Understanding the data: extraction, optimization and interpretation of logical rules

Włodzisław Duch
Rafał Adamczak
Krzysztof Grąbczewski
Karol Grudziński
Norbert Jankowski
Antoine Naud

Computational Intelligence Laboratory,
Department of Computer Methods,
Nicholas Copernicus University,

Grudziądzka 5, 87-100 Toruń, Poland.

e-mail: duch@phys.uni.torun.pl

WWW: http://www.phys.uni.torun.pl/~duch

Plan

1. Intro: understanding the data and knowledge discovery

2. Logical explanations:
3. Overview of methodology: rule extraction, optimization, calculation of probabilities
    - Neural methods of knowledge extraction
    - C-MLP2LN, Constructive MLP converted to Logical Network
    - S-MLP, Search-based MLP
    - SSV, Separability Split Value decision tree
    - FSM, Feature Space Mapping - fuzzy logic, prototypes
4. Prototype-based explanation: SBL, Similarity Based Learner

5. Visualization-based explanation:
    - PCI, Probabilistic confidence intervals
    - IMDS, Interactive multidimensional scaling
6. Some knowledge discovered

7. Example: system for analysis of psychometric questionnaires

8. Open problems

Understanding the data and knowledge discovery

More methods of classification than datasets to classify.
Computational intelligence (CI) methods: developed by statistics, patter recognition, machine learning, neural networks, logics, numerical taxonomy, visualization and other experts.

Neural networks are universal approximators/classifiers but are they good tools for real applications?

- Machine Learning (ML) camp: black box classifiers (such as NN) are unacceptable.
- Knowledge accessible to humans: symbols, similarity to prototypes, visualization.

What type of explanation is satisfactory?
Interesting cognitive psychology problem.

Exemplar and prototype theories of categorization: humans remember examples of each category or create a prototype out of many examples.

Both are true, logical rules are the highest form of summarization.

Types of explanation:

- logic-based: symbols and rules
- exemplar-based: prototypes and similarity
- visualization-based: maps, diagrams, relations

Wider implications

- Understanding what Computational Intelligence (CI) system has learned.
- Use of symbolic knowledge in neural networks: knowledge-based neurocomputing, domain knowledge for initialization, structuring.
- Use of distributed representations in symbolic systems for knowledge acquisition, association and generalization.

Use of various forms of knowledge in one system is still an open question.

Logical explanations

Logical rules, if simple enough, are preferred by humans.

- Explanations 'why' are in some applications necessary
- Rules may expose limitations of neural approximations.
- Rules may sometimes be more accurate than NN and other CI methods.
- Only relevant features are used in rules.
- Overfitting is easy to control, usually few parameters only.
- Rules forever!

Are rules indeed the only way to understand the data?

- IF the number of rules is relatively small AND
- IF the accuracy is sufficiently high.
- THEN rules `may be` an optimal choice.

Types of logical rules:

Crisp logic rules: for continuos *x* use linguistic variables (predicate functions):

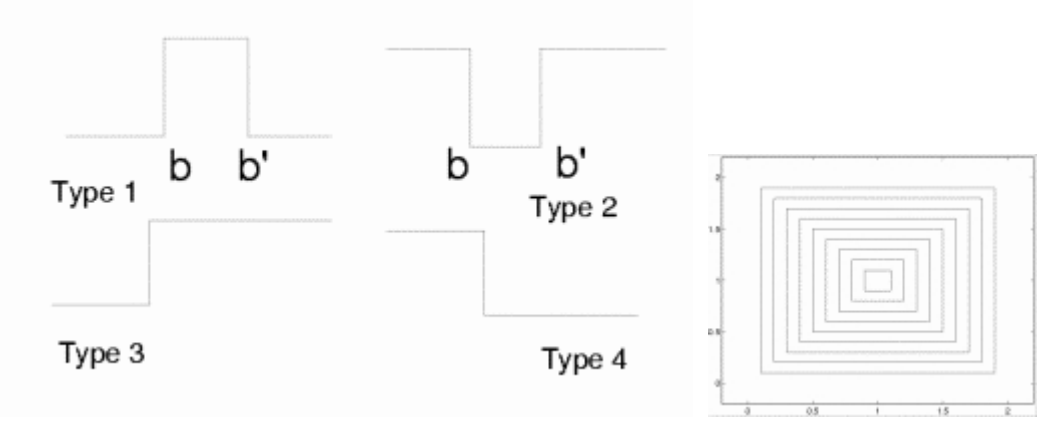
$s_k(x) \equiv$ True $[X_k \leq x \leq X'_k]$, for example:

small(x) = True{x|x<1}
medium(x) = True{x|x ∈ [1,2]}
large(x) = True{x|x2}

Linguistic variables are used in crisp (propositional, Boolean) rules:

IF small(height) AND red(hat) THEN (X is Brownie) ELSE IF ... ELSE ...

Rectangular membership functions, step functions are used for partitioning of the input space.
Decision regions: hyperrectangular (cuboidal).



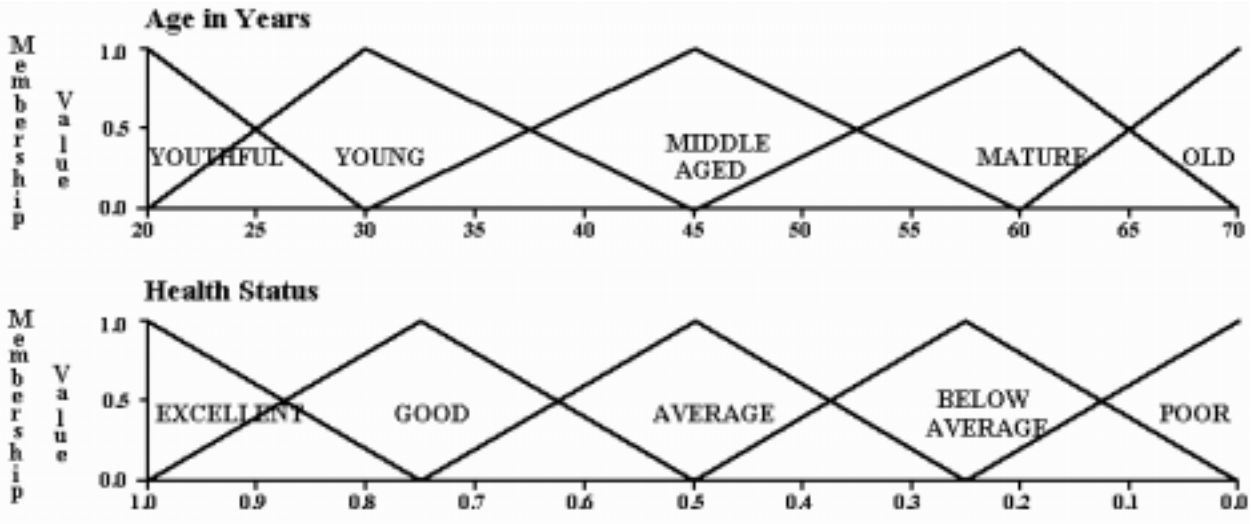
Decision trees provide crisp rules applied in a specific order.

If hyperrectangular regions are too simple, rules are not accurate; allow linear combinations of some inputs *x*.

The number of problems that one may analyze using crisp logic may be limited.

Fuzzy logic rules:

triangular,
trapezoidal,
Gaussian
and other type of membership (truth degree) functions

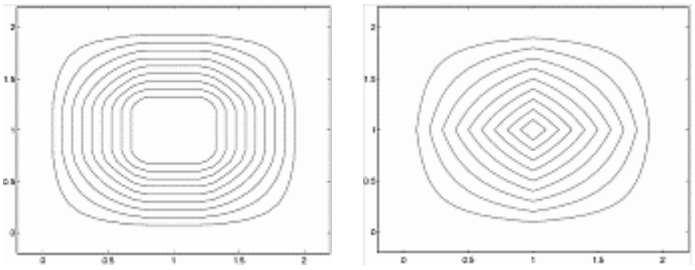


Fuzzy logic: separable functions - products of one-dimensional factors:

$$\mu_C(X) = \prod_{i=1}^{N} \mu_{C_i}(X_i); \text{degree of } X \in C$$

Many other possibilities exist to produce *N*-dimensional membership functions.

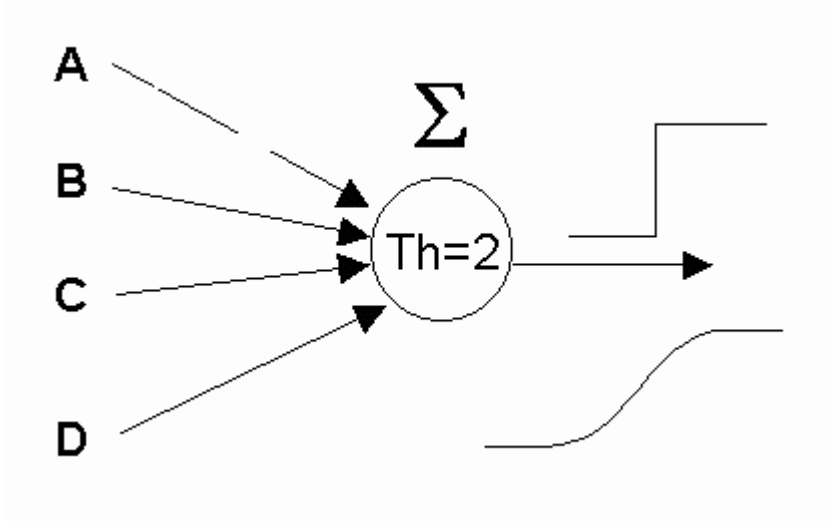
$$IF(\mu_C(X) > Th \text{ THEN } Fact_k = TRUE)$$



Triangular and trapezoidal membership functions give such countours in 2D for different *Th*

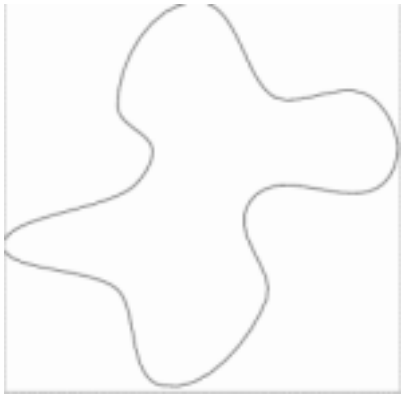
Rough logic: trapezoidal shapes, borders may be non-linear.

- **M-of-N rules**: M conditions out of N are true.
  Natural for neural systems, for example, if 2 logical conditions out of 4 are true:



IF 2 conditions out of {*A,B,C,D*} are true THEN (X is Brownie)
ELSE IF ... ELSE ...

Clusterization: may require arbitrary, complex decision border shapes.



Granulation: covering with simpler shapes, corresponding to many rules.

IF $X \subseteq C$) THEN $Fact_k$ = TRUE
Simple rules - only if non-linear feature transformations are used.

Crisp logic rules are most desirable; try them first,
but remember ...

- only one class is predicted $P(C_i|X,M) = 0$ or 1
  black-and-white picture may be inappropriate in many applications

- reliable crisp rules may reject some cases as unclassified
  tradeoff: reliability (confidence in rules) - rejection rate

- discontinuous cost function allow only non-gradient optimization.

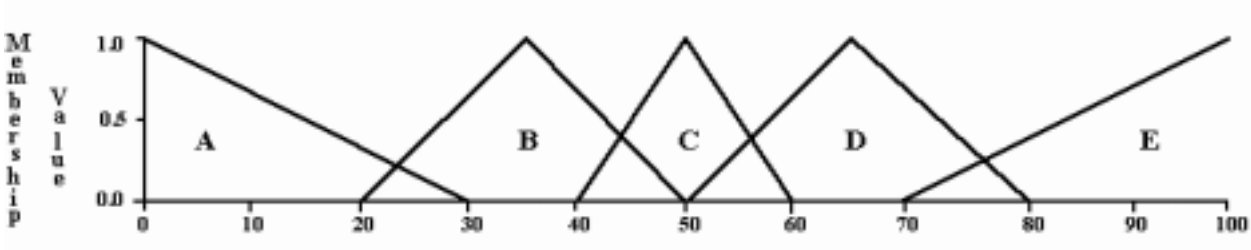
Fuzzy rules - continuous membership functions:

- continuous classification probabilities $P(C_i|X,M)$;
- all vectors classified (some with small probability);
- gradient-based optimization possible;

but remember ...

- not so comprehensible as the crisp rules;

- danger of overparameterization - more complex rules, additional position/shape parameters.



- Fixed set of membership functions with predetermined shapes - bad idea.

Curse of dimensionality:
k linguistic variables in d dimensions give k^d areas.
Context-dependent linguistic variables - adapt membership functions in each rule.
Effect: clusters of different sizes at different input areas.

Problems with rule-based classification models:

- Interpretation of crisp rules may be misleading.
- Crisp rules may be unstable against small perturbations of input values.
- Rule-based classifiers may be unstable - small change in the dataset leads to a large change in structure of complex sets of rules.
- Fuzzy rules do not estimate real probabilities.
- How to find the best fuzziness/precision tradeoff ?

Knowledge accessible to humans:

- symbols and rules, crisp and fuzzy;
- similarity to prototypes;
- visualization - exploratory data analysis.

First rule extraction/application is considered; than some remarks on prototype-based and visualization-based methods are made.

Overview of rule-based methodology

Methodology of rule extraction: many decisions depend on particular application

1. Select linguistic variables $s_k(X_k,X'_k)$ true if x in $[X_k,X'_k]$; for discrete features define subsets.
    - If the number of input feature is very high try feature selection methods first.
    - Neural networks may aggregate several inputs providing new features.
    - For continuous features decision trees and neural networks perform automatic discretization.
2. Select the **simplicity/accuracy** tradeoff.
    - Simplest sets of rules with acceptable error should be found first; they are the most comprehensible.
    - Sets of rules with growing complexity and accuracy may be found.
    - Rules covering a few cases only are usually rejected but in some applications domain experts may find them useful.
3. Extract rules from data using neural, machine learning or statistical techniques.

4. Repeat the procedure until a stable set of rules is found.

5. Explore the **reliability/rejection** rate tradeoff optimizing rule set.
    - Reliable rules make few errors but may reject some case.
    - Optimize linguistic variables (X_k,X'_k intervals) using the rules extracted.
6. Find optimal degree of fuzzification to calculate probabilities.
   Fuzzification may be introduced during optimization.

How to optimize sets of logical rules

Regularization of classification models (for example, network or tree pruning) allows to explore **simplicity-accuracy tradeoff.**

Next step: exploring the **confidence-rejection rate tradeoff.**

Define confusion matrix $F(C_i,C_j|M)$ counting the number of cases from class C_j assigned by the set of rules M to the class C_i.

Define weighted combination of the number of errors and the "predictive power" of rules:

$$E(M) = \gamma \sum_{i \neq j} F(C_i, C_j|M) - \mathrm{Tr}\, F(C_i, C_j|M)$$

This should be minimized without constraints; it is bound by -N (number of all training vectors).

Sets of rules M are parameterized by X_k, X'_k intervals.
For γ=0 predictive power of rules is maximized.
Rules that make fewer errors on the training data should be more reliable.

Cost function $E(M; \gamma)$ allows to reduce the number of errors to zero (large γ) for rules M that reject some instances.

Optional risk matrix may be used:

$$\min_M \left[\sum_{i \neq j} \mathbf{R}(C_i, C_j) F(C_i, C_j|M) \right]$$

If the confusion matrix $F(C_i,C_j|M)$ is discontinuous non-gradient minimization methods should be used (simplex, simulated annealing etc).

How to use logical rules to calculate probabilities

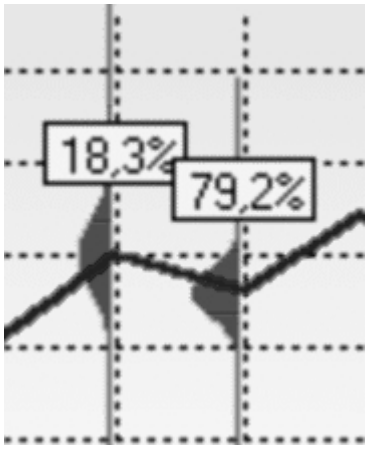
Data from measurements/observations are not precise.
Finite data resolution - Gaussian error distribution:
x - $G_x=G(y;x,s_x)$, where G_x is a Gaussian (fuzzy) number.

Given a set of logical rules $\{\Re\}$ apply them to input data $\{G_x\}$.
Use Monte Carlo sampling to recover $p(C_i \mid X; \{\Re\})$ - this may be used with any classifier.

Analytical estimation of this probability is based on cumulant function:

$$\rho(a-x) = \int_{-\infty}^{a} G(y;x,s_x)dy =$$

$$\frac{1}{2}\left[1+\mathrm{erf}\left(\frac{a-x}{s_x\sqrt{2}}\right)\right] \approx \sigma(\beta(a-x))$$



Approximation better than 2% for
$\beta = 2.4/\sqrt{2}s_x$

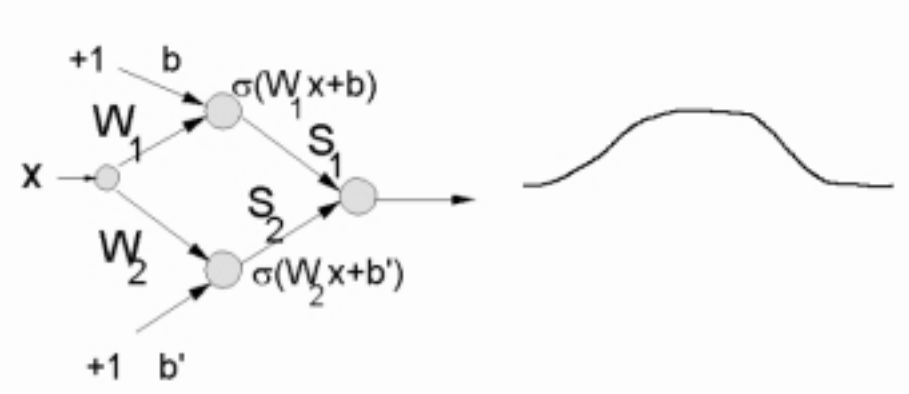
The rule $R_a(x) = \{xa\}$ is true for G_x with probability:

$$p(R_a(G_x)=T) = \int_{a}^{+\infty} G(y;x,s_x)dy \approx \sigma(\beta(x-a))$$

If the logistic function is used instead of the error function the exact error distribution is $\sigma(x)(1-\sigma(x))$; for $s^2=1.7$ it is within 3.5% identical with Gauss.

$$p(R_{a,b}(G_x)=T) \approx \sigma(\beta(x-a)) - \sigma(\beta(x-b))$$

Soft trapezoidal membership functions realized by L-units are obtained.



Fuzzy logic with such functions is equivalent to crisp logic with G_x; realized by neural networks with logistic transfer functions.

For conjunctive rule with many independent conditions:
$R = r_1 \wedge r_2 \wedge \ldots r_N$ the probability $p(C_i|X)$ is a product of

$$p(R_{ab}(G_x) = \mathrm{T}) = \int_a^b G(y; x, s_x)dy = \frac{1}{2}\left[\mathrm{erf}\left(\frac{b-x}{s_x\sqrt{2}}\right) - \mathrm{erf}\left(\frac{a-x}{s_x\sqrt{2}}\right)\right]$$

If rules are overlapping and conditions are correlated formula leading to Monte Carlo results is:

$$P(x \in C) = \sum_{R \in 2^{R_C}} (-1)^{|R|+1} P\left(x \in \bigcap R\right)$$

2^{Rc} are all subsets of the set of classification rules for class C
$|R|$ is the number of rules.

This is **not a fuzzy approach!**

Here small receptive fields are used, in fuzzy approach typically 2-5 large receptive fields define linguistic variables.

Benefits:

1. Probabilities instead 0, 1 crisp rule decisions.
2. Vectors that were not classified by crisp rules have now non-zero probabilities.
3. Dispersions s_x may be treated as adaptive parameters of the model M.
4. Gradient methods may be used for large-scale optimization.

$$E(M, s_x) = \frac{1}{2}\sum_X \sum_i (p(C_i|X; M) - \delta(C(X), C_i))^2$$

Alternative approaches: flexible matching in machine learning.

Overview of the neural methods of knowledge extraction

The trouble with doing something right the first time
is that nobody appreciates how difficult it was.
Anonymous

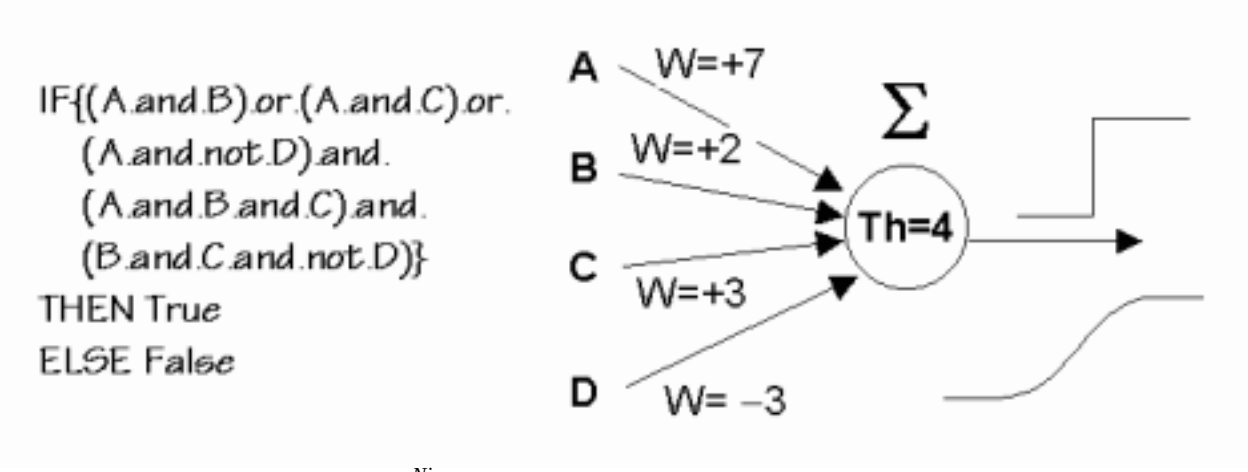
Review and comparison of many rule extraction methods:
R. Andrews, J. Diederich, A.B. Tickle, "A Survey and Critique of Techniques for Extracting Rules from Trained Artificial Neural Networks," Knowledge-Based Systems vol. 8, pp. 373-389, 1995.
Neural rule extraction algorithms differ in:

a. the "expressive power" of the extracted rules (types of rules extracted);
b. the "quality" of the extracted rules (accuracy, fidelity comparing to the underlying network, comprehensibility and consistency of the extracted rules);
c. the "translucency" of the method - analysis of individual nodes versus analysis of the total network function;
d. the algorithmic complexity of the method;
e. specialized network training schemes;
f. the treatment of linguistic variables.

Early papers:
K. Saito, R. Nakano, "Medical diagnostic expert system based on PDP model", Proc. of IEEE Int. Conf. on Neural Networks (San Diego CA), Vol 1 (1988) 255-262
Restrictions on the form of rules, the maximum number of positive and negative conditions, the depth of the breadth-first search process, including only conditions that were present in the training set.
KT algorithm: L.M. Fu, "Neural networks in computer intelligence", McGraw Hill, New York, 1994
Local method, conjunctive rules, depth of search is restricted. Network weights help to limit the search tree.

SUBSET algorithm

G. Towell, J. Shavlik, "Extracting refined rules from knowledge-based neural networks". Machine Learning 13 (1993) 71-101
Analyze incoming weights of hidden and output neurons.
Consider all possible subsets of incoming weights W_i, positive or negative.
Find all combinations > Th
Example:



Problem: number of subsets is 2^{Ninp}.
Exponentially growing number of possible conjunctive propositional rules.
Partial solution: restrict the number of antecedents, subsets or rules using some heuristics.
inputs with largest weights are analyzed first, combinations of two largest weights follow, until the maximum number of antecedent conditions is reached.

RuleNet

C. McMillan, M.C. Mozer, P. Smolensky, "Rule induction through integrated symbolic and subsymbolic processing". In: J. Moody, S. Hanson, R. Lippmann, eds, Advances in NIPS 4, Morgan Kaufmann, San Mateo, CA 1992
J.A. Alexander, M.C. Mozer, "Template-based algorithms for connectionist rule extraction". In: G. Tesauro, D. Touretzky, T. Leen, eds, Advances in NIPS 7, MIT Press, Cambridge, MA, 1995
Used to find *M* of *N* rules and propositional rules.
Make hypothesis and test them - training algorithm, called „The Connectionist Science Game", consists of 3-steps:

1. Train RuleNet network.
2. Extract symbolic rules using weight analysis.
3. Inject rules back into the network.

RulNet: 3 layer network, input, condition units and output action units.
Use weight templates exploring large spaces of candidate rules.
Only discrete-valued features, specific architecture for string-to-string mapping, for example character strings, not a general technique.

M-of-N method

G. Towell, J. Shavlik, "Extracting refined rules from knowledge-based neural networks". Machine Learning 13 (1993) 71-101
Rules of the form:
IF *M* of *N* antecedents are true THAN ....
Sometimes more compact and comprehensible than conjunctive rules.
Used in KBANN (Knowledge-Based ANN) networks, where symbolic knowledge is used to specify initial weights.

7. For each hidden and output unit form groups of similarly-weighted links.
8. Set all link weights to average of the group.
9. Eliminate groups that do not affect the output.
10. Use prototype weight templates (corresponding to symbolic rules) for comparison with the weight vectors.
11. Freeze weights, reoptimize biases.
12. Form single rule for each hidden and output unit.

IF(*M* of *N* antecedents (A_1, A_2 ... A_N) are true) THEN ...

Newer work: *M* of *N*3 algorithm:
R. Setiono, "Extracting *M* of *N* Rules from Trained Neural Networks", Transactions on Neural Networks 11 (2000) 512-519
Penalty term to prune the network, inputs should be binary.

REAL (Rule Extraction As Learning)

M. W. Craven, J.W. Shavlik, "Using sampling and queries to extract rules from trained neural networks". In: Proc. of the Eleventh Int. Conference on Machine Learning, New Brunswick, NJ. Morgan Kaufmann 1994, pp. 37-45
Rule extraction = learning logical function that approximates the target (neural network) function.

- Get new example,
- use existing rules to classify it,
- if wrong add a new rule based on this example,
- check if the extended set of rules still agree with NN.

Rules: IF ... THEN ... ELSE, M-of-N

VIA (Validity Interval Analysis)

S. Thrun, "Extracting rules from artifcial neural networks with distributed representations". In: G. Tesauro, D. Touretzky, T. Leen, eds, Advances in Neural Information Processing Systems 7. MIT Press, Cambridge, MA, 1995
Extract rules mapping inputs directly to the outputs, try to capture what does the network do, global method.

5. Assign arbitrary „validity intervals" to all NN units
   Restrictions on the input/activation values of units.
6. Refine the intervals by changing those that are never activated.
7. Analyze the intervals and derive rules.

Rules: IF ... THEN ... ELSE
Numerous rules, too specific. Has not been used much?

RULENEG

E. Pop, R. Hayward, J. Diederich, "RULENEG: extracting rules from a trained ANN by stepwise negation", QUT NRC technical report, December 1994;
R. Hayward, C. Ho-Stuart, J. Diederich and E. Pop, "RULENEG: extracting rules from a trained ANN by stepwise negation", QUT NRC technical report, January 1996
Forms conjunctive rules, one per input pattern.
For input pattern that is not correctly classified by the existing set of rules:
For i =1..N
Determine class of (x_1, ... NOT.x_i, ... x_N)
If the class has changed add $R = R$.AND.x_i

BRAINNE

S. Sestito, T. Dillon, "Automated knowledge acquisition". Prentice Hall (Australia), 1994
Network of M inputs and N outputs is changed to a network of M+N inputs and N outputs and retrained.
Original inputs that have weights which change little correspond to the most important features.

DEDEC

A.B. Tickle, M. Orlowski, J. Diederich, "DEDEC: decision detection by rule extraction from neural networks", QUT NRC technical report, September 1994
Rule extraction: find minimal information distinguishing a given pattern from others from the NN point of view.

Rank the inputs in order of importance - determine the importance of input features, using input weights.
Select clusters of cases with important features (using k-NN) and use only those features to derive rules.
Learn rules using symbolic induction algorithm.

RULEX

R. Andrews, S. Geva, "Rule extraction from a constrained error back propagation MLP". Proc. 5th Australian Conference on Neural Networks, Brisbane, Queensland 1994, pp. 9-12
Special MLP network, using local response units - combination of sigmoids in one dimension, forming ridges.
Disjoint regions of the data one hidden unit.
Similar to symmetric trapezoid neurofuzzy approach.
Trained with Constrained Backpropagation (some weights are kept fixed).
Inserting and refining rules is possible.
Propositional Rules:
IF $Ridge_1$ is active and $Ridge_2$ is active and .... THEN $Class_k$
Works for continuos & discrete inputs.

TREPAN

M. W. Craven, J.W. Shavlik, "Extracting tree-structured representations of trained networks". In: D. Touretzky, M. Mozer, M. Hasselmo, eds, Advances in NIPS 8, MIT Press, Cambridge, MA 1996.
Decision tree instead of rules - inductive algorithm.
NN treated as „oracle" answering queries.
Queries may be incomplete patterns.
Oracle determines class labels, is used to selects splits of nodes and to check if a tree node covers a single class only.
Tree expansion: best-first method, with node splits representing binary and M-of-N rules.
Spilt: partition input space to increase separation of input patterns into classes.
Nodes evaluated by: % cases reaching it times the % of errors in the node.
Split selected only after 1000 cases considered.
Thanks to oracle - works better than other inductive algorithms.
Conclusion: if a black box classifier works well on your data and rule-based description is required - use it as oracle!

Successive Regularization

M. Ishikawa, "Rule extraction by succesive regularization". In: Proc. of 1996 IEEE Int. Conf. on Neural Networks. Washington, 1996, pp. 1139-1143.
Structural learning with forgetting (SLF):
MLP with Laplace-type regularizing term:

$$E(W) = \frac{1}{2}\sum_{p}\left(Y^{(p)} - M(X^{(p)}; W)\right)^2 + \lambda \sum_{i,j} |W_{ij}|$$

$(X^{(p)}, Y^{(p)})$ - question-response patterns p;
W_{ij} - connection weight between units i and j.

Selective forgetting: only weights smaller than some threshold are included in the regularizing term.

This term leads to a constant decay of smaller weights.
Small weights are pruned and a skeletal network emerges.
Clarification of hidden units: 0/1 outputs forced by penalty term
$c \Sigma_i \min (1-h_i,h_i)$

Successive regularization:
Start from rather large λ, get dominant rules first.
Fix the parameters of this part of the network.
Decrease λ, train network = more connections left, more rules.
Skeletal structure + 0/1 outputs of hidden units = each node is represented as a logical function of nodes in the adjacent lower layer.
Good method but requires many experiments to find good initial network.

Other neural methods

- P. Geczy and S. Usui, "Rule extraction from trained neural networks". Int. Conf. on Neural Information Processing, New Zealand, Nov.1997, Vol. 2, pp. 835-838
  Train the network.
  Replace resulting weights by resulting 0, +1 and -1
  Extract logical functions performed by the network.
- H. Tsukimoto, "Extracting Rules from Trained Neural Networks" , Transactions on Neural Networks 11 (2000) 377-389
  Approximation of MLPs by by Boole'an functions.
  Network function is approximated by lower order logical polynomials.
  Results are not too good.
- R. Setiono and H. Liu, "**Neurolinear**: From neural networks to oblique decision rules". Neurocomputing (in print).
  Oblique decision rules, linear combination of inputs.
- R. Setiono, "Extracting rules from neural networks by **pruning and hidden-unit splitting**". Neural Computation, Vol. 9, No. 1, pp. 205-225.
  Prune the network to get small number of inputs connected to a hidden unit.
  Split the hidden node treating it as a few output units, each activation as a target value.
  Add new hidden layer, train and prune.
  Few results so far.

Neural rule extraction methods developed in our group

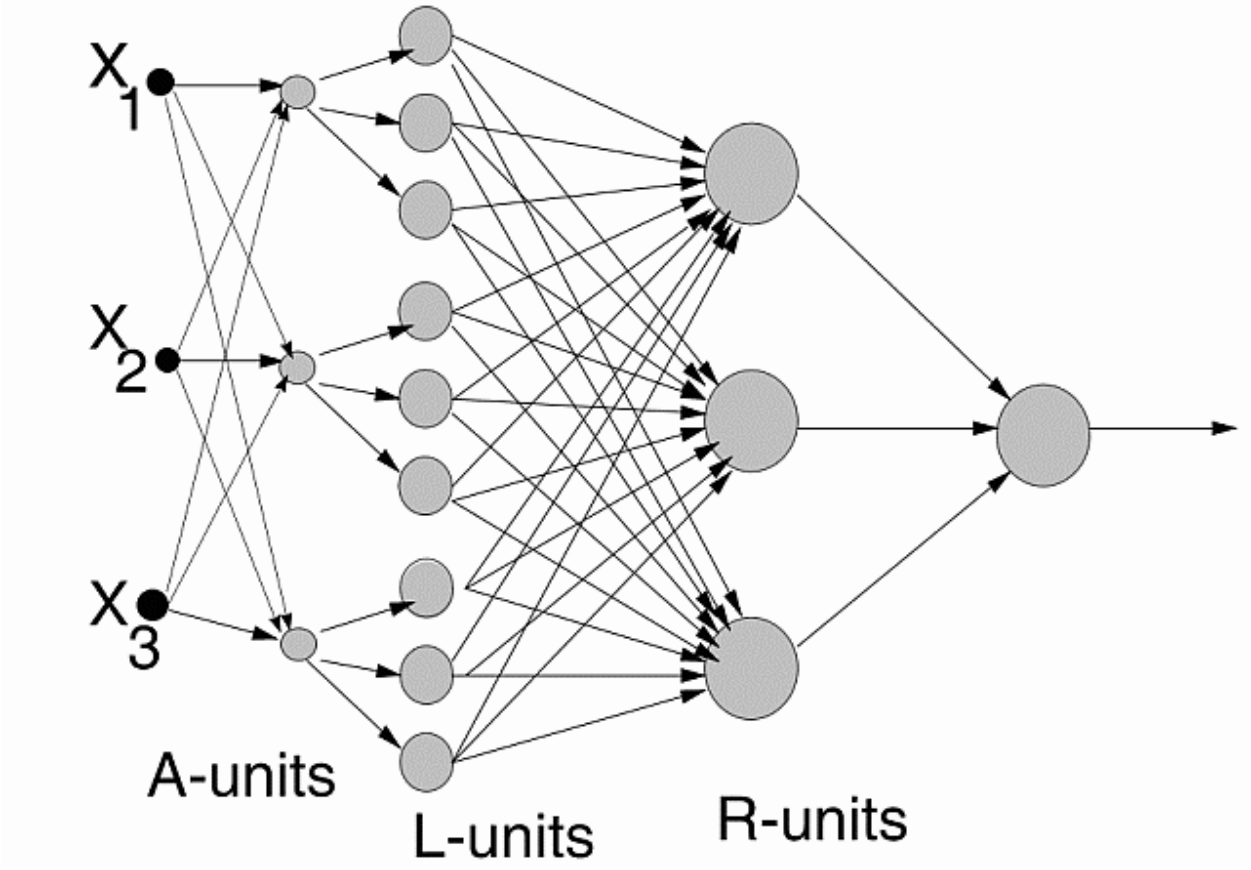
Several practical rule-extraction methods developed in our group:

1. Modified constrained constructive C-MLP2LN method

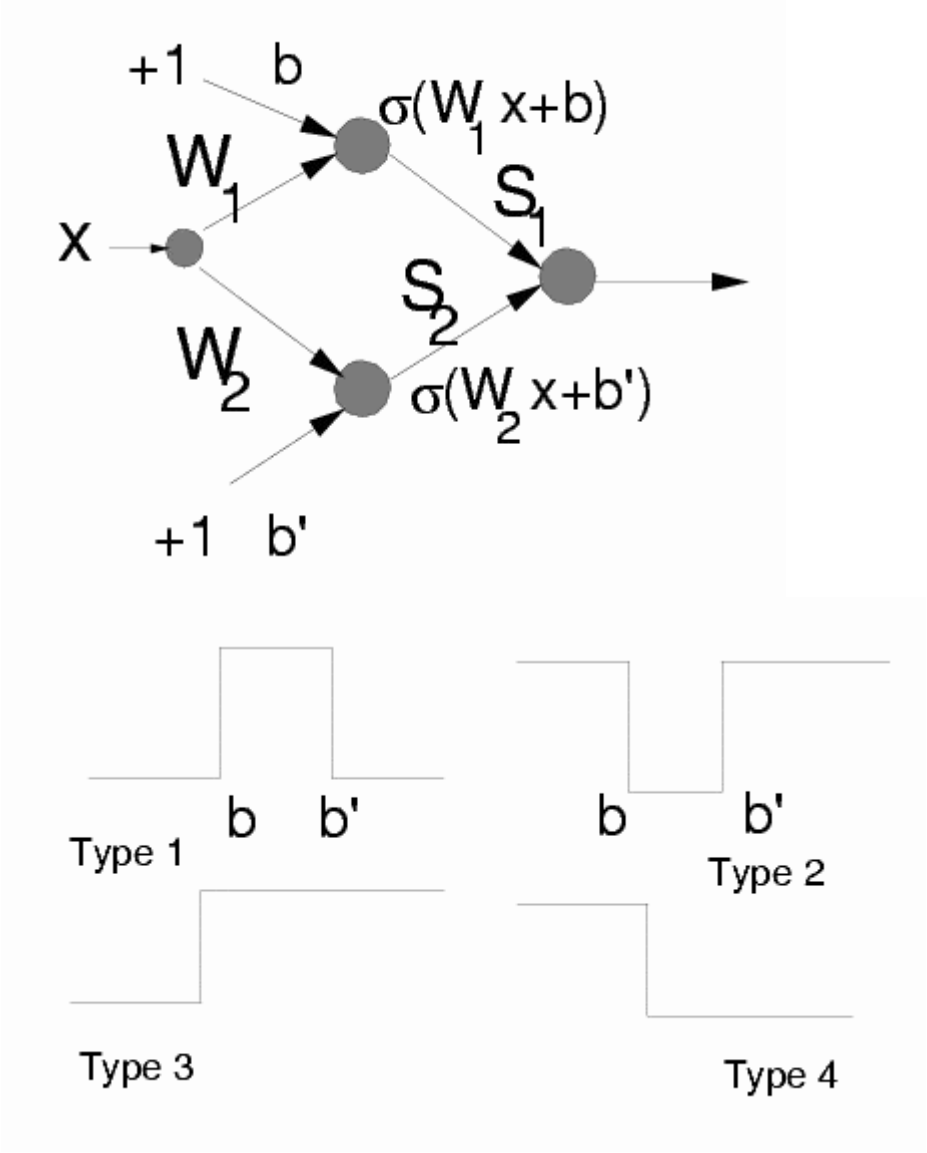
Simplify the network leaving only 0, ±1 weights, use special linguistic units for input discretization.

C-MLP2LN, Constructive MLP converted to Logical Network

Architecture: Aggregation, Linguistic variables and Rule layers; one output per class.



Aggregation: used to combine and discover new useful features, no constraints.



L-units: providing intervals for fuzzy or crisp membership functions, made from 2 neurons, only biases are adaptive parameters here.
Without L-units decision borders will be hyperplanes, combinations of inputs - sometimes it may be advantageous.

Constraint MLP cost function

$$E(W) = \frac{1}{2}\sum_{p}\sum_{k}\left(Y_k^{(p)} - \mathbf{F}_k\left(X^{(p)};W\right)\right)^2 + \frac{\lambda_1}{2}\sum_{i,j}W_{ij}^2 + \frac{\lambda_2}{2}\sum_{i,j}W_{ij}^2(W_{ij}-1)^2(W_{ij}+1)^2$$

First term: standard quadratic function (or any other)
Second term: weight decay & feature selection.
Third term: from complex to simple hypercuboidal classification decision regions for crisp logic

(for steep sigmoids).

Different regularizers may be used.

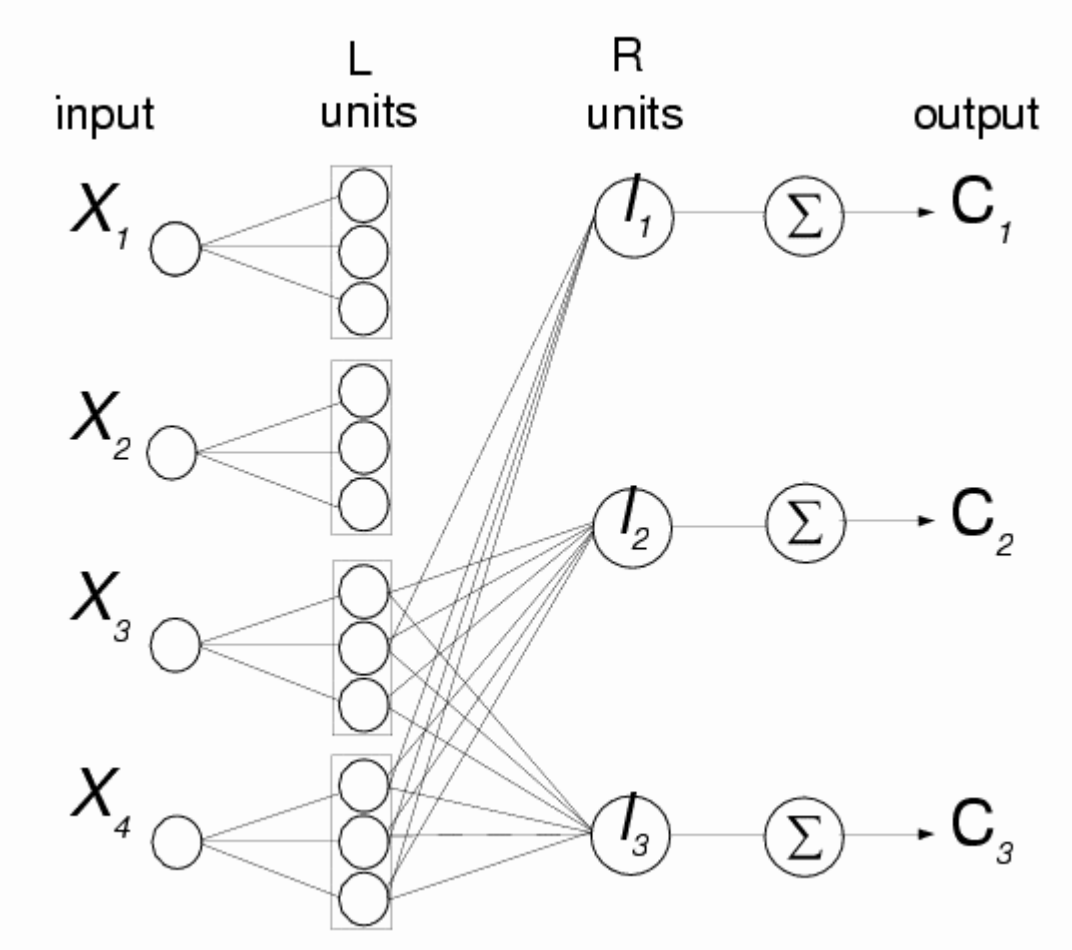
$$|W_{ij}||W_{ij}^2 - 1| \text{ cubic}$$

$$|W_{ij}| + |W_{ij}^2 - 1| \text{ quadratic}$$

$$\sum_{k=-1}^{+1} |W_{ij} + k| - |W_{ij} - \frac{1}{2}| - |W_{ij} + \frac{1}{2}| - 1$$

Different error functions may be used: quadratic, entropy based etc.

Increase the slope of sigmoidal functions during learning to get rectangular decision borders.



Another approach: increase *a* in the regularization term:

$$\sum_{ij} W_{ij}^{\ 2} \left(W_{ij} - a\right)^2$$

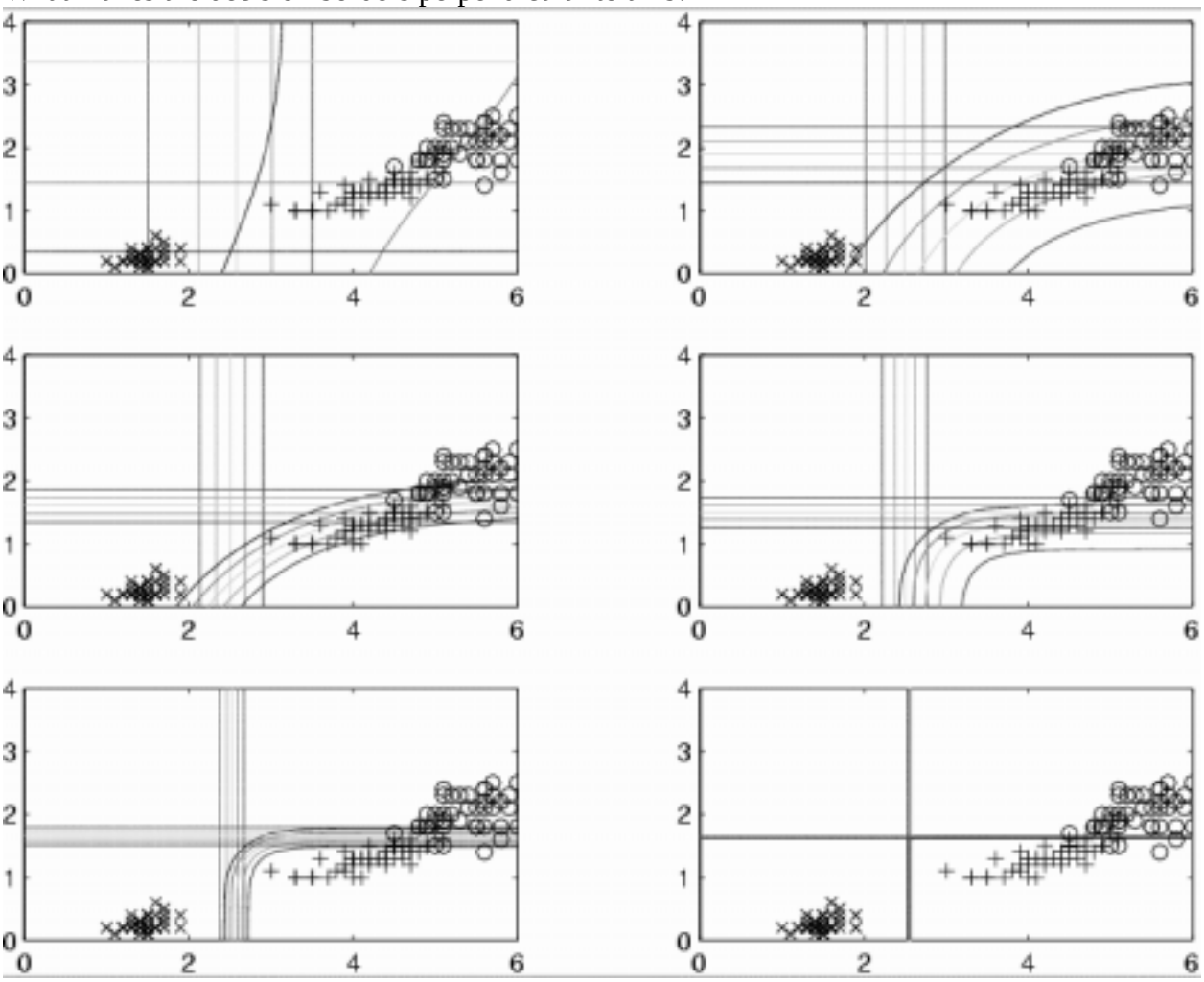
This prunes the network leaving large weights, which is equivalent to increasing the slope.

$$\sigma\left((W\cdot X+\theta)/T\right)=\sigma\left(\left(\frac{W}{\|W\|}\cdot X+\frac{\theta}{\|W\|}\right)\|W\|/T\right)$$

$$=\sigma\left((W'\cdot X+\theta')/T'\right)$$

Without logical inputs this allows large but non-equal weights.

What makes the decision borders perpendicular to axis?



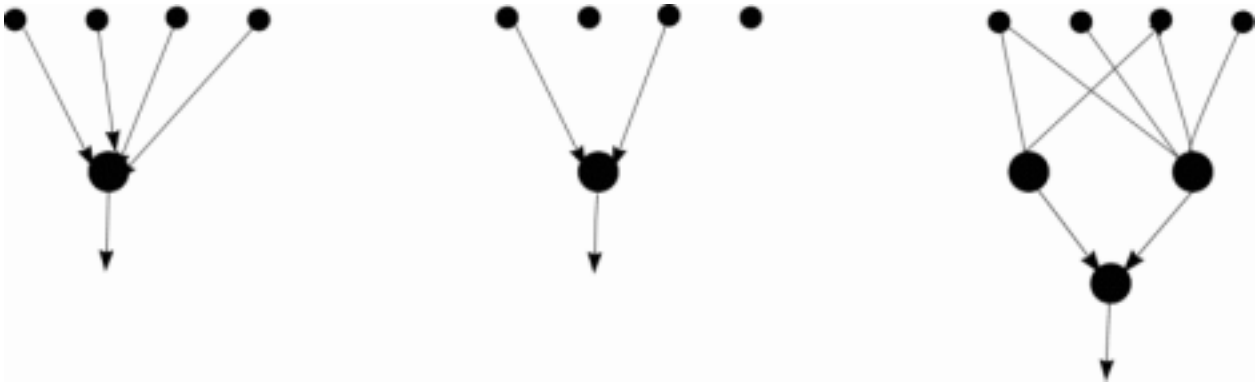
Logical rules from MLP: simplify the network by enforcing weight decay and other constraints.

Strong and weak regularization allows to explore **simplicity-accuracy tradeoff**.

- Constructive C-MLP2LN algorithm: faster, train one R-unit at a time.
- Add one neuron and train it, freezing the existing skeleton network.
- The network first grows, then shrinks; stop when the number of new vectors per one new neuron becomes too small.

Many equivalent sets of rules may be found.

Non-gradient optimization methods - closer to global optimum, better rules?
So far poor results but more experiments are needed - use Alopex?



MLP2LN network: Iris example, step by step

Architecture of the network:

- 4 L units,
- 1 hidden layer with 1 unit for each class,
- 3 output units

Learning process

- Network initialization by random weights
- Parameters:

- Learning 0.2, Forcing zeros 0.00001, Forcing ones 0, Sigmoid Slope 2
- Learning process 2000 cycles

- Learning 0.2, Forcing zeros 0. 0001, Forcing ones 0, Sigmoid Slope 2
- Learning process 1000 cycles

- Learning 0.2, Forcing zeros 0. 0005, Forcing ones 0, Sigmoid Slope 2
- Learning process 1000 cycles

- Learning 0.1, Forcing zeros 0.0, Forcing ones 0.0005, Sigmoid Slope 2
- Learning process 1000 cycles

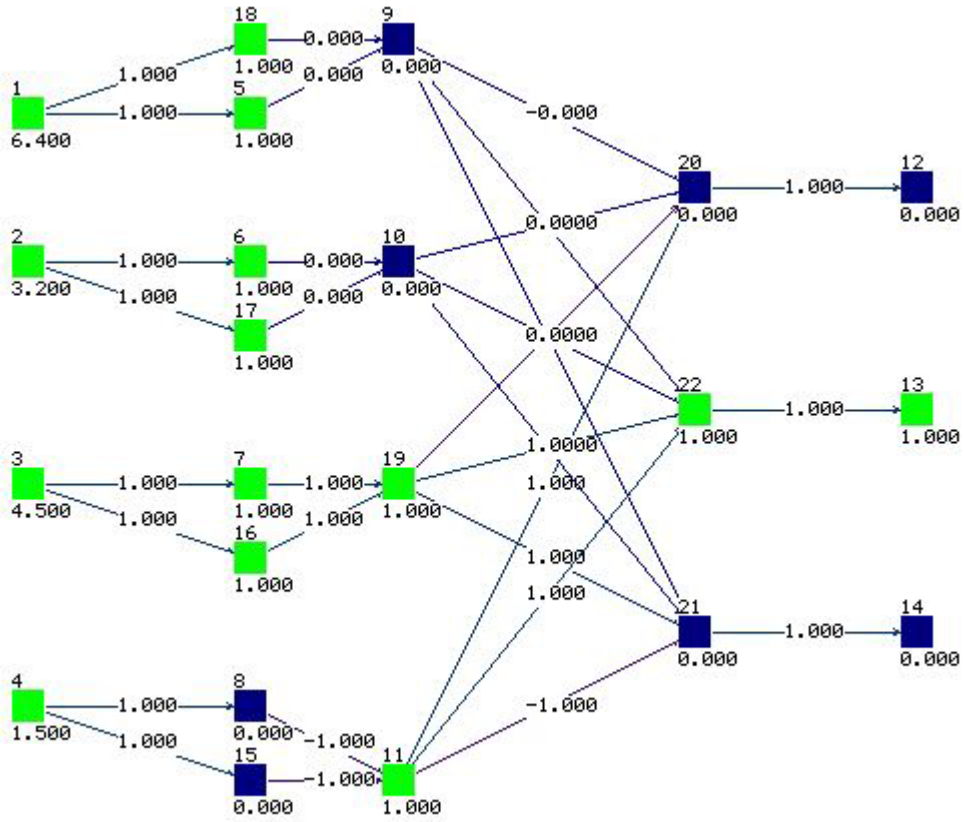
- Learning 0.1, Forcing zeros 0.0, Forcing ones 0.001, Sigmoid Slope 2
- Learning process 1000 cycles

- Learning 0.01, Forcing zeros 0.0, Forcing ones 0.01, Sigmoid Slope 4
- Learning process 1000 cycles

- Learning 0.001, Forcing zeros 0.0, Forcing ones 0.1, Sigmoid Slope 4
- Learning process 1000 cycles

- Learning 0.0001, Forcing zeros 0.0, Forcing ones 0.1, Sigmoid Slope 6
- Learning process 1000 cycles

- Learning 0.0, Forcing zeros 0.0, Forcing ones 0.0, Sigmoid Slope 1000
- Learning process 1 cycle



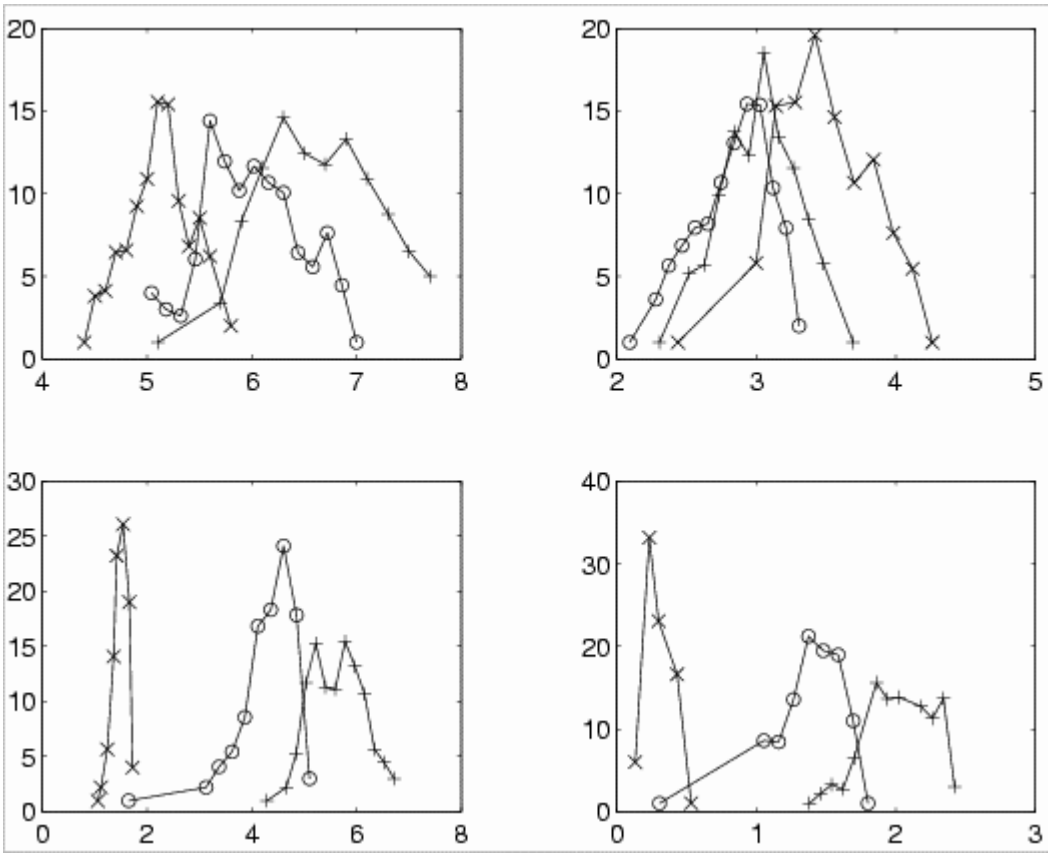
Final network structure with L-units.

IF ($x_3 \leq 2.5$ && $x_4 \leq 1.7$) Iris setosa

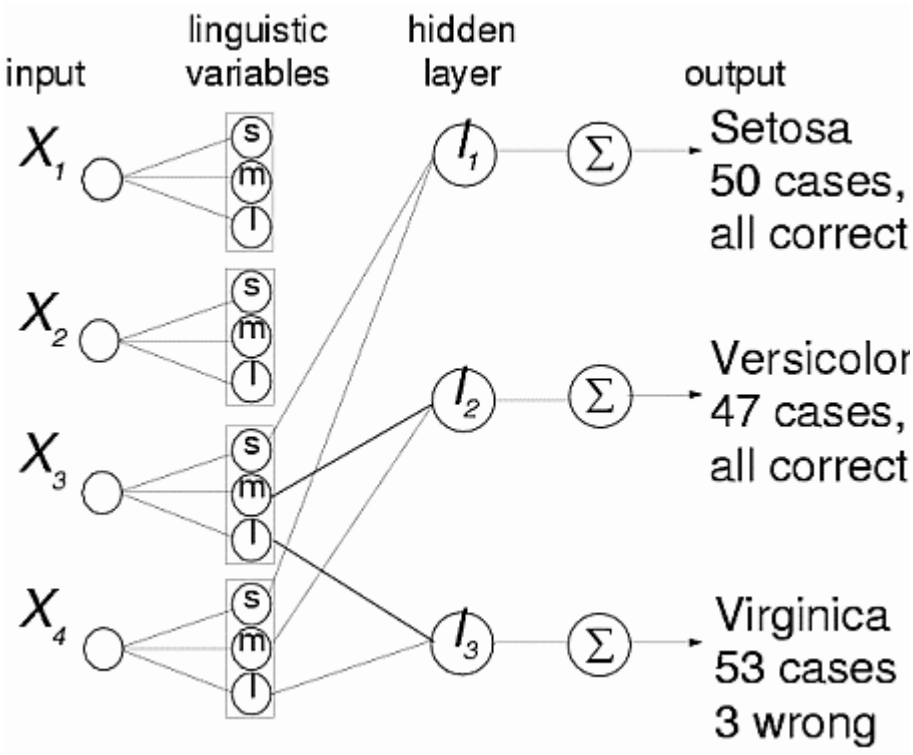
IF ($x_3 > 2.5$ && $x_4 \leq 1.7$) Iris versicolor

IF ($x_3 > 2.5$ && $x_4 > 1.7$) Iris virginica

Start from histograms instead of L units

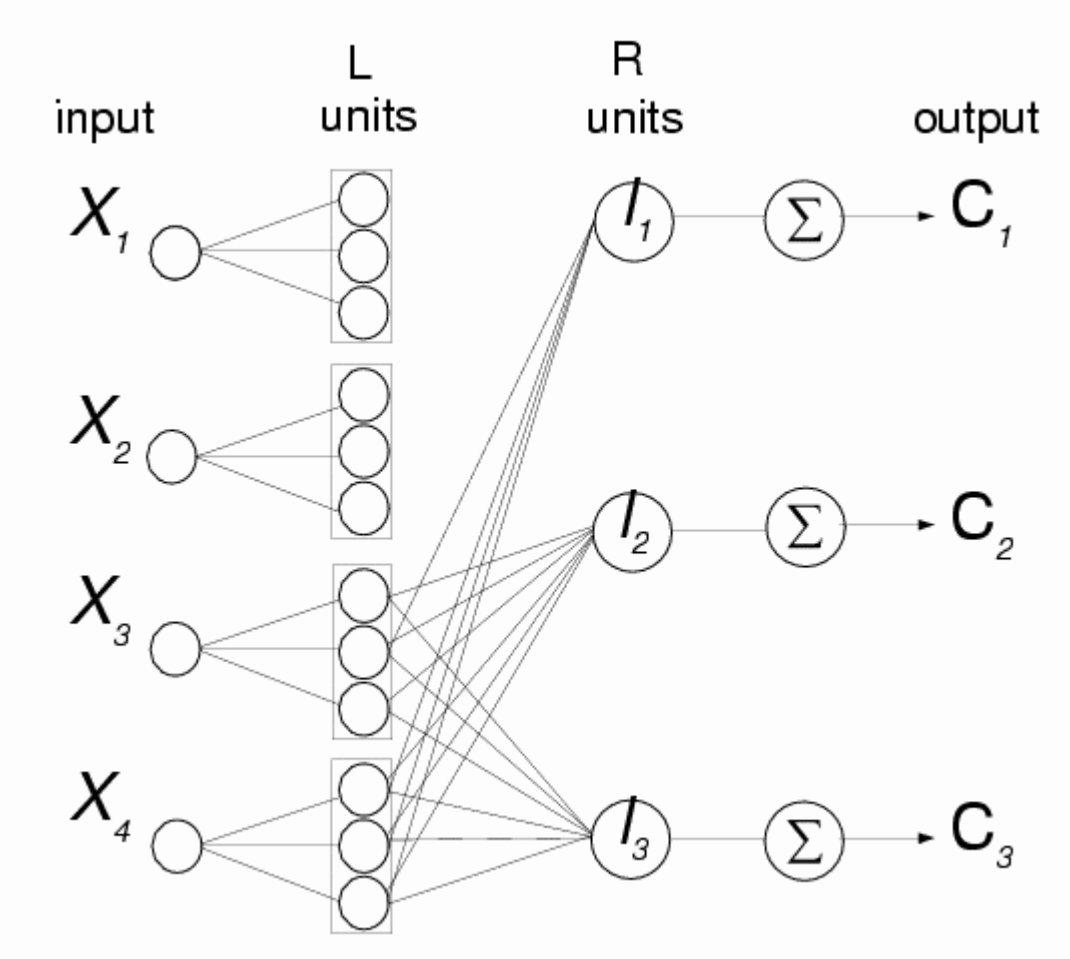


Final result starting from histograms, without L-units:



$$\text{setosa if } x_3 = \text{s} \vee x_4 = \text{s}$$
$$\text{versicolor if } x_3 = \text{m} \wedge x_4 = \text{m}$$
$$\text{virginica if } x_3 = \text{l} \vee x_4 = \text{l}$$

With lower regularization parameters - more complex network:



With stronger regularization - only x_3 is left

IF ($x_3 \leq 2.5$) Iris setosa (100%)
IF ($x_3 > 4.8$) Iris virginica (92%)
ELSE Iris versicolor (94%)

Overall accuracy: 95.3%

Summary

- Constructive algorithm is fast and requires little experimentation with network contruction.
- Sets of rules of different complexity may be created .
- Sets of rules of different rejection rate/reliability are constructed.

PL=x3=Petal Length; PW=x4=Petal Width

PVM Rules: accuracy 98% in leave-one-out and overall

Setosa	PL <3
Virginica	PL > 4.9 OR Petal Width > 1.6
Versicolor	ELSE

C-MLP2LN rules:
7 errors, overall 95.3% accuracy

Setosa	PL < 2.5	100%
Virginica	PL > 4.8	92%
Versicolor	ELSE	94%

Higher accuracy rules: overall 98%

Setosa	PL <2.9	100%
Virginica	PL>4.95 OR PW>1.65	94%
Versicolor	PL∈ [2.9,4.95] & PW∈ [0.9,1.65]	100%

100% reliable rules reject 11 vectors, 8 virginica and 3 versicolor:

Setosa	PL <2.9	100%
Virginica	PL>5.25 OR PW>1.85	100%
Versicolor	PL∈ [2.9,4.9] & PW<1.7	100%

Summary of the Iris rules:

Method	Accuracy	Reference
PVM 1 rule	97.3	Weiss
CART (dec. tree)	96.0	Weiss
FuNN	95.7	Kasabov
NEFCLASS	96.7	Nauck et.al.
FuNe-I	96.7	Halgamuge
PVM 2 rules	98.0	Weiss, optimal result, corresponds to about 96% in CV tests
C-MLP2LN	98.0	Duch et.al.
SSV	98.0	Duch et.al.
Grobian (rough)	100	Browne; overfitting

Refs are in:
W. Duch, R. Adamczak and K. Grabczewski, Methodology of extraction, optimization and application of crisp and fuzzy logical rules. IEEE Transactions on Neural Networks, xxx

2. Search-based MLP method (S-MLP)

Standard MLP architecture;

Weights/biases are all integers or discretized, start from integer weights/biases.

Start from $W_{ij} = 0$, $bias_i = -0.5$, change by 1.
Use beam search techniques instead of backpropagation.

Good results in classification and rule extraction
simple to program
so far used only for a few datasets.

Results for the Ljubljana cancer dataset.

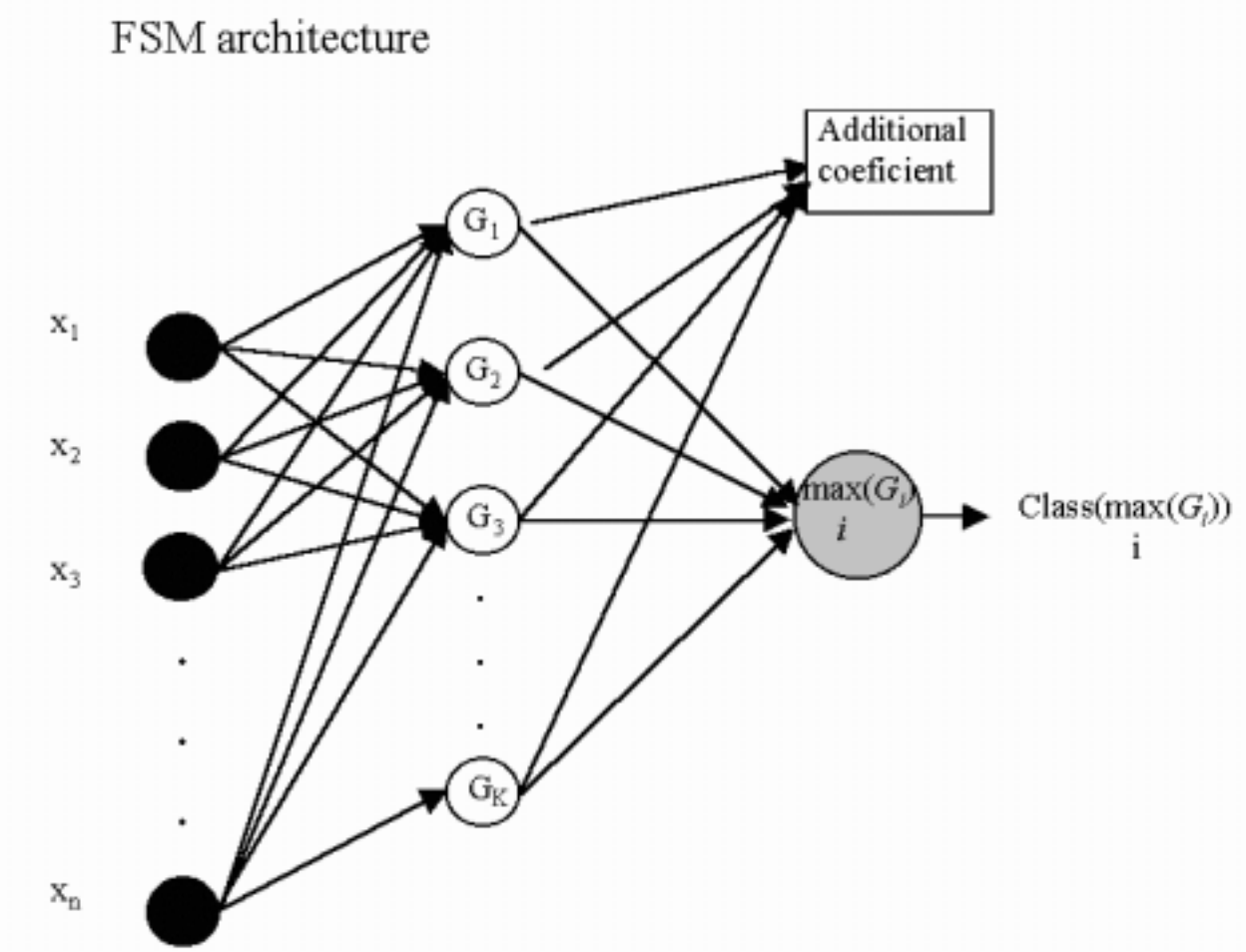
Method	Accuracy %
S-MLP, 2 weights per step	83.6
C-MLP2LN, 2 rules	78.0
Assistant-86	78.0
CART	77.3
CART, PVM, C-MLP2LN single rule	76.2
Naive Bayes rule	75.9
MLP with backpropagation	71.5
AQ15	66-72
Weighted network	68-73.5
Default class	70.3
LERS (rough sets)	67.1
k-NN, k=1	65.3

10-fold cross validation results for the appendicitis data.

Dataset and method	Accuracy
S-MLP	89.7
PVM, C-MLP2LN (logical rules)	89.6
RIAC (prob. inductive)	86.9
MLP+backpropagation	85.8
CART, C4.5 (dec. trees)	84.9
Bayes rule (statistical)	83.0

FSM, Feature Space Mapping neurofuzzy network

Method based on FSM (Feature Space Mapping) neurofuzzy network.
Crisp rules: FSM + rectangular transfer functions.
Fuzzy rules: FSM + context-dependent fuzzy membership functions.



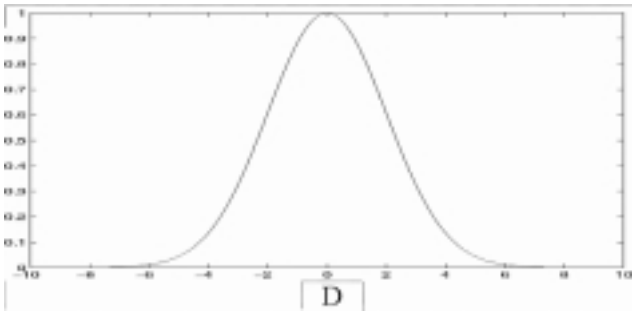
Transfer function

$$G(\mathbf{x};\mathbf{D},\sigma)=\prod_i G_i(x_i;D_i,\sigma_i)$$

Adaptive parameters D, W and n×n matrices Σ^p (rotations and rescaling)

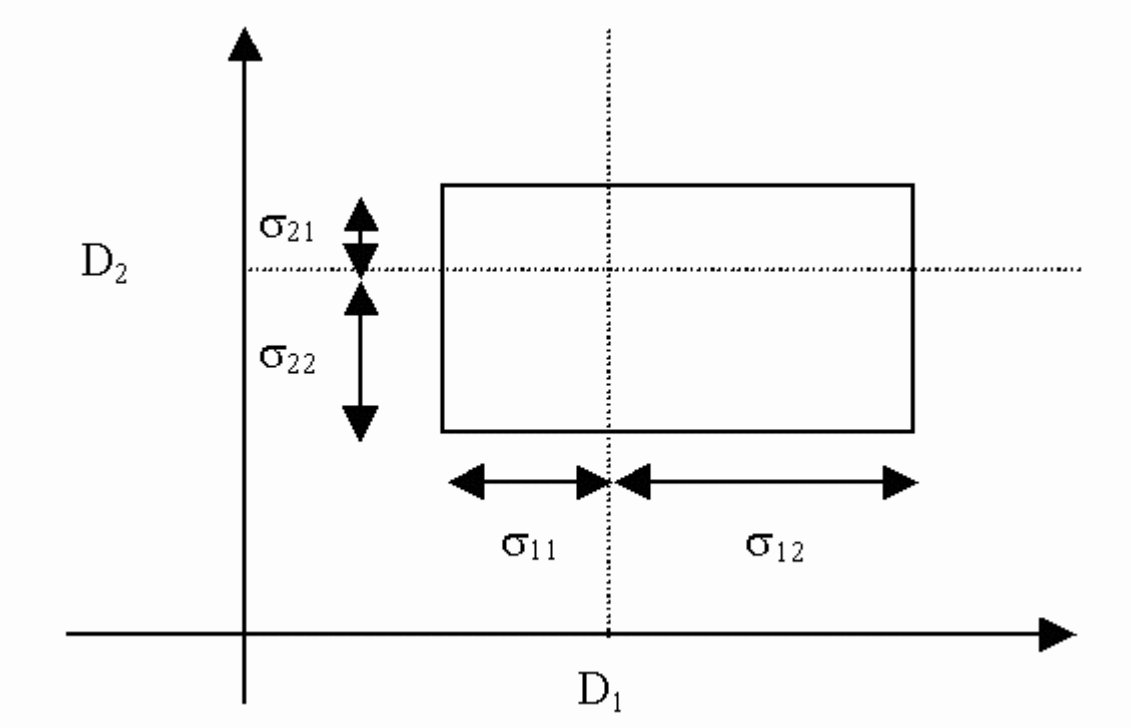
Examples of transfer function
Gauss function

$$G(\mathbf{x};\mathbf{D},\sigma) = e^{-\left(\frac{\|\mathbf{x}-\mathbf{D}\|}{\sigma}\right)^2} = \prod_{i=1}^{n} e^{-\left(\frac{x_i - D_i}{\sigma_i}\right)^2}$$



Rectangular function

$$G(x_i;D_i,\sigma_1,\sigma_2) = \begin{cases} 1 : x_i - D_i < 0 \wedge D_i - x_i < \sigma_{i1} \\ 1 : x_i - D_i \geq 0 \wedge x_i - D_i < \sigma_{i2} \\ 0 : else \end{cases}$$



Bicentral functions – soft trapezoidal functions

$$s(\mathbf{X};\mathbf{D},\Delta) = \prod_{i=1}^{N} \sigma(X_i - D_i)(1 - \sigma(X_i - D_i - \Delta_i))$$

New node conditions

$$S_1(\mathbf{x}^i) = \begin{cases} 1 : Class(\min_k \| \mathbf{x}^i - \mathbf{D}_k \|) = C^i \\ 0 : else \end{cases}$$

$$S_2(\mathbf{x}^i) = \begin{cases} 1 : Class(\max_k (G(\mathbf{x}^i; \mathbf{D}_k, \sigma_k))) = C^i \\ 0 : else \end{cases}$$

Adaptation of parameters

$$D_i = D_i + \frac{\gamma}{m}(x_i - D_i)$$

$$\sigma_i = \sigma_i + \kappa \frac{(1 - G(\mathbf{x}; \mathbf{D}, \sigma)) | x_i - D_i |}{1 + \frac{\tau - \tau_n}{\Lambda}}$$

$$m = m + 1$$

Logical rules for the Iris problem using FSM network

FSM network with rectangular transfer function

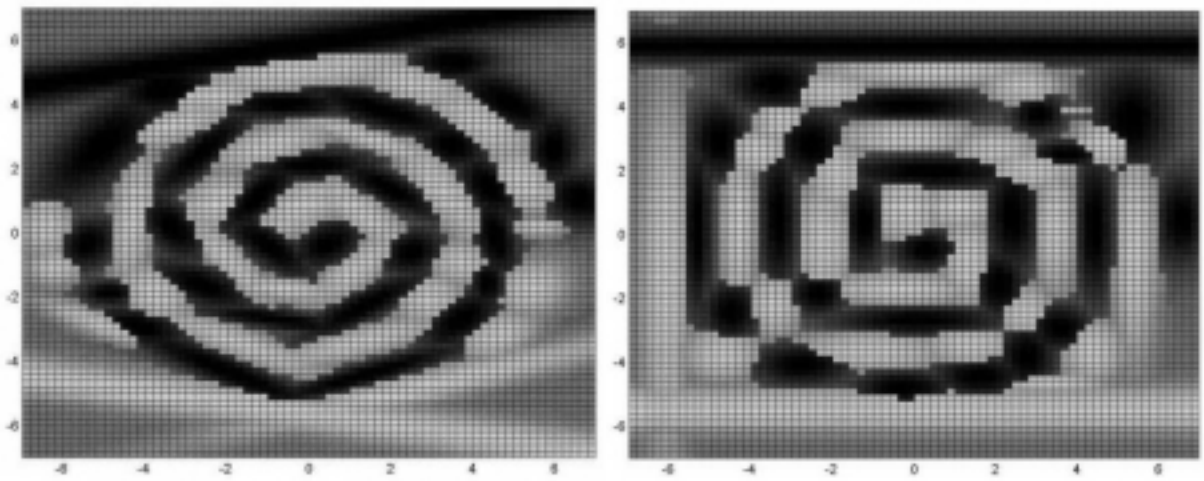
R1: (rule 1)
C4 (feature 4)
-4.89 Iris_setosa
+0.61 Iris_setosa

R2
C3 C4
0.66 0.65 Iris_versicolor
4.90 1.51 Iris_versicolor

5 incorrect classifications

2 Spiral data

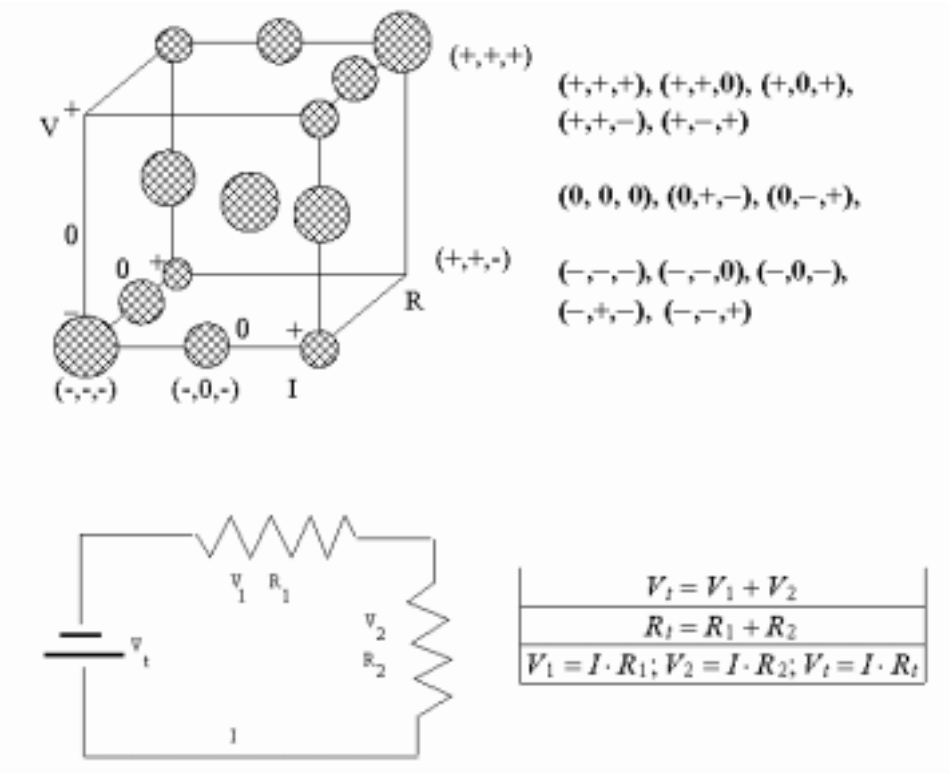
FSM network with Gaussian functions, 53 neurons, and FSM network with Gaussian functions, rotations enabled, 59 neurons



Localized separable functions may be treated as prototypes.

Other applications of FSM: as neural network, neurofuzzy system, prototype-based system or heuristics for search-based reasoning.

Example: any law of the form A=B*C or A=B+C, here Ohm's law V=I*R, has 13 true facts, 14 false facts.



Overview of decision-tree based methods

General remarks:

Decision Trees (DT) are simple to use, use a few parameters, provide simple rules.
Most DT are univariate, axis-parallel.
Oblique trees use linear combinations of input features.

D - training set partitioned into D_k subsets by some tests *T*.
Stop(D_k)=True if assumed leaf purity is reached.

- If Stop(*D*) the tree is a leaf associated with the most frequent class in D.
- Test *T* has mutually exclusive outcomes T_i, $i = 1...K$, subset D_i is composed from cases for which T_i=True.
- Splitting criterion is defined *S(T(x))*.
- For a discrete attribute test *A*=?
- *A*<t for a continuous attribute *A*;
  if *A* has values $v_1 < v_2 < ... v_N <$ check all $t = (v_i + v_{i+1})/2$;
  select the best *S*(*T*(*t*))

Trees are pruned to improve generalization and to generate simpler rules.

CART, Classification and Regression Tree

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984) Classification and Regression Trees", Wadsworth.
Split criterion is based on *Gini*(node) index:

$$Gini = 1 - \sum_i p_i^2$$

p_i is the probability of class *i* vectors in the node.
For each possible split calculate *Gini*, select split with minimum impurity.
Use minimal cost-complexity pruning, rather sophisticated.

DB-CART - added boosting and bagging.

Boosting: making a committee of many classifiers trained on the same training data, with re-weighted wrongly classified cases.

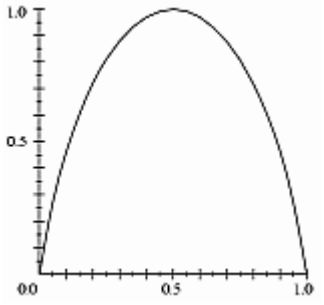
Bagging, bootstrap aggregating: making a committee of many classifiers trained on subsets of data created from the training set by bootstrap sampling (i.e. drawing samples with replacement).
Commercial version of CART and IndCART: different ways of handling missing values and pruning.

C 4.5

Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufmann.
C 4.5 splitting criterion is the gain ratio:
for C classes and fraction $p(D;j)=p(C_j|D)$ in j-th class
the number of information bits the set D contains is:

$$Info(D) = -\sum_{j=1}^{C} p(D,j) \times \log_2(p(D,j))$$

For 2 classes information (vertical) changes with $p(D;1)$=1-$p(D;2)$ reaching max. for 0.5
Info = expected number of bits required to encode a randomly selected training case.



Information gained by a test T with k possible values is:

$$Gain(D,T) = Info(D) - \sum_{i=1}^{k} \frac{|D_i|}{|D|} \times Info(D_i)$$

Max. for tests separating D into one-dimensional subsets; attributes with many values are always selected.
Use information gain ratio instead: gain divided by the split information

$$Split(D,T) = -\sum_{i=1}^{k} \frac{|D_i|}{|D|} \times \log_2\left(\frac{|D_i|}{|D|}\right)$$

Improvements of continuous attribute treatment in C5:

- The Minimum Description Length (MDL) principle is used: minimize theory and exceptions costs
  Modify $Gain(D;T)$ - $\log_2 (N\text{-}1)/|D|$
- Threshold t is chosen to maximize gain.

C4.5 rule generation algorthm, used usually before pruning.
Convert each tree path to a rule:
IF $Cond_1$ AND $Cond_2$ ... AND $Cond_n$ THEN class C

- Remove conditions which are not useful.
- Remove empty rules and identical rules.
- Group all rules according to classes.
- Delete rules if the accuracy of the whole set of rules for the class is not lowered.
- Ordered the rules to minimize false positive errors.
- Try to delete rules in turn if accuracy of the whole ruleset on the training set is not lowered.

Z. Zheng, "Scaling Up the Rule Generation of C4.5". Proc. of PAKDD'98, Berlin: Springer

Verlag, 348-359, 1998.
Rules are frequently more accurate and simpler than trees, especially if generated from pruned trees.

ANN-DT - Decision Trees from Neural Networks

G.P. J. Schmitz, C. Aldrich, and F.S. Gouws, "ANN-DT: An Algorithm for Extraction of Decision Trees from Artificial Neural Networks". Transactions on Neural Networks 10 (1999) 1392-1401

Train an MLP or RBF model
Generate more data interpolating input points in the neighborhood of the training data (equivalent to adding noise).
Use NN as an oracle to predict class.
Create DT using CART criteria or alternative criteria (correlation between variation of the network and variation of the attribute) to analyze attribute significance.
Prune the network using CART approach.
A few results so far, first good NN should be created.

OC - Oblique Classifier

Many variants of the oblique tree classifiers: CART-LC, CART-AP, OC-1, OC!LP, OC-1AP ...
For some data results are significantly better, trees are smaller, but rules are less comprehensible - combinations of inputs are used.
There is no comparison between neural methods of rule extraction (with aggregation) and oblique trees so far.

Inductive methods

R. Michalski, "A theory and methodology of inductive learning". Artificial Intelligence 20 (1983) 111-161.
StatLog project book:
D. Michie, D.J. Spiegelhalter and C.C. Taylor, "Machine learning, neural and statistical classification". Elis Horwood, London 1994
Many inductive methods have been proposed in machine learning.

PVM

S. Weiss, 1988
Maximize predictive accuracy of a single condition rule, make exhaustive or heuristic search.
Try combinations of 2 conditions.
Expensive but for small datasets finds very simple rules.

RISE - Rule Induction from a Set of Exemplars (Domingos 1996)

Exemplars are maximally specific rules.

- Loop over rules;
- find the nearest example from the same class not yet covered;
- try to generalize existing rule covering the new case - compute change in accuracy and accept it unless classification decreases;
- if no rule is generalized stop.

Use hybrid similarity function, good for nominal and numerical attributes.

SSV, Separability Split Value decision tree

SSV separability criterion: separate maximum number of pairs from different classes minimizing the number of separated pairs from the same class.

$$\mathrm{LS}(s,f,D) = \begin{cases} \{x \in D : f(x) < s\} & \text{if } f \text{ is continuous} \\ \{x \in D : f(x) \notin s\} & \text{otherwise} \end{cases}$$

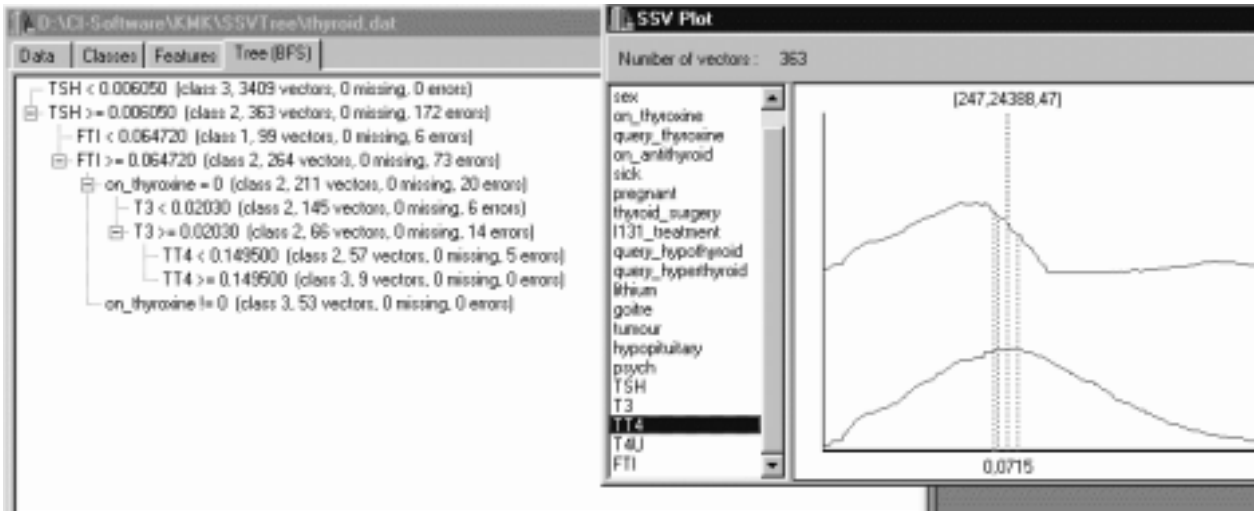
$$\mathrm{RS}(s,f,D) = D - \mathrm{LS}(s,f,D)$$

$$\mathrm{SSV}(s) = 2 \sum_{c \in C} |LS(s,f,D) \cap D_c| \cdot |RS(s,f,D) \cap (D - D_c)|$$

$$- \sum_{c \in C} \min(|LS(s,f,D) \cap D_c|, |RS(s,f,D) \cap D_c|)$$

Simple, automatic; gives useful linguistic variables; deals with discrete and continuous features; handles missing values.

Applications: discretization, feature selection, rules, decision trees.



Each node of the tree is described by:

- the split condition
- the number of vectors in the node (satisfying the condition)
- the number of missing values within that vectors for the split feature
- the number of erroneously classified vectors.

The SSV plot shows criterion values against split values for the feature selected in the list on the left. The plot lines show the following:

- red - the number of errors if we add the split to the tree
- green - the first part of SSV - the number of correctly separated pairs
- blue - the second part of SSV - the number of separated pairs from the same class

Remarks:

- The numbers above the SSV plot lines show the values of the red, green and blue curves for the best split value for the presented feature
- The value below the plot is the best split value for the presented feature
- SSV estimates separability, so it can significantly differ from the error curve (red line)
- Simple, automatic, easy to program.
- Accurate and simple logical rules were obtained using SSV.
- Always use it first

Prototype-based explanation

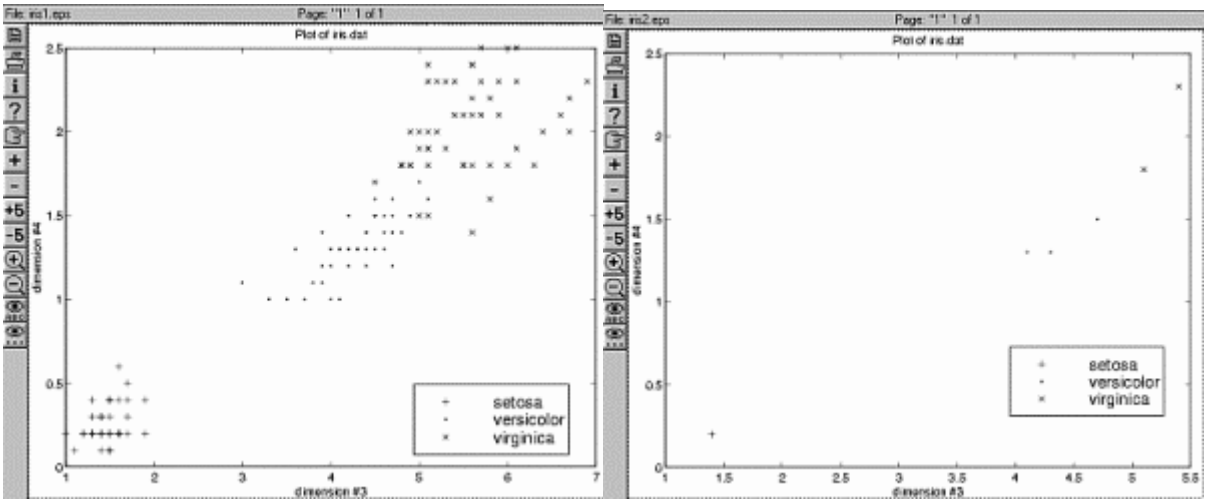
Select the best prototypes - "supermans".

SBL, Similarity Based Learner

Simplest approach: select references in *k*-nearest neighbor method.

SBL - performs all kind of similarity-based evaluations and optimizations.

Example: Original Iris data and 6 prototypes giving the same accuracy of classification



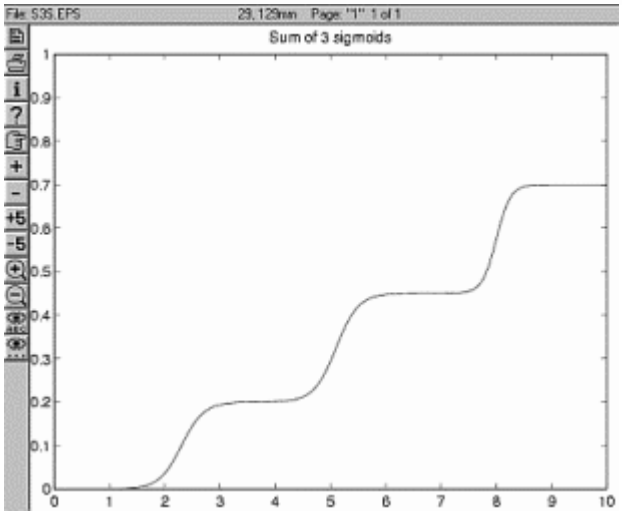
Display decision borders around prototypes – all depends on the type of similarity functions used.

How to use Similarity Based Methods of logical rule extraction?

Rules possible with:

- Variants of nearest neighbor methods with special distance functions (sums of sigmoids)

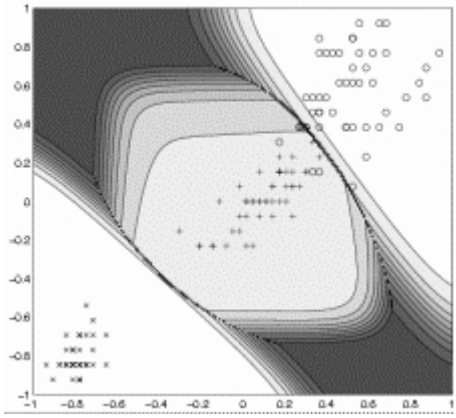
$$d(\mathbf{A}_i, \mathbf{B}_i) = \sum_j \alpha_{ij} \sigma\left(\beta_{ij} |\mathbf{A}_i - \mathbf{B}_i| - \gamma_{ij}\right)$$



Minimize in-class distance and maximize between-class distance: well known technique in statistics.
Neural-like realization with such distance function.

- Neural k-NN with large exponents in Minkovsky's distance

$$D(\mathbf{X}, \mathbf{Y}; s)^{\alpha} = \sum_i^N s_i d(\mathbf{X}_i, \mathbf{Y}_i)^{\alpha}$$



Iris case for α=7; for large α decision borders are rectangular.

Visualization-based explanation

Explanatory data analysis - show the data.
Overview of visualization methods: if time permits ...
SOM - most popular, trying to classify/display at the same time, but poorly.

6. PCI, Probabilistic Confidence Intervals

- May be used with any classifier.
- Shows the probabilities in the neighborhood of the case analyzed for all/each feature.
- Allows to evaluate reliability of classification, but not to explain the data.

Presented on separate pages by Norbert Jankowski.

IMDS, Interactive database exploration using multidimensional scaling

Data topography preserving mapping method: MDS (Least Squares Scaling)

- Minimization of a Stress function as:

$$S(x) = \sum_{i<j}^{N_t} w_{ij} \cdot \left(D_{ij} - d_{ij}\right)^2$$

  where w_{ij} are weights allowing to control which distances are to be better preserved.
  using a gradient descent method (steepest descent, conjugate gradient, quasi-Newton, ...)

  Our choice: steepest descent with 2nd order optimization of the step-size along the gradient

- Relative Stress expression to map N_m new data points:

$$S_r(x) = \sum_{i<j}^{N_m} w_{ij} \cdot \left(D_{ij} - d_{ij}\right)^2 + \sum_{i=1}^{N_m} \sum_{j=N_m+1}^{N_t} w_{ij} \cdot \left(D_{ij} - d_{ij}\right)^2$$

- Locally weighted Stress expression to force preservation of distances close to a chosen point P_c:

  Multiply previous weight w_{ij} by a Gaussian-like term centered on P_c, decreasing when the mean distance

$D_{cij} = (D_{ci} + D_{cj})/2$ between D_{ij} end points and point P_c is increasing:

$$w_{cij} = w_{ij} \cdot e^{-D_{cij}^2 / 2\sigma^2}$$

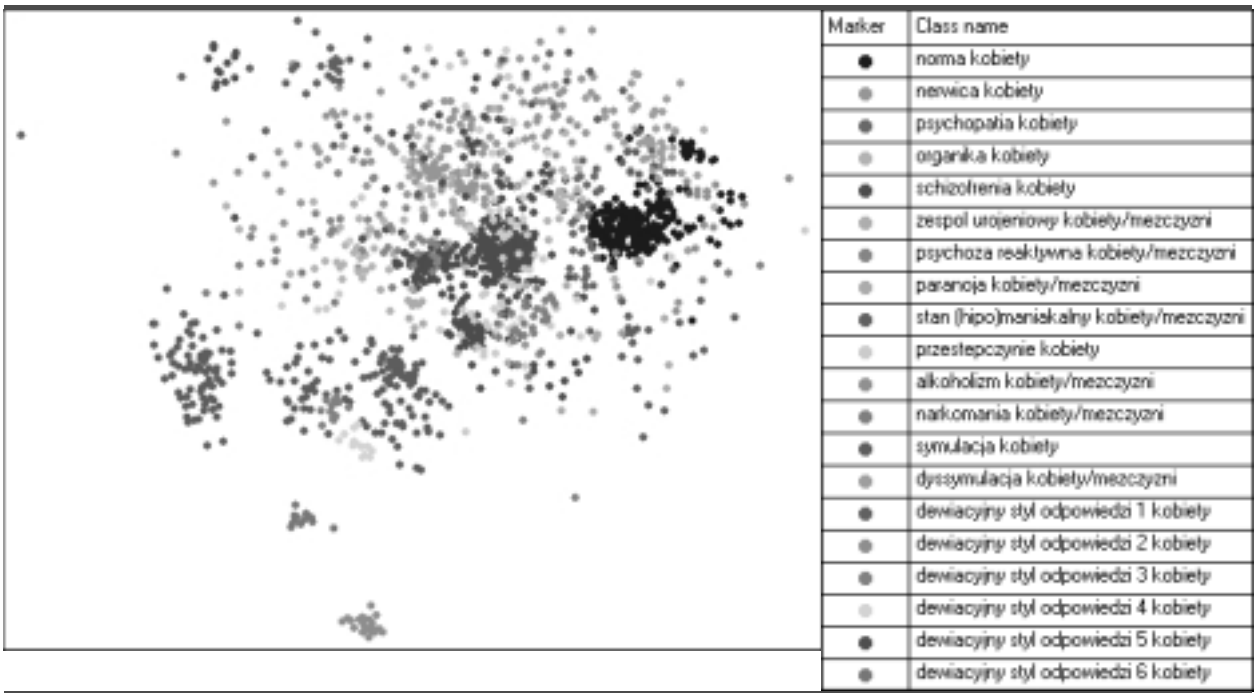
Example: Real life database visualization

Psychometric MMPI test: patients as samples, numerical factors as attributes

Two datasets: Men / Women.

- Women dataset: 1611 samples, 13 attributes, 20 classes
- Men dataset: 1716 samples, 13 attributes, 20 classes

Women dataset



Metric MDS mapping of the Women database.
$S_0 = 0.075$ (PCA initialization) $S_{conv} = 0.024$.

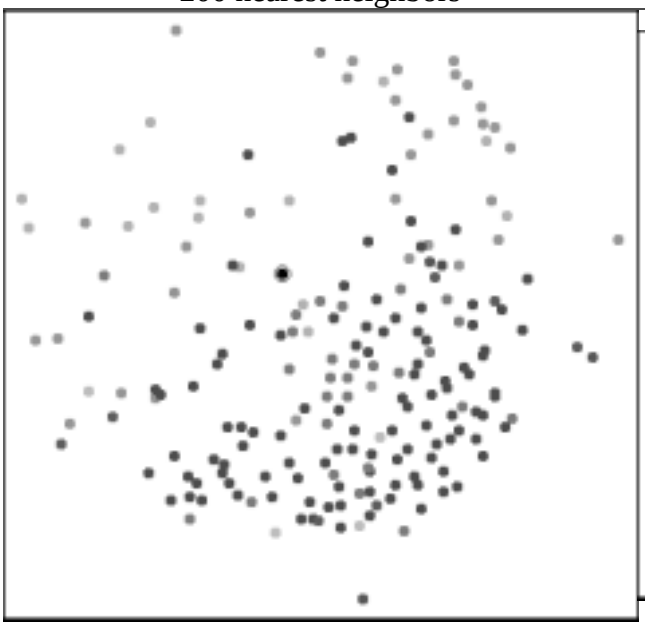
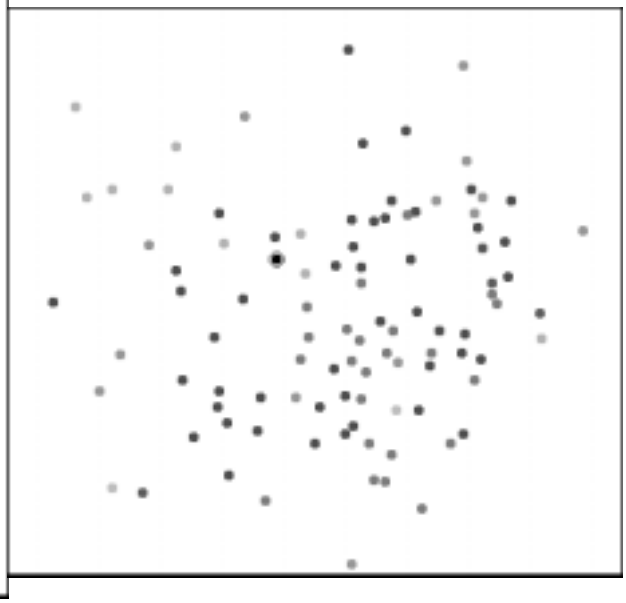
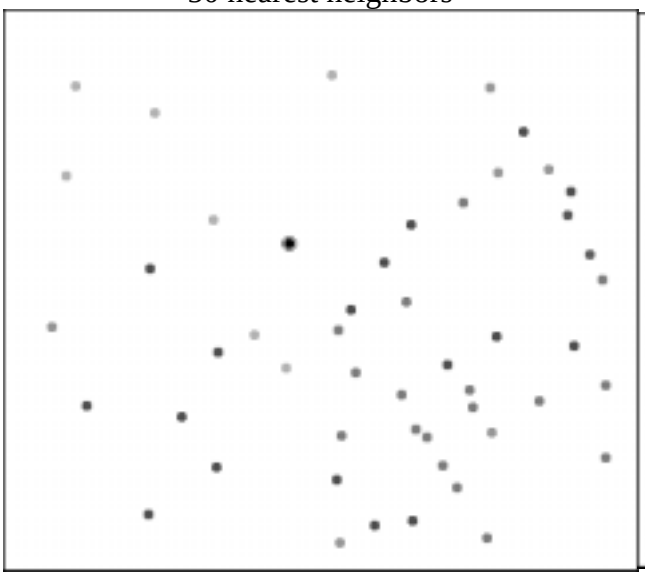
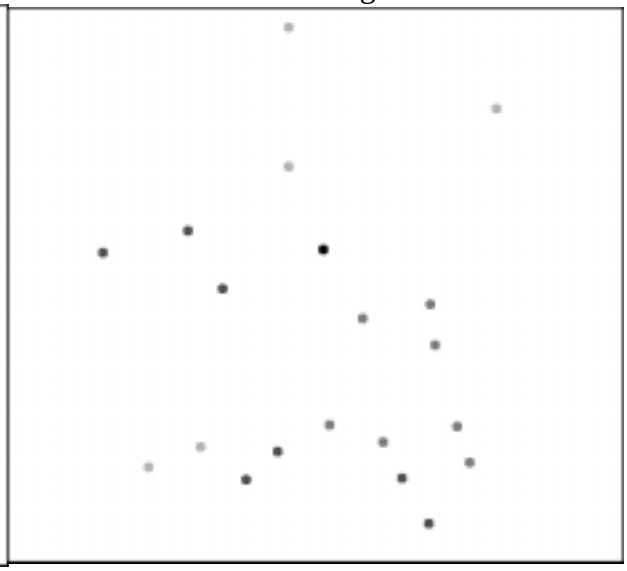
Focusing on data point 'p554' from class 'organika'

- Purpose: View (Understand) why this data is classified into class 'organika'.

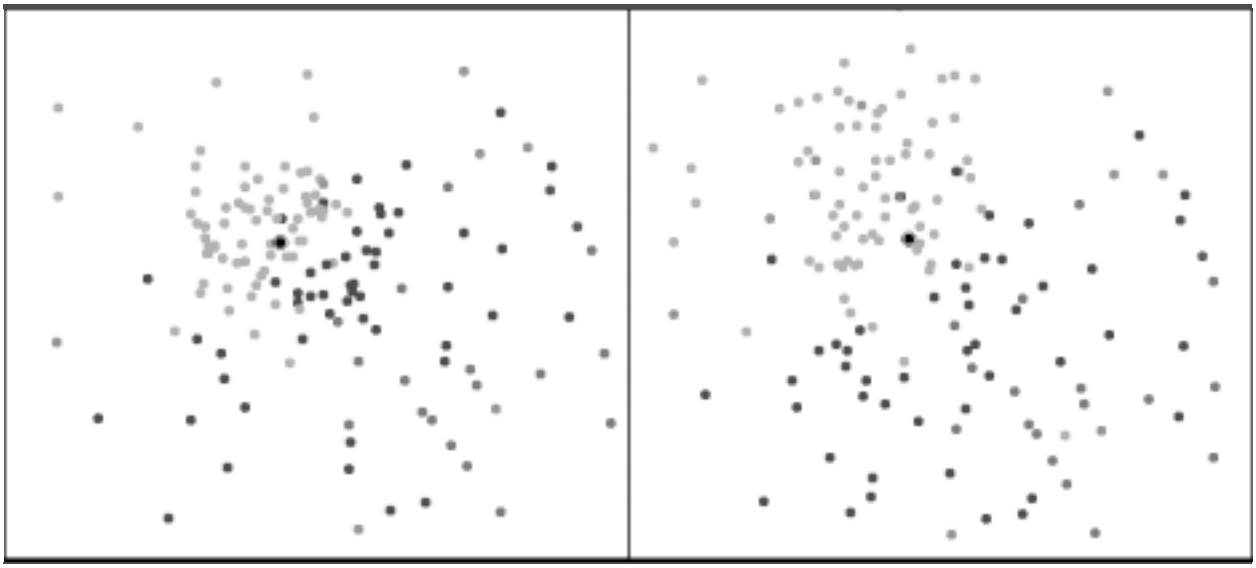
- Classified using **IncNet** neural network, for which features 2, 4 and 7 are sufficient to classify correctly class 'organika'.

- To avoid interference from noisy dimensions, only those dimensions (2,4,7) were used for the MDS mapping,
-
  - Progressive zooming by mapping successively the 200, 100, 50 and 20 nearest data

points (interactively selected) from point 'p554' (marked by a black circled dot).

200 nearest neighbors	100 nearest neighbors
	
S_{conv} = 0.02695 (random initialization, trial 6)	S_{conv} = 0.14635 (random init., trial 24)
50 nearest neighbors	20 nearest neighbors
	
S_{conv} = 0.02849 (random initialization, trial 2)	S_{conv} = 0.01899 (random initialization, trial 1)

Visualization of IncNet classifier's decision borders



The 50 nearest neighbors with 100 Gaussian (σ =1) points classified	The 50 nearest neighbors with 100 Gaussian (σ =2) points classified

1 - Generation of 100 new points from a Gaussian distribution centered at p554,
2 - Classification of the new points using IncNet classifier,
3 - Addition of the new points to the 100 nearest neighbors map using relative mapping (each point is mapped separately).

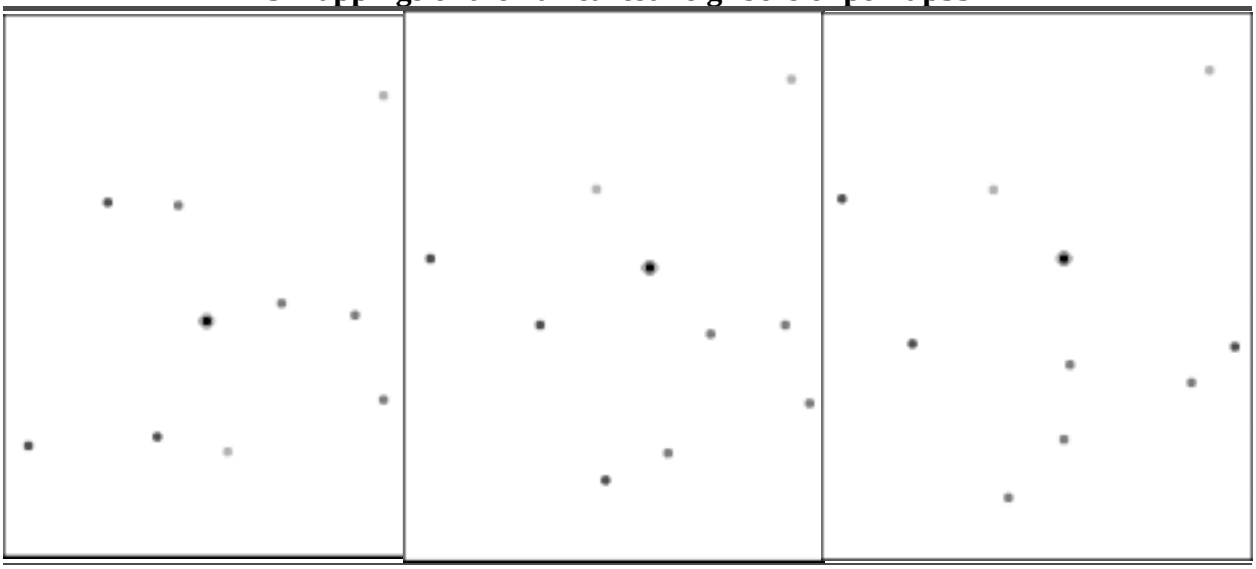
Sensitivity to initial configuration:

Initialization of the configuration:

- Initialization using the first 2 principal components (SVD of the coordinates matrix),
- Random initialization repeated a number of times,

Our strategy: Initialize using PCA and 20 random trials and then keep the best run.

3 mappings of the 10 nearest neighbors of point p554



S_{conv} = 0.03904 (PCA initialization)	S_{conv} = 0.023181 (random initialization, trial 1)	S_{conv} = 0.023176 (random initialization, trial 2)

Features of MDS mapping for database visualization

- When using local minimization method, initial configuration is of crucial importance,
- Small differences in final Stress value may correspond to noticeably different displayed configuration,
- Interactive User Interface implies fast mapping algorithm,
- Reliable mapping implies performant minimization, which implies lengthy procedures,
- A compromise must be found between mapping speed and mapping quality.

Features of our MDS mapping software (prototype GUI)

- On-line mapping: seeing how the configuration evolves during mapping,
- Possibility to add new points to an existing map using ***relative mapping***,
- Interactive selection of a subset of points: in a rectangle, on a disc of given radius, a N-dimensional sphere,
- Dataset display transformation: translation, rotation, horizontal or vertical flipping and **zooming**,
- Generation of new points in Gaussians for their classification allows to see classifiers decision borders,
- 'Batch' mapping option: Map in 1 run (all data points together) / Map in x runs (batches of N_t / x points).

Some knowledge discovered

Iris – comparison was already made;

4 measurements in cm, petals and sepals, for example:

5.1,3.5,1.4,0.2, Iris-setosa
4.9,3.0,1.4,0.2, Iris-setosa
4.7,3.2,1.3,0.2, Iris-setosa
6.3,3.3,4.7,1.6, Iris-versicolor
4.9,2.4,3.3,1.0, Iris-versicolor
5.8,2.7,4.1,1.0, Iris-versicolor
6.3,2.9,5.6,1.8, Iris-virginica
6.5,3.0,5.8,2.2, Iris-virginica
6.5,3.0,5.5,1.8 Iris-virginica

Mushrooms

The Mushroom Guide clearly states that there is no simple rule for determining the edibility of these mushrooms; no rule like „leaflets three, let it be„ for Poisonous Oak and Ivy.

8124 cases, 22 symbolic attributes, up to 12 values each, equivalent to 118 logical features.
51.8% represent edible, the rest non-edible mushrooms.

Example:

edible, convex, fibrous, yellow, bruises, anise, free, crowded, narrow, brown, tapering, bulbous, smooth, smooth, white, white, partial, white, one, pendant, purple, several, woods

poisonous, convex, smooth, white, bruises, pungent, free, close, narrow, white, enlarging, equal, smooth, smooth, white, white, partial, white, one, pendant, black, scattered, urban

edible, convex, fibrous, yellow, bruises, anise, free, crowded, narrow, brown, tapering, bulbous, smooth, smooth, white, white, partial, white, one, pendant, purple, several, woods

edible, flat, smooth, white, bruises, almond, free, crowded, narrow, pink, tapering, bulbous, smooth, smooth, white, white, partial, white, one, pendant, purple, several, woods

edible, bell, smooth, white, bruises, almond, free, close, broad, white, enlarging, club, smooth,

smooth, white, white, partial, white, one, pendant, black, scattered, meadows

poisonous, convex, smooth, white, bruises, pungent, free, close, narrow, white, enlarging, equal, smooth, smooth, white, white, partial, white, one, pendant, black, scattered, urban

poisonous, convex, smooth, white, bruises, pungent, free, close, narrow, pink, enlarging, equal, smooth, smooth, white, white, partial, white, one, pendant, black, several, urban

poisonous, convex, smooth, white, bruises, pungent, free, close, narrow, pink, enlarging, equal, smooth, smooth, white, white, partial, white, one, pendant, brown, scattered, urban

Safe rule for edible mushrooms:

odor = (almond.or.anise.or.none) $\wedge$ spore-print-color = $\neg$ green — 48 errors, 99.41% correct
This is why animals have such a good sense of smell!
Other odors: creosote, fishy, foul, musty, pungent or spicy

Rules for poisonous mushrooms - 6 attributes only

R_1) odor = $\neg$ (almond $\vee$ anise $\vee$ none); — 120 errors, 98.52%
R_2) spore-print-color = green — 48 errors, 99.41% correct
R_3) odor = none $\wedge$ stalk-surface-below-ring = scaly $\wedge$ stalk-color-above-ring = $\neg$ brown — 8 errors, 99.90%
R_4) habitat = leaves $\wedge$cap-color = white — no errors!

R_1 + R_2 are quite stable, found even with 10% of data using CMLP2LN;
R_3 and R_4 may be replaced by other rules:

R'$_3$): gill-size = narrow $\wedge$ stalk-surface-above-ring = (silky $\vee$ scaly)
R'$_4$): gill-size = narrow $\wedge$ population = clustered

Only 5 attributes used ! These rules were found using SSV.

Method	Rules	Antecedents	Accuracy %
RULENEG	300	8087	91.00
REAL	155	6603	98.00
DEDEC	26	26	99.80
TREX	3	13	100
C4.5 (decision tree)	3	3	99.80
RULEX	1	3	98.52
Successive Regularization	1	4	99.41
Successive Regularization	2	22	99.90
Successive Regularization	3	24	100.00

What chemical receptors in the nose realize such discrimination?
What does it tell us about evolution?

Ljubliana breast cancer

286 cases, 201 no recurrence cancer events (70.3%), 85 are recurrence (29.7%) events.
9 attributes, symbolic with 2 to 13 values.

Single rule:

$$\text{involved nodes} = \neg[0,2] \wedge \text{Degree-malignant} = 3$$

with else condition gives over 77% in crossvalidation;
best systems do not exceed 78% accuracy (insignificant difference).

All knowledge contained in the data is:
if more than 2 nodes were involved and it is highly malignant there will be recurrence.

Wisconsin breast cancer

699 cases, 458 benign (65.5%), 241 (34.5%) malignant.
9 attributes, integers 1-10, one attribute missing in 16 cases.

The simplest rules, large regularization:

IF $f_2 \geq 7 \vee f_7 \geq 6$ THEN malignant (95.6%)

Overall accuracy (including ELSE condition) is 94.9%.

f_2 - uniformity of cell size; f_7 - bland chromatin

Hierarchical sets of rules with increasing accuracy may be build

C-MLP2LN gives 5 initial rules for malignant cases.

R_1: $f_2<6 \wedge f_4<4 \wedge f_7<2 \wedge f_8<5$ (100)%

R_2: $f_2<6 \wedge f_5<4 \wedge f_7<2 \wedge f_8<5$ (100)%

R_3: $f_2<6 \wedge f_4<4 \wedge f_5<4 \wedge f_7<2$ (100)%

R_4: $f_2 \in [6,8] \wedge f_4<4 \wedge f_5<4 \wedge f_7<2 \wedge f_8<5$ (100)%

R_5: $f_2<6 \wedge f_4<4 \wedge f_5<4 \wedge f_7 \in [2,7] \wedge f_8<5$ (92.3)%

The last rule covers 39 cases, including 3 errors.

Confusion matrix: $\begin{bmatrix} 238 & 3 \\ 25 & 433 \end{bmatrix}$, with (malignant, benign)

Overall accuracy 96%.

More accurate set of rules:

R_1:	$f_2<6 \wedge f_4<3 \wedge f_8<8$		(99.8)%
R_2:	$f_2<9 \wedge f_5<4 \wedge f_7<2 \wedge f_8<5$		(100)%
R_3:	$f_2<10 \wedge f_4<4 \wedge f_5<4 \wedge f_7<3$		(100)%
R_4:	$f_2<7 \wedge f_4<9 \wedge f_5<3 \wedge f_{7\in}[4,9] \wedge f_8<4$		(100)%
R_5:	$f_{2\in}[3,4] \wedge f_4<9 \wedge f_5<10 \wedge f_7<6$	$\wedge f_8<8$	(99.8)%

R_1 and R_5 misclassify the same 1 benign vector.
ELSE condition makes 6 errors, overall reclassification accuracy 99.00%

In all cases features f_3 and f_6 (uniformity of cell shape and bare nuclei) are not important, f_2 and f_7 being the most important.

100% reliable set of rules rejects 51 cases (7.3%).

For malignant class 4 rules are obtained;

For the benign cases rules are obtained by negation $\neg (R_1 \vee R_2 \vee R_3 \vee R_4)$,

followed by optimization of intervals.

Results from the **10-fold (stratified) crossvalidation** - accuracy of rules is hard to compare without the test set

Method	**% accuracy**
IncNet	**97.1**
3-NN, Manhattan	**97.1≠ 0.1**
Fisher LDA	**96.8**
MLP+backpropagation	**96.7**
LVQ (vector quantization)	**96.6**
Bayes (pairwise dependent)	**96.6**
FSM (density estimation)	**96.5**
Naive Bayes	**96.4**
Linear Discriminant Analysis	**96.0**
RBF	**95.9**
CART (decision tree)	**94.2**
LFC, ASI, ASR (decision trees)	**94.4-95.6**
Quadratic Discriminant Analysis	**34.5**

The Hypothyroid dataset

Data from Machine Learning Database repository, UCI
3 classes: hypothyroid, hiperthyroid, normal;
training vectors 3772 = 93+191+3488
test vectors 3428 = 73+177+3178
21 attributes (medical tests), 6 continuos

Optimized rules: 4 errors on the training set (99.89%), 22 errors on the test set (99.36%)

primary hypothyroid:	TSH30.48 & FTI <64.27	97.06 %
primary hypothyroid:	TSH=[6.02,29.53] & FTI <64.27 & T3< 23.22	100%
compensated:	TSH 6.02 & FTI[64.27,186.71] & TT4=[50, 150.5) & On_Tyroxin=no & surgery=no	98.96 %
no hypothyroid:	ELSE	100%

4 continuous attributes used and 2 binary.

Method	% training	% test	Reference
C-MLP2LN rules + ASA opt.	99.9	99.36	our group
CART	99.8	99.36	Weiss
PVM	99.8	99.33	Weiss
IncNet	99.7	99.24	our group
MLP init+ a,b opt.	99.5	99.1	our group
C-MLP2LN rules	99.7	99.0	our group
Cascade correlation	100.0	98.5	Schiffmann
BP + local adapt. rates	99.6	98.5	Schiffmann
BP+genetic opt.	99.4	98.4	Schiffmann
Quickprop	99.6	98.3	Schiffmann
RPROP	99.6	98.0	Schiffmann
3-NN, Euclides, 3 features used	98.7	97.9	our group
1-NN, Euclides, 3 features used	98.4	97.7	our group
Best backpropagation	99.1	97.6	Schiffmann
1-NN, Euclides, 8 features used	--	97.3	our group
Bayesian classif.	97.0	96.1	Weiss
BP+conjugate gradient	94.6	93.8	Schiffmann
1-NN Manhattan, std data		93.8	our group
default: 250 test errors		92.7	
1-NN Manhattan, raw data		92.2	our group

NASA Shuttle

Training set 43500, test set 14500, 9 attributes, 7 classes
Approximately 80% of the data belongs to class 1, only 6 vectors in class 6.

Rules from FSM after optimization: 15 rules, train 99.89%, test 99.81% accuracy.

32 rules obtained from SSV give 100% train, 99.99% test accuracy (1 error).

Method	% training	% test	Reference
SSV, 32 rules	100	99.99	our group
NewID decision tree	100	99.99	Statlog
Baytree decision tree	100	99.98	Statlog
CN2 decision tree	100	99.97	Statlog
CART	99.96	99.92	Statlog
C4.5	99.96	99.90	Statlog
FSM, 15 rules	99.89	99.81	our group
MLP	95.50	99.57	Statlog
k-NN	99.61	99.56	Statlog
RBF	98.40	98.60	Statlog
Logistic DA	96.06	96.17	Statlog
LDA	95.02	95.17	Statlog
Naive Bayes	95.40	95.50	Statlog
Default	78.41	79.16	

More examples of logical rules discovered are on our rule-extraction WWW page
http://www.phys.uni.torun.pl/kmk/projects/rules.html

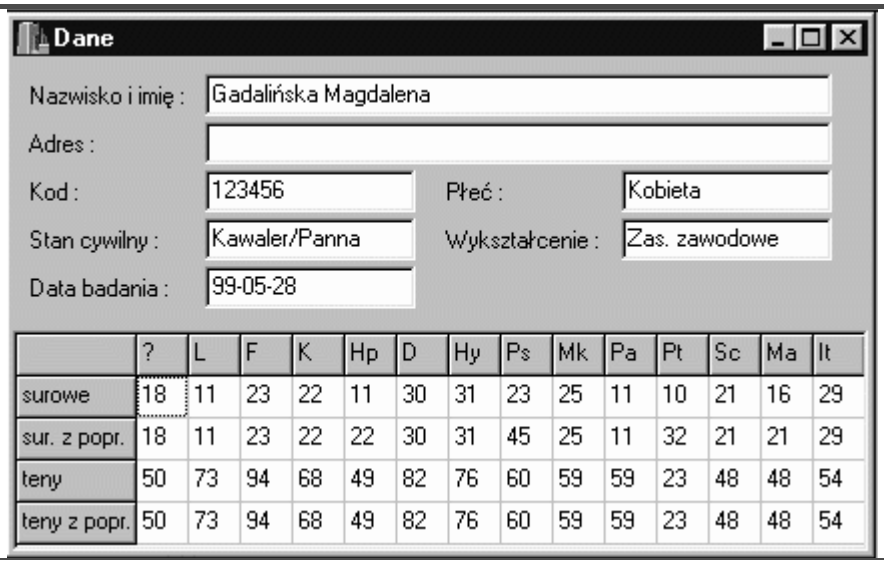
Most people do not publish explicit rules!

Analysis of psychometric questionnaires

Example of an expert system generated with the help of analysis of psychometric data

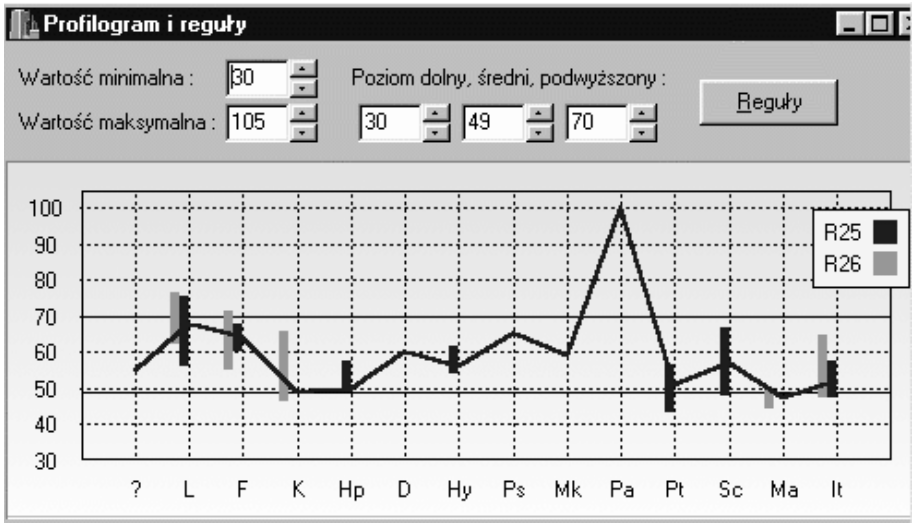
- Start from computerized test or scanning the paper forms.
  MMPI test has 550 questions; any similar test may be computerized.
- Store results in a database for future reference

Compute coefficients (scales) measuring different tendencies. MMPI scales 1-4 used for control, next 10 coefficients are clinical scales: hypochondria, depression, hysteria, psychopathy, paranoia, schizophrenia etc.



	?	L	F	K	Hp	D	Hy	Ps	Mk	Pa	Pt	Sc	Ma	It
surowe	18	11	23	22	11	30	31	23	25	11	10	21	16	29
sur. z popr.	18	11	23	22	22	30	31	45	25	11	32	21	21	29
teny	50	73	94	68	49	82	76	60	59	59	23	48	48	54
teny z popr.	50	73	94	68	49	82	76	60	59	59	23	48	48	54

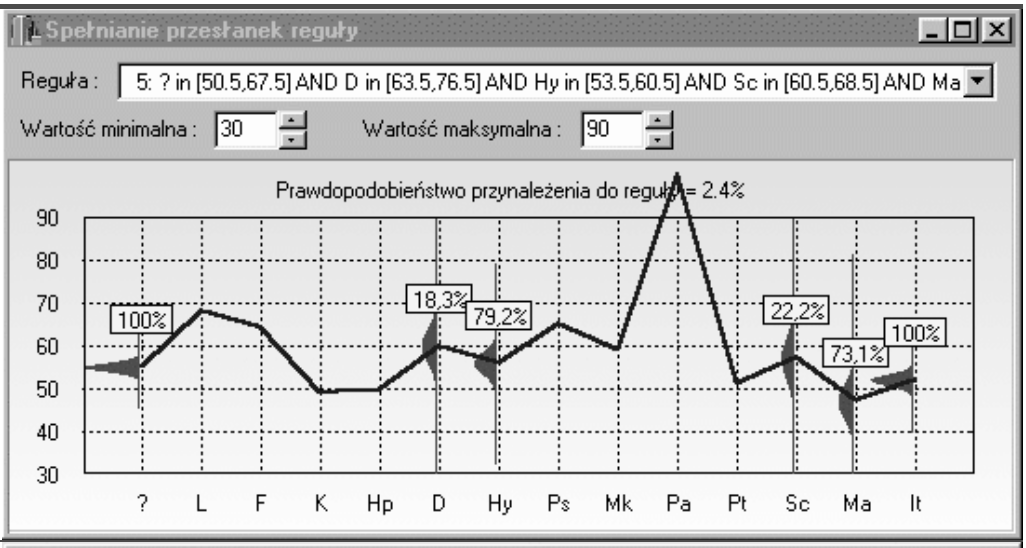
Display scales in a „psychogram", interpreted by skilled psychologists diagnosing specific problems; show rules that are true for this case. Rules are derived from data collected in the Academic Psychological Clinic of Nicholas Copernicus University and in several psychiatric hospitals around Poland.



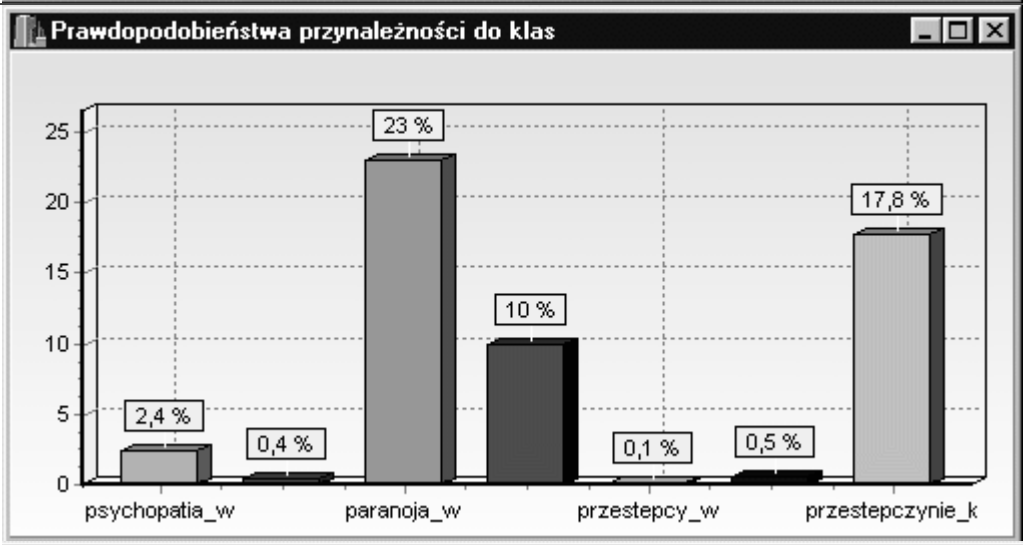
Two datasets used, woman and man, over 1600 cases each, 27 classes (normal, neurotic, drug addicts, schizophrenic, psychopaths, organic problems, malingerers, persons with criminal tendencies etc.).

2-3 rules per class found, a total of 50-100 rules.

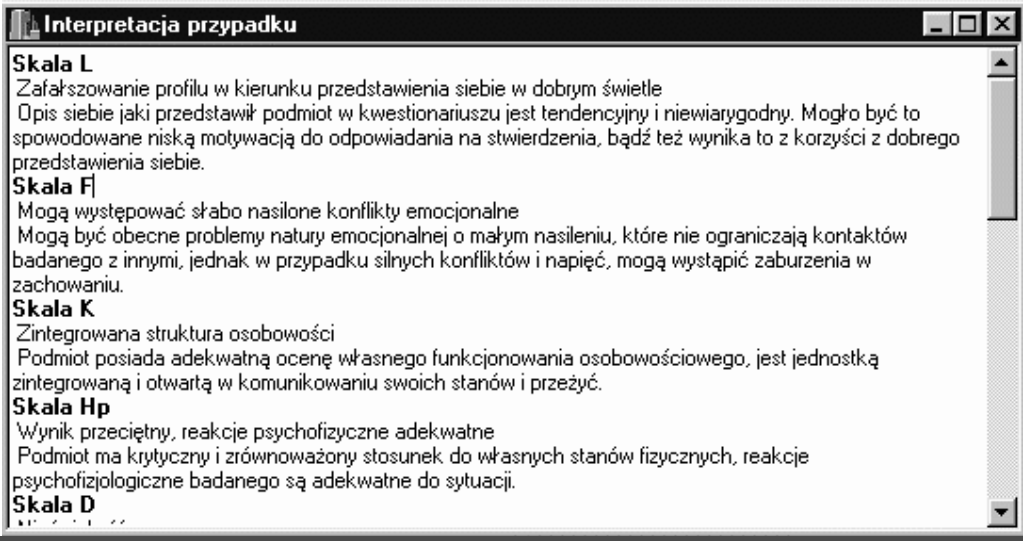
Analyze how each rule fits to the case; vary uncertainty of input measurement (optimal uncertainty has been calculated by minimization of generalization error).



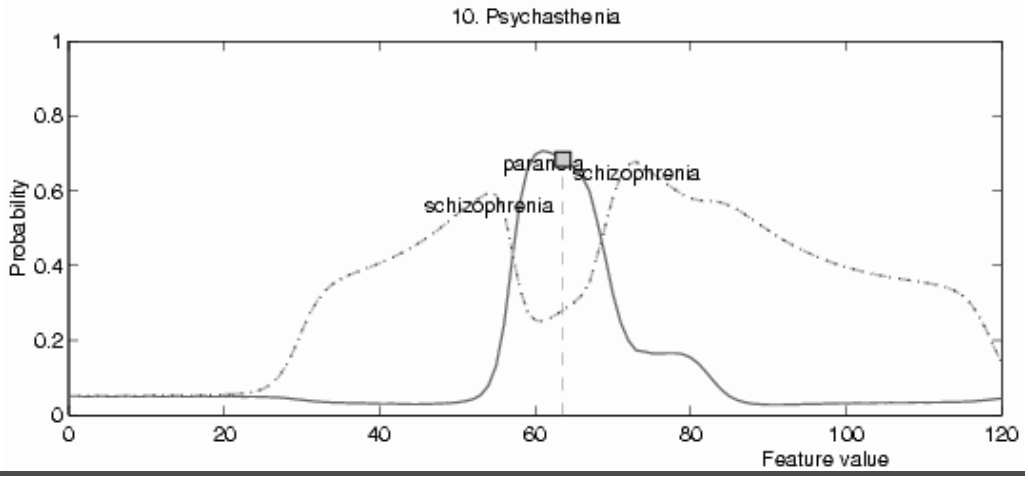
Show probabilities of different diagnoses, graph their dependence on the uncertainity of inputs.



Show verbal interpretation of cases and rules.



- If probability of new classes quickly grows with the assumed uncertainty of the measurement analyze probabilistic confidence levels.



Multidimensional scaling (MDS) allows to see the case in relation to known cases.
Probabilities of different diagnoses may be interpolated to show change of the mental health over time.

Probabilistic confidence levels allow to see detailed changes.
Rules are very important here, allowing for detailed interpretation.
Rules generated using SSV classification tree and FSM neural network.

System	Data	# rules	Accuracy	Fuzzy
C4.5	Women	55	93.0%	93.7%
	Men	61	92.5%	93.1%
FSM	Women	69	95.4%	97.6%
	Men	98	95.9%	96.9%

10-fold crossvalidation gives 82-85% correct answers with FSM (crisp unoptimized rules), and 79-84% correct answers with C4.5.

Fuzzification improves FSM crossvalidation results to 90-92%.

Some questions:

How good are our experts?

How to measure the correctness of such system?

Can we provide useful information if diagnosis is not reliable?

How to deal with several disease - automatic creation of new classes?

Open problems

In real world projects training and finding optimal networks is not our hardest problem ...
Good methods to discover rules exist although proving that simplest sets of rules have been discovered is usually not possible.

Discovering hierarchical structure in the data:

- basic tests are performed first and hypothesis made;
- only the tests necessary to confirm initial hypothesis are made;
- if confirmed no further tests are made; if not more tests are made;
- the data contain large groups of missing values.

Dealing with unknown values.

- values that are not known or have been corrupted in the measurement process (questions not answered);
- values that have not been measured on purpose (questions not asked).

Constructing new, more useful features.
Constructing theories allowing to reason about data – from partial knowledge of subproblems, derived from analysis of datasets, to systematic reasoning.

Constructing new and modifying existing classes.
Building complex systems interacting with humans.

References

Most papers are available from these pages
http://www.phys.uni.torun.pl/kmk/publications.html
http://www.phys.uni.torun.pl/~duch/cv/papall.html

CMLP2LN

- Duch W, Adamczak R, Grąbczewski K, *A new methodology of extraction, optimization and application of crisp and fuzzy logical rules*. IEEE Transactions on Neural Networks (in print, 2000)
- Duch W, Adamczak R, Grąbczewski K, Jankowski N (2000) *Neural methods of knowledge extraction* , Control and Cybernetics (in print)
- Duch W, Grąbczewski K, Jankowski N, Adamczak R (2000) *Optimization and interpretation of rule-based classifiers*. Intelligent Information Systems IIS'2000, Physica Verlag (Springer), pp. 1-13
- Duch W, Adamczak R, Grąbczewski K, Żal G (1999) *Hybrid neural-global minimization method of logical rule extraction*, J. of Advanced Computational Intelligence, 3: 1-9
- Duch W, Adamczak R, Grąbczewski K (1998) *Extraction of logical rules from backpropagation networks.* Neural Processing Letters 7: 1-9
- Kasabov N, Kozma R, Duch W (1998) *Rule extraction from linguistic rule Networks and from Fuzzy Neural Networks: Propositional versus Fuzzy Rules*, 4th International Conference on Neural Networks and their Applications, March 11-13, 1998, Marseille, France, pp. 403-406
- Duch W, Adamczak R, Grąbczewski K, Żal G (1998) *A hybrid method for extraction of logical rules from data*. Second Polish Conference on Theory and Applications of Artificial Intelligence, Łódź, 28-30 Sept. 1998, pp. 61-82
- Duch W, Adamczak R, Grąbczewski K (1997) *Constraint MLP and density estimation for extraction of crisp logical rules from data.* ICONIP'97, New Zealand, Nov.1997, pp. 831-834
- Duch W, Adamczak R, Grąbczewski K (1997) *Extraction of crisp logical rules using constrained backpropagation networks*, International Joint Conference on Neural Networks (IJCNN'97), Houston, Texas, 9-12.6.1997, pp. 2384-2389
- Duch W, Adamczak R, Grąbczewski K, Ishikawa M, Ueda H (1997) *Extraction of crisp logical rules using constrained backpropagation networks - comparison of two new approaches*, Proc. of the European Symposium on Artificial Neural Networks (ESANN'97), Bruge 16-18.4.1997, pp. 109-114
- Duch W, Adamczak R, Grąbczewski K (1996) *Extraction of logical rules from training data using backpropagation networks* The 1st Online Workshop on Soft Computing, 19-30.Aug.1996, pp. 25-30
- Duch W, Adamczak R, Grąbczewski K (1996) *Extraction of logical rules from training data using backpropagation networks* CAI'96, First Polish Conference on Theory and

Applications of Artificial Intelligence, Łódź, 19-21.12.1996, pp. 171-178

FSM

- Duch W, *Platonic model of mind as an approximation to neurodynamics*, in: Brain-like computing and intelligent information systems, ed. S-i. Amari, N. Kasabov (Springer, Singapore 1997), chap. 20, pp. 491-512
- Duch W, Adamczak R, Jankowski N (1997) *New developments in the Feature Space Mapping model*, Third Conference on Neural Networks and Their Applications, Kule, October 1997, pp. 65-70
- Duch W (1996) *From cognitive models to neurofuzzy systems - the mind space approach.* Systems Analysis-Modelling-Simulation 24 (1996) 53-65
- Duch W and Adamczak R (1996) *Feature Space Mapping network for classification*. Proceedings of the Second Conference on Neural Networks and their applications, Orle Gniazdo, 30.IV-4.V.1996, pp. 125-130
- Duch W, Jankowski N, Naud A, Adamczak R (1995) *Feature Space Mapping: a neurofuzzy network for system identification*. Proc. of. Engineering Applications of Neural Networks (EANN'95), Helsinki 21-23.08.1995, pp. 221-224
- Duch W and Diercksen GHF (1995) *Feature Space Mapping as a universal adaptive system* Comp. Phys. Comm. **87**: 341-371

SSV Decision Tree

- Grąbczewski K and Duch W (2000) *The separability of split value criterion*. 5th Conference on Neural Networks and Soft Computing, Zakopane, June 2000, pp. 201-208
- Grąbczewski K, Duch W (1999) *A general purpose separability criterion for classification systems*, 4th Conference on Neural Networks and Their Applications, Zakopane, May 1999, pp. 203-208

Search-based MLP

- Duch W, Grąbczewski K (1999) *Searching for optimal MLP*, 4th Conference on Neural Networks and Their Applications, Zakopane, May 1999, pp. 65-70

SBL prototype based explanations

- Duch W (2000) *Similarity based methods: a general framework for classification, approximation and association*. Control and Cybernetics (in print)
- Grudziński K, Duch W (2000) *SBL-PM: A Simple Algorithm for Selection of Reference Instances for Similarity Based Methods*. Intelligent Information Systems IIS'2000, Physica Verlag (Springer), pp. 99-108

Interactive MDS visualization

- Naud A and Duch W (2000) *Interactive data exploration using MDS mapping*. 5th Conference on Neural Networks and Soft Computing, Zakopane, June 2000, pp. 255-260
- Duch W and Naud A (1996) *Multidimensional scaling and Kohonen's self-organizing*

maps . Proceedings of the 2nd Conf. on Neural Networks and their applications, Orle Gniazdo, 30.IV-4.V.1996, pp. 138-143

- Duch W (1995) *Quantitative measures for the self-organized topographical mapping*. Open Systems and Information Dynamics **2**:295-302

Other issues

- Duch W, Grudziński K and Stawski G (2000) *Symbolic features in neural networks*. 5th Conference on Neural Networks and Soft Computing, Zakopane, June 2000, pp. 180-185

- Duch W, Adamczak R, Hayashi Y (2000) *Eliminators and classifiers* , ICONIP-2000, 7th International Conference on Neural Information Processing (submitted June 2000)

- Duch W and Hayashi Y (2000) *Computational intelligence methods and data understanding*. International Symposium on Computational Intelligence, Kosice - Slovakia, August 2000 (14 p, in print)

- Duch W, Adamczak R, Grąbczewski K, *Neural optimization of linguistic variables and membership functions*. International Conference on Neural Information Processing (ICONIP'99), Perth, Australia, Nov. 1999, Vol. II, pp. 616-621

- Duch W, Adamczak R, Grąbczewski K (1999) *Optimization of logical rules derived by neural procedures*, 1999 International Joint Conference on Neural Networks, Washington, July 1999, paper no. 741 (6 pages)

- Duch W, Korczak J (1999) *Optimization and global minimization methods suitable for neural networks*, Neural Computing Surveys (submitted, in revision)

Applications

- Adamczak R and Duch W (2000) *Neural networks for structure-activity relationship problems*. 5th Conference on Neural Networks and Soft Computing, Zakopane, June 2000, pp. 669-674

- Duch W, Adamczak R, Grąbczewski K, Żal G, Hayashi Y (2000) *Fuzzy and crisp logical rule extraction methods in application to medical data*. In: P.S. Szczepaniak, P.J.G. Lisboa, J. Kacprzyk (eds.), Fuzzy systems in Medicine. Physica - Verlag, Springer 2000, pp. 593-616

- Duch W, Adamczak R, Grąbczewski K (1999) *Neural methods for analysis of psychometric data.*, Proc. of the Intern. Conference EANN'99, Warsaw, 13-15.09.1999, pp. 45-50

- Duch W, Adamczak R, Grąbczewski K, Jankowski N, Żal G (1998) *Medical diagnosis support using neural and machine learning methods*, Proc. of the Intern. Conference EANN'98, Gibraltar, 10-12.06.1998, pp. 292-295

- Duch W, Adamczak R, Grąbczewski K, Żal G (1998) *Hybrid neural-global minimization logical rule extraction method for medical diagnosis support*, Intelligent Information Systems VII, Malbork, Poland, 15-19.06.1998, pp. 85-94

- Duch W, Adamczak R, Grąbczewski K (1997) ***Extraction of logical rules from medical datasets***, 3rd Conf. on Neural Networks and Their Applications, Kule 1997, pp. 707-712