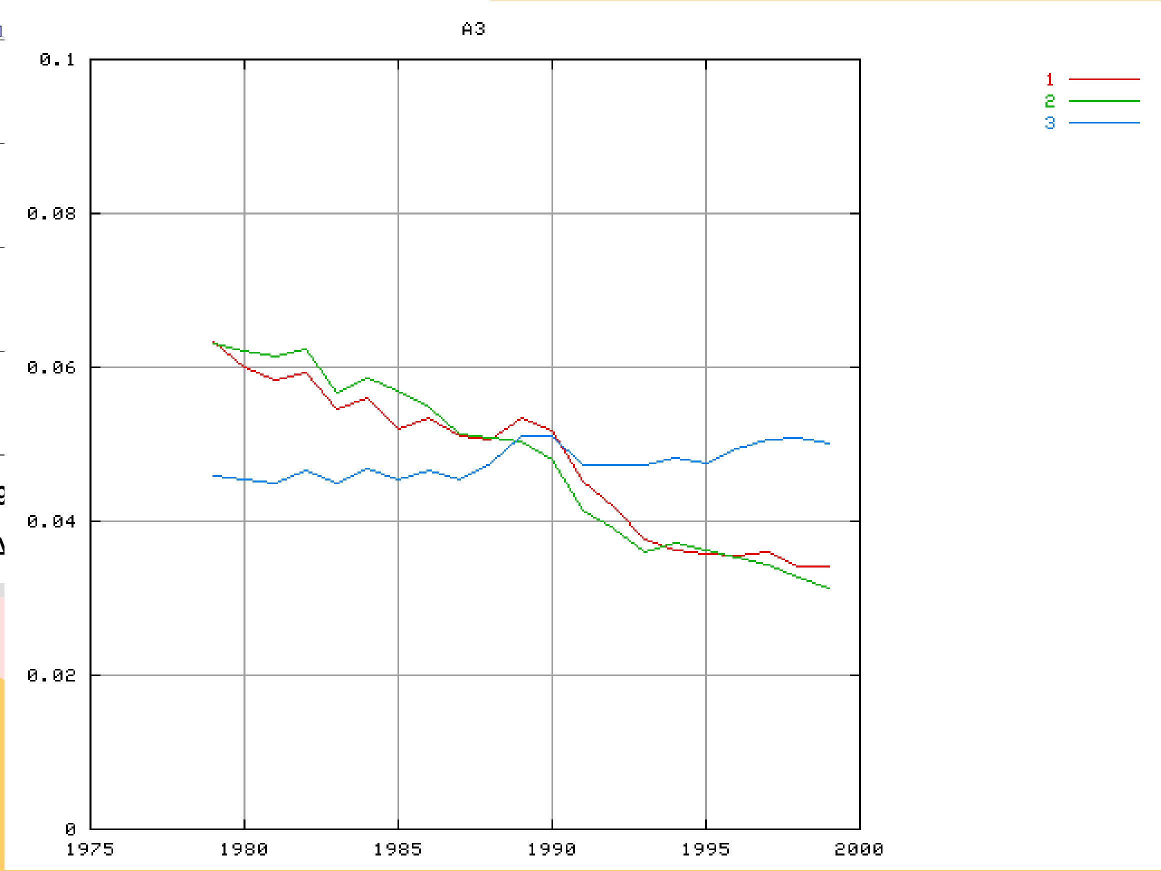
Where ? When ?
How many ?

Casualty CAS7999
(~7 mil.injuries,
16 variables)

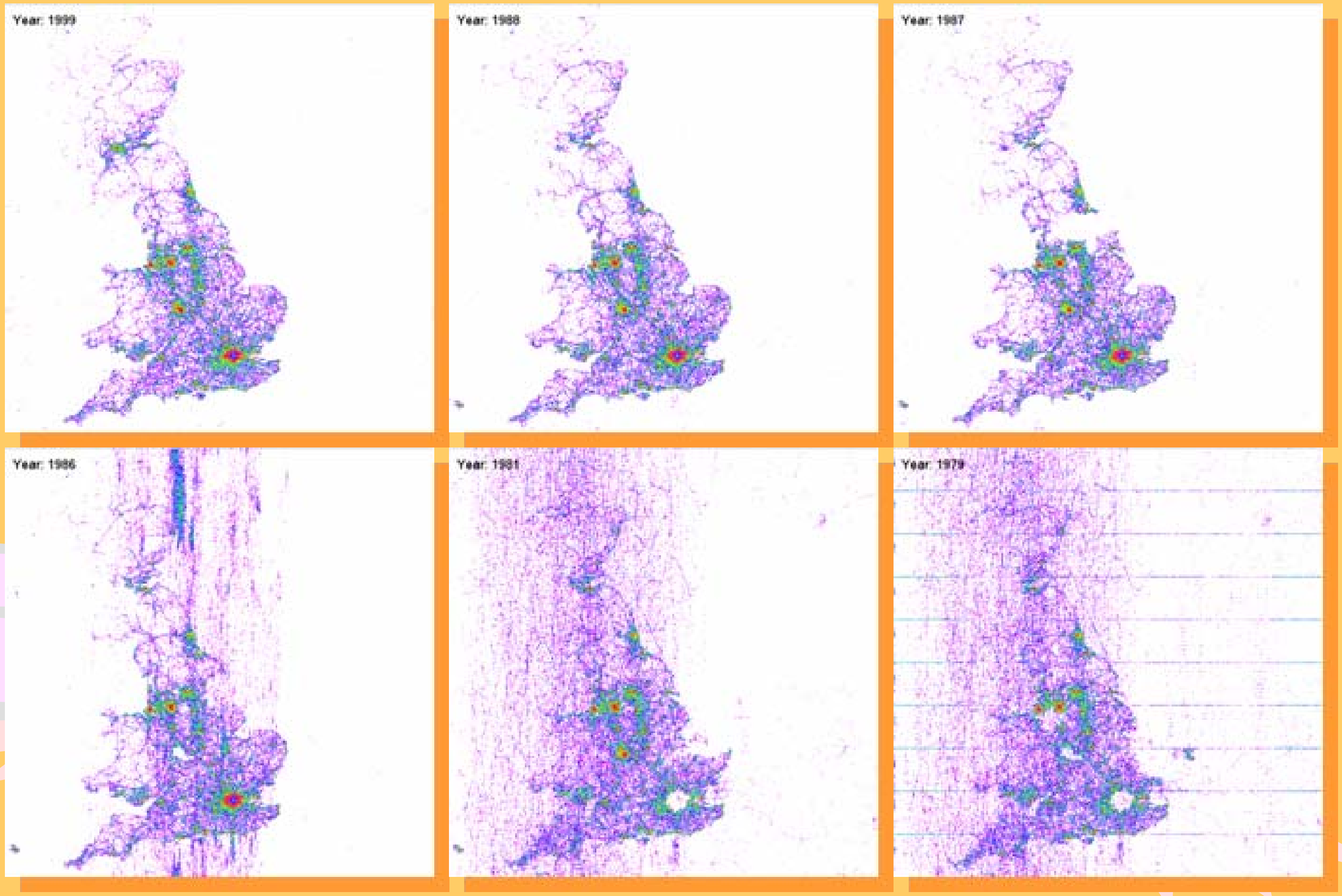
Who was injured ?
What injuries ? ...

Data understanding





Data quality: Accident location



Data preparation

- There are 51 police force areas in UK
- For each area we count the number of accidents in each:
  - Year
  - Month
  - Day of Week
  - Hour of Day

Data preparation

YEAR																					
pfc	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
a	10023	9431	9314	8965	8655	9014	9481	9069	8705	8829	9399	9229	8738	8199	7453	7613	7602	7042	7381	7362	6905
b	6827	6895	6952	7032	6778	6944	6387	6440	6141	5924	6331	6233	5950	6185	5910	6161	5814	6263	5881	5855	5780
c	2409	2315	2258	2286	2022	2169	2212	2096	1989	1917	2137	2072	2032	1961	1653	1526	1552	1448	1521	1408	1234

MONTH												
pfc	jan	feb	mar	apr	may	jun	jul	aug	sep	oct	nov	dec
a	72493	67250	77434	73841	78813	78597	80349	74226	79362	85675	84800	76282
b	2941	2771	3145	3317	3557	3668	3988	4048	3822	3794	3603	3481
c	9261	8574	9651	9887	10649	10590	10813	11299	10810	11614	10884	10306

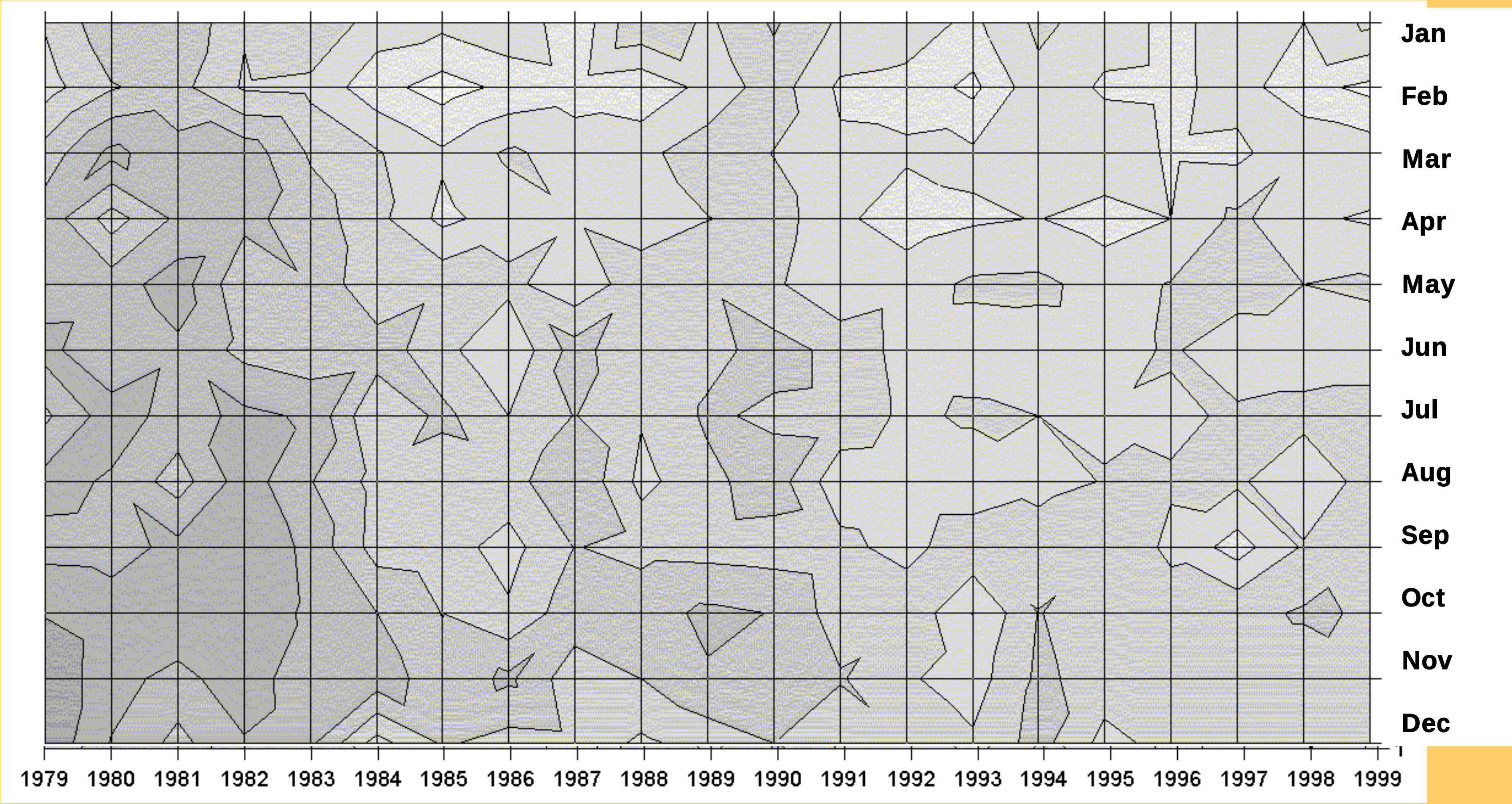
DAY OF WEEK							
12	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
a	96666	132845	137102	138197	142662	155752	125898
b	5526	5741	5502	5679	6103	7074	6510
c	15350	17131	16915	17116	18282	21000	18544

HOUR																		
pfc	0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	23
a	794	626	494	242	166	292	501	1451	2284	...	3851	3538	2557	2375	1786	1394	1302	1415
b	2186	1567	1477	649	370	521	1004	4099	7655	...	11500	11140	7720	7129	5445	4396	3946	4777
c	2468	1540	1714	811	401	399	888	3577	8304	...	12112	12259	8701	7825	6216	4809	4027	4821

Simple visualization of short time series

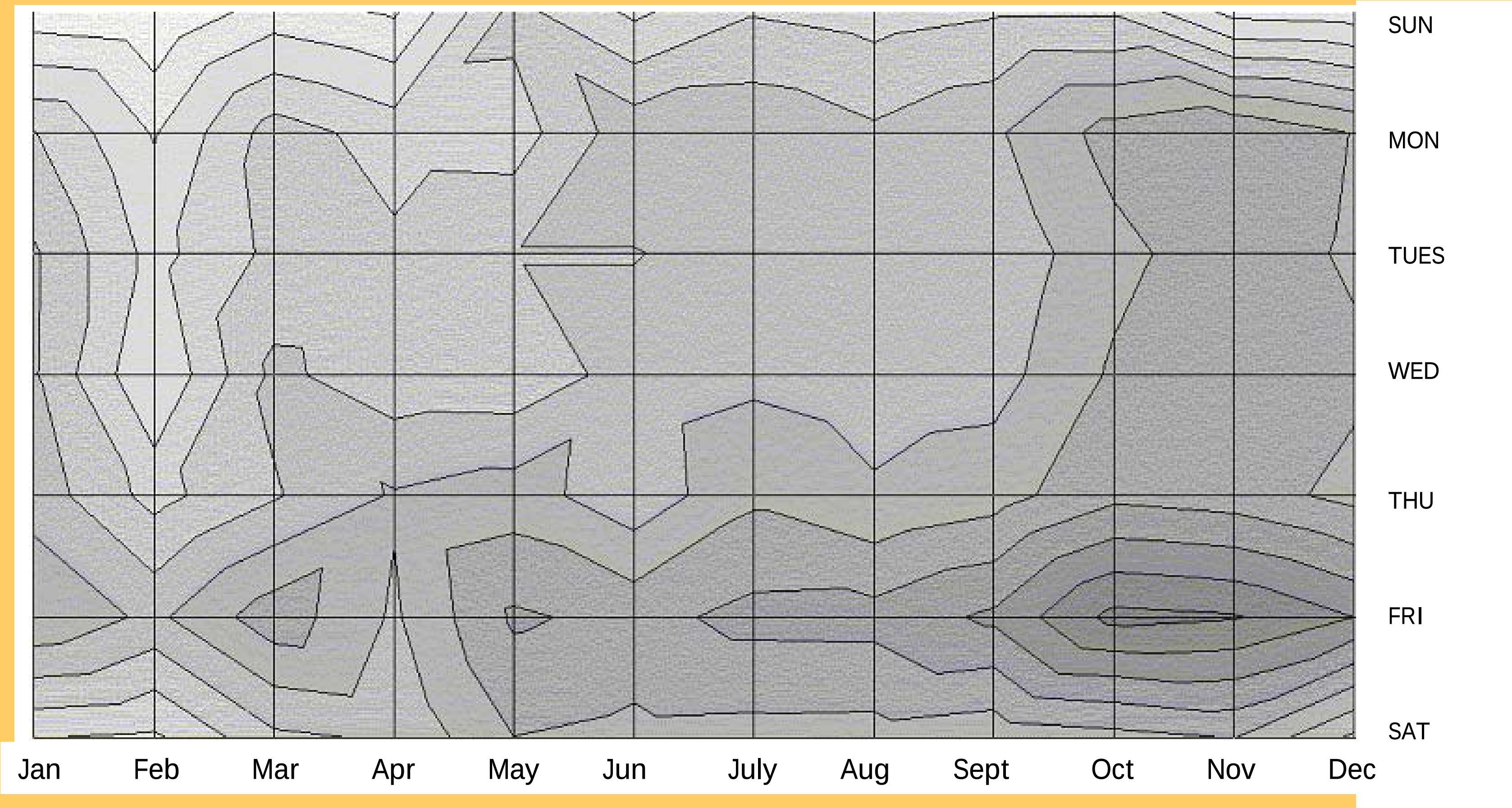
- Used for data understanding
- Very informative and easy to understand format
- UK traffic accident analysis: Distributions of number of accidents over different time periods (year, month, day of week, and hour)

Year/Month distribution



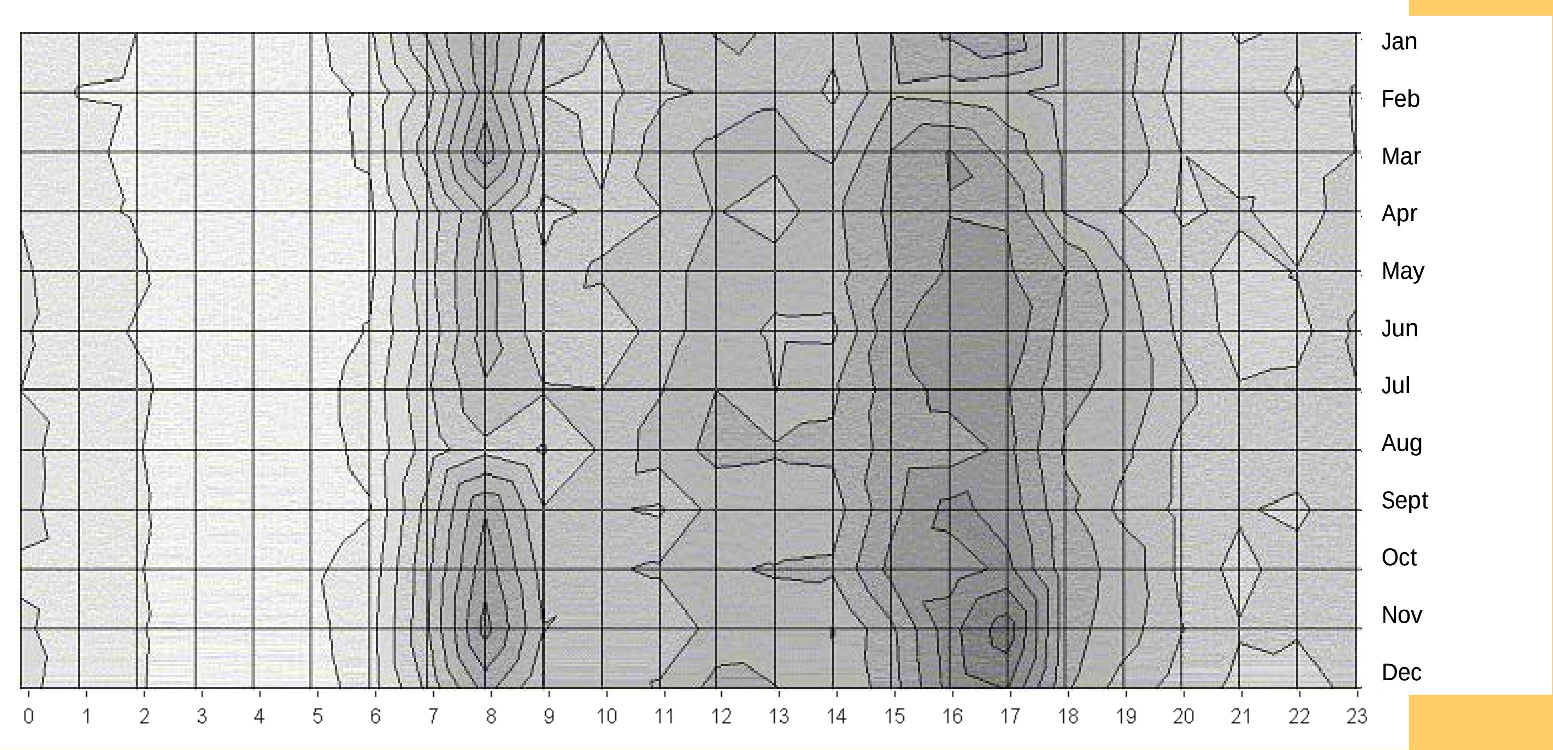
Darker color - MORE accidents

Day of Week/Month distribution



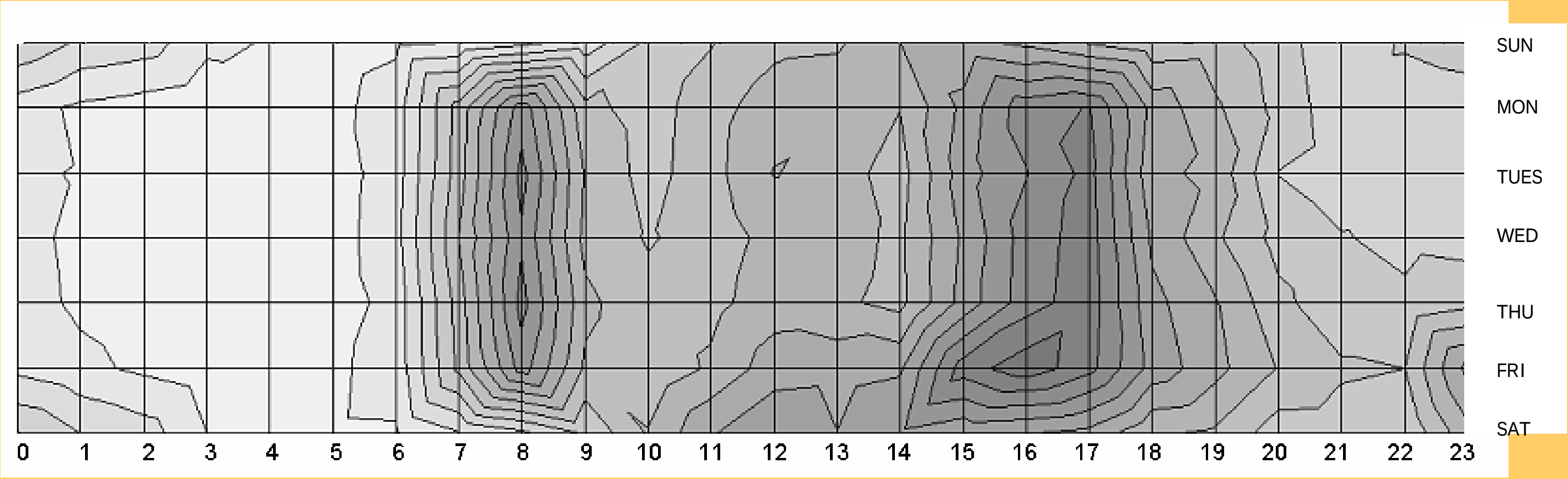
All weekdays (Mon – Fri) are worse in deep winter, Friday the worst

Hour/Month distribution



1. More Accidents at “Rush Hour”, Afternoon Rush hour is the worst
2. More holiday traffic (less rush hour) in August

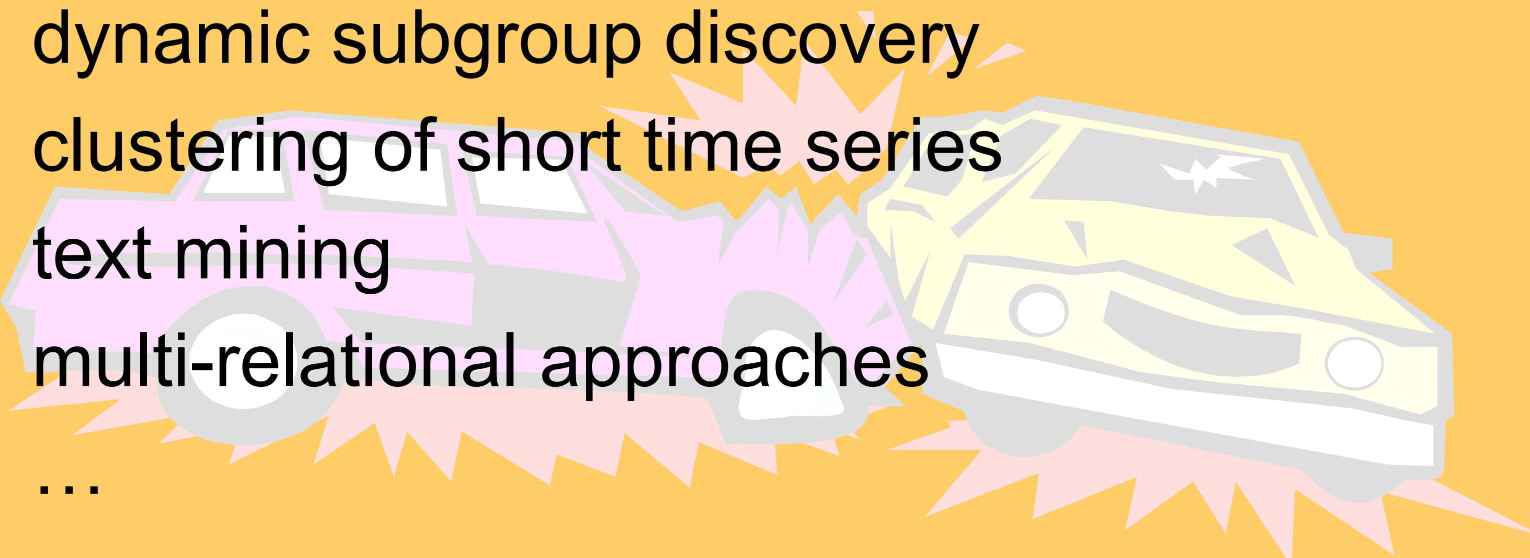
Day of Week/Hour distribution



1. More Accidents at “Rush Hour”, Afternoon Rush hour is the worst and lasts longer with “early finish” on Fridays
2. More leisure traffic on Saturday/Sunday

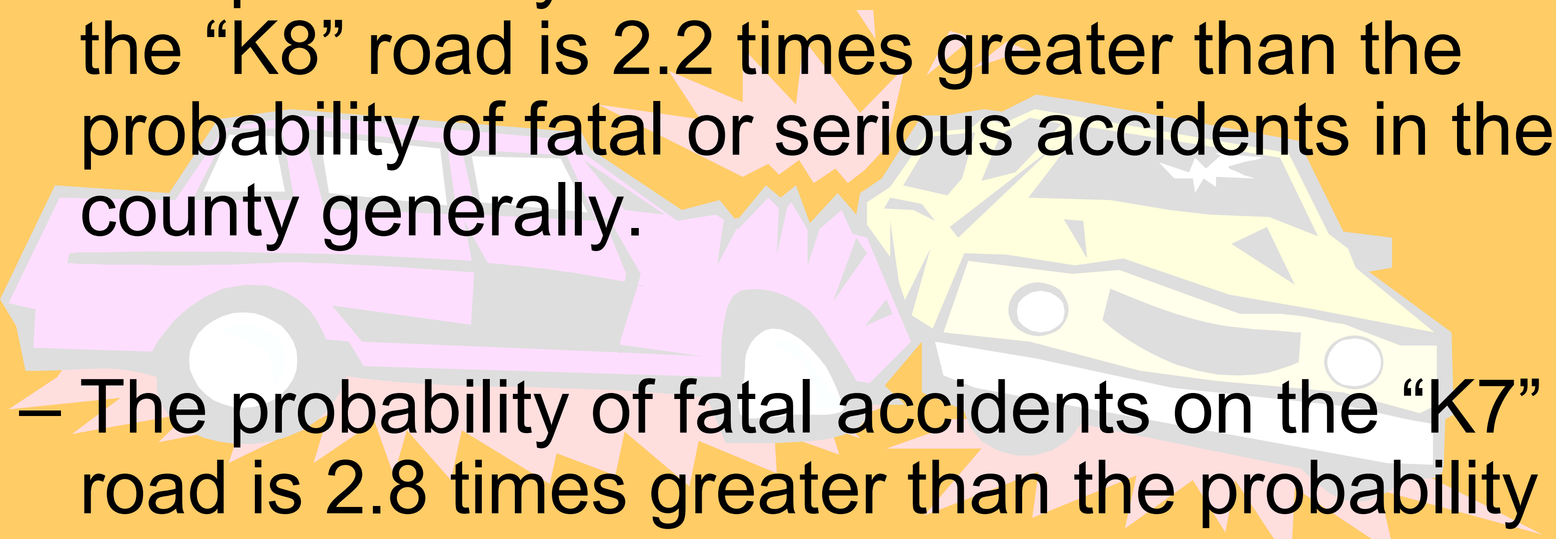
Traffic: different modeling approaches

- association rule learning
- static subgroup discovery
- dynamic subgroup discovery
- clustering of short time series
- text mining
- multi-relational approaches
- …



Some discovered association rules

- Association rules: Road number and Severity of accident
  - The probability of a fatal or serious accident on the “K8” road is 2.2 times greater than the probability of fatal or serious accidents in the county generally.
  - The probability of fatal accidents on the “K7” road is 2.8 times greater than the probability of fatal accidents in the county generally (when the road is dry and the speed limit = 70).



Analysis of documents of European IST project

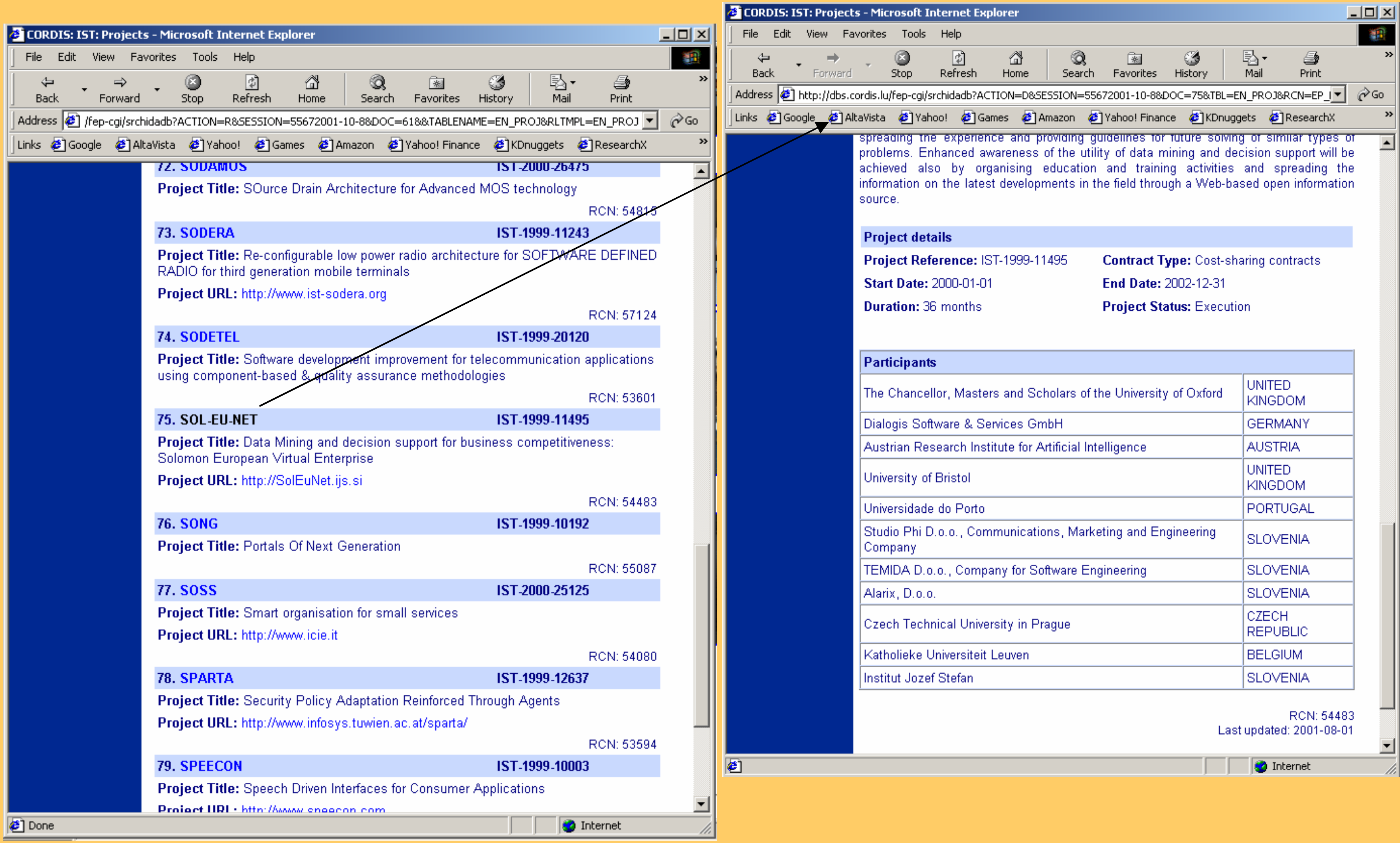
Data source:

- List of IST project descriptions as 1-2 page text summaries from the Web (database www.cordis.lu/)
- IST 5FP has 2786 projects in which participate 7886 organizations

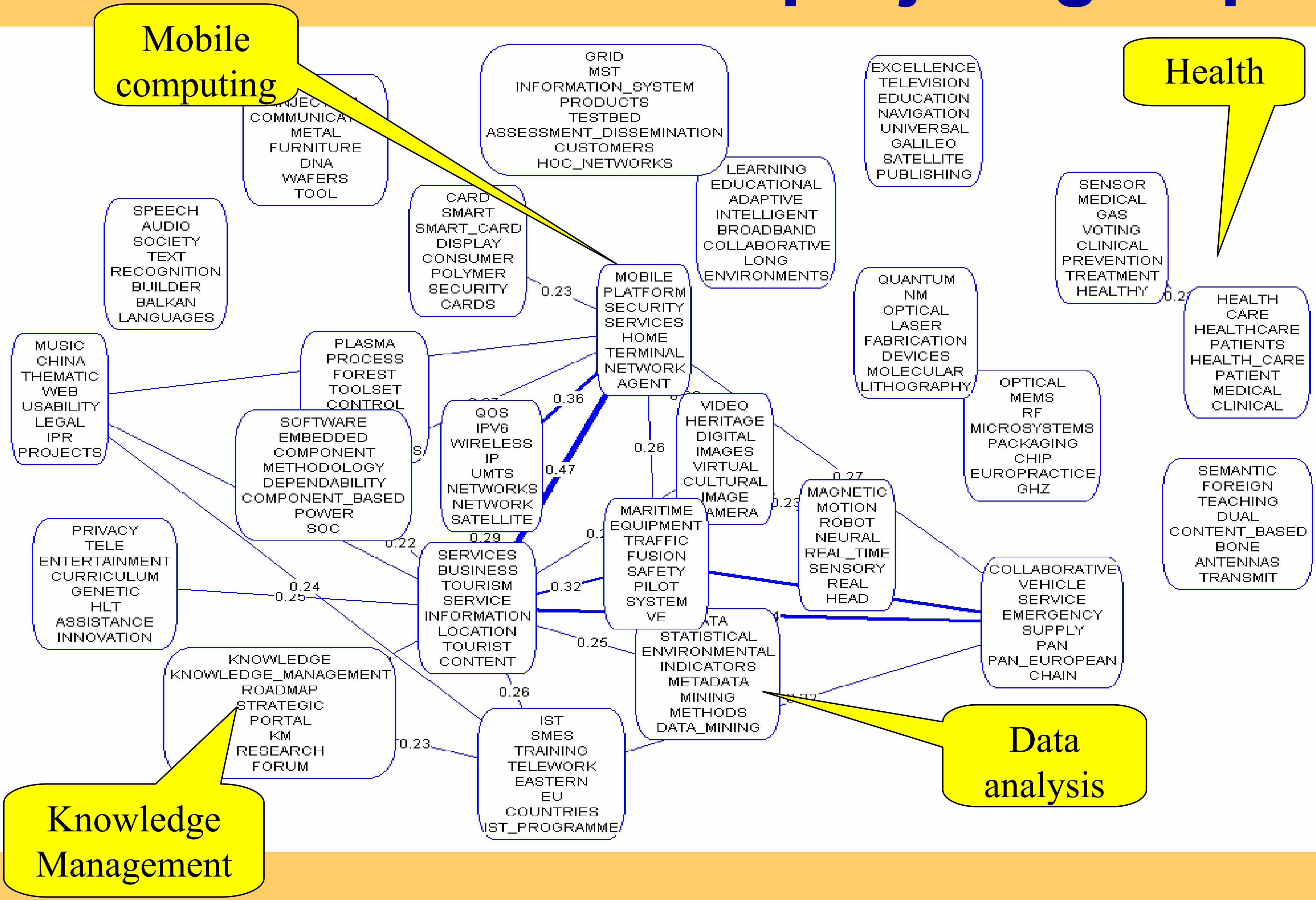
Analysis tasks:

- Visualization of project topics
- Analysis of collaboration
- Connectedness between organizations
- Community/clique identification
- Thematic consortia identification
- Simulation of 6FP IST

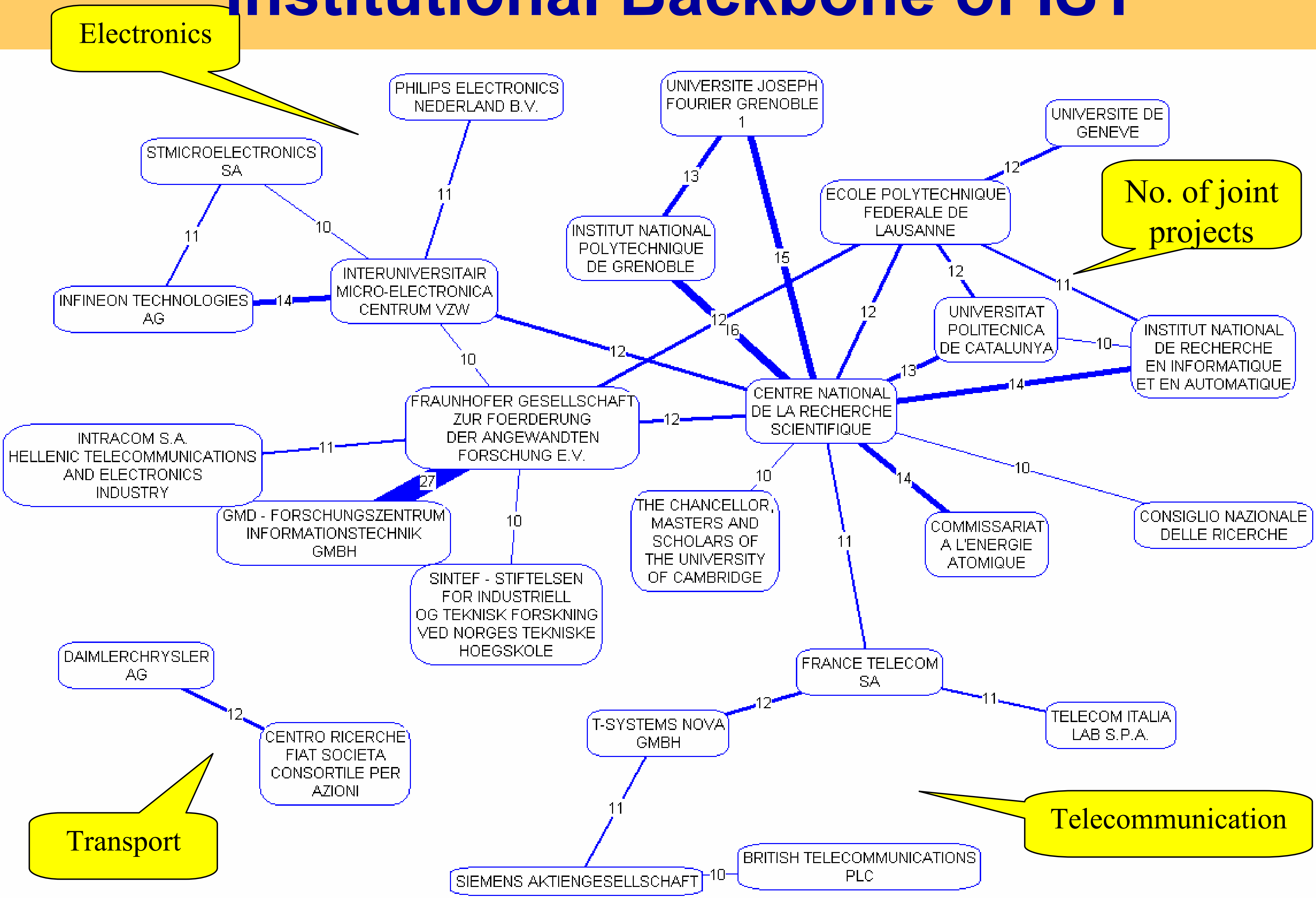
Analysis of documents of European IST project



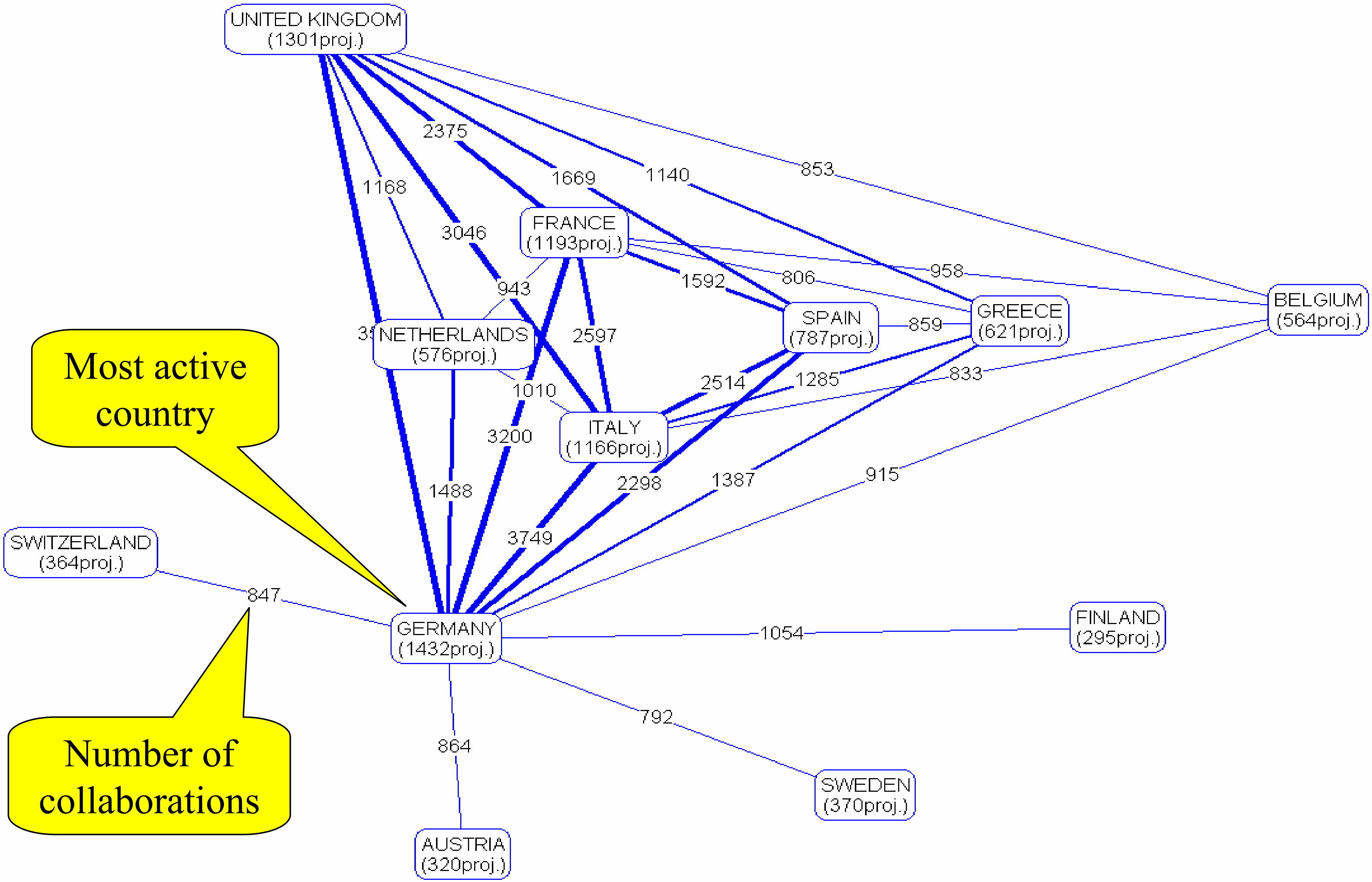
Visualization into 25 project groups



Institutional Backbone of IST



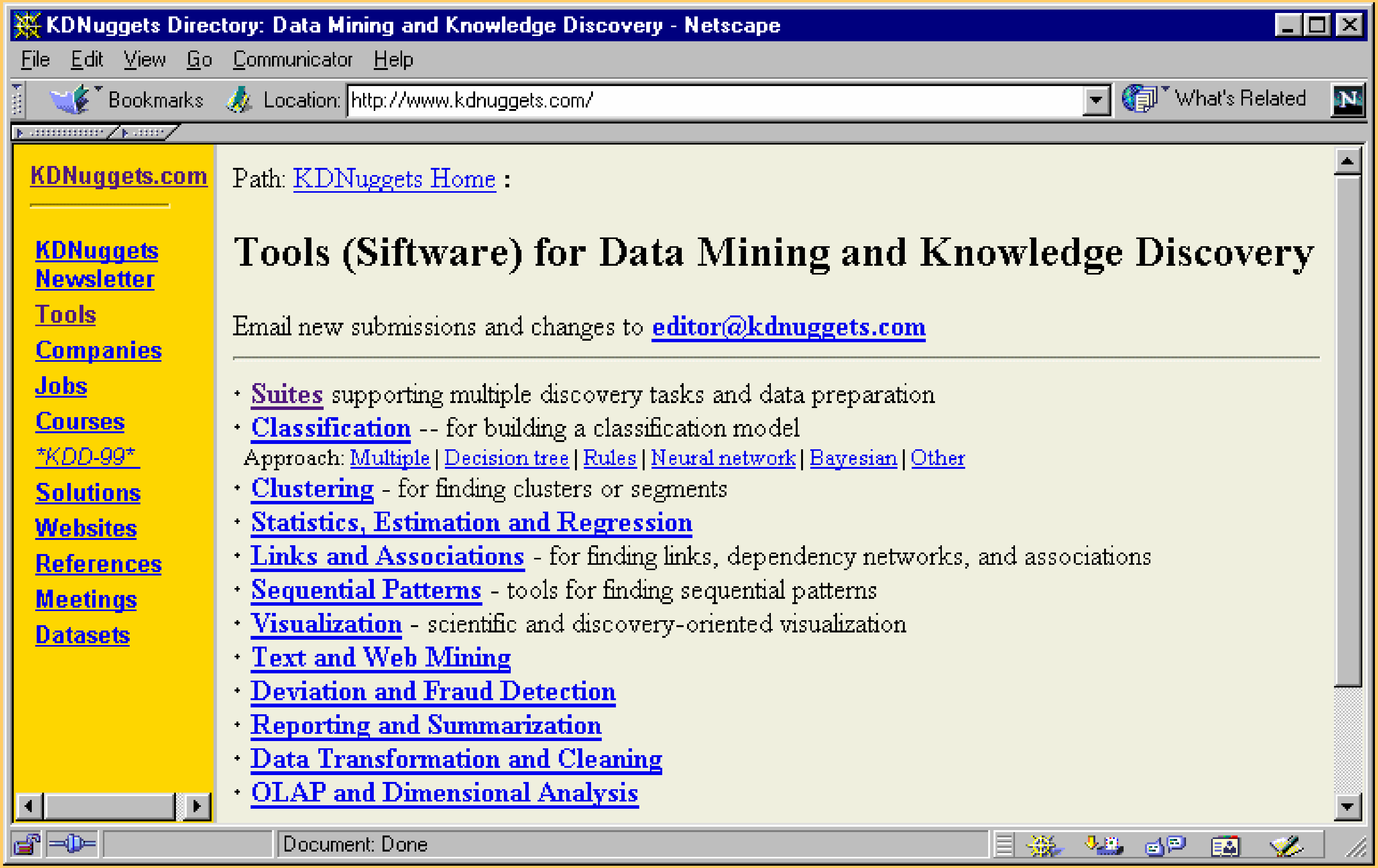
Collaboration between countries (top 12)



Part I. Introduction

- Data Mining and the KDD process
- Examples of discovered patterns and applications
- Data mining tools and visualization

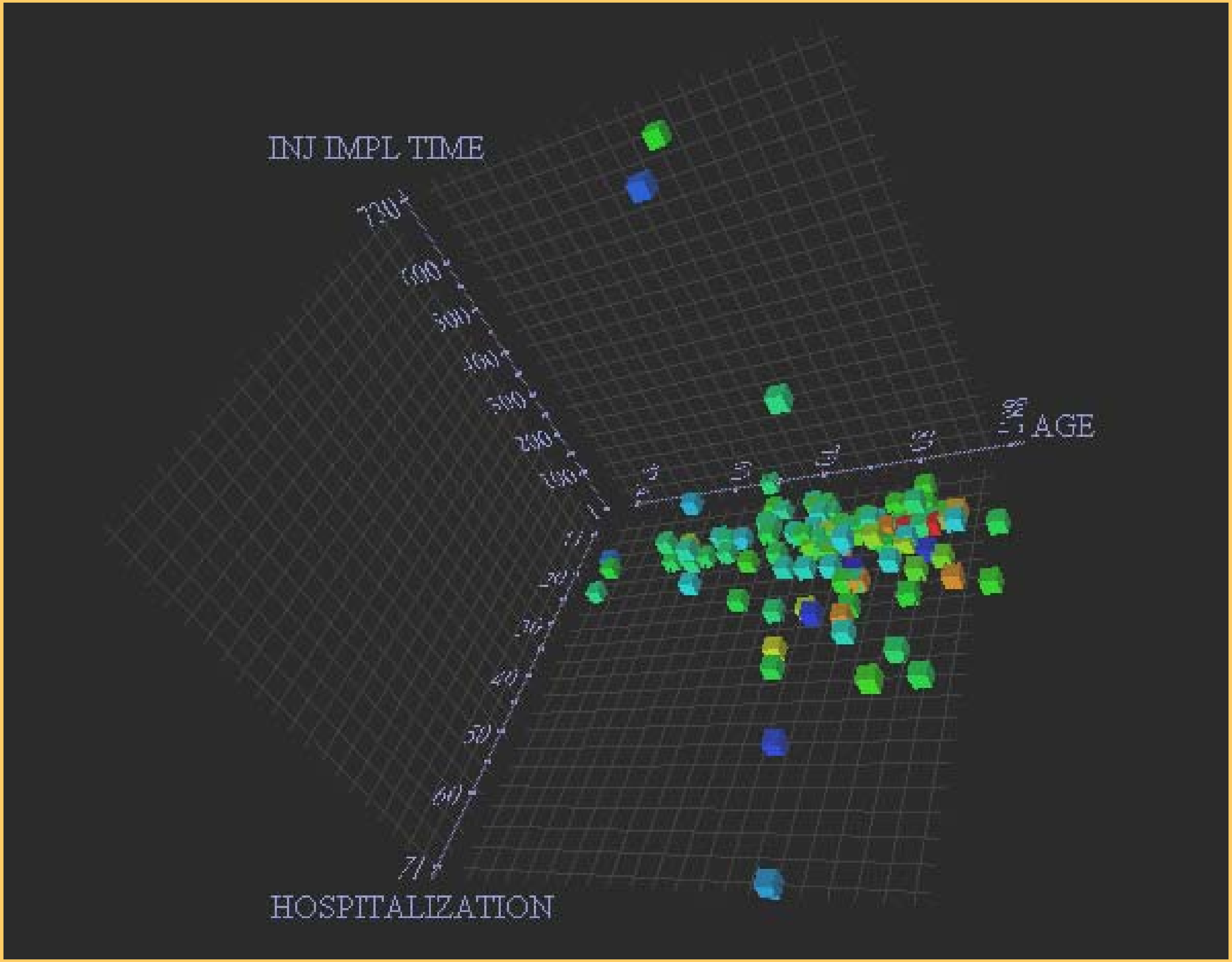
DM tools


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

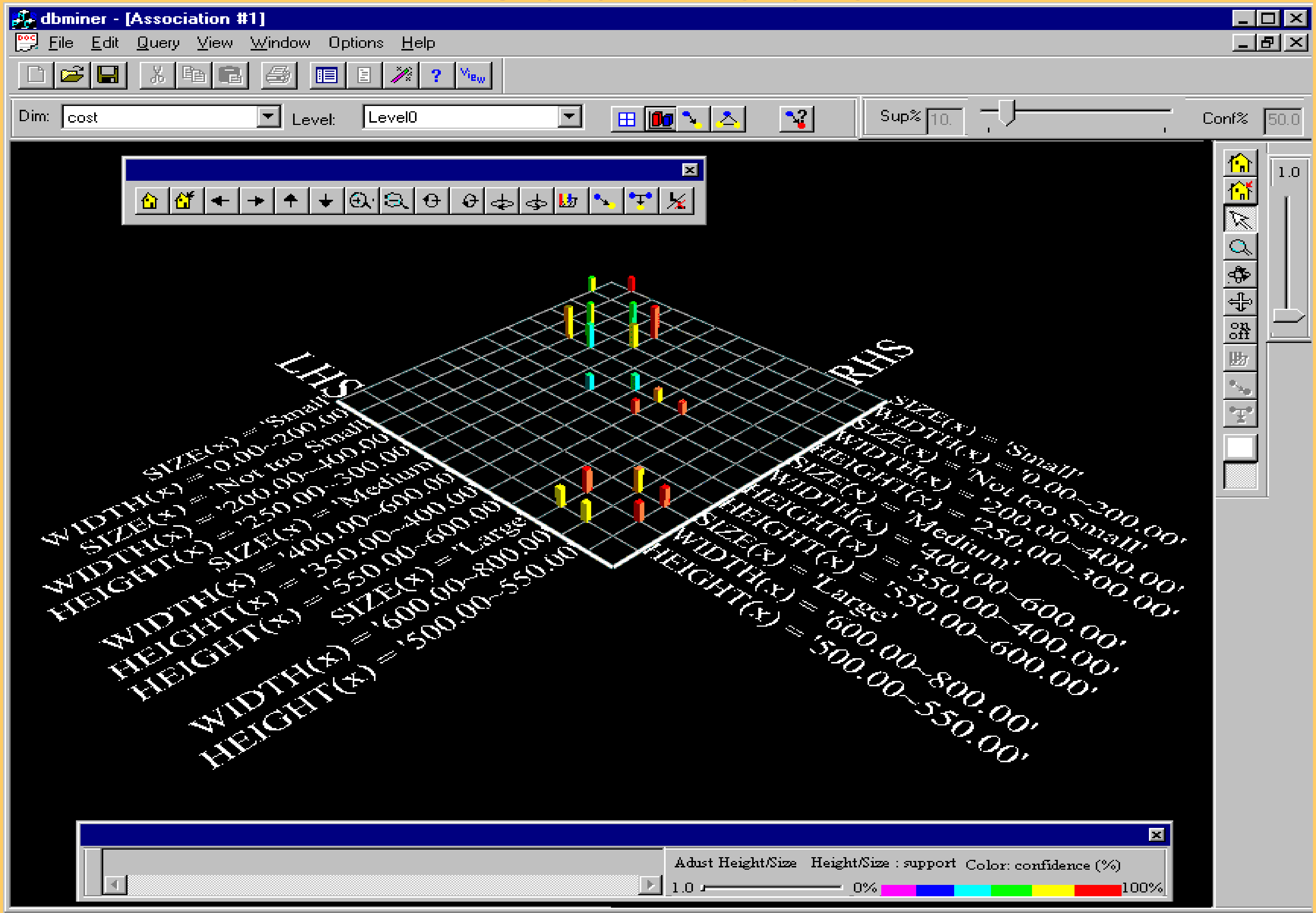
Visualization

- 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

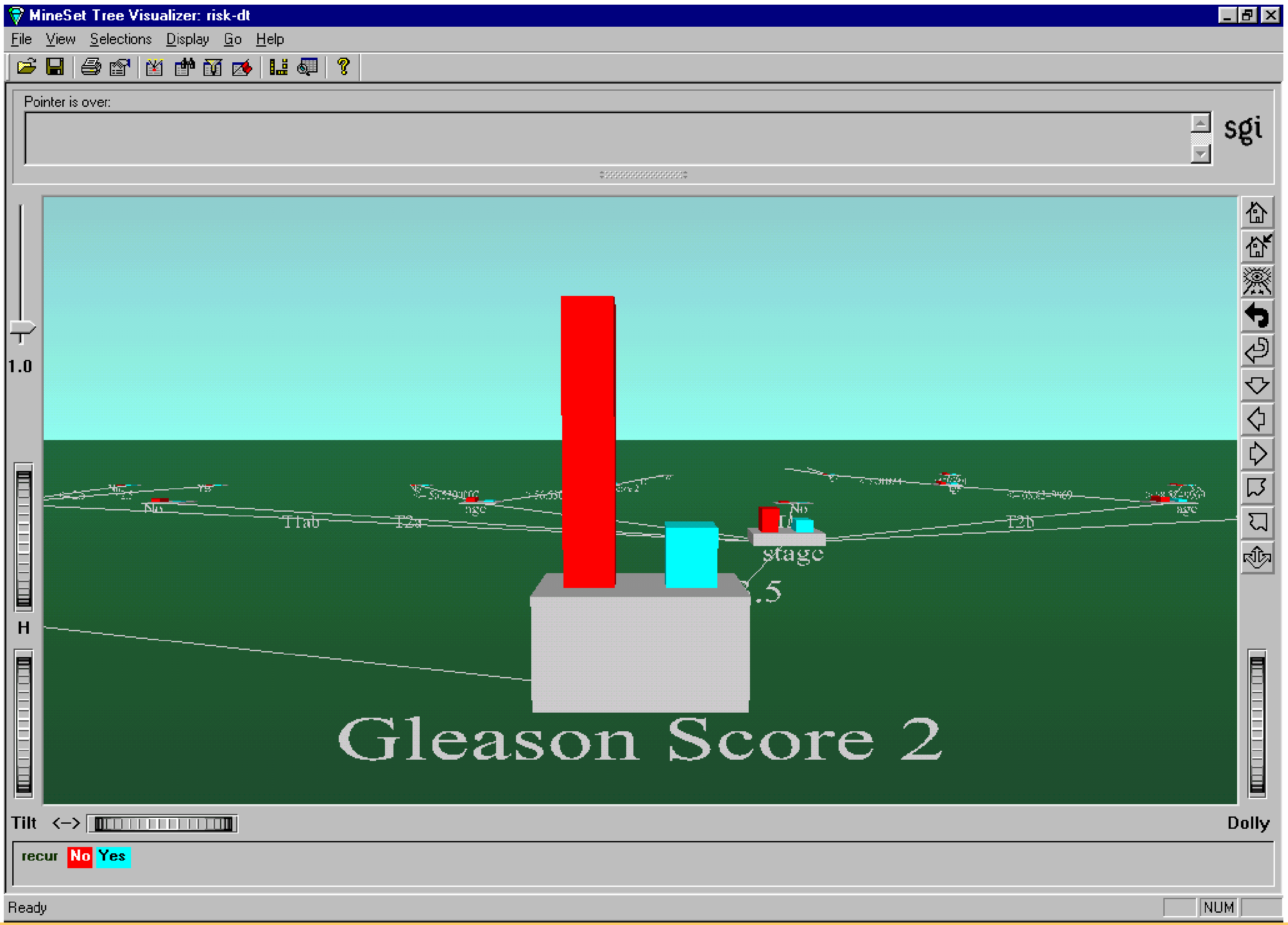
Data visualization: Scatter plot



DB Miner: Association rule visualization



MineSet: Decision tree visualization



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

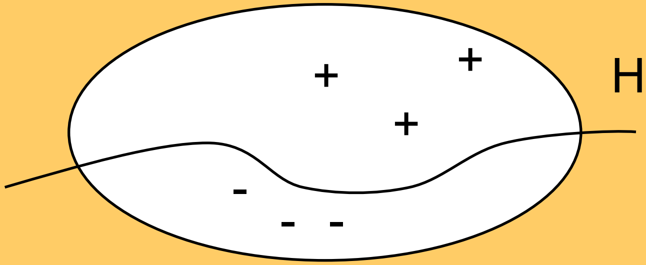
Part II: Standard Data Mining Techniques

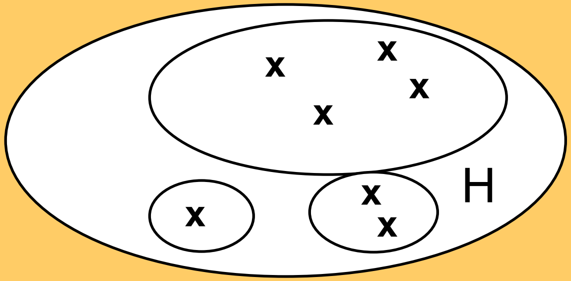
Classification of Data Mining techniques

- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

Types of DM tasks

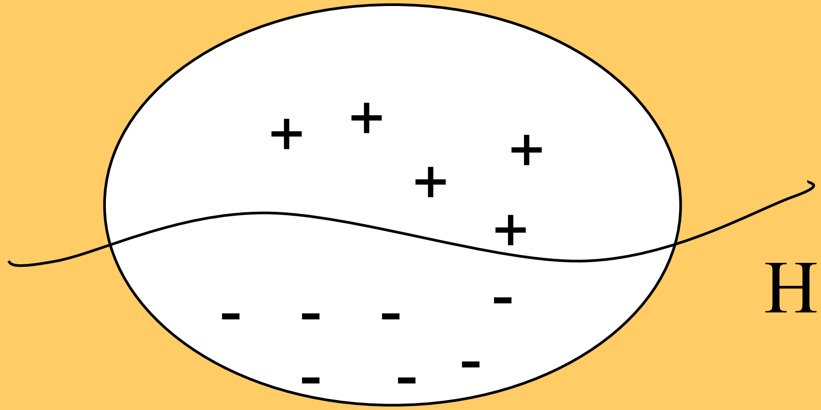
- **Predictive DM:**
  - Classification (learning of rules, decision trees, ...)
  - Prediction and estimation (regression)
  - Predictive relational DM (ILP)
- **Descriptive DM:**
  - description and summarization
  - dependency analysis (association rule learning)
  - discovery of properties and constraints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



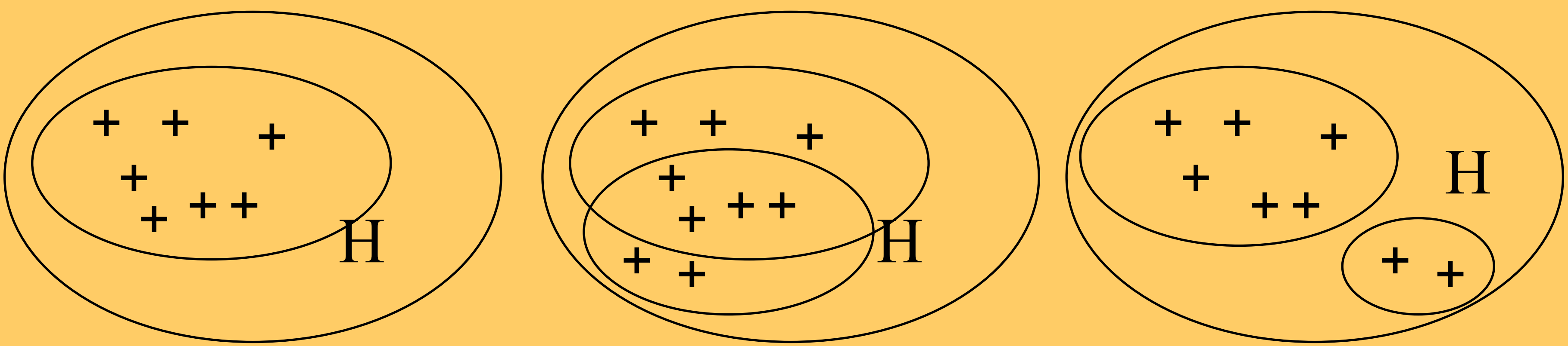


Predictive vs. descriptive induction

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 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 unabeled 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 II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

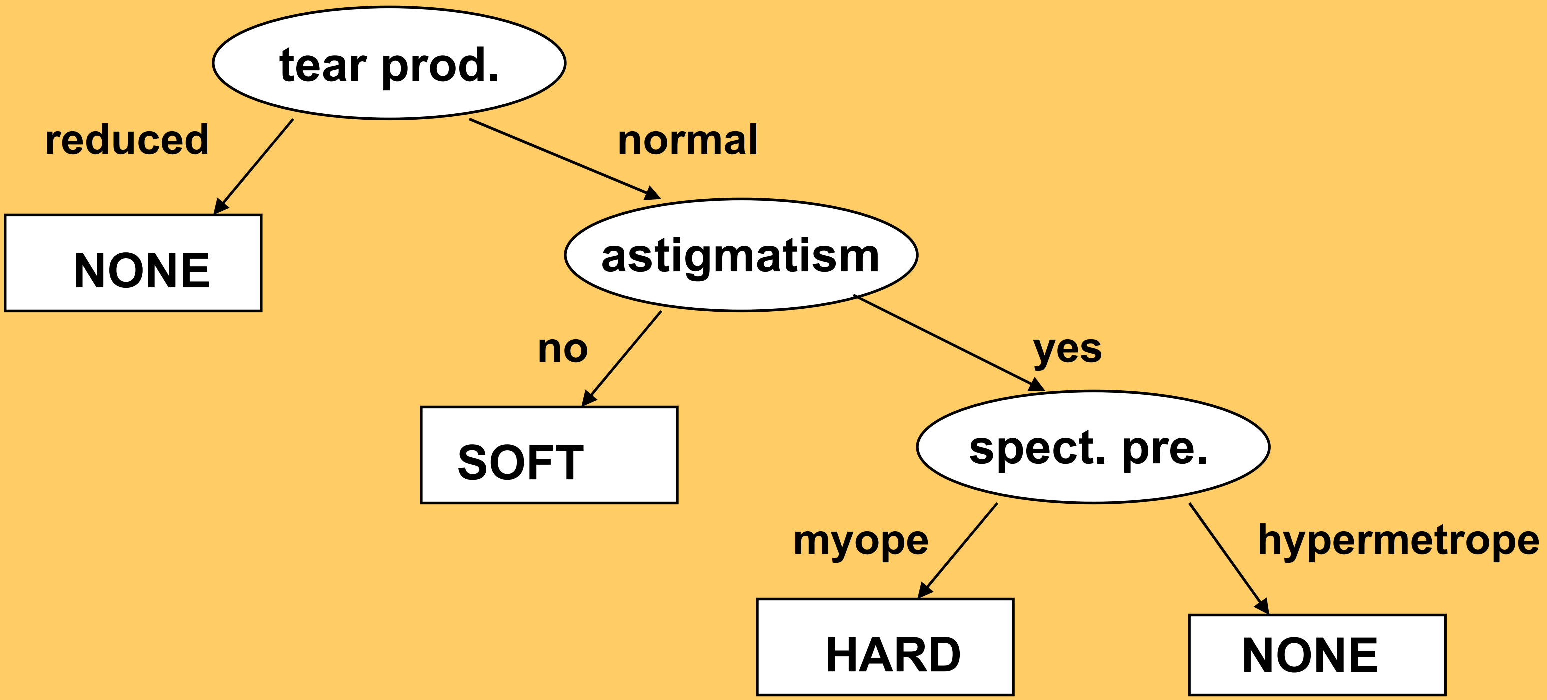
Predictive DM - Classification

- 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, ...

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

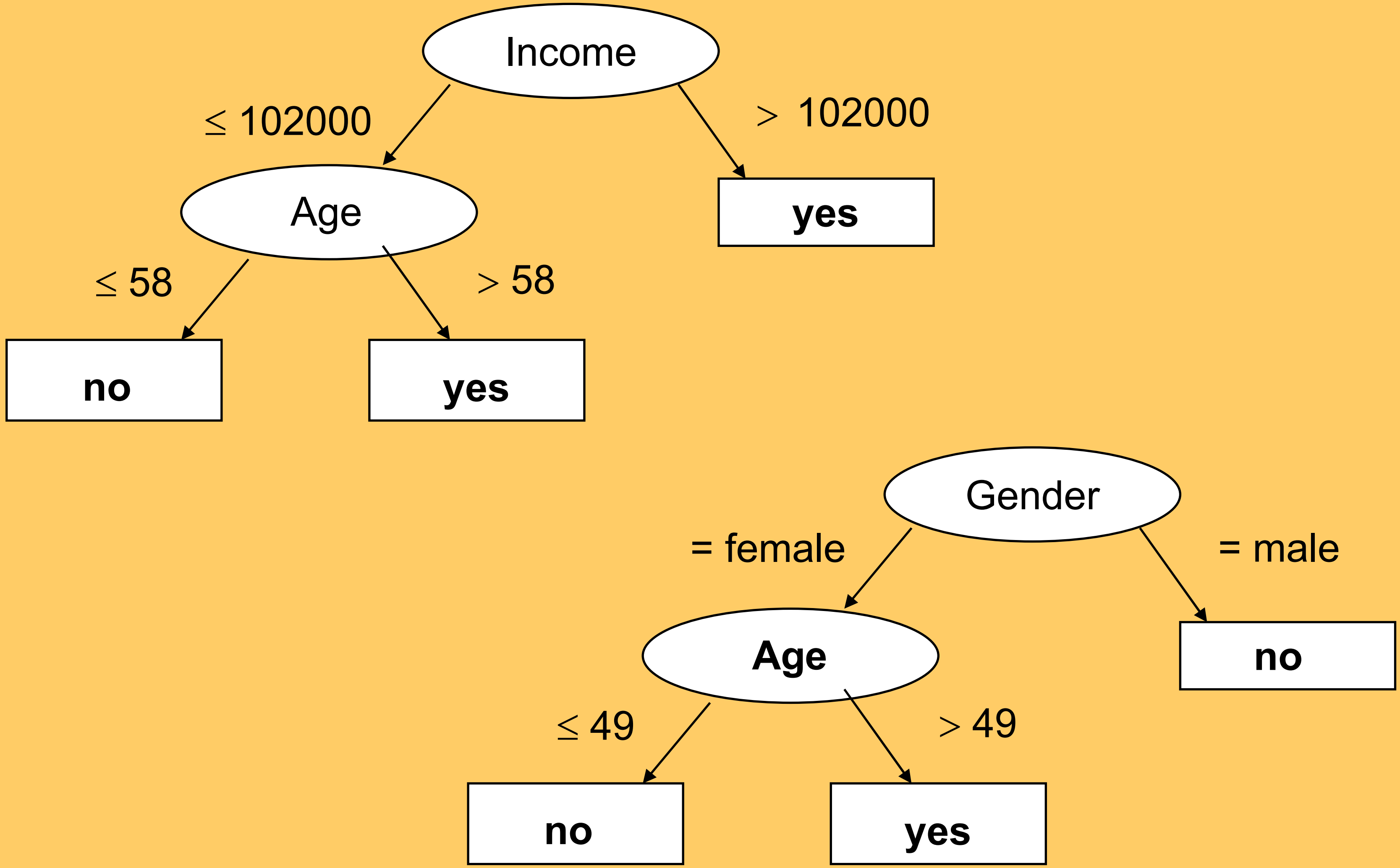
Decision tree for contact lenses recommendation



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

Induced decision trees



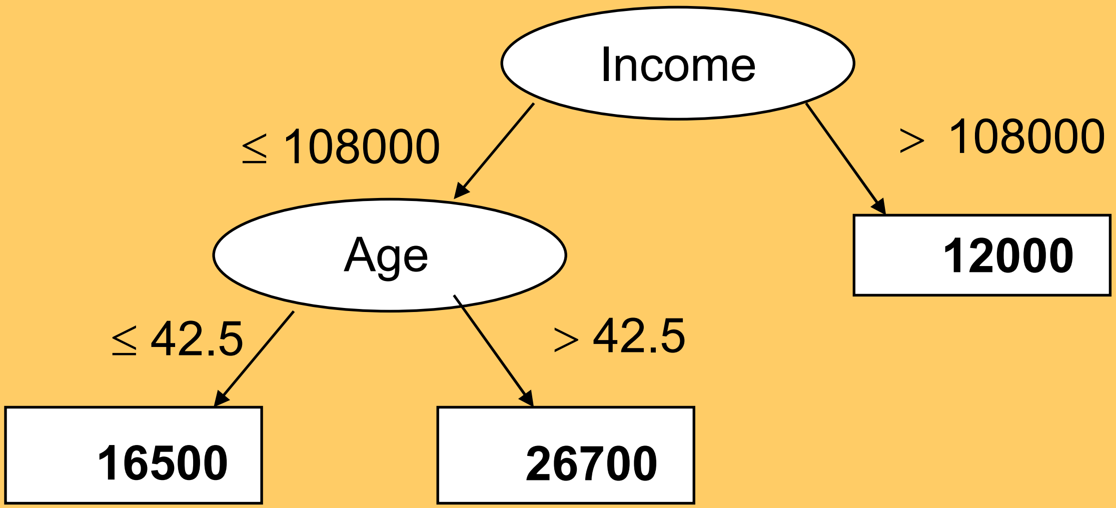
Predictive DM - Estimation

- 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, ...

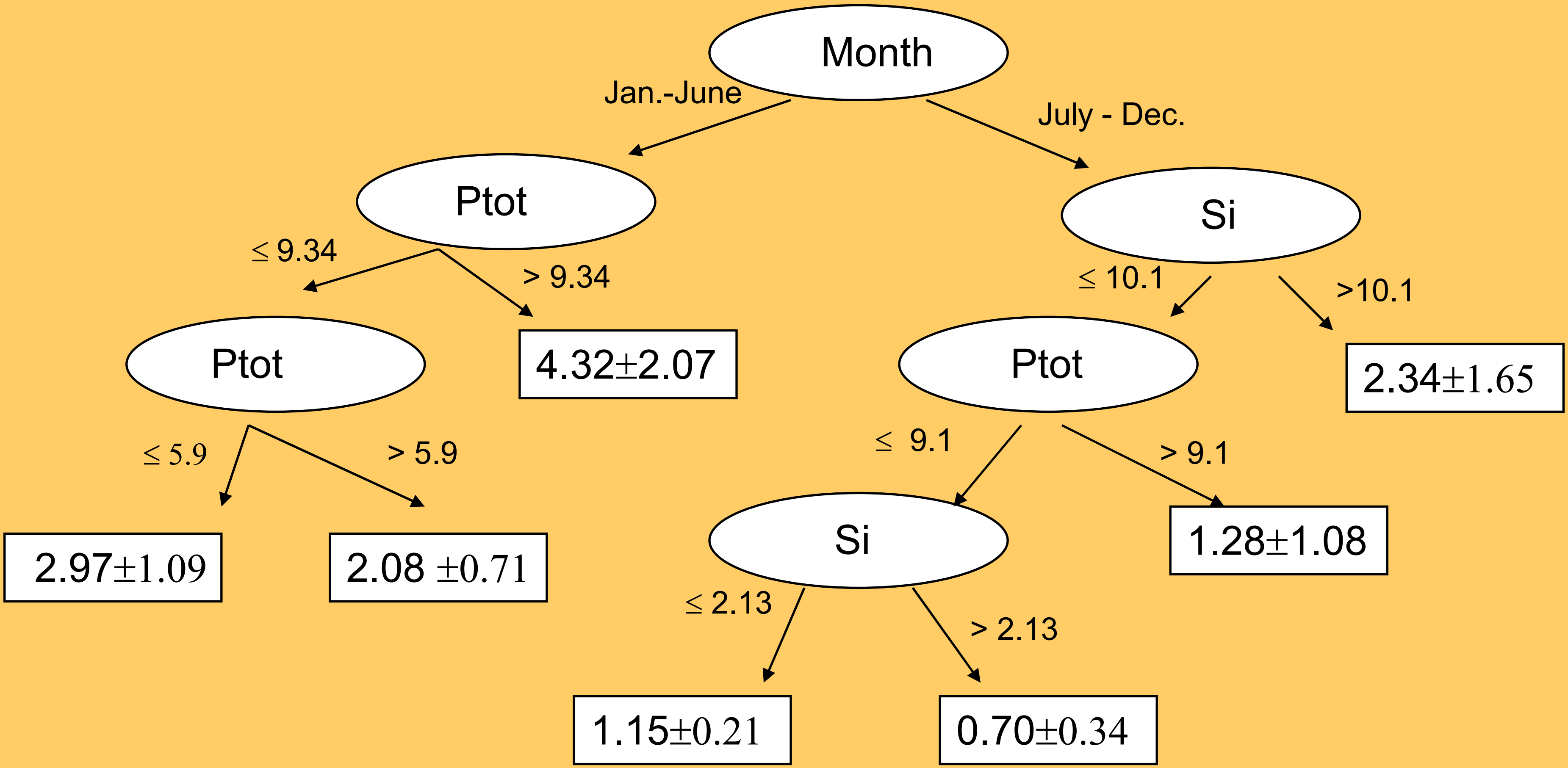
Illustrative example: Customer data

Customer	Gender	Age	Income	Spent	
c1	male	30	214000	18800	
c2	female	19	139000	15100	
c3	male	55	50000	12400	
c4	female	48	26000	8600	
c5	male	63	191000	28100	
O6-O13	…	…	…	…	
c14	female	61	95000	18100	
c15	male	56	44000	12000	
c16	male	36	102000	13800	
c17	female	57	215000	29300	
c18	male	33	67000	9700	
c19	female	26	95000	11000	
c20	female	55	214000	28800	

Customer data: regression tree



Predicting algal biomass: regression tree



Part II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

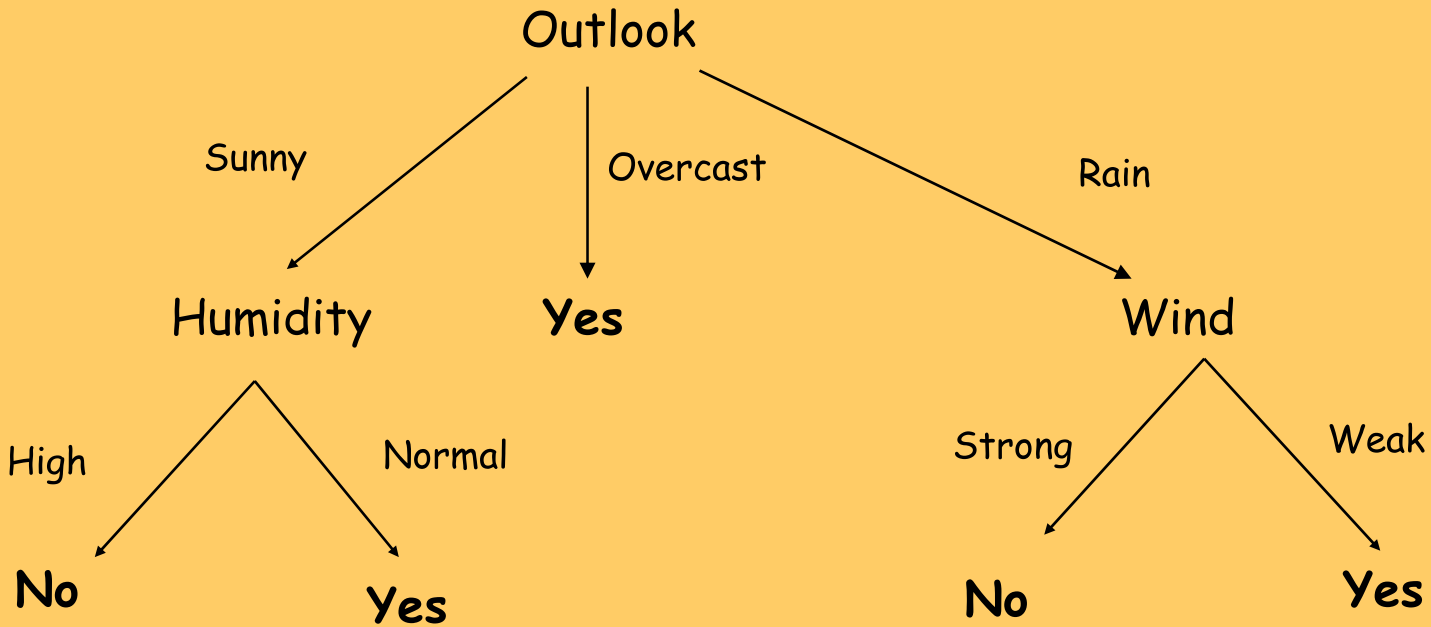
Decision tree learning

- Top-Down Induction of Decision Trees (TDIDT, Chapter 3 of Mitchell’s book)
- decision tree representation
- the ID3 learning algorithm (Quinlan 1986)
- heuristics: information gain (entropy minimization)
- overfitting, decision tree pruning
- brief on evaluating the quality of learned trees (more in Chapter 5)

PlayTennis: Training examples

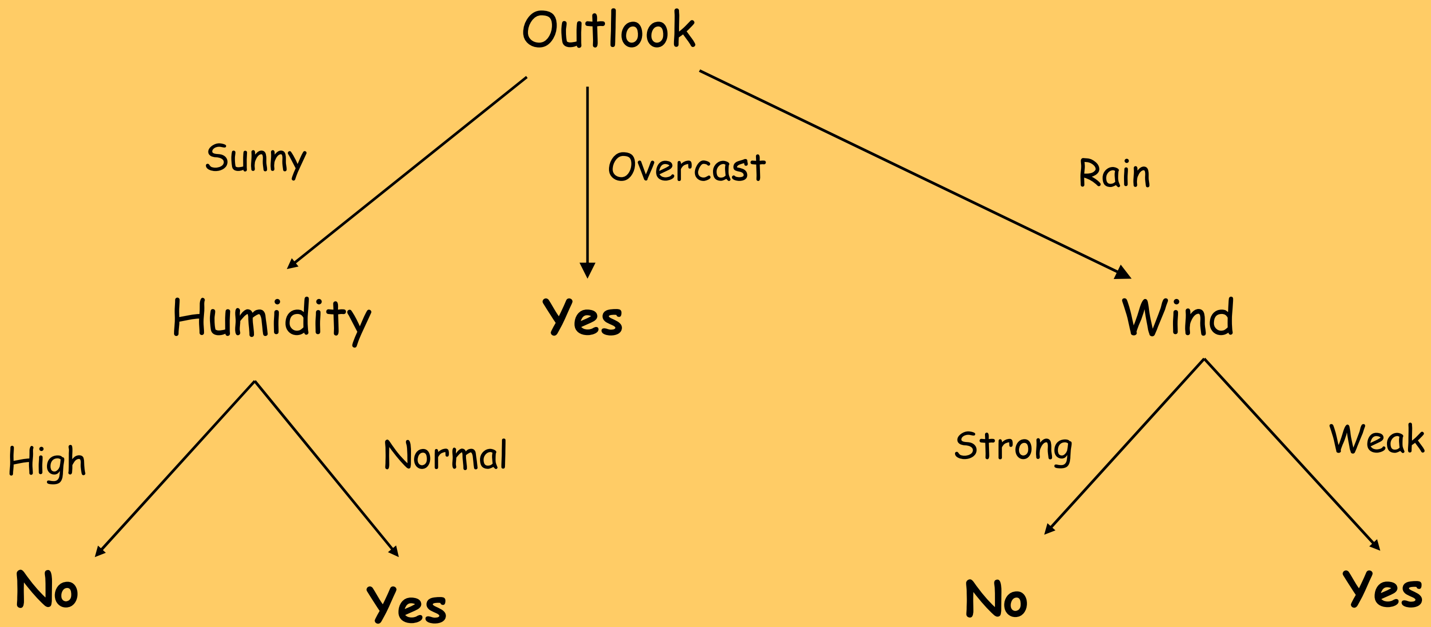
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal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

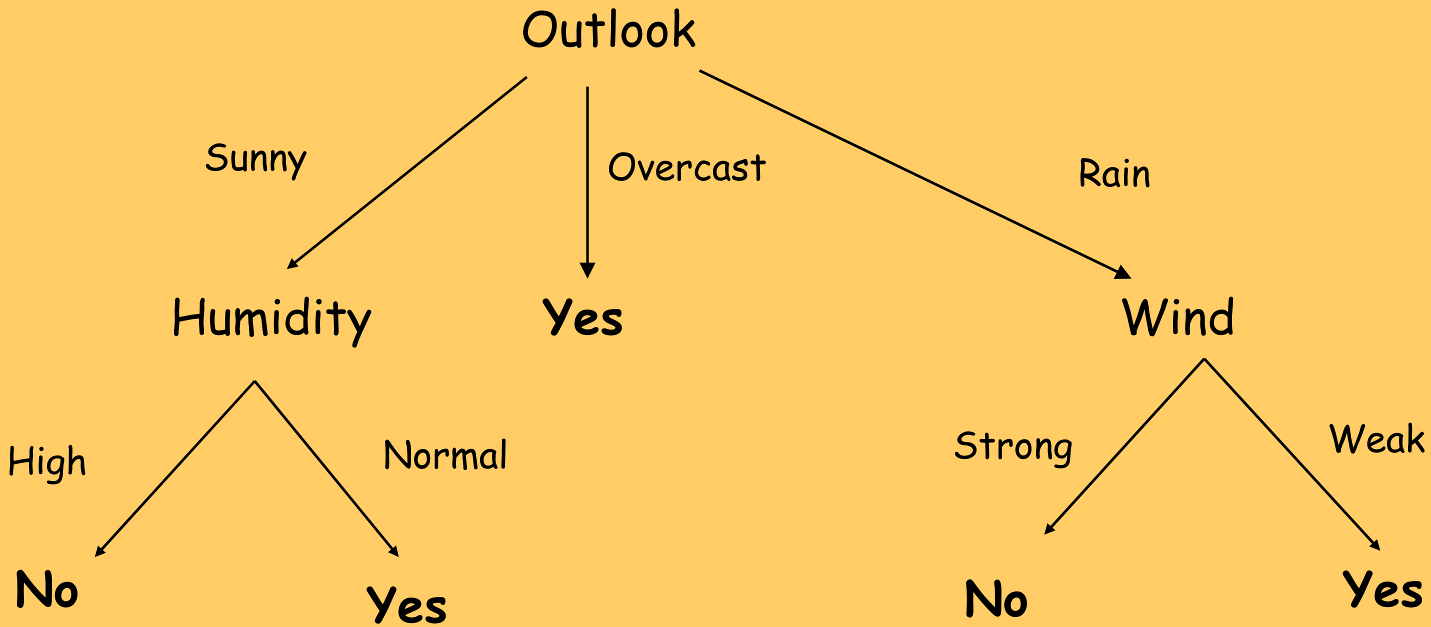
V (Outlook=Overcast)

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)
- 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

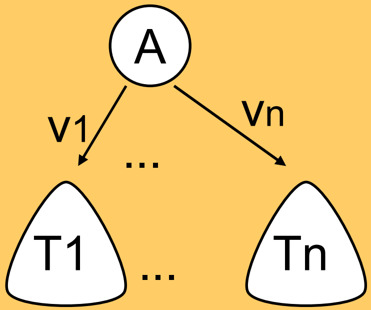
PlayTennis = No, because Outlook=Sunny $\wedge$ Humidity=High

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 C_j
  - then label the root with C_j
  - 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**
    - construct decision tree T:



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

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

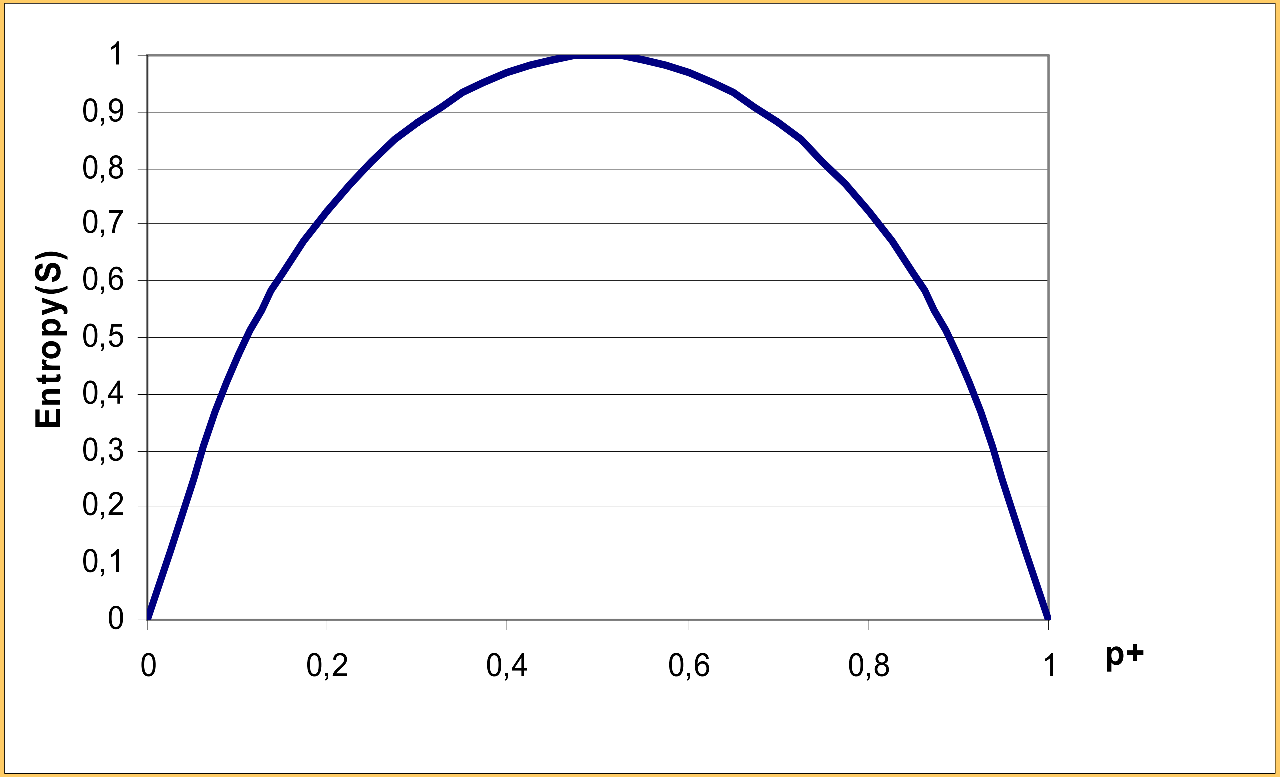
$\mathbf{p_c}$ - prior probability of class $\mathbf{C_c}$
(relative frequency of $\mathbf{C_c}$ in **S**)

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

Humidity?
- High → [3+, 4-] E=0.985
- Normal → [6+, 1-] E=0.592

Wind?
- Weak → [6+, 2-] E=0.811
- Strong → [3+, 3-] E=1.00

Information gain search heuristic

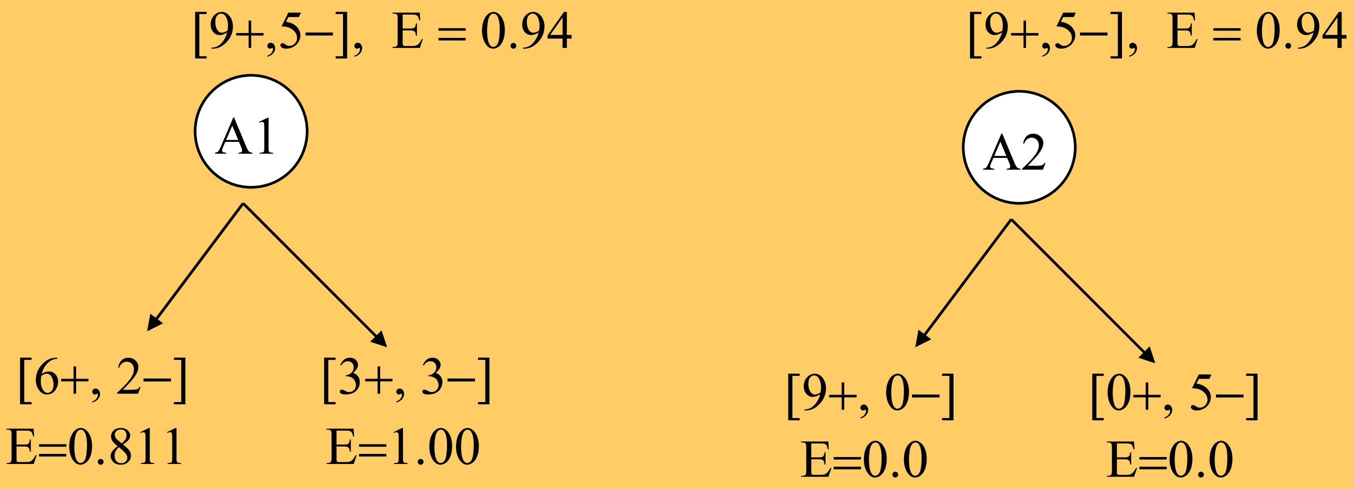
- **Information gain** measure is aimed to minimize the number of tests needed for the classification of a new object
- **Gain(S,A)** – expected reduction in entropy of S due to sorting on A

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

- **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}

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

  - 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**

Play tennis: 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

Play tennis: 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 of positive examples in set c :

$$p(+|c) = \frac{n^+(c)}{n(c)}$$

- Laplace estimate *:

$$p(+|c) = \frac{n^+(c)+1}{n(c)+2} \qquad p(+|c) = \frac{n^+(c)+1}{n(c)+k}$$

- *m*-estimate **:

$$p(+|c) = \frac{n^+(c)+m \cdot p_a(+)}{n(c)+m}$$

* k is number of classes, for k=2: uniform distribution assumption of 2 classes

** m is weight given to prior (i.e. number of 'virtual' examples)

Probability estimates: Intuitions

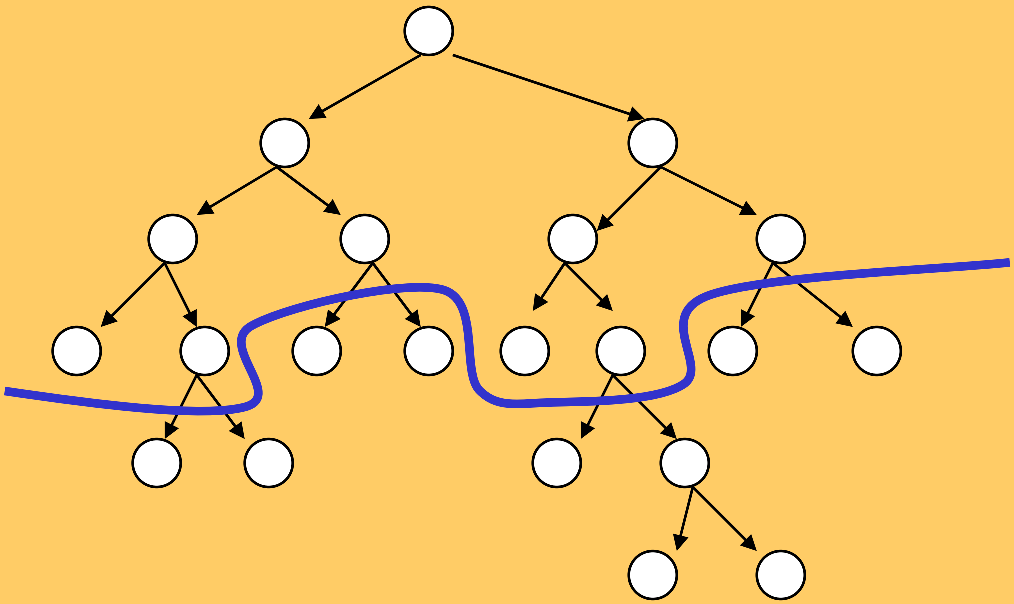
- An experiment with N trials, n successes
- Estimating the probability of success of next trial
- **Relative frequency: n/N**
  - reliable when the number of trials is large
  - unreliable with small samples, e.g., 1/1 = 1
- **Laplace: (n+1)/(N+2), or (n+1)/(N+k),** k classes
  - assumes a uniform distribution of classes
- **m-estimate: (n + m.p_a)/(N+m)**
  - prior probability of success p_a, user-defined parameter m (weight given to prior, i.e. number of 'virtual' examples)

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

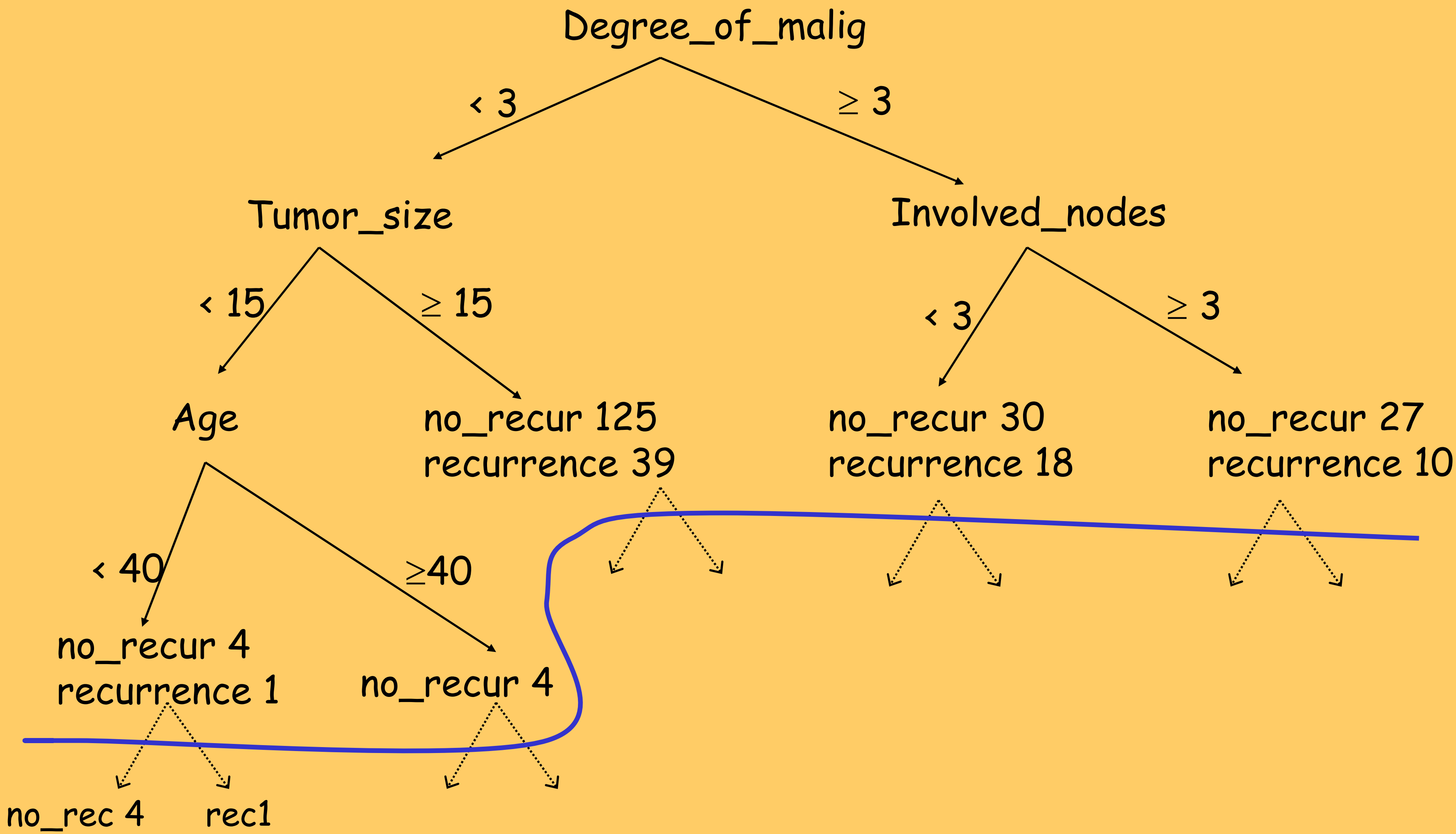
Sources of imperfection

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

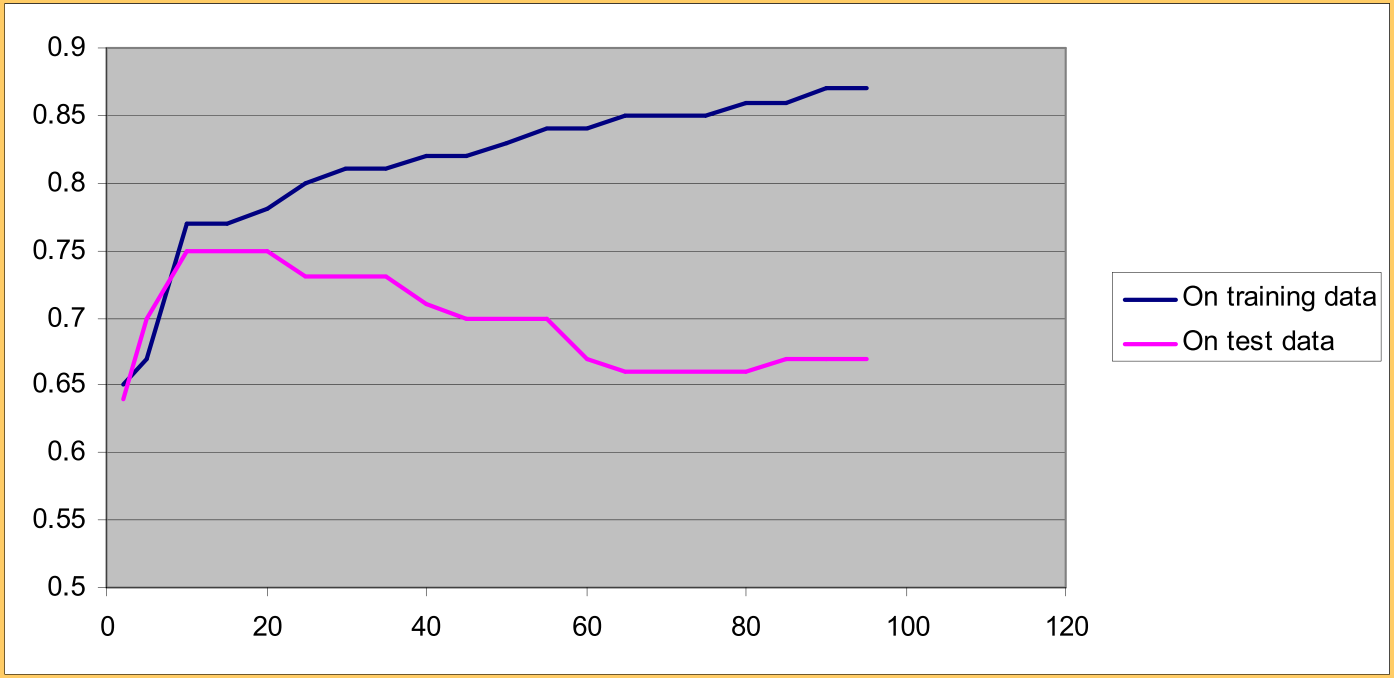


Accuracy and error

- 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



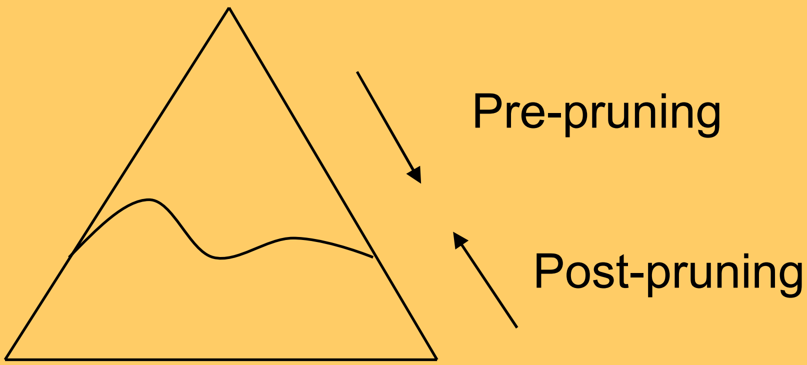
- Question: how to prune optimally?

Overfitting

- Consider error of hypothesis h over:
  - training data T: ErrorT(h)
  - entire distribution D of data: ErrorD(h)
- Hypothesis h $\in$ H overfits training data T if there is an alternative hypothesis h’ $\in$ H such that
  - ErrorT(h) < ErrorT(h’), and
  - ErrorD(h) > ErrorD(h’)
- Prune decision trees to avoid overfitting T

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

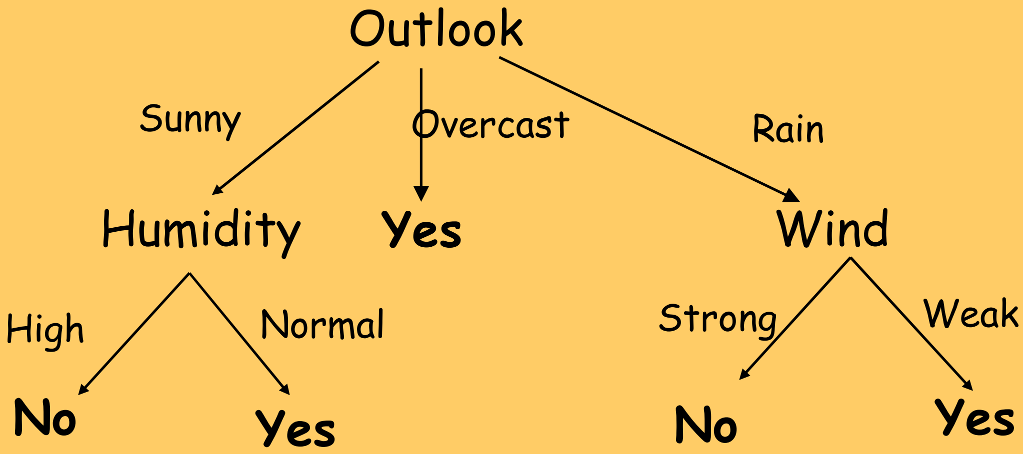


- forward pruning considered inferior (myopic)
- post pruning makes use of sub trees

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))

PlayTennis: Converting a tree to 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

Rule post-pruning (Quinlan 1993)

- Very frequently used method, e.g., in C4.5
- 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

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: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

Rule learning

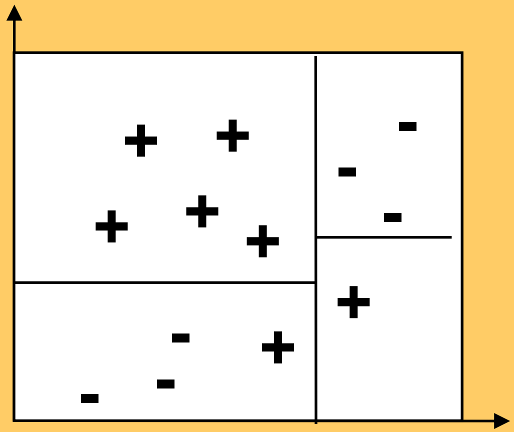
- Rule set representation
- Two rule learning approaches:
  - Learn decision tree, convert to rules
  - Learn set/list of rules
    - Learning an unordered set of rules
    - Learning an ordered list of rules
- Heuristics, overfitting, pruning

Predictive DM - Classification

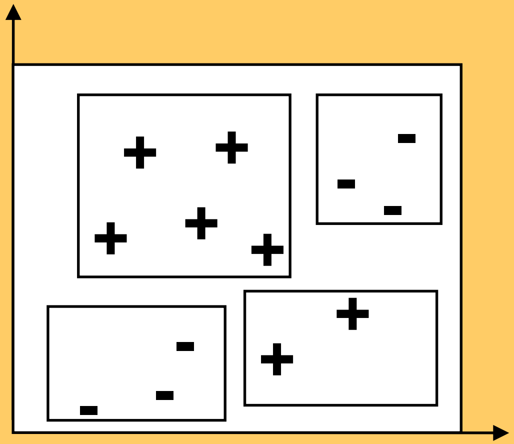
- data are objects, characterized with attributes - objects belong to different classes (discrete labels)
- given the objects described by attribute values, induce a model to predict different classes
- decision trees, if-then rules, ...

Decision tree vs. rule learning: Splitting vs. covering

- Splitting (ID3)



- Covering (AQ, CN2)



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 BestClass [ClassDistr]

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

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

Consumer data: classification rules

Unordered rules (independent, may overlap):

Income $>$ 108000 **=>** BigSpender = yes

Age $\geq$ 49 & Income $>$ 57000 **=>** BigSpender = yes

Age $\leq$ 56 & Income $<$ 98500 **=>** BigSpender = no

Income $<$ 51000 **=>** BigSpender = no

33 $<$ Age $\leq$ 42 **=>** BigSpender = no

DEFAULT BigSpender = yes

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

Contact lense: classification rules

- tear production=reduced **=>** lenses=NONE

  [S=0,H=0,N=12]
- 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]
- tear production=normal & astigmatism=yes & spect. pre.=hypermetrope **=>** lenses=NONE

  [S=0,H=1,N=2]

Unordered rulesets

- rule Class IF Conditions is learned by first determining Class and then Conditions
  - NB: ordered sequence of classes C1, …, Cn in RuleSet
  - But: unordered (independent) execution of rules when classifying a new instance: all rules are tried and predictions of those covering the example are collected; voting is used to obtain the final classification
- if no rule fires, then DefaultClass (majority class in E)

Contact lense: decision list

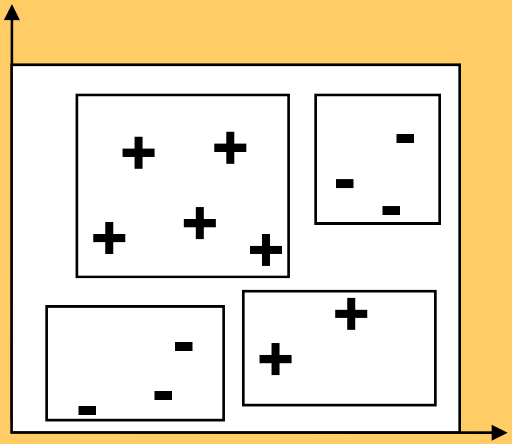
Ordered (order dependent) rules :

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 set of rules: if-then-else decision lists

- 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})

Original covering algorithm (AQ, Michalski 1969,86)



Basic covering algorithm

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

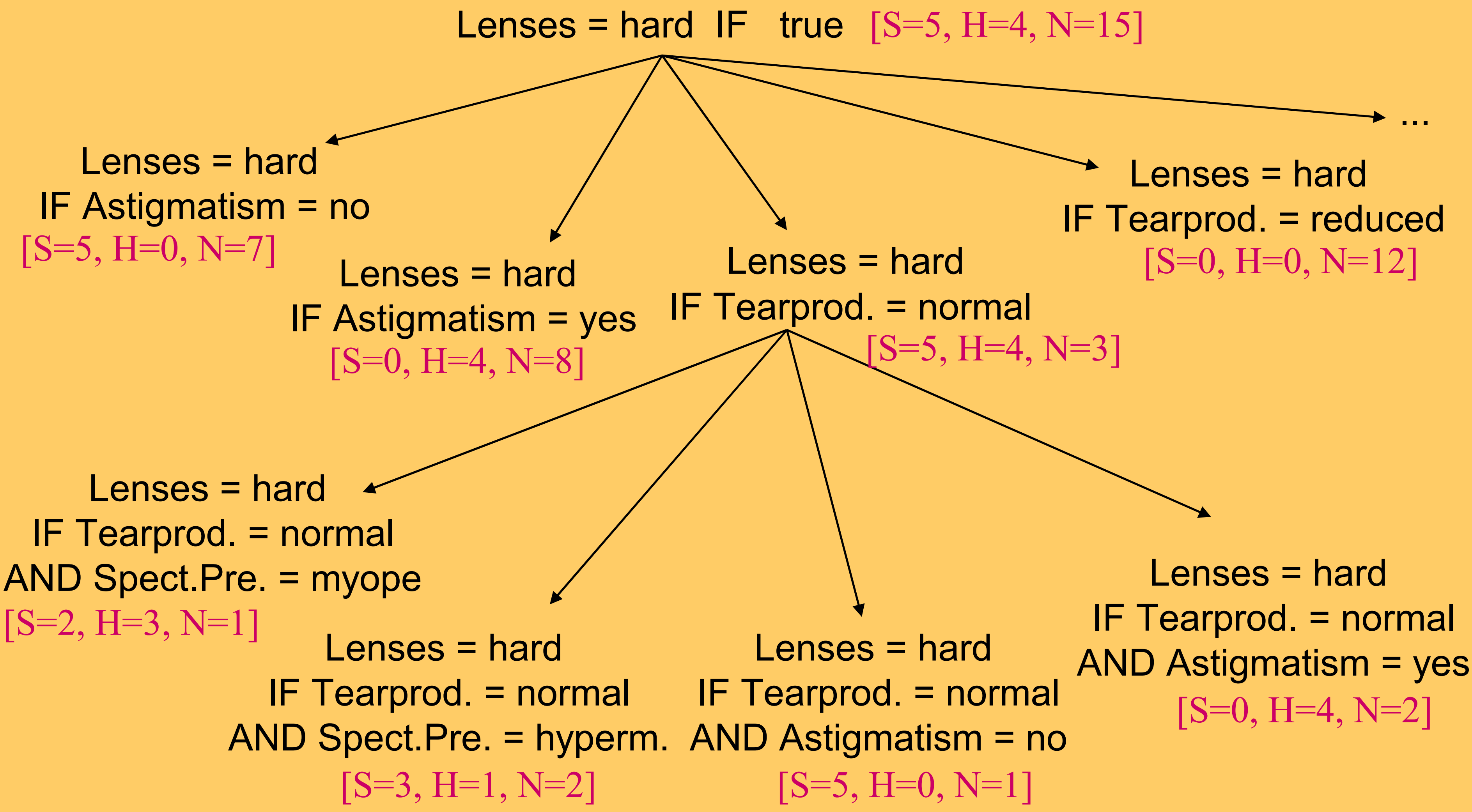
Learning unordered set of rules

- RuleBase := empty
- **for** each class C_i **do**
  - E_i := P_i U N_i, RuleSet(C_i) := empty
  - **repeat** {learn-set-of-rules}
    - R := **`Class = C`**$_i$ **`IF Conditions`**, `Conditions` := true
    - **repeat** {learn-one-rule}
      R' := **`Class = C`**$_i$ **`IF Conditions AND Cond`**
      (general-to-specific beam search of Best R')
    - **until** stopping criterion is satisfied
      (no negatives covered or Performance(R') < ThresholdR)
    - add R' to RuleSet(C_i)
    - delete from P_i all positive examples covered by R'
  - **until** stopping criterion is satisfied (all positives covered or Performance(RuleSet(C_i)) < ThresholdRS)
- RuleBase := RuleBase U RuleSet(C_i)

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
- 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 heuristic search

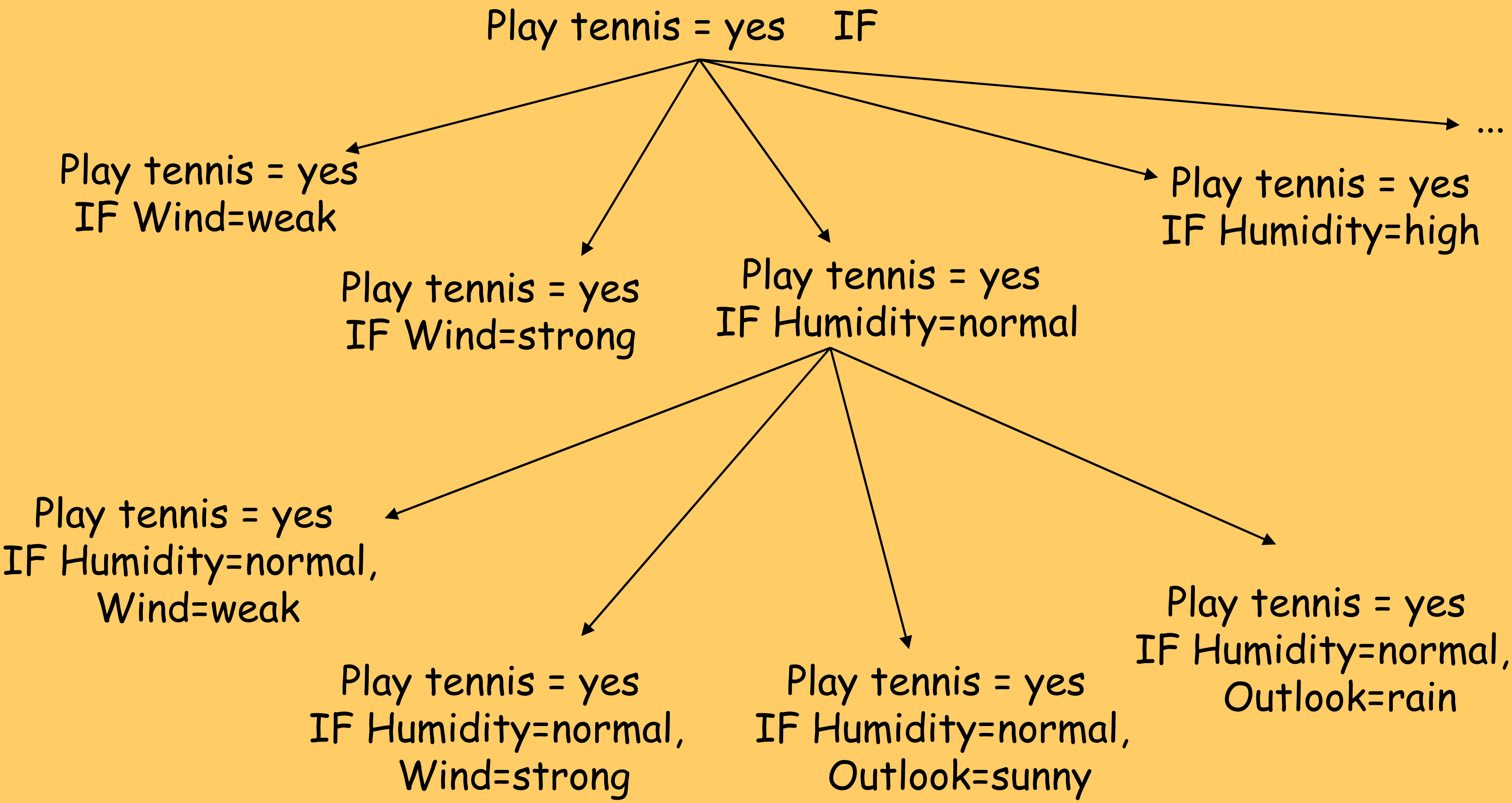


Learn-one-rule:

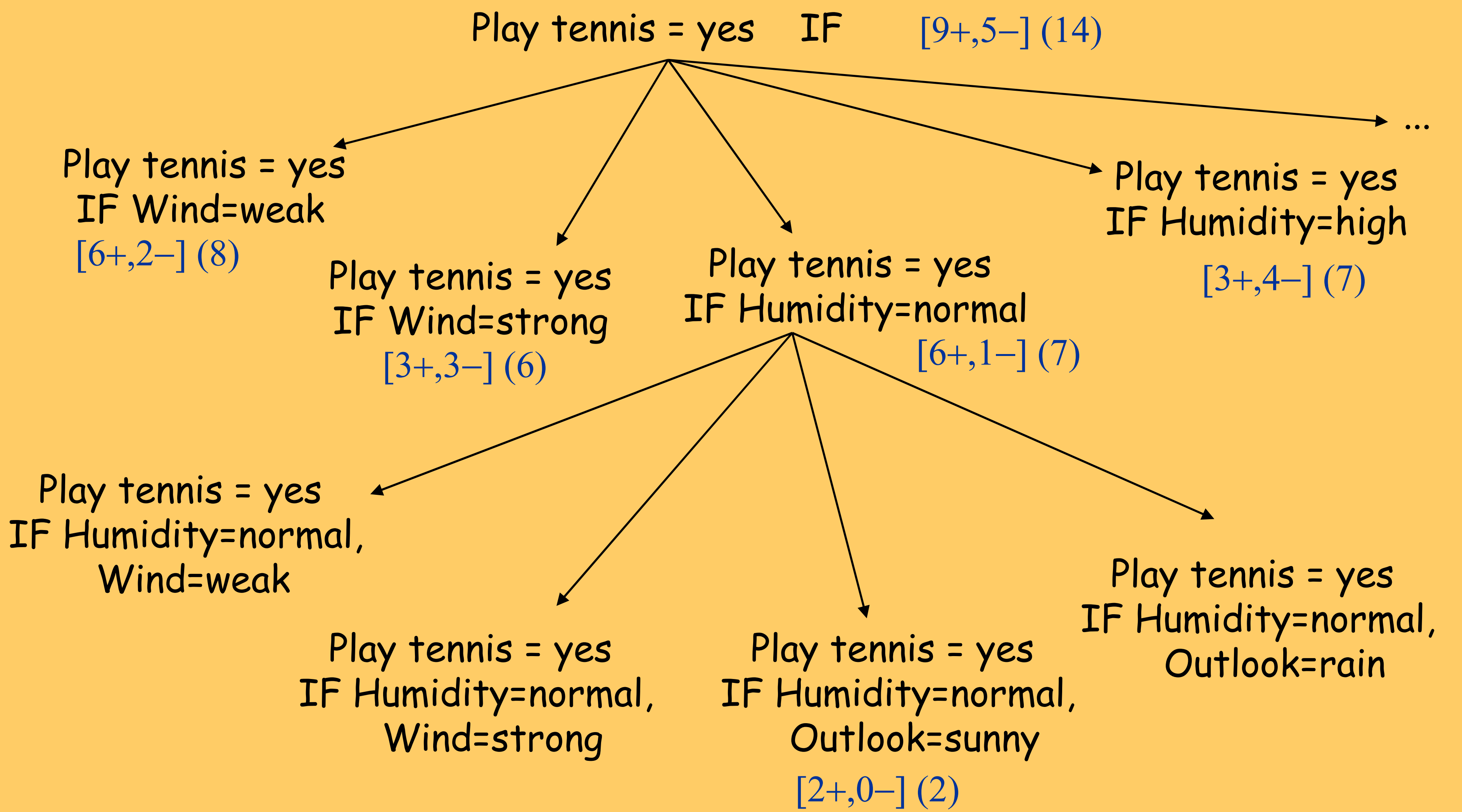
PlayTennis training examples

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

Learn-one-rule as search: PlayTennis example



Learn-one-rule as heuristic search: PlayTennis example



Heuristics for learn-one-rule: PlayTennis example

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

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

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

Estimating **accuracy** with **probability:**

A(Ci $\leftarrow$ Conditions) = p(Ci | Conditions)

Estimating **probability** with **relative frequency:**

covered pos. ex. / all covered ex.

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

Probability estimates

- **Relative frequency** of covered positive examples:
  - problems with small samples

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

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

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

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

- ***m*-estimate** :
  - special case: p(+)=1/k, m=k
  - takes into account prior probabilities $p_a(C)$ instead of uniform distribution
  - independent of the number of classes k
  - m is domain dependent (more noise, larger m)

$$= \frac{n(Cl.Cond)+m.p_a(Cl)}{n(Cond)+m}$$

Rule learning: summary

- **Hypothesis construction**: find a set of *n* rules
  - usually simplified by *n* separate rule constructions
- **Rule construction**: find a pair (Class, Body)
  - e.g. select rule head (class) and construct rule body
- **Body construction**: find a set of *m* features
  - usually simplified by adding to rule body one feature at a time

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 $WRAcc(R) = p(Cond).(p(Cl|Cond) - p_a(Cl))$

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

- Unlike the ordered case of standard CN2 where rules are interpreted in an `IF-THEN-ELSE` fashion, in the unordered case and in CN2-SD 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:

```
class=bird ← legs=2 & feathers=yes     [13,0]
class=elephant ← size=large & flies=no [2,10]
class=bird ← beak=yes                  [20,0]
```

[35,10]

Two-legged, feathered, large, non-flying animal with a beak? bird !

Performance metrics

- Confusion matrix, contingency table
- Heuristics for guiding the search
- Rule evaluation measures

Confusion matrix and Contingency table

	Predicted positive	Predicted negative	
Positive examples	**True pos. TP**	**False neg. FN**	Pos
Negative examples	**False pos. FP**	**True neg. TN**	Neg
	PredPos	PredNeg	N

	Body is true (Cd)	***Body is false (¬Cd)***	
Head is true (Cl)	**n(Cl.Cd)** ***true positives***	**n(Cl.¬Cd)** ***false negatives***	**n(Cl)**
Head is false (¬Cl)	**n(¬Cl.Cd)** ***false positives***	**n(¬Cl¬Cd)** ***true negatives***	**n(¬Cl)**
	n(Cd)	**n(¬Cd)**	**N**

TP = n(Cl.Cd)

- p(Cl.Cd) = n(Cl.Cd) / N
- ...

Confusion matrix and rule (in)accuracy

- 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

What is “high” accuracy?

- Rule accuracy should be traded off against the “default” accuracy 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 ←Cond) = p(Cl | Cond) – p(Cl)

Weighted relative accuracy

- If a rule covers a single example, its accuracy 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)]

Remarks on rule evaluation measures

- 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 - appropriate measure for use in descriptive induction, e.g., association rule learning

Contingency table

	Body is true (Cd)	*Body is false (¬Cd)*	
Head is true (Cl)	**n(Cl.Cd)** *true positives*	**n(Cl.¬Cd)** *false negatives*	**n(Cl)**
Head is false (¬Cl)	**n(¬Cl.Cd)** *false positives*	**n(¬Cl¬Cd)** *true negatives*	**n(¬Cl)**
	n(Cd)	**n(¬Cd)**	**N**

- p(Cl.Cd) = n(Cl.Cd) / N etc.

Rule evaluation measures

- ***Coverage***

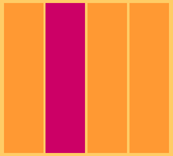
  $\mathrm{Cov}(\mathrm{Cl} \leftarrow \mathrm{Cond}) = p(\mathrm{Cond})$

- ***Support* = *frequency***

  $\mathrm{Sup}(\mathrm{Cl} \leftarrow \mathrm{Cond}) = p(\mathrm{Cl.Cond})$

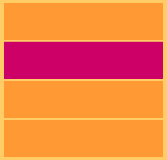
- ***Rule accuracy* = *confidence* = *precision***

  $\mathrm{Acc}(\mathrm{Cl} \leftarrow \mathrm{Cond}) = n(\mathrm{Cl.Cond})/n(\mathrm{Cond}) = p(\mathrm{Cl} \mid \mathrm{Cond})$

- ***Sensitivity* = *recall of positives*** (TPr)

  $\mathrm{Sens}(\mathrm{Cl} \leftarrow \mathrm{Cond}) = n(\mathrm{Cl.Cond}) / n(\mathrm{Cl}) = p(\mathrm{Cond} \mid \mathrm{Cl})$

- ***Specificity* = *recall of negatives***

  $\mathrm{Spec}(\mathrm{Cl} \leftarrow \mathrm{Cond}) = n(\neg\mathrm{Cl}\neg\mathrm{Cond}) / n(\neg\mathrm{Cl})$

  $= p(\neg\mathrm{Cond} \mid \neg\mathrm{Cl})$

Other measures

- ***Relative sensitivity***
  - $\text{RSens}(\text{Cl} \leftarrow \text{Cond}) = p(\text{Cond} \mid \text{Cl}) - p(\text{Cond})$
- ***Relative specificity***
  - $\text{RSpec}(\text{Cl} \leftarrow \text{Cond}) = p(\neg\text{Cond} \mid \neg\text{Cl}) - p(\neg\text{Cond})$
- ***Weighted relative sensitivity***
  - $\text{WRSens}(\text{Cl} \leftarrow \text{Cond}) = p(\text{Cl})[p(\text{Cond} \mid \text{Cl}) - p(\text{Cond})]$
- ***Weighted relative specificity***
  - $\text{WRSpec}(\text{Cl} \leftarrow \text{Cond}) =$
    $= p(\neg\text{Cl})[p(\neg\text{Cond} \mid \neg\text{Cl}) - p(\neg\text{Cond})]$
- ***THEOREM*: WRSens(R) = WRSpec(R) = WRAcc(R), where**
  - $\text{WRAcc}(\text{Cl} \leftarrow \text{Cond}) = p(\text{Cond})[p(\text{Cl} \mid \text{Cond}) - p(\text{Cl})]$

Part II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

Descriptive DM

- Often used for preliminary 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)

Part II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

Subgroup Discovery

Given: a population of individuals and a 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 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

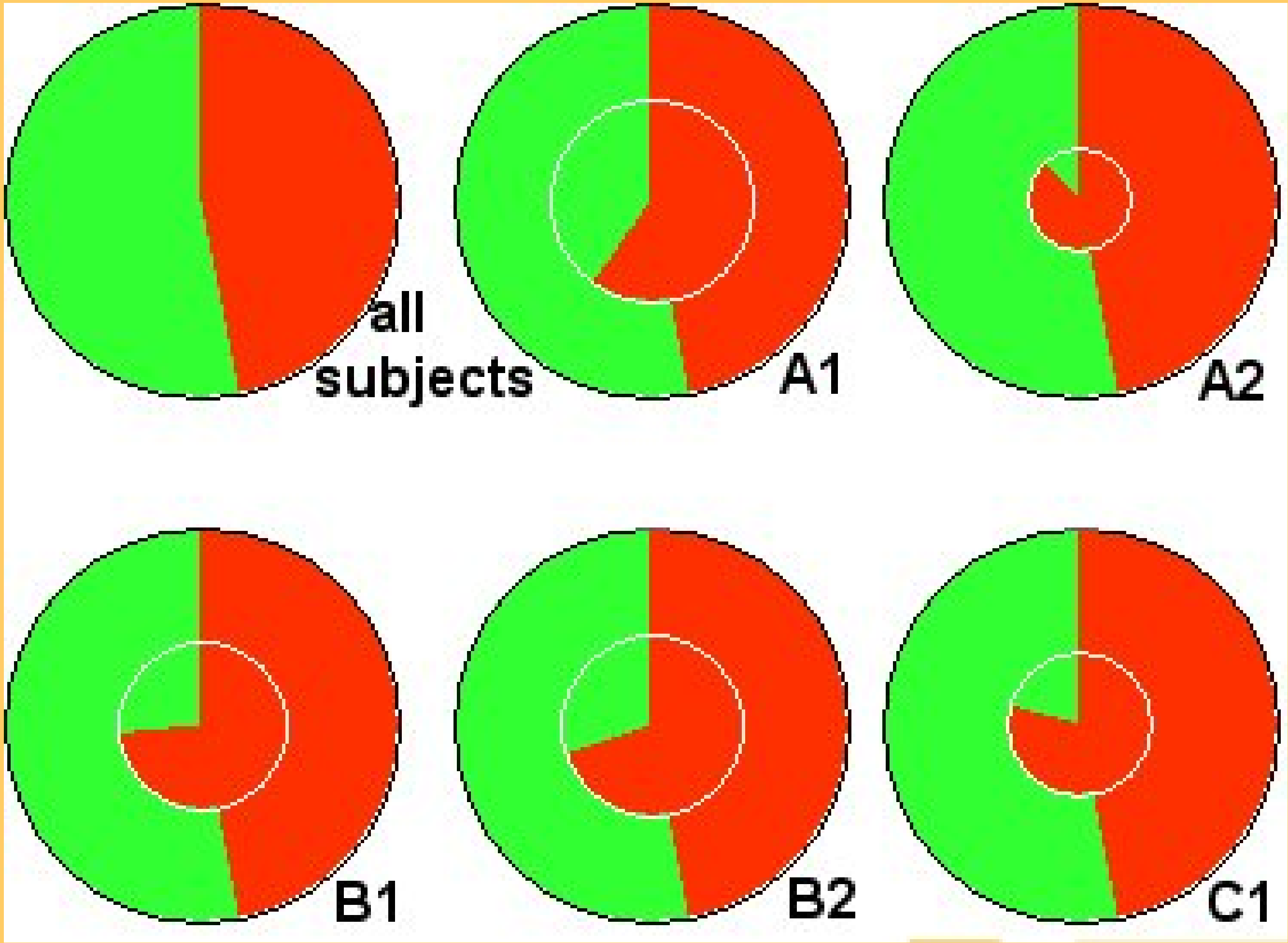
Subgroup Discovery: Medical Case Study

- **Find and characterize population subgroups with high CHD risk** (Gamberger, Lavrac, Krstacic)
- **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

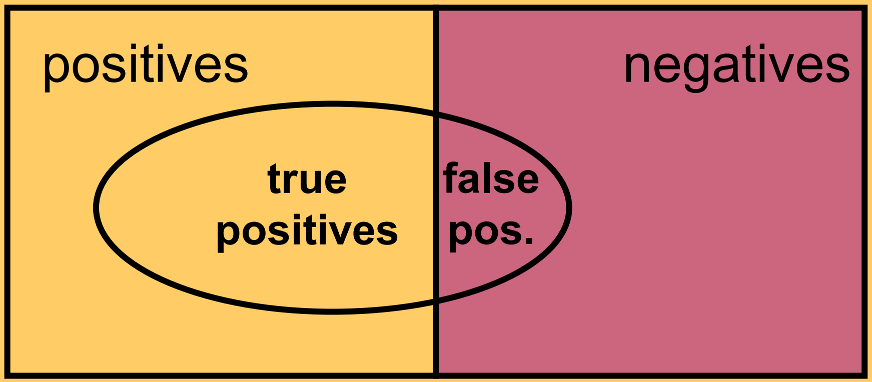


Subgroups of patients with CHD risk

[Gamberger, Lavrac & 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
    - **Multiplicative weights:** `w(e,i) =` $\texttt{gamma}^{\texttt{i}}$ `, 0<gamma<1`
      `note: gamma = 1` ➔ find the same (first) rule again and again
      `gamma = 0` ➔ behaves as standard CN2

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 II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - Hierarchical clustering

Association Rule Learning

Rules: X =>Y, if X then Y

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

Transactions:

itemsets (records)

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

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

Given: a set of transactions D

Find: all association rules that hold on the set of transactions that have support > MinSup and confidence > MinConf

Procedure:

- find all large itemsets Z, Sup(Z) > MinSup
- split every large itemset Z into XY, compute Conf(X,Y) = Sup(X,Y)/Sup(X),
  if Conf(X,Y) > MinConf then X **=>**Y
  (Sup(X,Y) > MinSup, as XY is large)

Part II: Standard Data Mining Techniques

- Classification of Data Mining techniques
- Predictive DM
  - Decision Tree induction
  - Learning sets of rules
- Descriptive DM
  - Subgroup discovery
  - Association rule induction
  - 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**

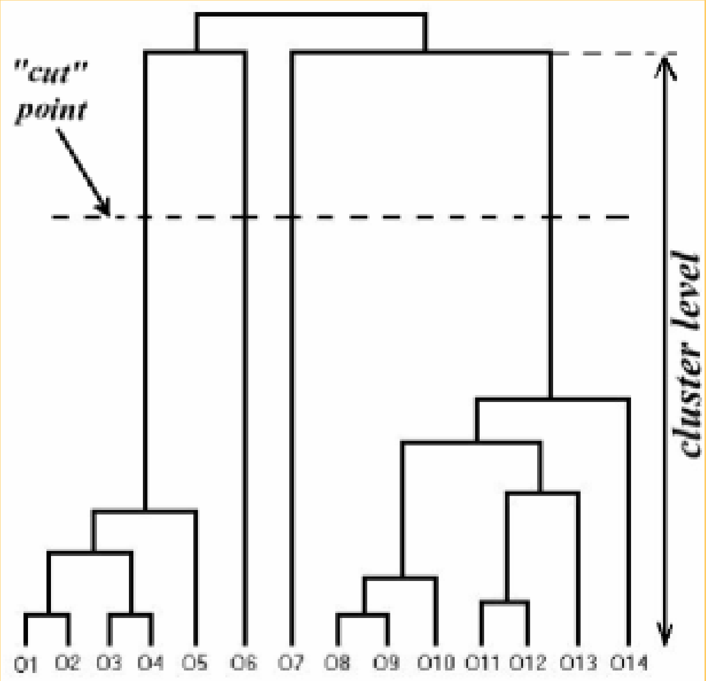
    $C_r = C_i \cup C_j$;

  determine *dissimilarities* between

    C_r and other clusters;

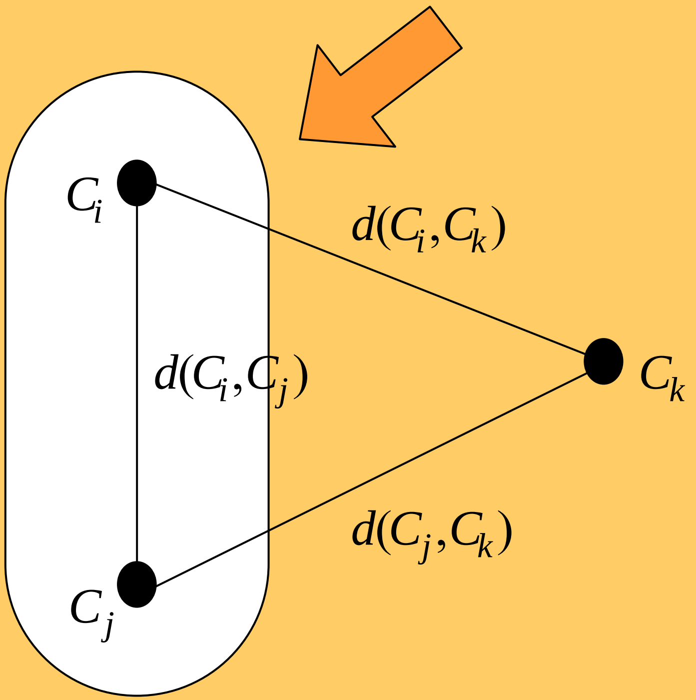
until one cluster left;

- Dendrogram:



Hierarchical clustering

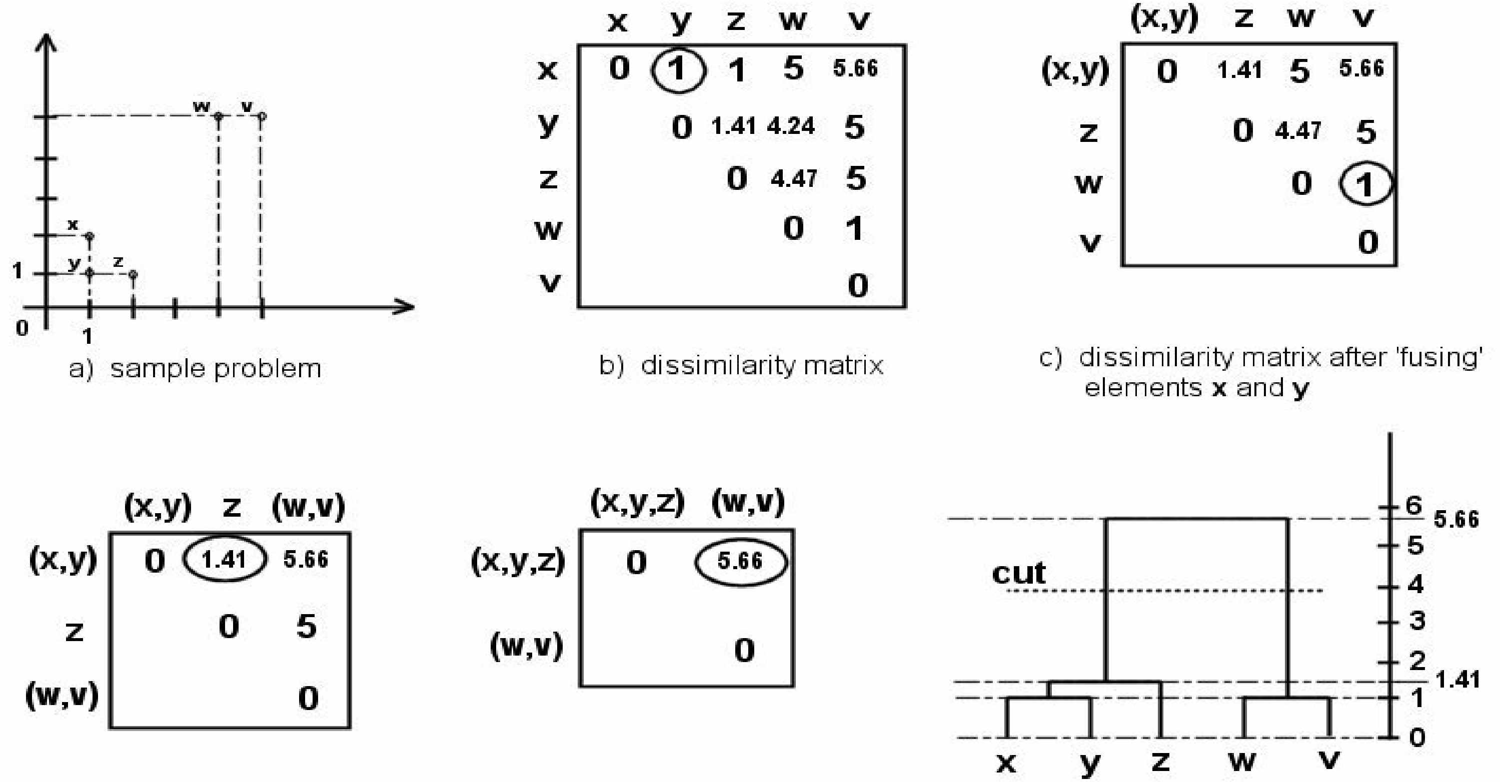
- Fusing the nearest pair of clusters



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

- Computing the dissimilarities from the "new" cluster

Hierarchical clustering: example



a) sample problem

b) dissimilarity matrix

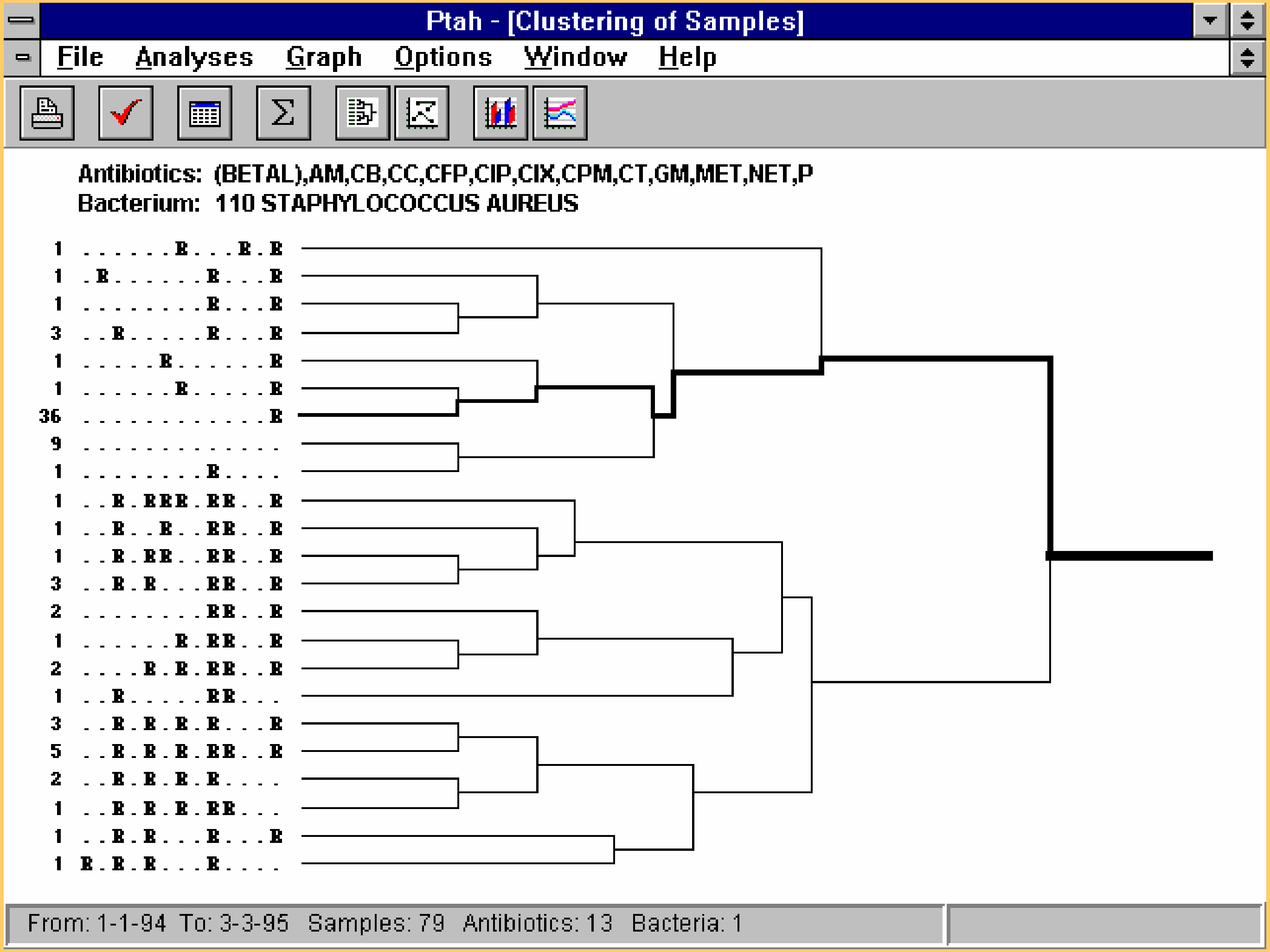
c) dissimilarity matrix after 'fusing' elements **x** and **y**

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 theraphy", IDAMAP book, 1997]

Part II: Summary

- Predictive DM:
  - classification, regression
  - trees, rules
  - splitting vs. covering
  - preventing overfitting
- Descriptive DM:
  - association rules
  - subgroup discovery
  - clustering

Part III: Evaluation

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

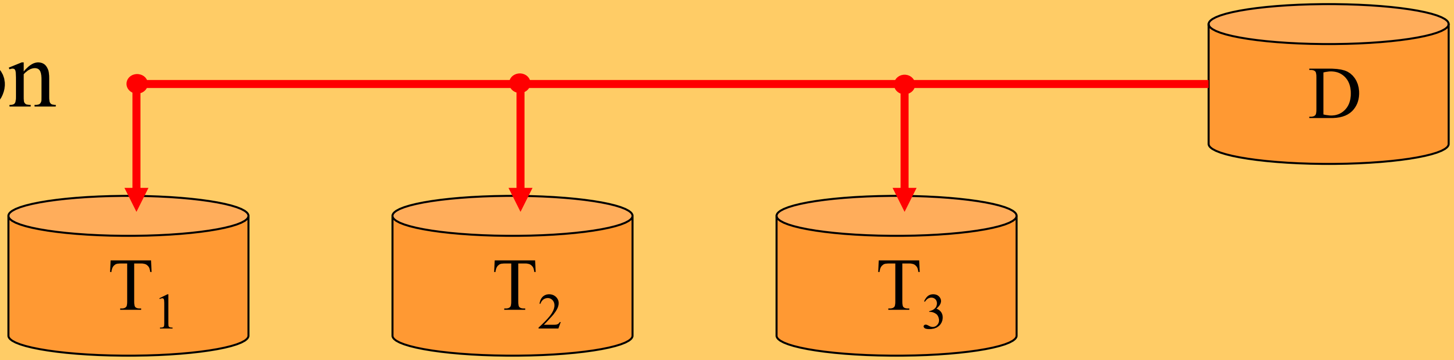
Evaluating hypotheses

- **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

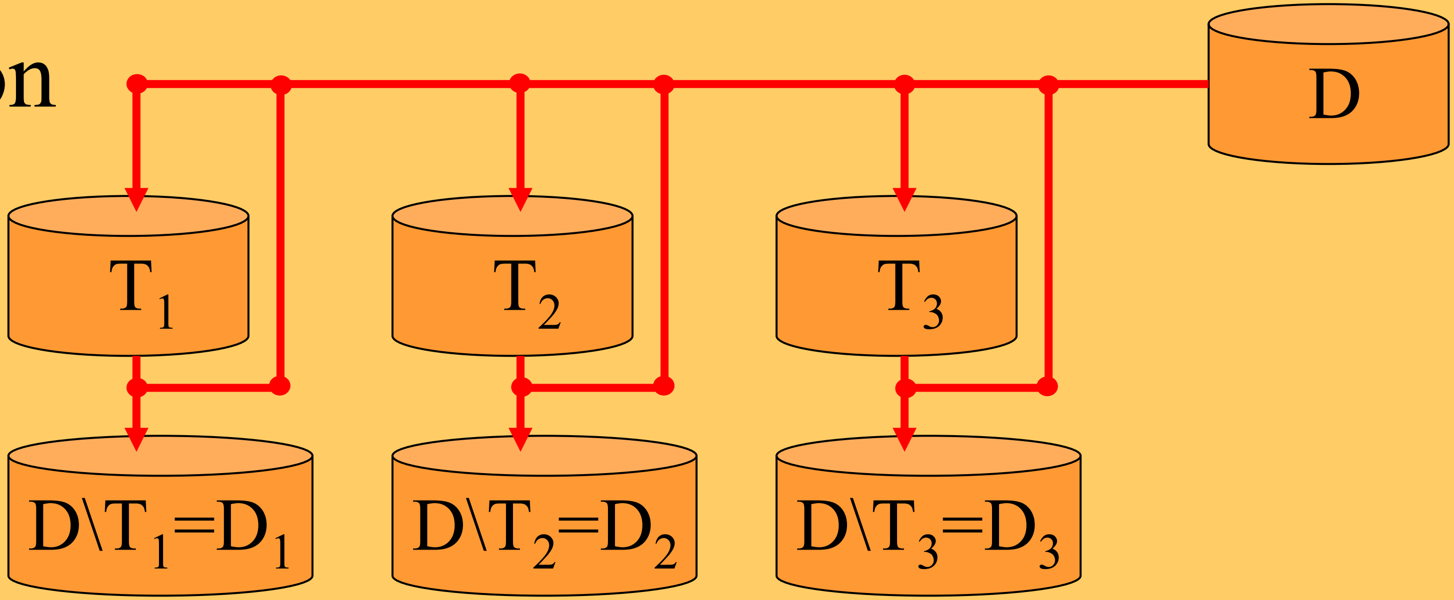
n-fold cross validation

- 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 = $D \setminus T_i$
  - 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

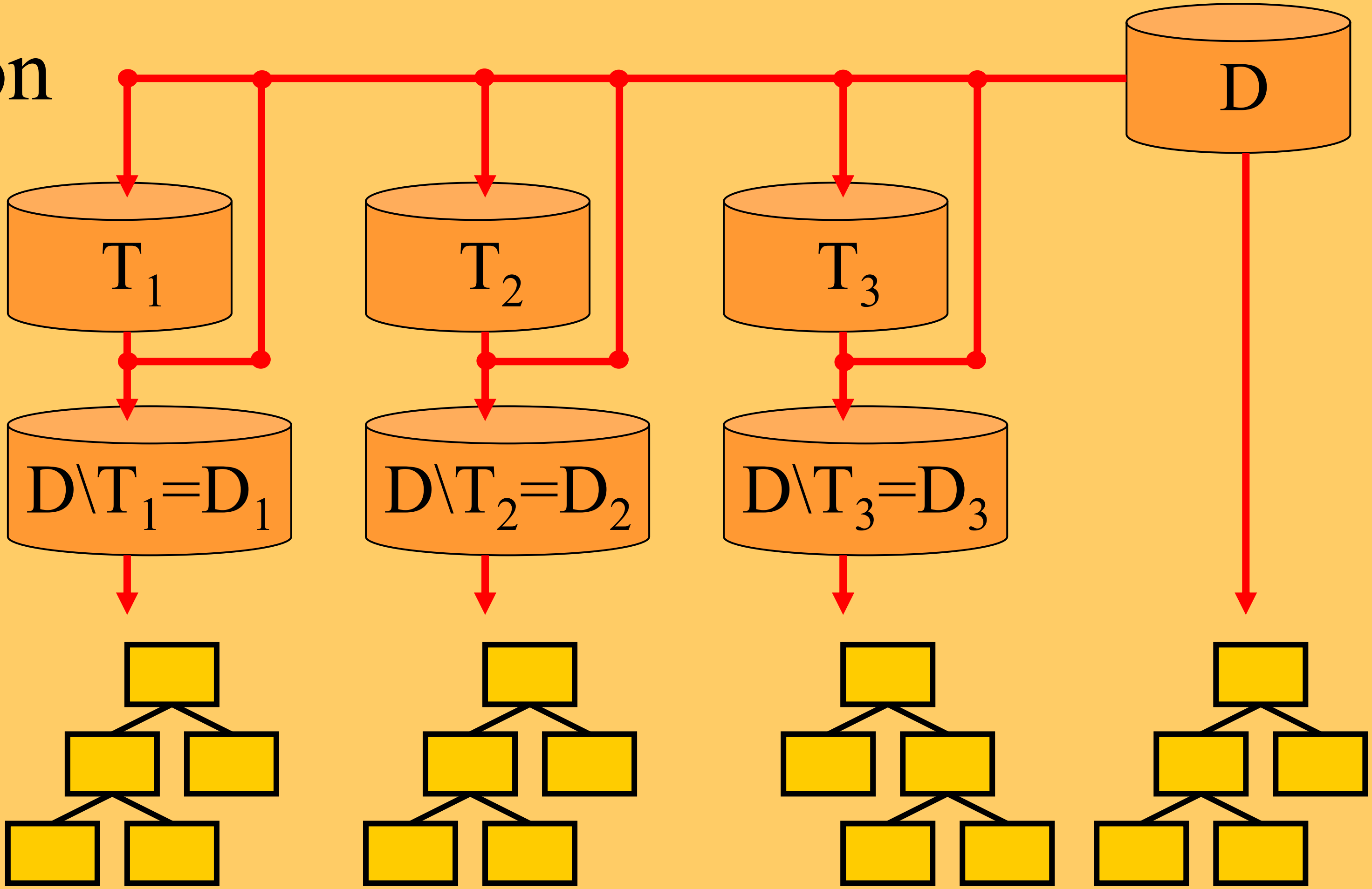
•Partition



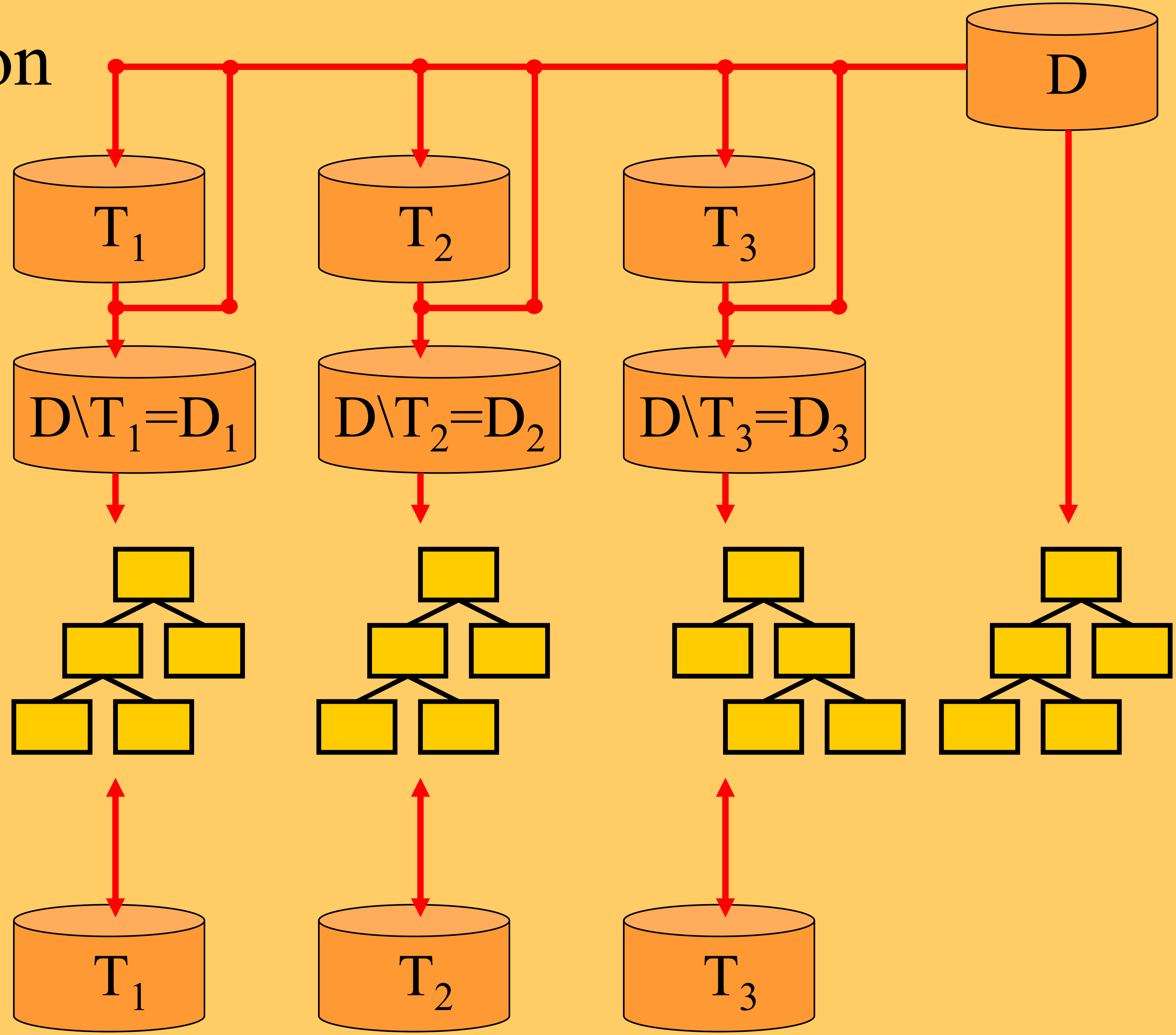
- Partition
- Train



- Partition
- Train



- Partition
- Train
- Test



(In)Accuracy

- Suppose two classifiers both achieve 80% accuracy on an evaluation dataset, are they always equally good?
  - e.g., classifier 1 correctly classifies 40 out of 50 positives and 40 out of 50 negatives; classifier 2 correctly classifies 30 out of 50 positives and 50 out of 50 negatives
  - on a test set which has more negatives than positives, classifier 2 is preferable;
  - on a test set which has more positives than negatives, classifier 1 is preferable; unless…
  - …the proportion of positives becomes so high that the ‘always positive’ predictor becomes superior!
- Conclusion: accuracy is not always an appropriate 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.
  - TP_1 = 40/50 = 80%
  - TP_2 = 30/50 = 60%
- ***False positive rate*** = #false pos. / #neg.
  - FP_1 = 10/50 = 20%
  - FP_2 = 0/50 = 0%
- ***ROC space*** has FP rate on X axis and TP rate on Y axis

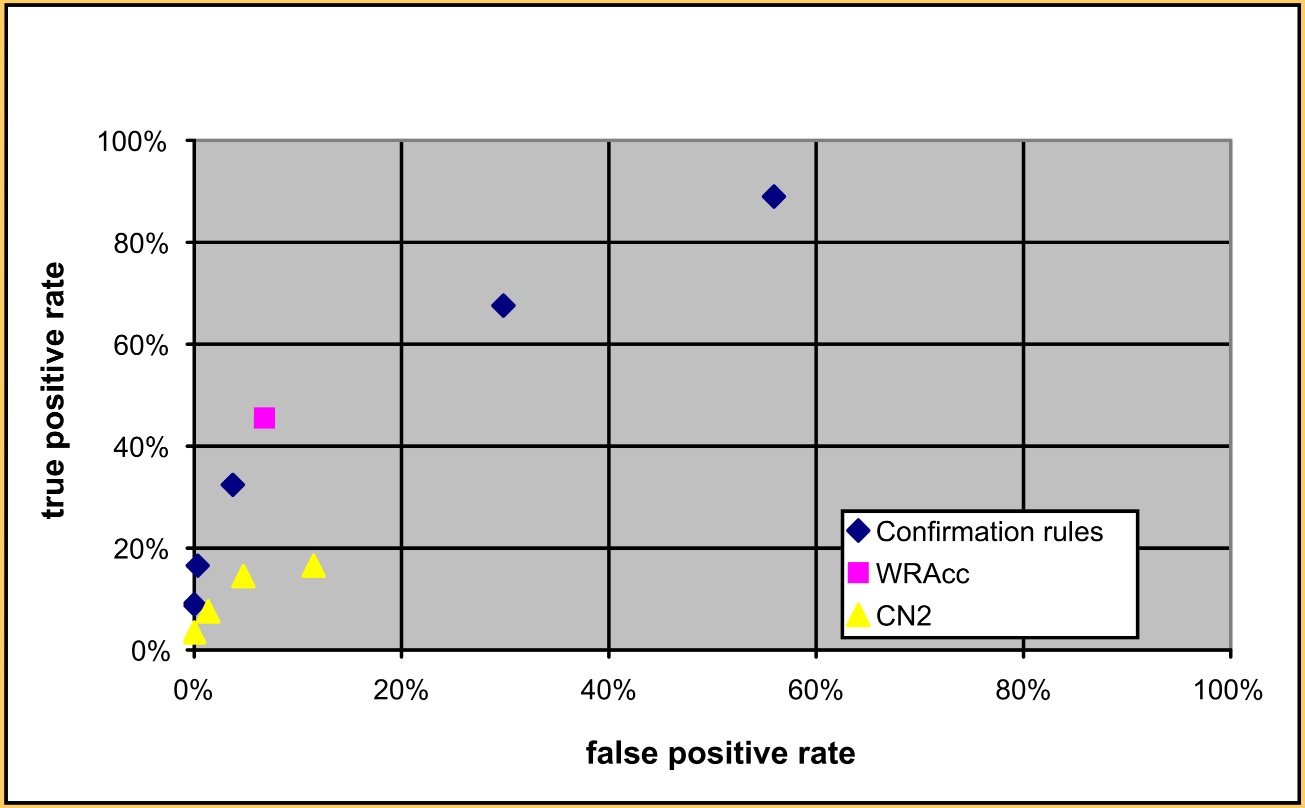
Classifier 1

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

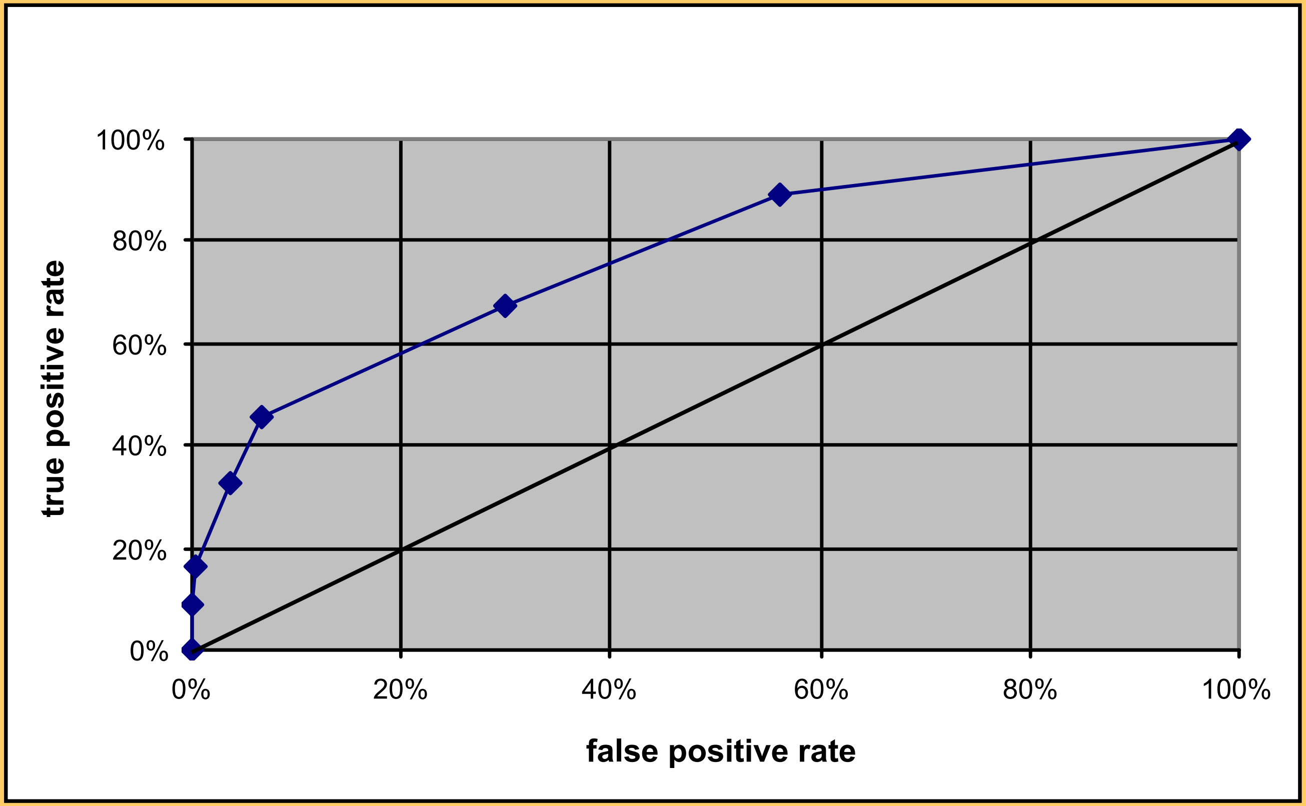
Classifier 2

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

The ROC convex hull



The ROC convex hull



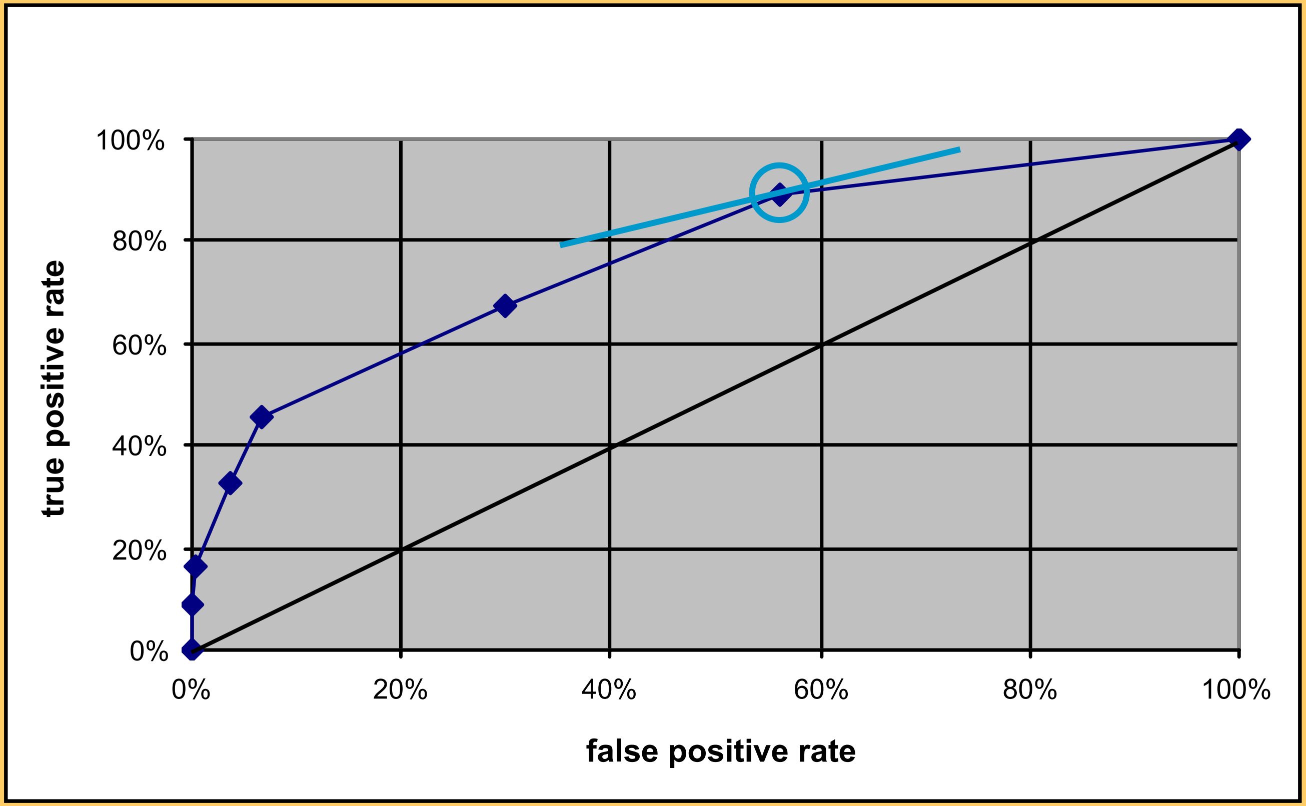
Choosing a classifier

$$\frac{FPcost}{FNcost} = \frac{1}{2}$$

$$\frac{Neg}{Pos} = 4$$

$$slope = \frac{4}{2} = 2$$

Choosing a classifier



$$\frac{FPcost}{FNcost} = \frac{1}{8}$$

$$\frac{Neg}{Pos} = 4$$

$$slope = \frac{4}{8} = .5$$

Rule evaluation measures

- ***Coverage***

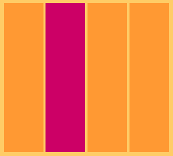
  $Cov(Cl \leftarrow Cond) = p(Cond)$

- ***Support* = frequency**

  $Sup(Cl \leftarrow Cond) = p(Cl.Cond)$

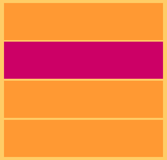
- ***Rule accuracy* = confidence = precision**

  $Acc(Cl \leftarrow Cond) = n(Cl.Cond)/n(Cond) = p(Cl \mid Cond)$



- ***Sensitivity* = recall of positives** (TPr)

  $Sens(Cl \leftarrow Cond) = n(Cl.Cond) / n(Cl) = p(Cond \mid Cl)$



- ***Specificity* = recall of negatives**

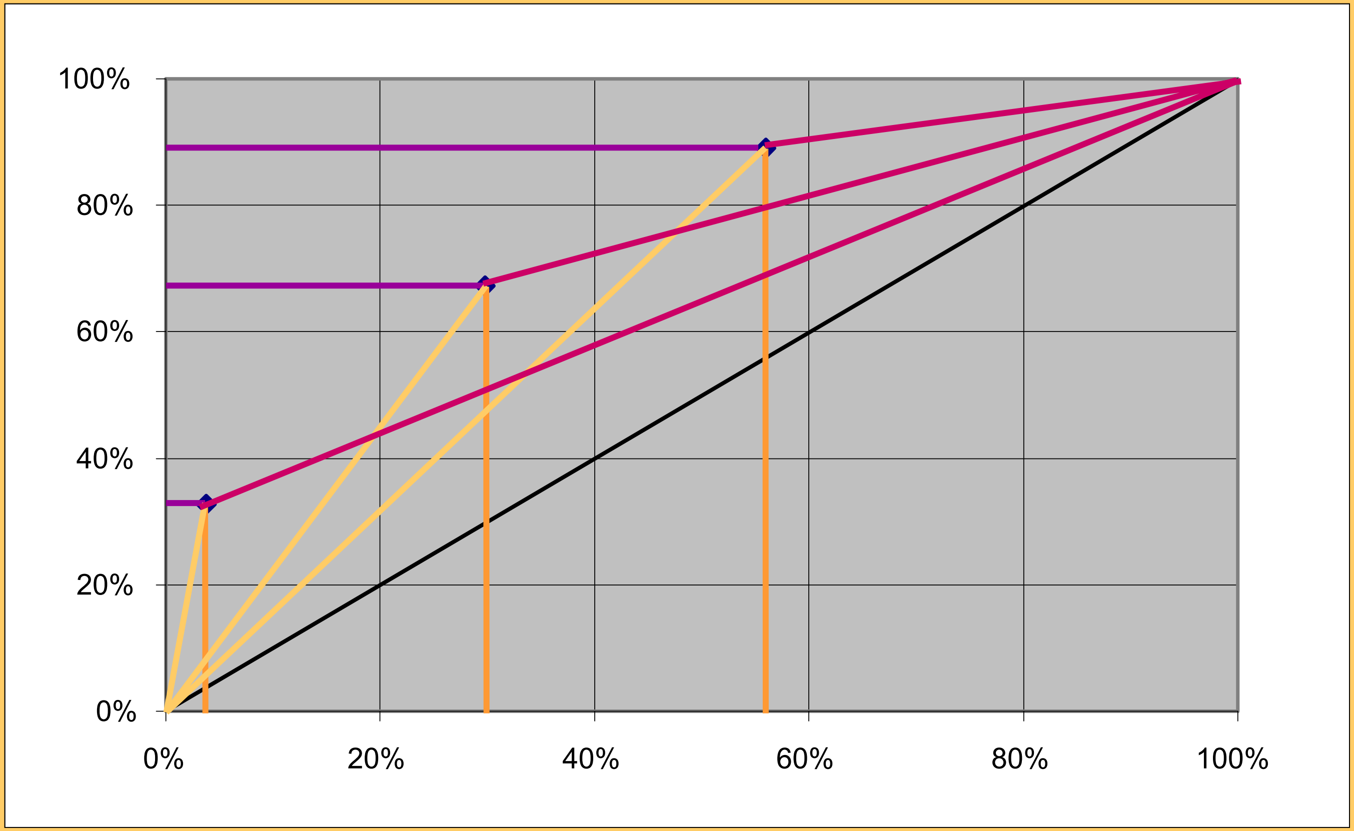
  $Spec(Cl \leftarrow Cond) = n(\neg Cl \neg Cond) / n(\neg Cl)$

  $= p(\neg Cond \mid \neg Cl)$

ML metrics in ROC space

true positive rate
= sensitivity
= recall
= TP/Pos

rule accuracy
= confidence
= precision
= TP/(TP+FP)



accuracy on
negatives
= TN/(TN+FN)

false positive rate =
1 – specificity
= 1 – TN/Neg =
FP/Neg

Part III: Summary

- 10-fold cross-validation is a standard classifier evaluation method used in machine learning
- ROC analysis very natural for rule learning and subgroup discovery
  - can take costs into account
  - here used for evaluation
  - also possible to use as search heuristic
- Upgrade to c>2 classes
  - full ROC analysis requires c(c–1) dimensions, distinguishing all pairwise misclassification types
  - can be approximated by c dimensions

Part IV: Relational Data Mining

- 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	Income	Age	Club Status	Response
...	...	...	...	...	...	...	...	...	...	...	...
3478	Smith	John	38, Lake Dr	Sampleton	34677	male	single	i60-70k	32	member	no-response
3479	Doe	Jane	45, Sea Ct	Invention	43666	female	married	i80-90k	45	non-member	response
...	...	...	...	...	...	...	...	...	...	...	...

Basic customer table.

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

Customer table for analysis.

ID	Zip	Sex	So St	In come	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
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
...	...	...	...	...	...	...	...

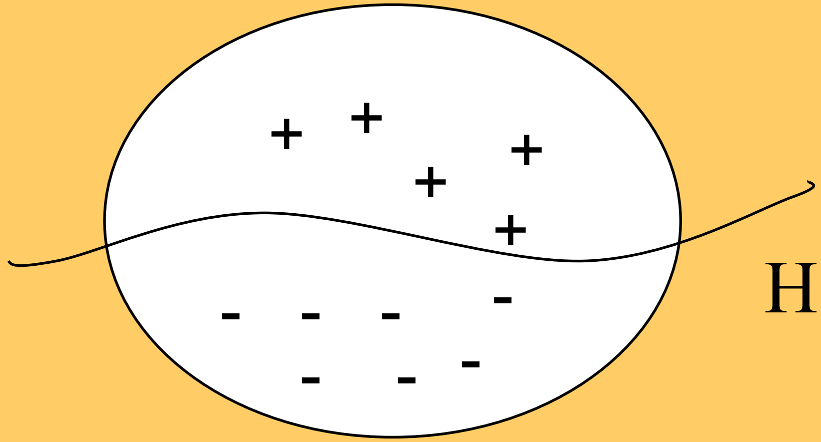
order				
Customer ID	Order ID	Store ID	Delivery Mode	Paymt Mode
...	...	...	...	...
3478	2140267	12	regular	cash
3478	3446778	12	express	check
3478	4728386	17	regular	check
3479	3233444	17	express	credit
3479	3475886	12	regular	credit
...	...	...	...	...

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

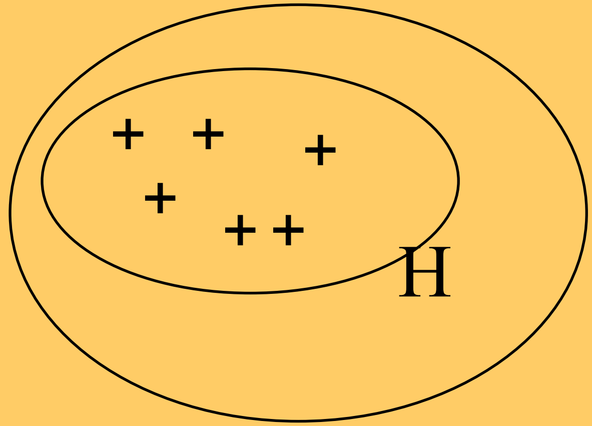
Relational representation of customers, orders and stores.

Basic Relational Data Mining tasks

Predictive RDM

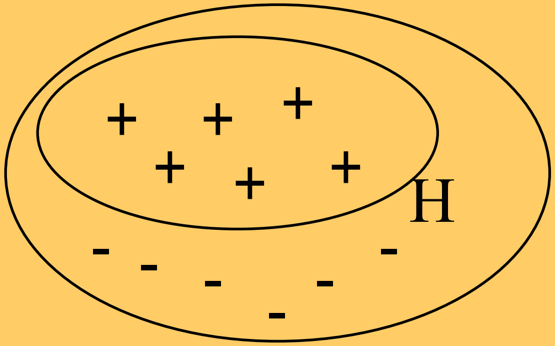


Descriptive RDM



Predictive ILP

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



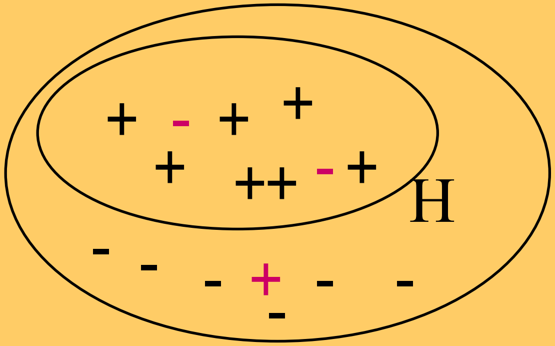
- **Find:**
  A hypothesis $H \in L_H$, such that (given B) H covers all positive and no negative examples

- In logic, **find** H such that
  - $\forall e \in E^+ : B \wedge H \models e$ (H is complete)
  - $\forall e \in E^- : B \wedge H \not\models e$ (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)

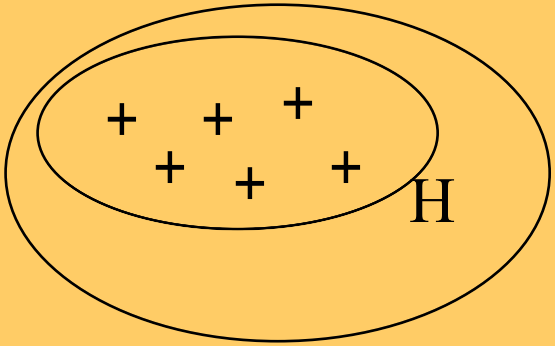


(instead of finding a hypothesis $H \in L_H$, such that (given *B*) *H* covers all positive and no negative examples)

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



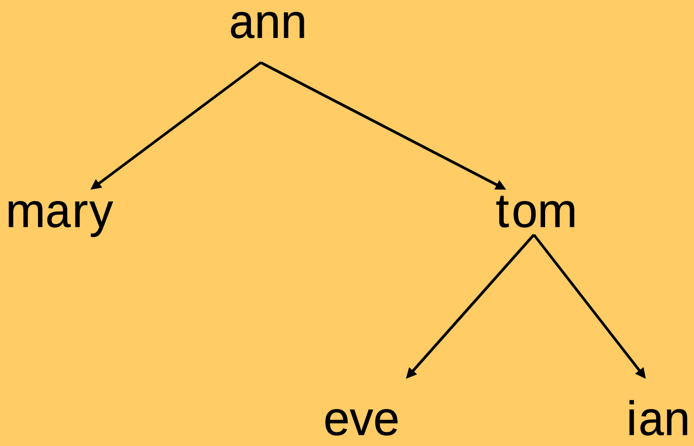
- In logic, **find** H such that $\forall c \in H$, c is true in some preferred model of $B \cup E$ (e.g., least Herbrand model $M(B \cup E)$)

- **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

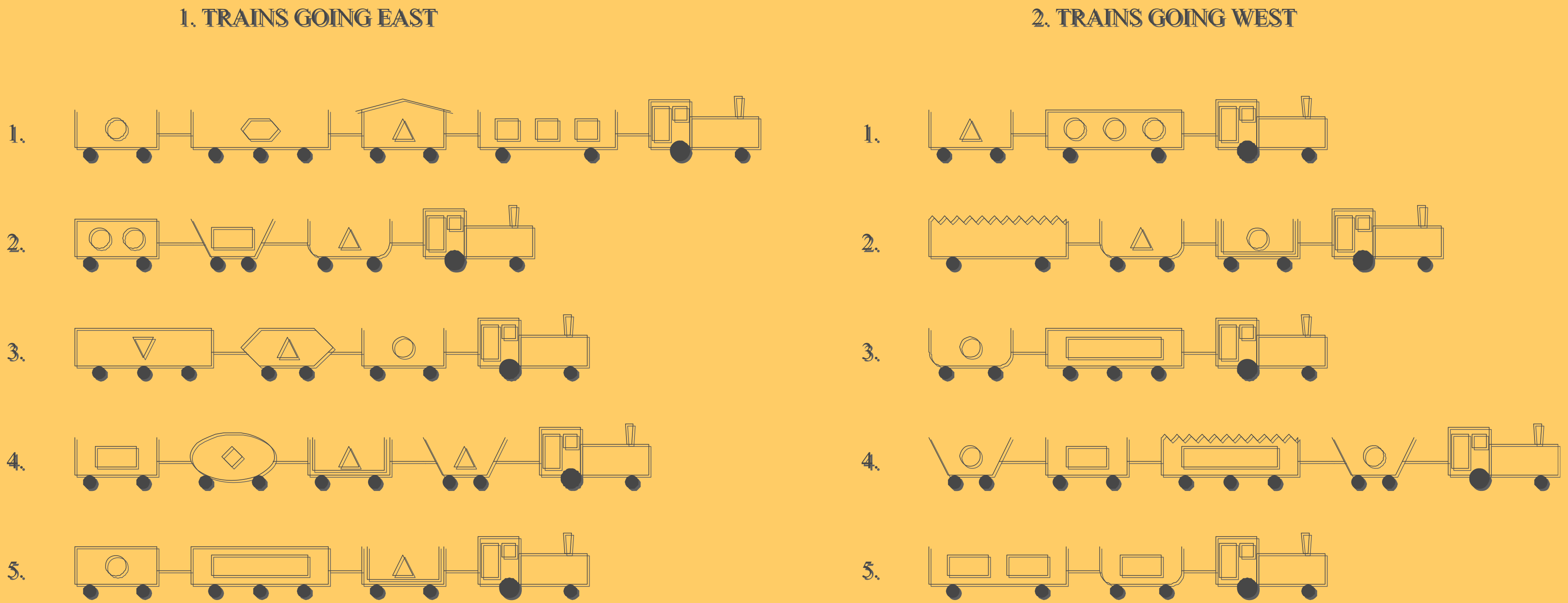
- **Predictive ILP**

```
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

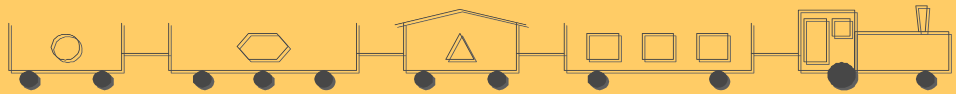
LOAD	CAR	OBJECT	NUMBER
l1	c1	circle	1
l2	c2	hexagon	1
l3	c3	triangle	1
l4	c4	rectangle	3
…	…	…	

TRAIN_TABLE

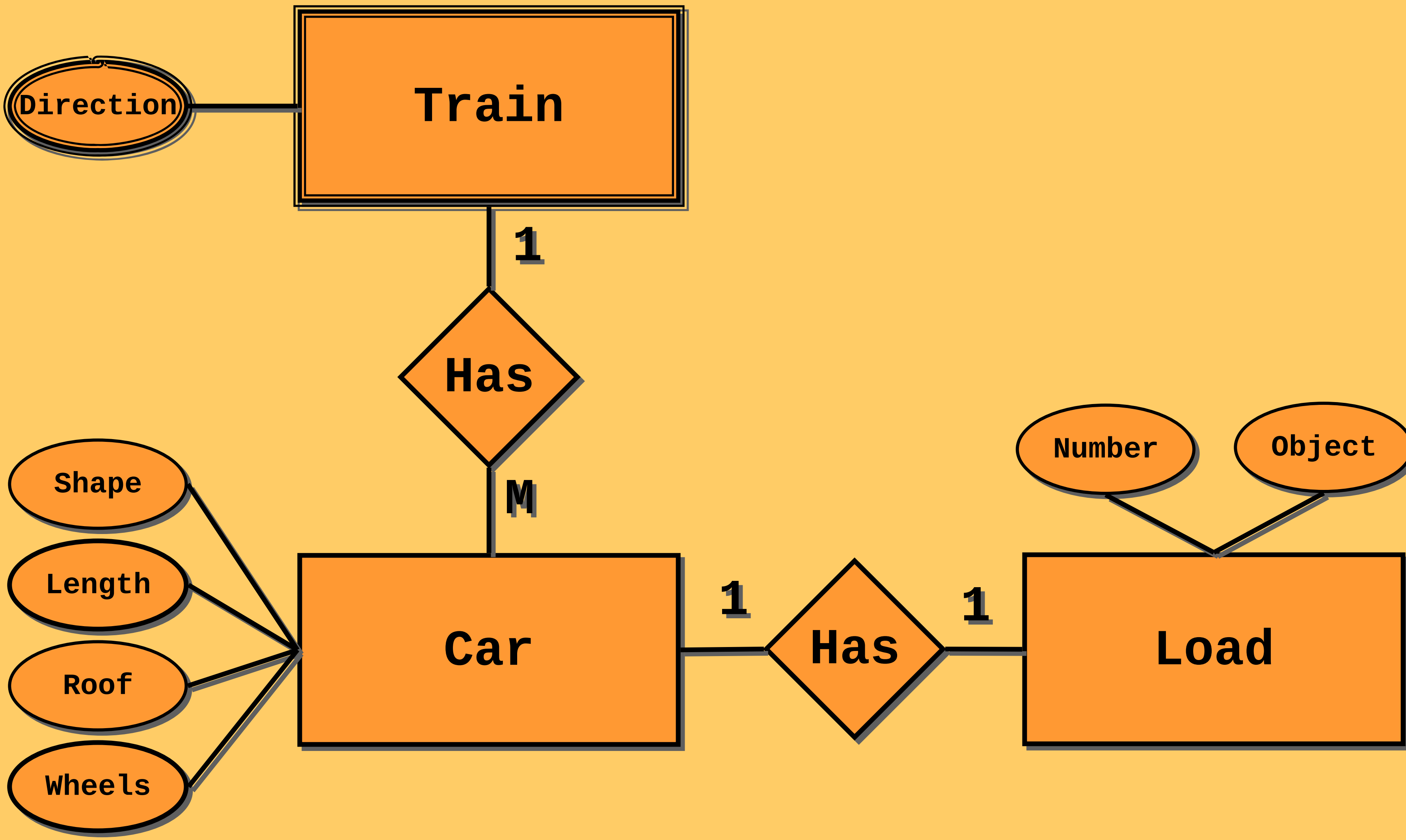
TRAIN	EASTBOUND
t1	TRUE
t2	TRUE
…	…
t6	FALSE
…	…

CAR_TABLE

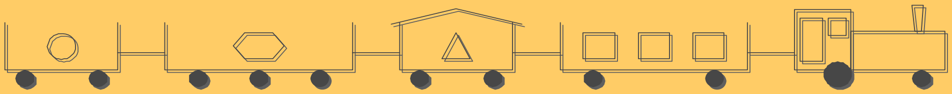
CAR	TRAIN	SHAPE	LENGTH	ROOF	WHEELS
c1	t1	rectangle	short	none	2
c2	t1	rectangle	long	none	3
c3	t1	rectangle	short	peaked	2
c4	t1	rectangle	long	none	2
…	…	…			…



ER diagram for East-West trains



ILP representation: Datalog ground facts



- Example:
  eastbound(t1).

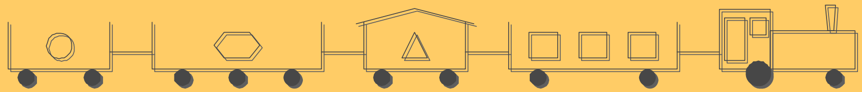
- Background theory:

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):

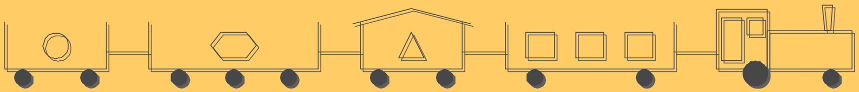
eastbound(T) :- car(T,C),short(C),not none(C).

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 IV: Relational Data Mining

- 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
$\oplus$	sue	eve	true	true	false	false	true	false	false
$\oplus$	ann	pat	true	false	false	false	true	false	false
$\ominus$	tom	ann	false	true	false	false	true	false	false
$\ominus$	eve	ann	true	true	false	false	false	false	false

- **Result of propositional rule learning:**

  Class = $\oplus$ 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).

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 IV: Relational Data Mining

- What is RDM?
- Propositionalization techniques
- Inductive Logic Programming
  - ILP as search
  - ILP techniques and implementations
    - Propositionalisation (LINUS, RSD)
    - Specialization techniques (MIS, FOIL, …)
      - Top-down search of refinement graphs
    - Generalization techniques (CIGOL, GOLEM)
      - Inverse resolution
      - Relative least general generalization
    - Combining top-down and bottom-up
      - Inverse entailment (PROGOL)

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

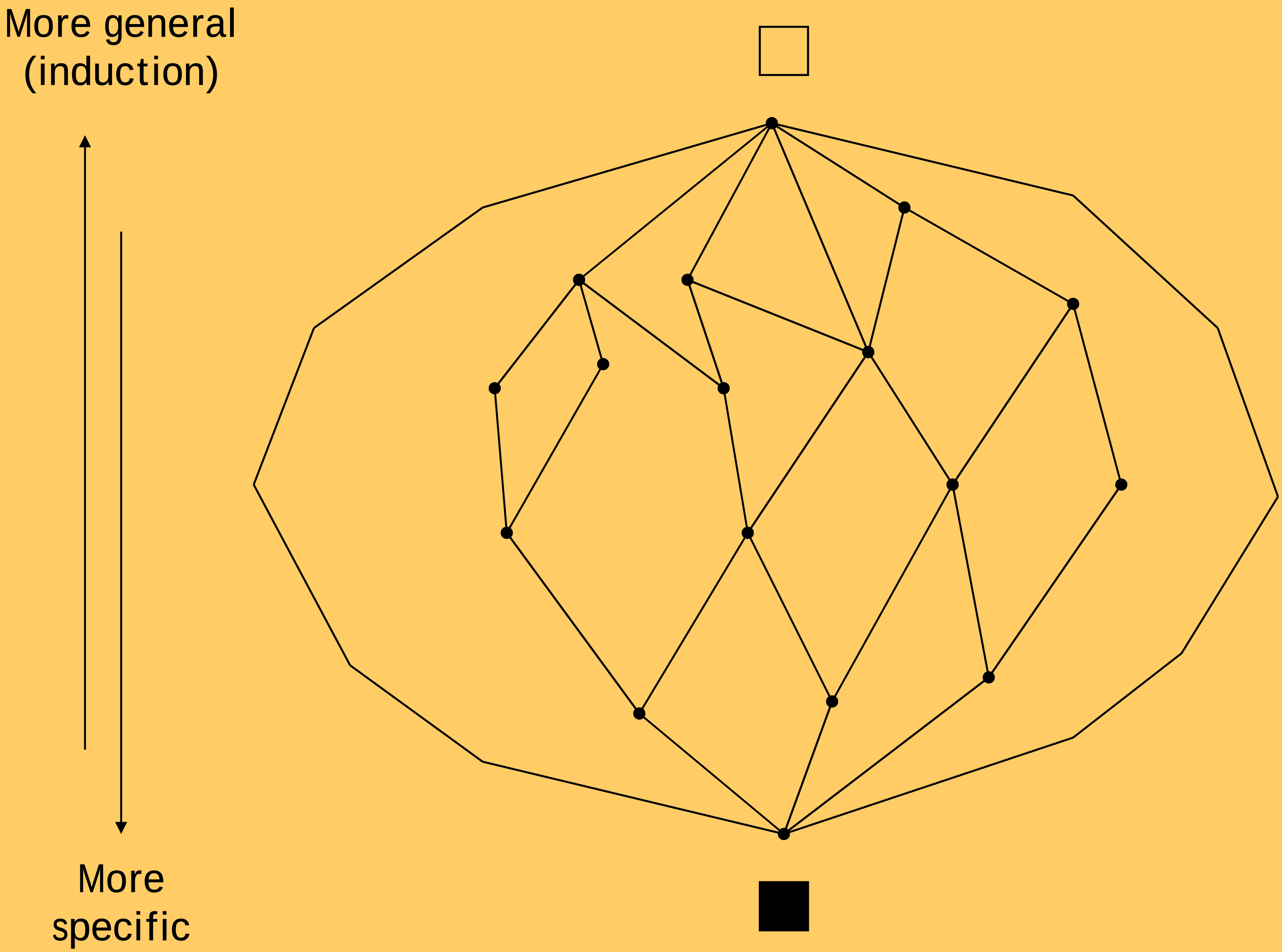
- **Semantic generality**
  Hypothesis H_1 is semantically more general than H_2 w.r.t. background theory B if and only if $B \cup H_1 \models H_2$
- **Syntactic generality or θ-subsumption**
  (most popular in ILP)
  - Clause c_1 θ-subsumes c_2 ($c_1 \geq_\theta c_2$)
    if and only if $\exists \theta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

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).	

daughter(X,Y) ←

daughter(X,Y) ← *X=Y*

…

daughter(X,Y) ← *parent(Y,X)*

…

daughter(X,Y) ← *parent(X,Z)*

daughter(X,Y) ← *female(X)*

daughter(X,Y) ← *female (X)* *female(Y)*

…

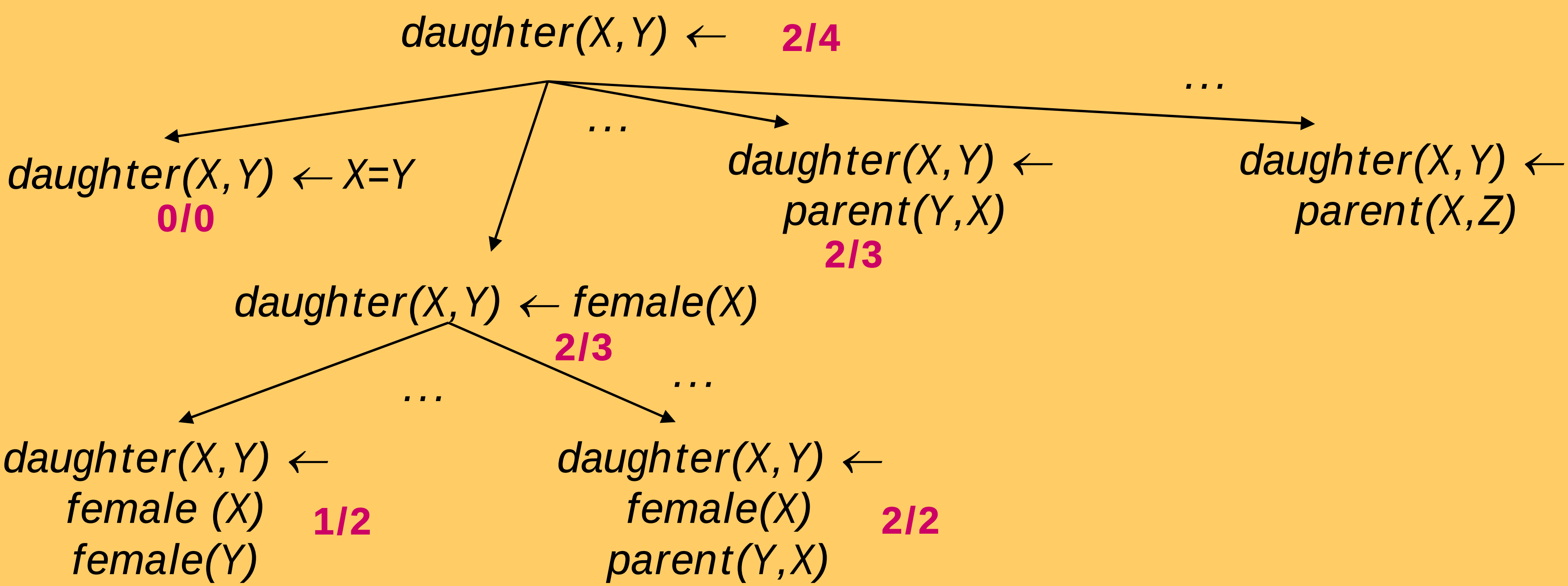
daughter(X,Y) ← *female(X)* *parent(Y,X)*

…

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 IV: 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

Part V:
Conclusions and Literature

Machine Learning and Statistics

- Both areas have a long tradition of developing inductive techniques for data analysis.
  - reasoning from properties of a data sample to properties of a population
- KDD = statistics + marketing ? No !
- KDD = statistics + ... + machine learning
- Use statistics for hypothesis testing and data analysis where many assumptions hold
  - about data independence, data distribution, random sampling, etc.
- Use machine learning hypothesis generation, possibly from small data samples

DM and Statistics …

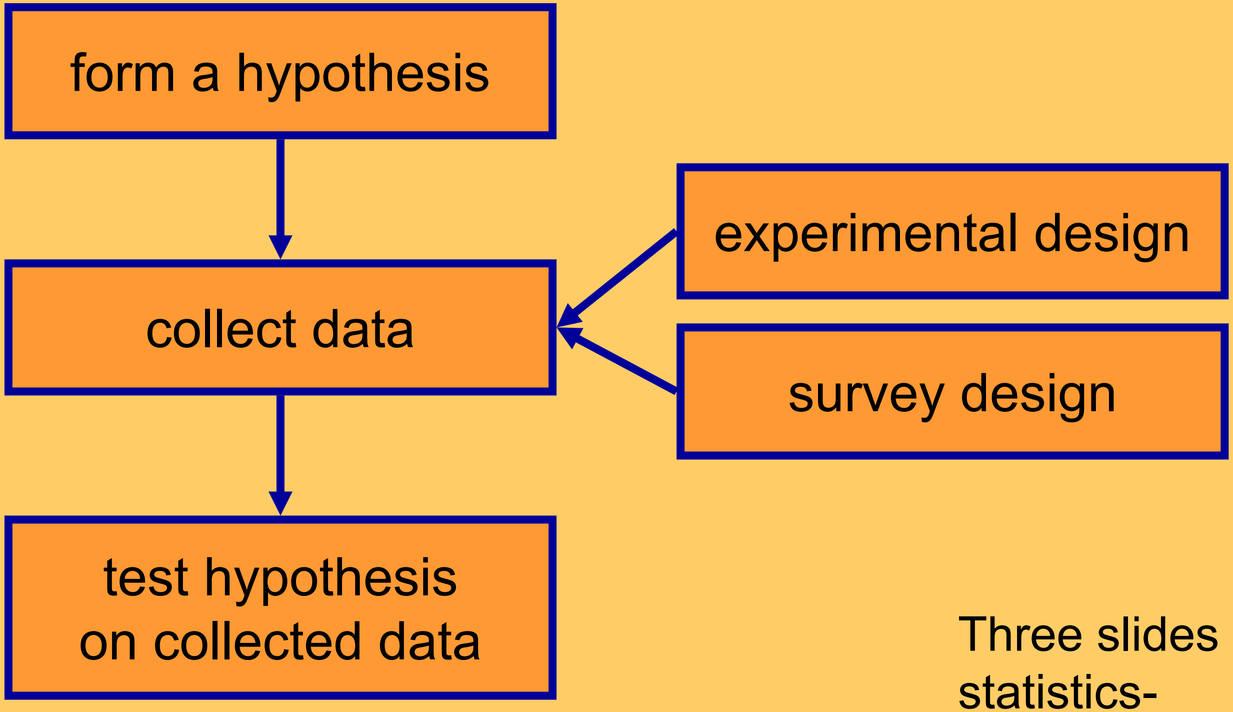
- KDD a broader view: provide tools to automate the entire process of data analysis, including statistician's art of hypothesis selection

  [Fayyad et al., *Comm ACM*]

- Eventually, what is done in DM could be done with statistics. Attractive in DM is the relative ease with which new insights can be gained (though not necessary interpreted)

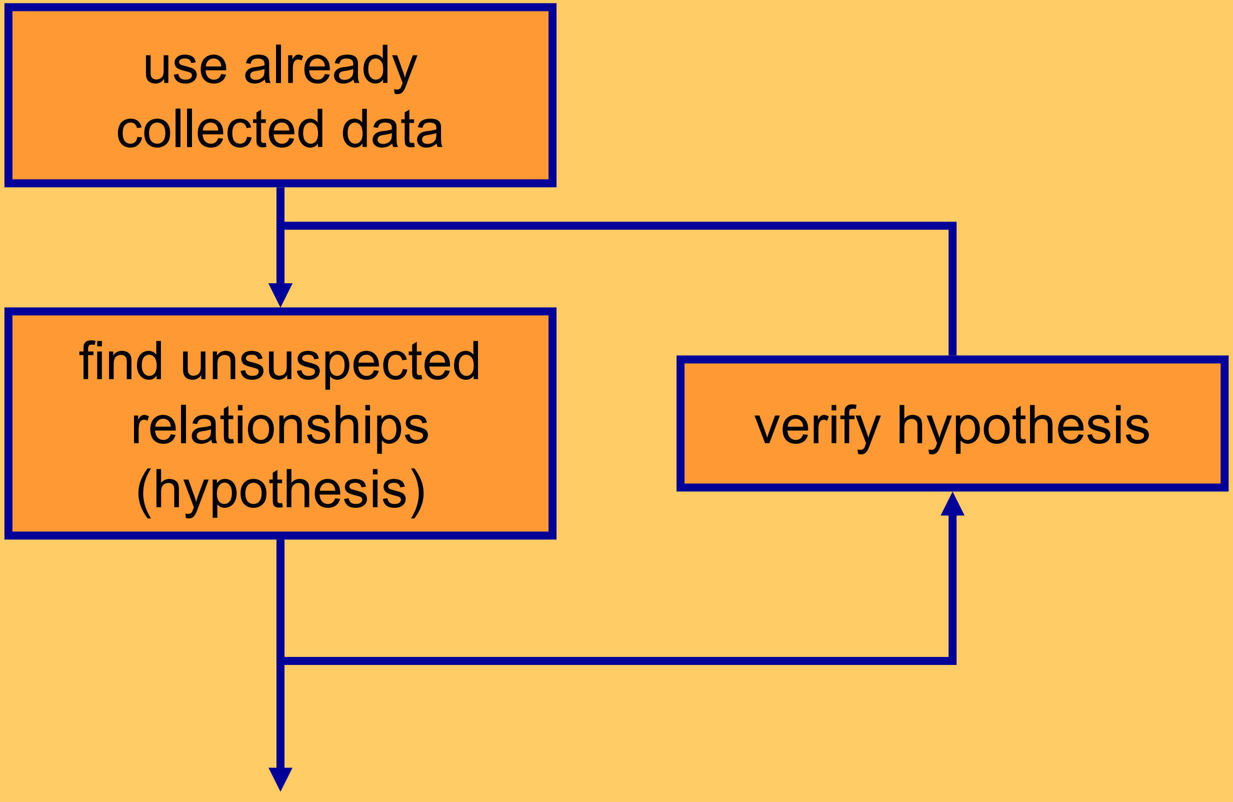
  [P Cabena et al., *Discovering data mining: from concept to implementation*, 1997]

Statistics:
Primary Data Analysis

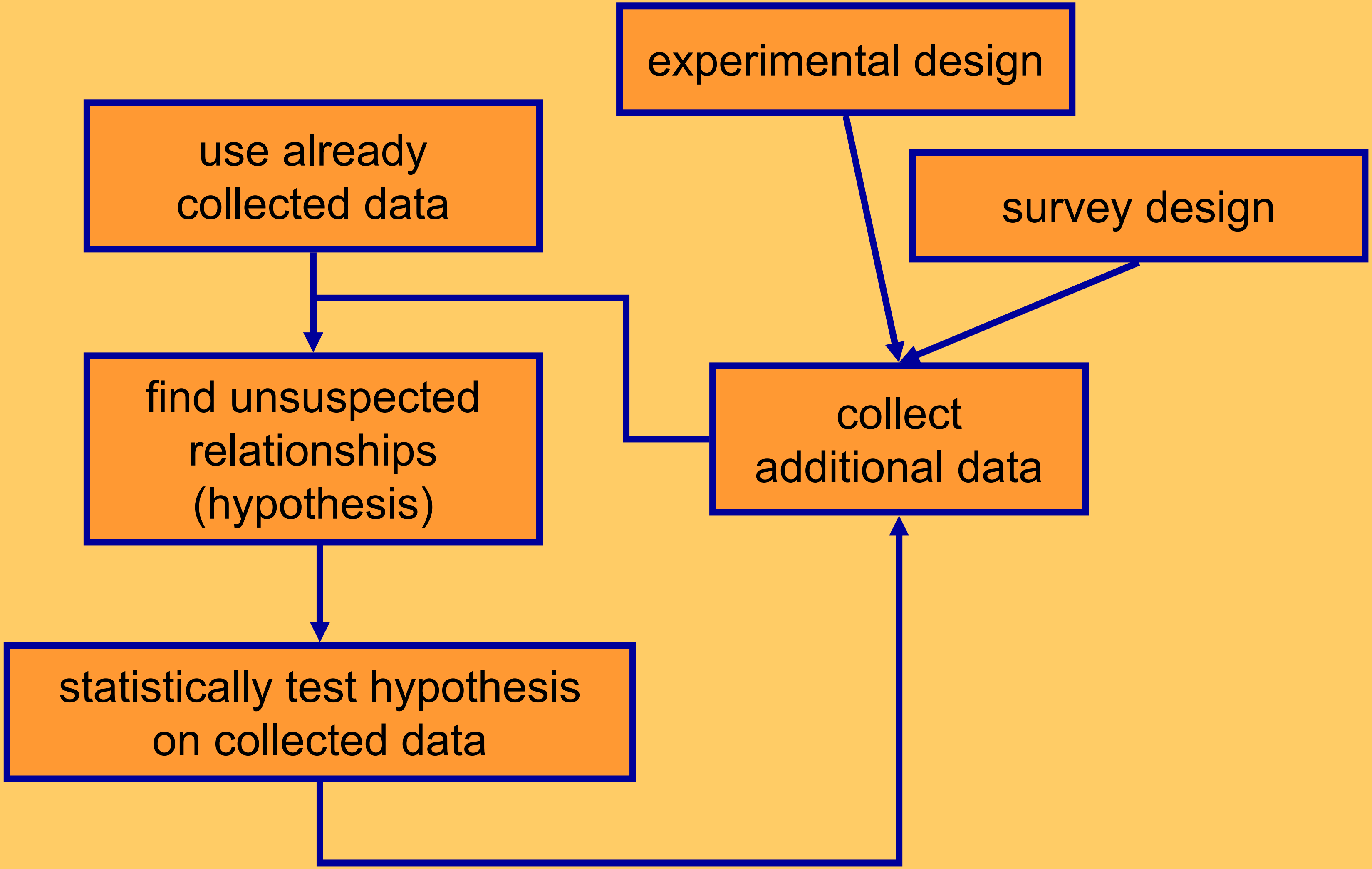


Three slides on statistics-machine learning relationship by Blaž Zupan

Data Mining: Secondary Data Analysis



Data analysis with DM and Statistics



Summary: Statistics vs. ML

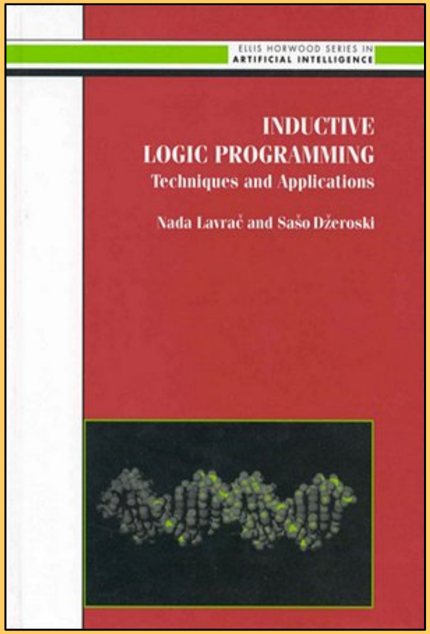
- Statistics and Machine Learning have long histories of developing inductive techniques for data analysis
- Statistics is particularly good when certain theoretical expectations about the data distribution, independence, random sampling, etc. are satisfied
- Machine Learning and Data Mining are particularly good when requiring generalizations that consist of easily understandable patterns

Literature: Rule induction and ILP

- Chapter “Rule Induction” by P. Flach and N. Lavrač in the book “Intelligent Data Analysis”, edited by Michael Berthold and David Hand , Springer 2003 (2nd edition)

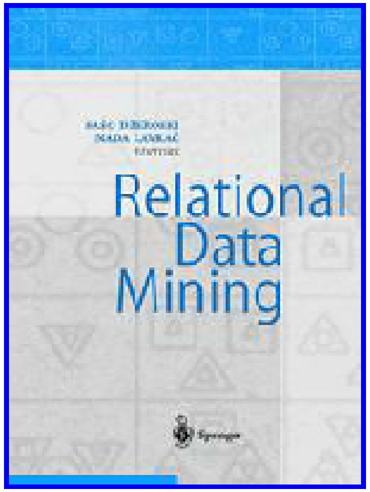
ILP: Techniques and Applications, Ellis Horwood 1994

- Description of LINUS and standard ILP techniques
- book by Lavrac and Dzeroski available at http://www-ai.ijs.si/SasoDzeroski/ILPBook/



Relational Data Mining, Springer 2001

- Recent developments in propositionalization (revisited LINUS and much more) – a chapter in RDM book
- http://www-ai.ijs.si/SasoDzeroski/RDMBook/



Relational Data Mining

Saso Dzeroski and Nada Lavrac, editors

Springer, Berlin, 2001

Front matter (foreword by Heikki Mannila , preface)

Table of contents (as it appears in the book - PDF, with abstracts - HTML)

Buy this book from Springer.

Acknowledgments

- Colleagues:
  - Peter Flach, Dragan Gamberger, Sašo Džeroski, Blaž Zupan (joint work, some slides borrowed)
  - Marko Grobelnik, Dunja Mladenić (Sol-Eu-Net)
- Funding agencies:
  - MVZT, EC (project Sol-Eu-Net)