Machine learning and data mining : An introductory overview

Hendrik Blockeel
KU Leuven, Department of Computer Science
& LIACS, Leiden University

Machine learning, data mining

- Machine learning & data mining are related fields -- we will not distinguish them here
- Both are very broad
- This talk aims at giving an overview of tasks, representations, and types of approaches

Machine learning, data mining

Machine learning :
Programs learn from experience
- analyze data about previous attempts to execute some task successfully / optimally
- draw conclusions
- incorporate these in the program that executes the task

Data mining :
Find patterns in data that can lead to knowledge
- analyze data in a (large) database
- find patterns
- interpret these patterns

Main focus is on analysis of data

Problem settings

- Problems vary with respect to:
  - the task
  - the format of the input (data/knowledge)
  - the format of the output
- These characteristics determine the set of learning methods that can be used
- Solution approaches further vary with respect to flexibility of use

1. Tasks

- Predictive learning (classification, regression)
- Identifying structure in the data (clustering, frequent pattern discovery)
- Building a model of the data

A. Predictive learning

- 2 types of predictive learning:
  - inductive:
    - learn a "predictive function" f:X->Y
    - f can later be used to predict Y from X for new instances
  - transductive: makes predictions directly from the data
- very different methods for these two

What kind of predictions?

- If Y (variable to predict) is
  - nominal: "classification"
  - numerical: "regression"
  - tuple (nominal/numerical components) or vector (numerical): "multi-target"
  - set of nominal values: "multi-label"
  - structured (time series, graph, ...): "structured output prediction"

The classic example: Classification

Outlook	Temp.	Humid.	Wind	Tennis?
Sunny	30	85	yes	no
Sunny	22	60	no	yes
Cloudy	21	65	yes	yes

Cloudy	25	65	yes	?

f(Cloudy,2,65,yes)

Regression

Outlook	Temp.	Humid.	Wind	Tennis?
Sunny	30	85	yes	no
Sunny	22	60	no	yes
Cloudy	21	65	yes	yes

Cloudy	?	65	yes	no

Multitarget regression

Outlook	Wind	Tennis	Temp.	Hum.
Sunny	yes	no	30	85
Sunny	no	yes	22	60
Cloudy	yes	yes	21	65

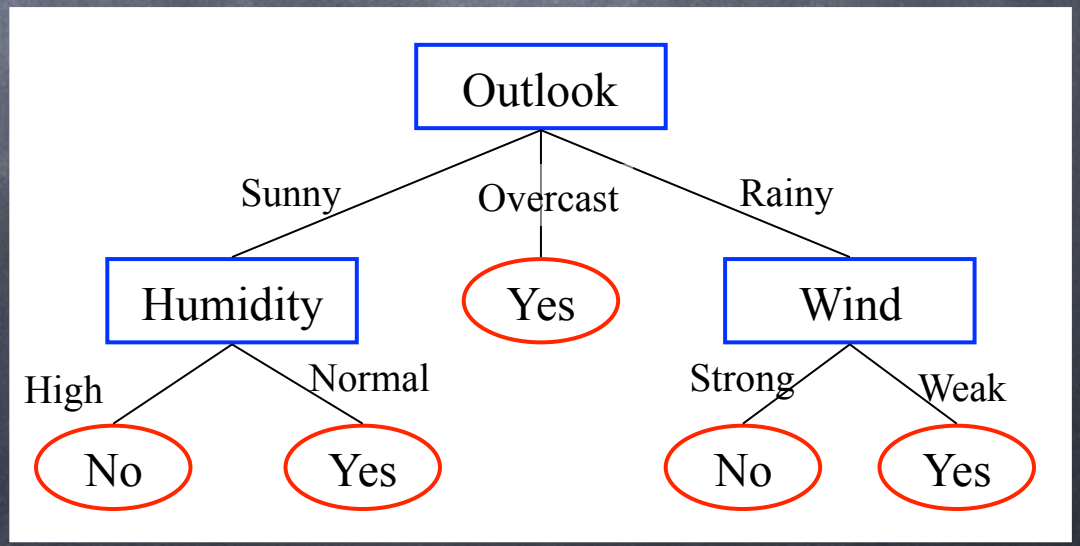
Cloudy	yes	yes	?	?

1) Induction : Learning a predictive function

- Given a dataset consisting of pairs (X_i,Y_i), learn a function f that maps X to Y, consistently with the dataset
- "consistently": typically not 100% consistent; minimize some loss function L
  - ideally, L reflects "error" on population
  - practically, L combines error on dataset with "complexity" of f
- main question: how to represent f? (see later)

Representations for predictive functions

- decision trees
- neural networks
- SVMs
- rule sets
- Naive Bayes model
- linear equation
- ...



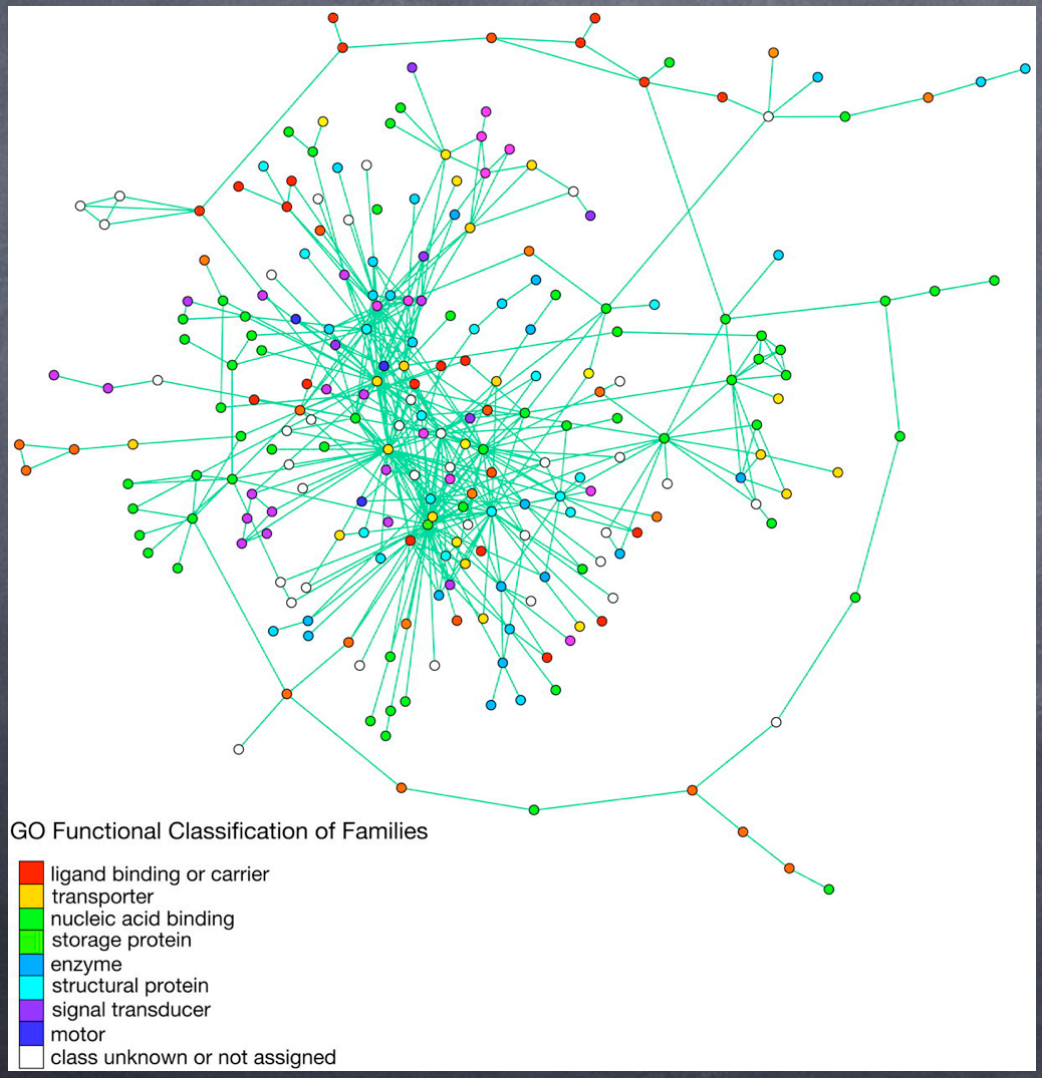
2) Transduction

- Given a dataset D consisting of (X,Y) pairs, where some values are missing, predict those values from D
- = "dataset completion", "imputation"
- Special case: always the same attribute is missing (the "target attribute")
- E.g.: recommender systems, protein function prediction, ...

Example: recommender systems

	I1	I2	I3	I4	I5	I6	I7	I8	I9
U1	?			5		1		3	
U2		1	5	5		2	5	?	
U3	2		?	2			2	5	
U4		3			5		3		
U5	3		3		4			4	
U6	?	?	?	?	5	1	?	?	?

Example: Protein function prediction



(from H. Rahmani)

Transduction methods

- nearest neighbor methods
  - find most similar known cases, predict unknown values to be similar to those
- matrix factorization
- ...

- Important point: analysis happens at the time of prediction ("lazy learning")

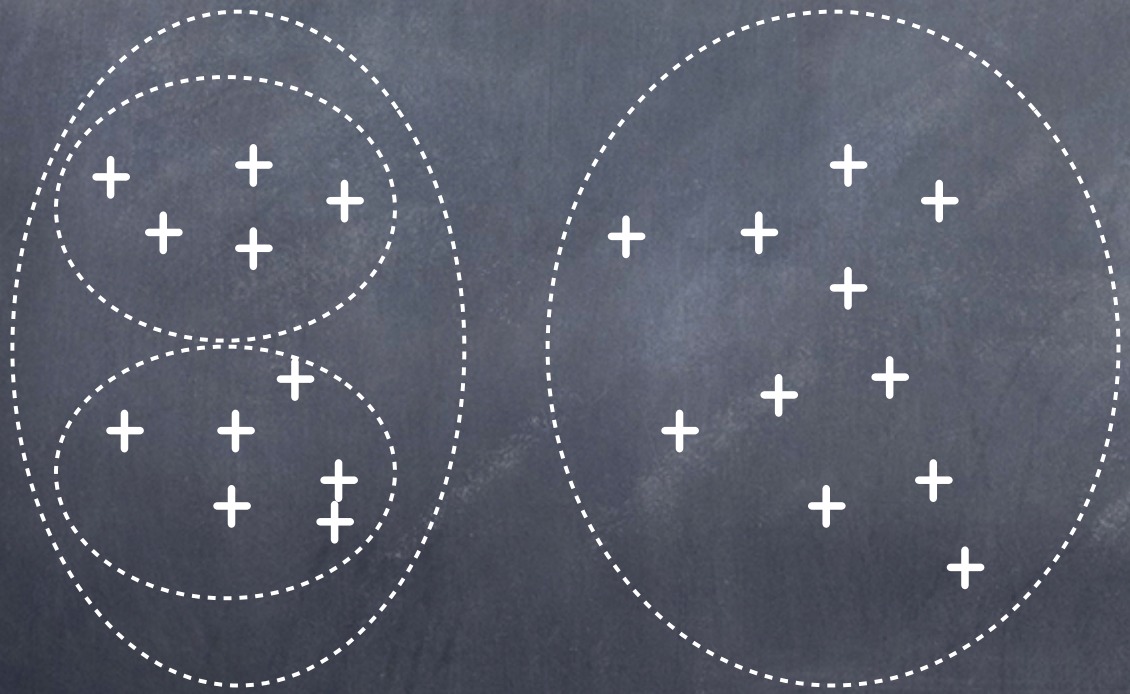
B. Identifying structure

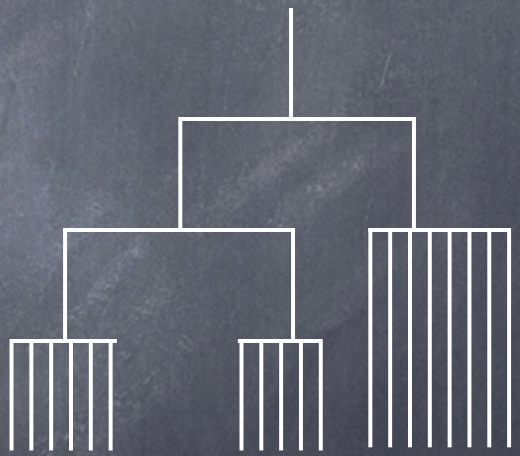
- Clustering
- Finding frequent patterns
- Finding low-dimensional structure (PCI, manifold learning, ...)
  - (illustrate PCI with image recognition)

Clustering

- Assuming data are not spread uniformly in the input space, try to identify "clusters" of data points
- Flat vs. hierarchical, extensional vs. intensional, ...

Clustering





Can describe clusters extensionally (listing elements), using Gaussians, ...

Outlier detection

- Find exceptions to the regular structure in the data
- Anomaly detection, outlier detection
- Used for, e.g., fraud detection
  - (well-known example: distribution of 1st digit in numbers related to financial transactions)

Frequent patterns

- A pattern is some local structure (on one or a few elements) that may frequently occur
  - e.g., itemsets: "bread and beer often bought together"
  - e.g., subsequences, subtrees, subgraphs that frequently occur

abdcbaccdaadcbdcbbdcadcbabbadcbdcbabdc

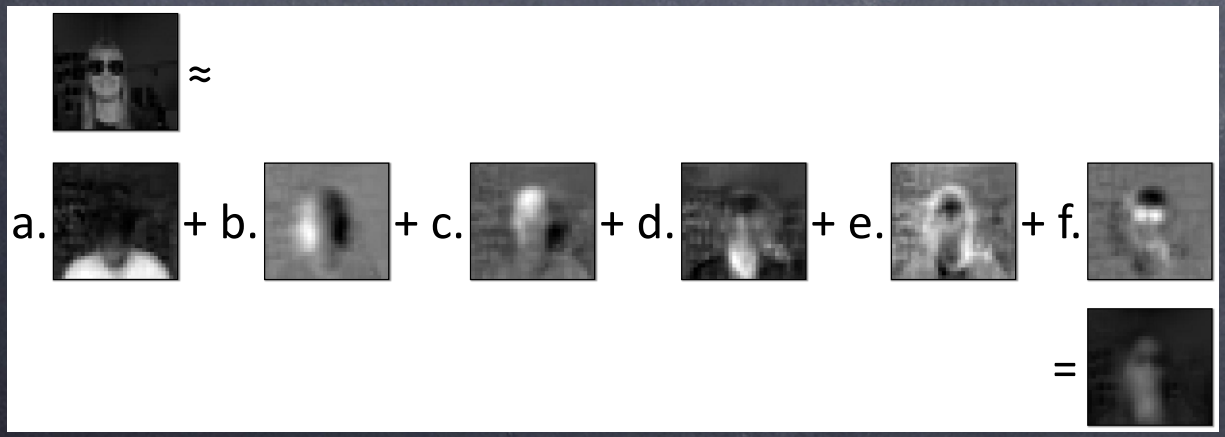
abdcbaccdaadcbdcbbdcadcbabbadcbdcbabdc

Dimensionality reduction

- Represent data in a (much) lower-dimensional space (e.g., visualize in 2 or 3 dimensions)
- Principal components analysis (PCA) : find plane with maximal "projected variance"
- Multi-dimensional scaling (MDS): represent in lower-D, preserving distances
- Manifold learning: similar, but subspace is non-linearly embedded in original space

Principle Component Analysis

- Analyze datasets to determine "directions" of principle components
- Decompose each data point into its principal components by projecting it onto those directions
- e.g.: face recognition



(from B. Verstrynge)

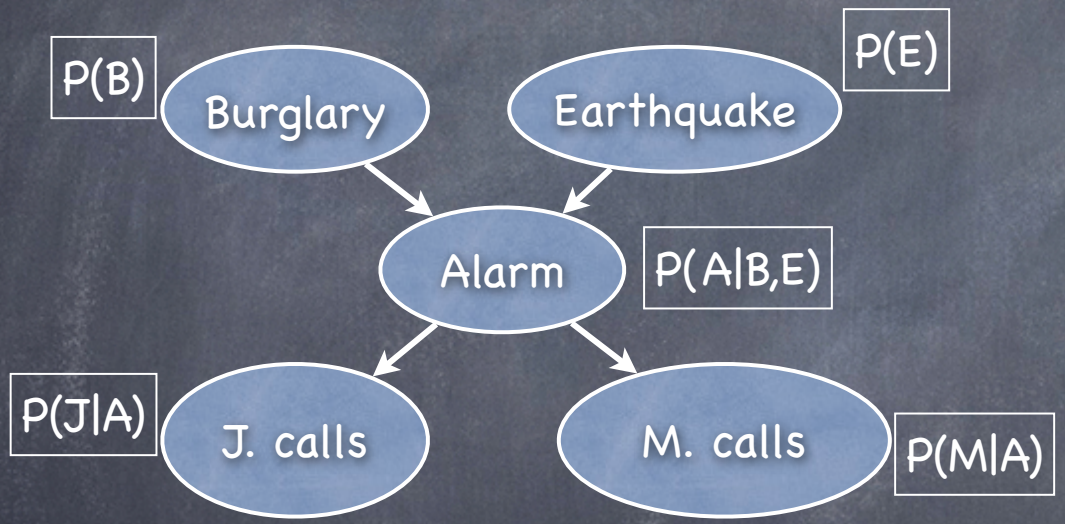
C. Building a model of the data

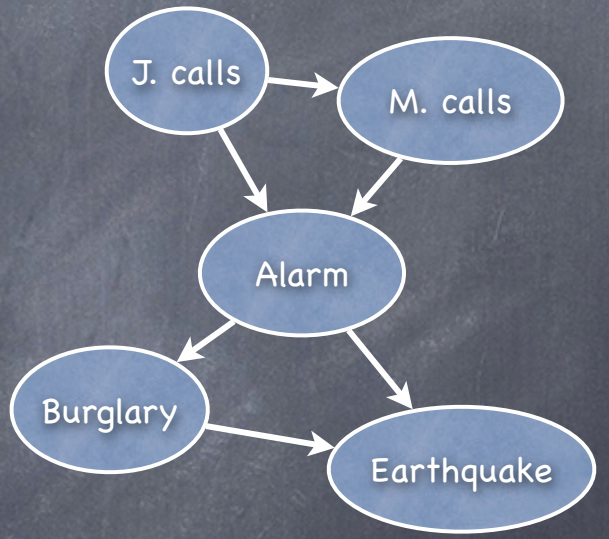
- The model typically allows us to
  - generate more data (generative model)
  - derive a predictive function
  - predict the occurrence of patterns
  - reason about data (what if ...)
- In some sense, this task generalizes over the previous ones (but: elicitation of knowledge from model may be just as difficult as from original data)

Probabilistic models

- Estimate the joint distribution of the data
- Important: need to decide on how to represent the answer
- Tabular representation, Bayesian network (directed), Markov network (undirected), ...

Bayesian networks: example

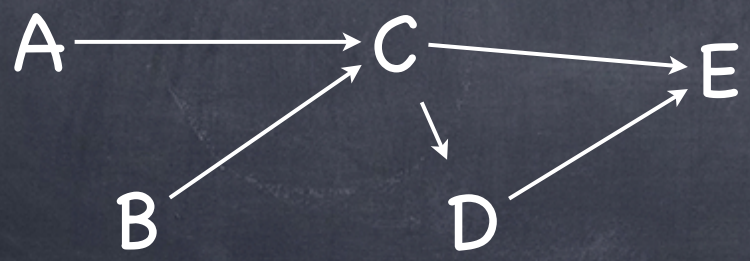




- tabulates (conditional) probabilities (CPD)
- joint prob. distr. (JPD) = product of all these
- compare to tabulating JPD: exponential in #vars

Causal structure

- Joint distribution models the population the data is drawn from
- Can tell us something about, e.g., correlations, but not causality
- There is some work on discovering the (likely) causal structure behind the data
- Note: Bayesian networks by themselves do **not** indicate causal structure



"Automatic programming"

- 80's, 90's: Some experiments with automatic construction of programs from examples
  - e.g., construct Prolog programs (sorting etc.) from I/O examples
  - genetic programming
- State of the art still limited to very simple programs, learnt with a lot of background knowledge (= auxiliary procedures/functions/predicates must be known)

```
qsort(X,Y) :- pivot(X,XL,XU), qsort(XL,YL), qsort(XU, YU), append(YL, YR, Y).
```

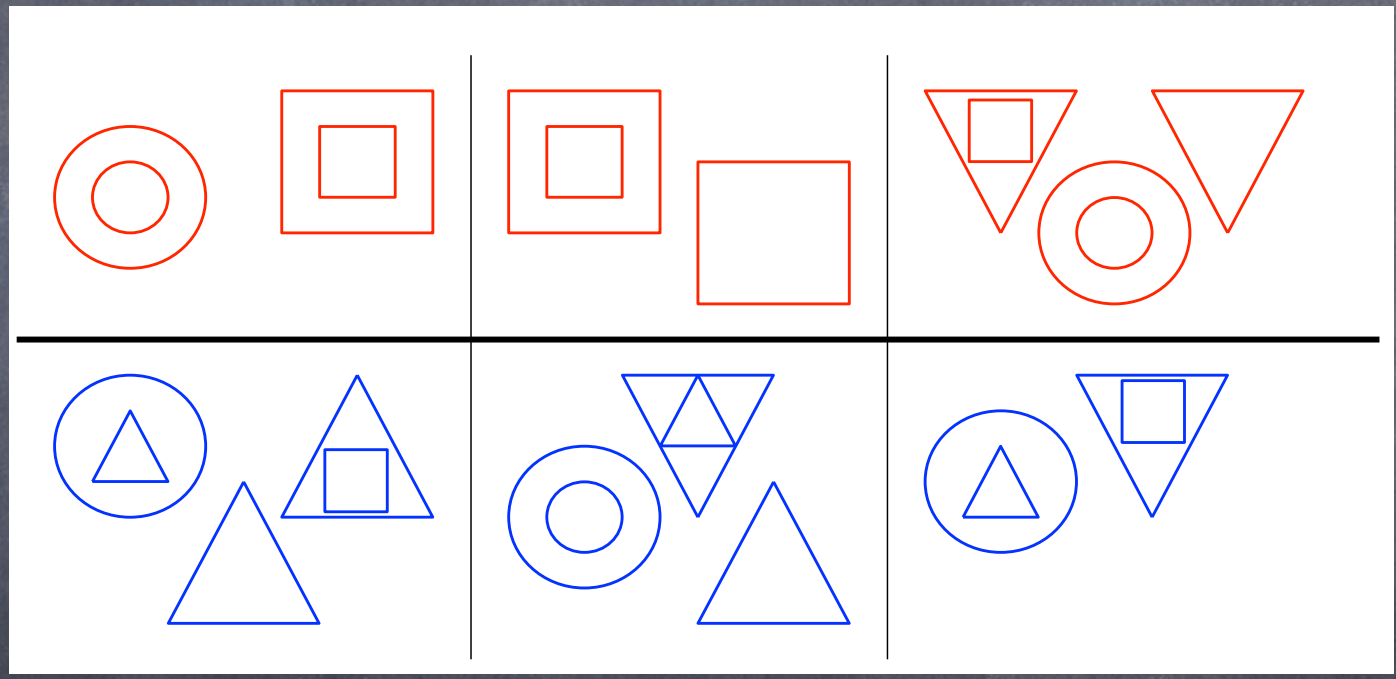
2. Input format

- The input of the learning system consists of
  - data: e.g., a set of examples
  - [background knowledge] : knowledge about the domain or about the problem
  - [task specification] : a more precise specification of a general task
  - [parameter settings] (method-dependent)

The data

- Standard format: a set of examples (X_i,Y_i) with X_i in R^n, Y_i in R^n (whole dataset = 1 table)
- elements are vectors or tuples (1 element = 1 row)
- typically assumed "iid" (independent and identically distributed)
- This does not cover cases where X or Y can have an internal structure (set, graph, ...) or are embedded in an external structure
- -> "relational learning"

Example: "Bongard problems"



- 1 drawing = set of objects: #elements not fixed, not ordered
- Information about one drawing cannot be embedded in a single tuple

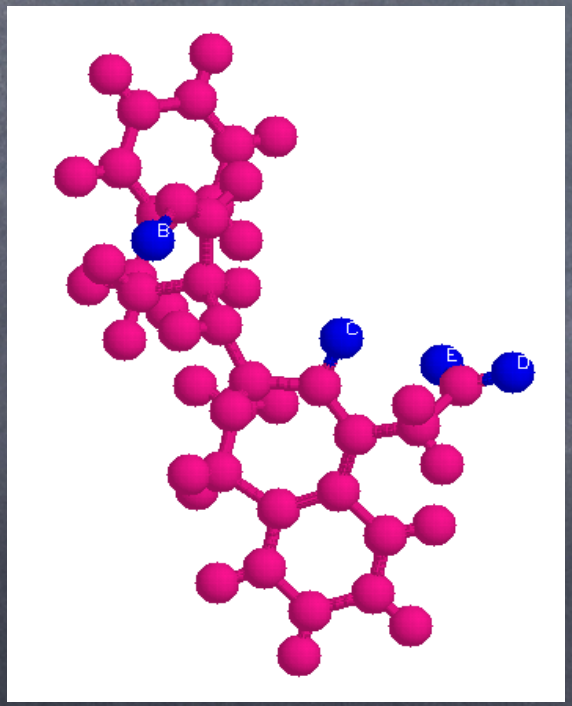
- Methods are often specific to one type of input data:
  - sequences, time series, ...
  - trees
  - graphs
  - relational database (multiple tables)
  - knowledge base

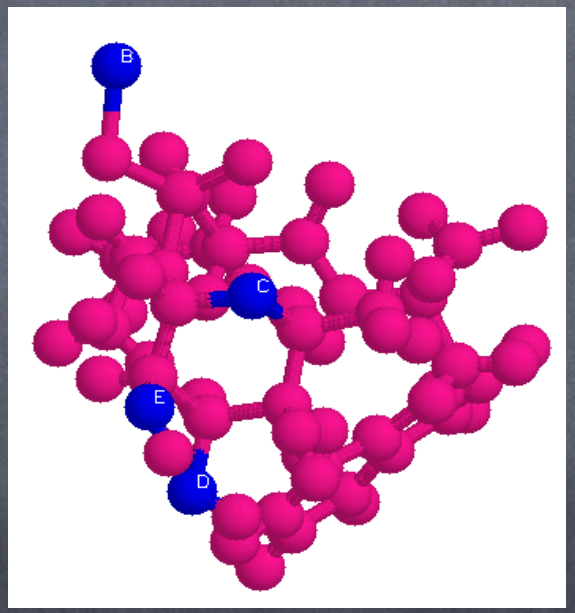
Background knowledge

- Sometimes, knowledge about the domain is available (e.g., molecules: chemical knowledge)
- How to incorporate this knowledge?
- First question: how to even represent the knowledge?
- logic-based formalisms

Finding pharmacophores

- Given some molecules that are active against a certain disease, find the largest common substructure in these molecules (likely pharmacophore)
- Finn et al., 1996: use inductive logic programming


B
C
E
D


B
C
E
D

Description of molecules & output

```
atm(m1,a1,o,2,3.4304,-3.1160,0.0489).
atm(m1,a2,c,2,6.0334,-1.7760,0.6795).
atm(m1,a3,o,2,7.0265,-2.0425,0.0232).
...
bond(m1,a2,a3,2).
bond(m1,a5,a6,1).
bond(m1,a2,a4,1).
bond(m1,a6,a7,du).
...
```

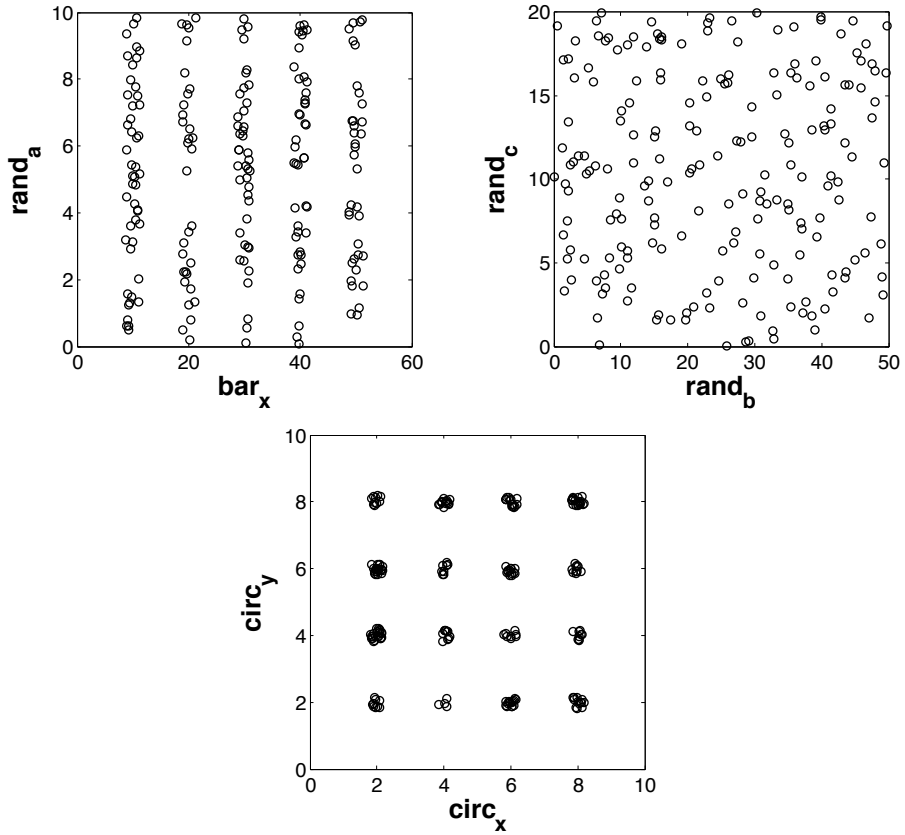
```
...
hacc(M,A):- atm(M,A,o,2,_,_,_).
hacc(M,A):- atm(M,A,o,3,_,_,_).
hacc(M,A):- atm(M,A,s,2,_,_,_).
hacc(M,A):- atm(M,A,n,ar,_,_,_).
zincsite(M,A):-
        atm(M,A,du,_,_,_,_).
hdonor(M,A) :-
          atm(M,A,h,_,_,_,_),
          not(carbon_bond(M,A)), !.
...
```

```
active(A) :- zincsite(A,B), hacc(A,C), hacc(A,D), hacc(A,E),
        dist(A,C,B,4.891,0.750), dist(A,C,D,3.753,0.750), dist(A,
        C,E,3.114,0.750), dist(A,D,B,8.475,0.750), dist(A,D,E,
        2.133,0.750), dist(A,E,B,7.899,0.750).
```

Task specification

- What type of patterns are you looking for?
  - when explicitly specified, sometimes called "language bias"
  - see pharmacophore example
- What exactly do you want to optimize?
- Any constraints the solution must fulfill?

Example: Clustering from example clusters



- high-dimensional space: multiple meaningful clusterings

- indicate what you consider "good" clusters by giving an example cluster

(from B. Verstrynge)

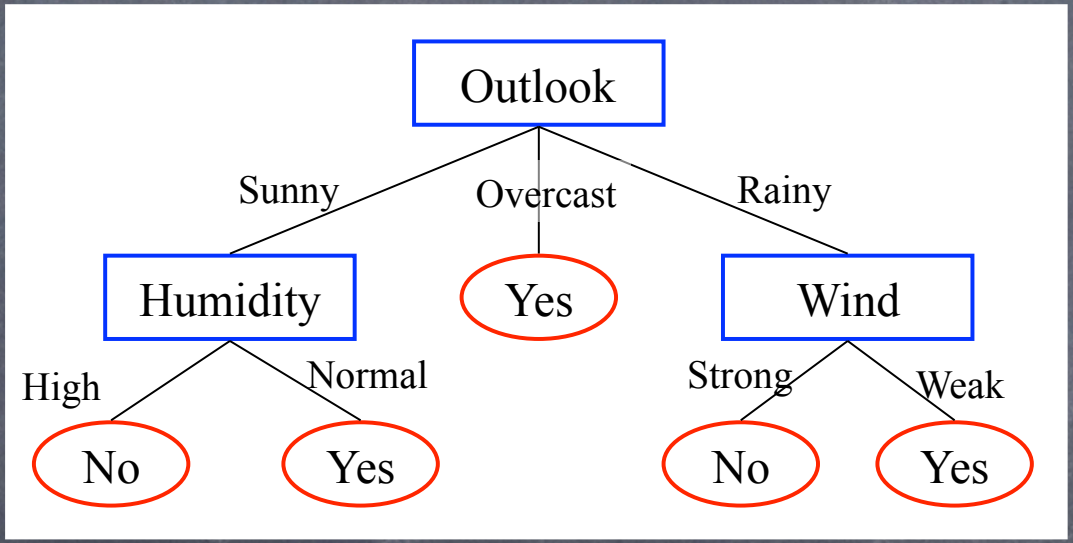
3. Output format

- Much variation in representation of
  - predictive function
  - detected patterns
  - model
- This representation affects interpretability

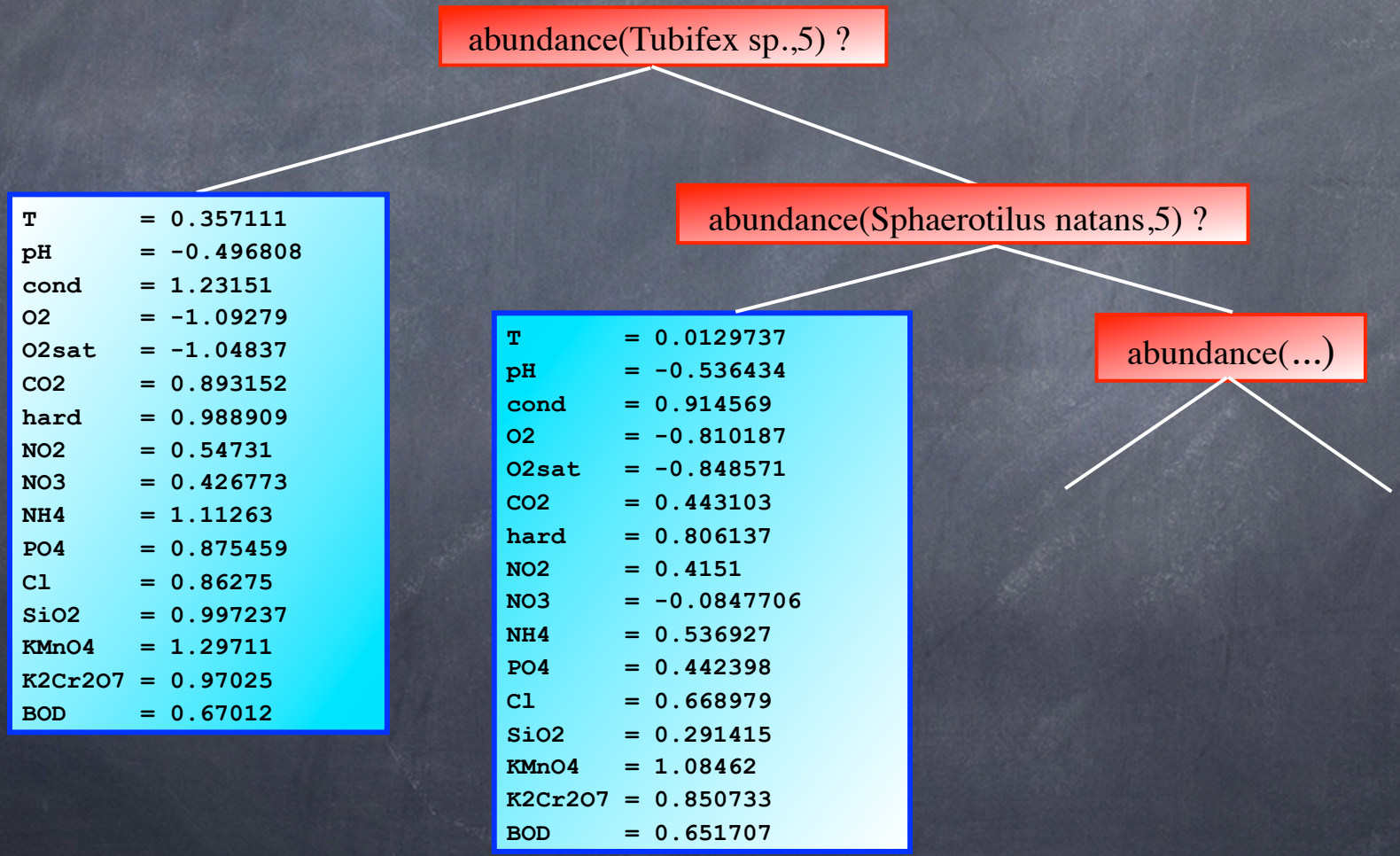
Predictive functions

- Neural networks, decision trees, rule sets, linear functions, SVMs, ...

Decision trees



Example: water quality prediction



Rule sets

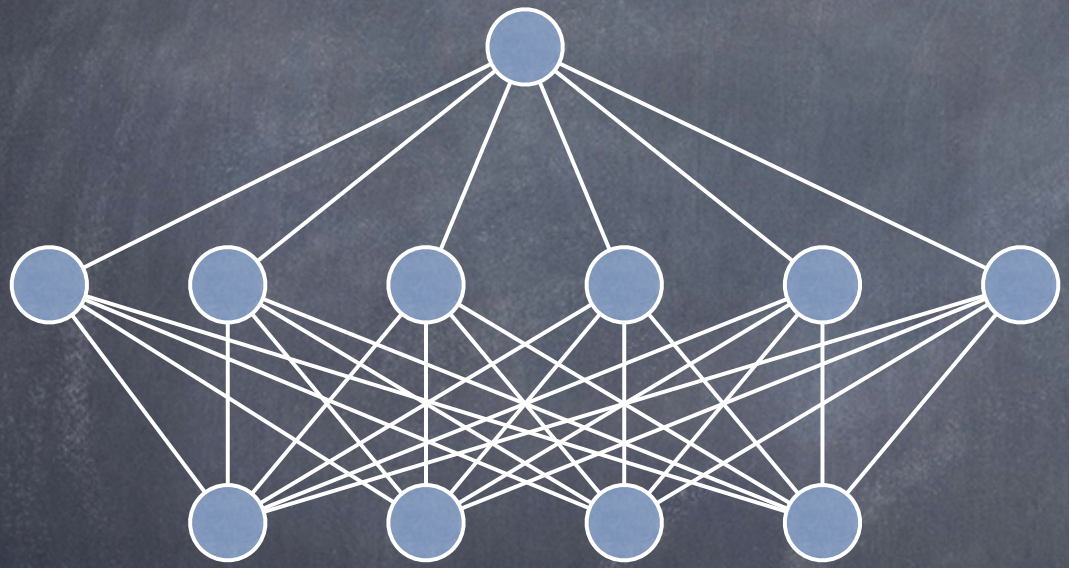
- Sets of "if-then rules", or "classification rules"
- similar for regression

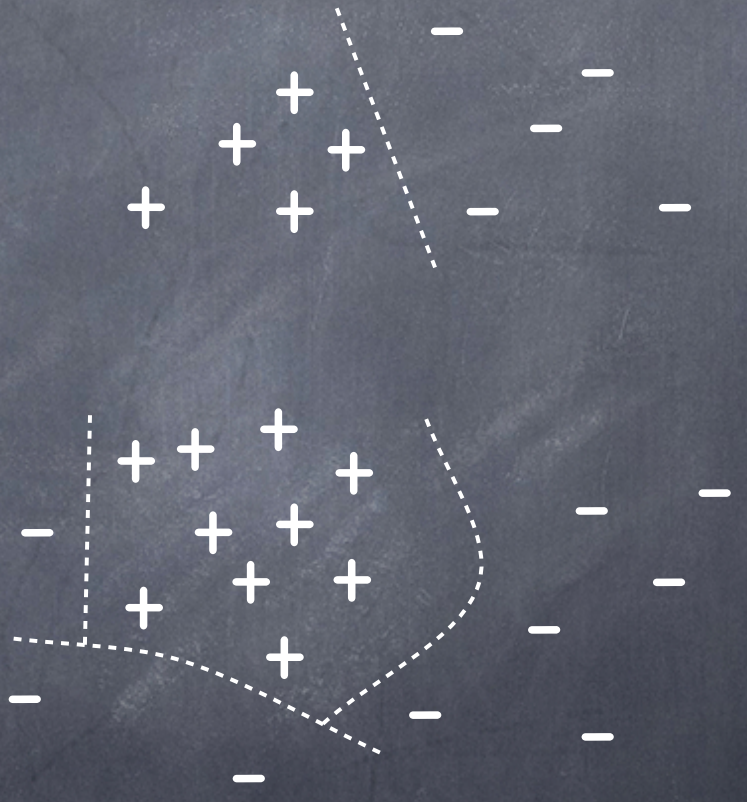
if Outlook = Sunny **and** Humidity = High **then** No
if Outlook = Sunny **and** Humidity = Normal **then** Yes
…

Equations

- polynomial or other functions
- e.g., standard linear regression from statistics
- can turn numerical predictions into (probabilistic) binary predictions: loglinear regression

Feedforward neural networks





uses non-linear transformations
of linear combinations of inputs

Support vector machines

$$f(\vec{x}) = \text{sgn}(\sum_{s_i \in SV} \alpha_i y_i \Phi(\vec{s}_i)\Phi(\vec{x}) + b) = \text{sgn}(\sum_{s_i \in SV} \alpha_i y_i K(\vec{s}_i, \vec{x}) + b)$$

separation defined by so-called "support vectors"

Naive Bayes

- A simple probabilistic model
- Look at each attribute independently; class where that value occurs frequently becomes more likely
- $P(Y|X_1, ..., X_n) \sim P(X_1|Y)\ P(X_2|Y)\ ...\ P(X_n|Y)\ P(Y)$

Patterns

- Clustering (extensional / intensional)
  - extensional = define clusters by enumerating elements
  - intensional = define by listing properties of elements
- itemsets (bread & cheese often bought together)
- association rules ("**if** bread & cheese **then** wine (40%)")
- ...

Models

- Probabilistic models
  - Bayesian networks, Markov networks, HMMs, dynamic Bayesian networks, ...
- Causal networks
- Grammars, automata, Gaussian processes, ...

4. Flexibility of approaches

- Least tunable, easiest to use: standard task, solved by getting an off-the-shelf system and "pushing the button"
  - e.g., indicate target variable, learn a decision tree
  - possibly setting some parameters
- Toolboxes with many learners inside are available: Weka, Orange, R, ...
- **= "standard" data mining**

- A bit more tunable: impose constraints on the answers you're looking for
  - "find all association rules that have beer & wine in the body, with confidence >50% and support >10%"
  - e.g., "find clusters, fulfilling the constraint that x and y must be in the same cluster"
  - given language L (defined by some grammar), find all patterns in it that hold in these data
- **= Constraint-based data mining**

Modeling

- Define the whole data analysis task in a formal, declarative way
  - as a SQL query (only possible for relatively simple data analysis)
  - as a query in some DM language
  - using a general-purpose modeling language
- e.g., Matlab programming, = mathematical modeling
- **= "declarative data mining"**

The current state of the art

- Currently considered "standard":
  - off the shelf methods
  - mathematical programming (linear algebra, statistics, optimization, ...; e.g., Matlab, R)
- Constraint-based and declarative data mining are topics of current research

5. Applications in software engineering?

- For a good overview, see: Du Zhang, Jeffry Tsai (eds.) (2005): Machine learning applications in software engineering
  - + Advances in ... (2007)
- Here just some small things & ideas...

Matching hardware and software

- Performance prediction for combinations of computer architecture & software, based on analyzing benchmark data
- Helps with answering the question: Given a program, e.g. to be run on an embedded system, what architecture should the processor have?

Optimizing compiler parameters

- Compilers have many optimization options
- With options give best performance for which optimization goals?
  - multiple optimizaton goals: construct Pareto front
- Analysis of combinations of existing options in compilers has yielded some interesting results
  - Some options never useful
  - For GCC, substantially better combinations found than the -O1, -O2, -O3 options

Software design...

- Some existing work at LIACS, Leiden: Analyzing correlation between #classes and #bugs in a software project

Other ideas

- Formulate bug detection as anomaly / outlier detection
  - given knowledge of design patterns, find patterns that are "almost the same" - may be likely bugs
- Find frequent patterns in software design (not necessarily currently recognized as design patterns), find deviations from these patterns

Things not discussed...

- reinforcement learning
- semi-supervised learning
- data preprocessing
- ...

Some reference materials

- Du Zhang, Jeffry Tsai (eds.) (2005): Machine learning applications in software engineering
- Encyclopedia of Machine Learning (Sammut & Webb, eds.): a very complete and detailed resource
- P. Flach, "Machine Learning: The Art and Science of Algorithms that Make Sense of Data": focuses on overview, insight & intuitions

Questions ?