Induction of Rules

Lecturer: JERZY STEFANOWSKI
Institute of Computing Sciences
Poznan University of Technology
Poznan, Poland
Lecture 6 /7
SE Master Course 2008/2009

Źródła

- Wykład częściowo oparty na moim wykładzie szkoleniowym dla COST Action Spring School on Data Mining and MCDA – Troina 2008 oraz wcześniejszych wystąpieniach konferencyjnych.
- Proszę także przeczytać stosowane rozdziały z mojej rozprawy habilitacyjnej – dostępna na mojej stronie WWW.
Outline of this lecture

1. Rule representation
2. Basic algorithms for rule induction – idea of „Sequential covering"
3. MODLEM → exemplary algorithm for inducing a minimal set of rules.
4. Classification strategies
5. Descriptive properties of rules
6. Explore → discovering a richer set of rules
7. Logical relations (ILP) and rule induction
8. Final remarks

Rules - preliminaries

- **Rules** → popular symbolic representation of knowledge derived from data;
  - Natural and easy form of representation → possible inspection by human and their interpretation.
- Standard form of rules
  IF Conditions THEN Class
- Other forms: Class IF Conditions; Conditions → Class

**Example:** The set of decision rules induced from PlaySport:

- if outlook = overcast then Play = yes
- if temperature = mild and humidity = normal then Play = yes
- if outlook = rainy and windy = FALSE then Play = yes
- if humidity = normal and windy = FALSE then Play = yes
- if outlook = sunny and humidity = high then Play = no
- if outlook = rainy and windy = TRUE then Play = no
Polish contribution – prof. Ryszard Michalski

• Father of Machine Learning and rule induction

Ryszard S. Michalski
(1937 - 2007)

ABC Chair Professor of Computational Sciences and Health Informatics
Director of the Center for Discovery Science and Health Informatics
George Mason University

The page has been created since January 1, 2006

More about fathers of machine learning

Rules – more preliminaries

- A set of rules – a disjunctive set of conjunctive rules.
- Also DNF form:
  - \textit{Class} IF \textit{Cond} \_1 \text{ OR } \textit{Cond} \_2 \text{ OR } \ldots \text{ Cond} \_m
- Various types of rules in data mining
  - Decision / classification rules
  - Association rules
  - Logic formulas (ILP)
  - Other $\rightarrow$ action rules, …
- \textbf{MCDA} $\rightarrow$ attributes with some additional preferential information and ordinal classes.

Why Decision Rules?

- Decision rules are more compact.
- Decision rules are more understandable and natural for human.
- Better for descriptive perspective in data mining.
- Can be nicely combined with background knowledge and more advanced operations, …

Example: Let $X \in \{0,1\}$, $Y \in \{0,1\}$, $Z \in \{0,1\}$, $W \in \{0,1\}$. The rules are:
  - if $X=1$ and $Y=1$ then 1
  - if $Z=1$ and $W=1$ then 1
  - Otherwise 0;
Decision rules vs. decision trees:

- Trees – splitting the data space (e.g. C4.5)

  Decision boundaries of decision trees

- Rules – covering parts of the space (AQ, CN2, LEM)

  Decision boundaries of decision rules

Rules – more formal notations

- A rule corresponding to class $K_j$ is represented as

  \[
  \text{if } P \text{ then } Q
  \]

  where $P = w_1$ and $w_2$ and … and $w_m$ is a condition part and $Q$ is a decision part (object $x$ satisfying $P$ is assigned to class $K_j$)

- Elementary condition $w_i \ (a \ rel \ v)$, where $a \in A$ and $v$ is its value (or a set of values) and $rel$ stands for an operator as $=, <, \leq, \geq, >$.

- $[P]$ is a cover of a condition part of a rule $\rightarrow$ a subset of examples satisfying $P$.
  - if $(a2 = \text{small}) \ and \ (a3 \leq 2) \ then \ (d = C1) \ \{x_1, x_7\}$
Rules - properties

- \( B \rightarrow \) a set of examples from \( K_j \).
- A rule if \( P \) then \( Q \) is discriminant in \( DT \) iff \( [P] = \bigcap [w] \subseteq B \),
- otherwise (\( P \cap B \neq \emptyset \)) the rule is partly discriminating
  - Rule accuracy (or confidence) \( ||[P \cap K]|/||[P]| \)
- Rule cannot not have a redundant condition part, i.e. there is no other \( P^* \subset P \) such that \( [P^*] \subseteq B \).

- Rule sets induced from \( DT \)
  - Minimal set of rules
  - Other sets of rules (all rules, satisfactory)

An example of rules induced from data table

<table>
<thead>
<tr>
<th>id.</th>
<th>( a_1 )</th>
<th>( a_2 )</th>
<th>( a_3 )</th>
<th>( a_4 )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>m</td>
<td>s</td>
<td>1</td>
<td>a</td>
<td>C1</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>f</td>
<td>w</td>
<td>1</td>
<td>b</td>
<td>C2</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>m</td>
<td>n</td>
<td>3</td>
<td>c</td>
<td>C1</td>
</tr>
<tr>
<td>( x_4 )</td>
<td>f</td>
<td>n</td>
<td>2</td>
<td>c</td>
<td>C1</td>
</tr>
<tr>
<td>( x_5 )</td>
<td>f</td>
<td>n</td>
<td>2</td>
<td>a</td>
<td>C2</td>
</tr>
<tr>
<td>( x_6 )</td>
<td>m</td>
<td>w</td>
<td>2</td>
<td>c</td>
<td>C2</td>
</tr>
<tr>
<td>( x_7 )</td>
<td>m</td>
<td>s</td>
<td>2</td>
<td>b</td>
<td>C1</td>
</tr>
<tr>
<td>( x_8 )</td>
<td>f</td>
<td>s</td>
<td>3</td>
<td>a</td>
<td>C2</td>
</tr>
</tbody>
</table>

Minimal set of rules
- if \( (a_2 = s) \land (a_3 \leq 2) \) then \( (d = C1) \) \( \{x_1, x_7\} \)
- if \( (a_2 = n) \land (a_4 = c) \) then \( (d = C1) \) \( \{x_3, x_4\} \)
- if \( (a_2 = w) \) then \( (d = C2) \) \( \{x_2, x_6\} \)
- if \( (a_1 = f) \land (a_4 = a) \) then \( (d = C2) \) \( \{x_5, x_8\} \)

Partly discriminating rule:
- if \( (a_1=m) \) then \( (d=C1) \) \( \{x_1, x_3, x_7 \ | \ x_6\} \) 3/4
How to learn decision rules?

- Typical algorithms based on the scheme of a sequential covering and heuristically generate a minimal set of rule covering examples:
  - see, e.g., AQ, CN2, LEM, PRISM, MODLEM, Other ideas – PVM, R1 and RIPPER).
- Other approaches to induce „richer” sets of rules:
  - Satisfying some requirements (Explore, BRUTE, or modification of association rules, „Apriori-like”).
  - Based on local „reducts” → boolean reasoning or LDA.
- Specific optimization, eg. genetic approaches.
- Transformations of other representations:
  - Trees → rules.
  - Construction of (fuzzy) rules from ANN.

Covering algorithms

- A strategy for generating a rule set directly from data:
  - for each class in turn find a rule set that covers all examples in it (excluding examples not in the class).
  - The main procedure is iteratively repeated for each class.
  - Positive examples from this class vs. negative examples.
  - This approach is called a covering approach because at each stage a rule is identified that covers some of the instances.
  - A sequential approach.
  - For a given class it conducts in a stepwise way a general to specific search for the best rules (learn-one-rule) guided by the evaluation measures.
Original covering idea (AQ, Michalski 1969, 86)

for each class $K_i$ do

$E_i := P_i \cup N_i$ (Pi positive, Ni negative example)

RuleSet($K_i$) := empty

repeat (find-set-of-rules)

find-one-rule $R$ covering some positive examples

and no negative ones

add $R$ to RuleSet($K_i$)

delete from $P_i$ all pos. ex. covered by $R$

until $P_i$ (set of pos. ex.) = empty

Find one rule:

Choosing a positive example called a seed.

Find a limited set of rules characterizing
the seed → STAR.

Choose the best rule according to LEF criteria.

Another variant – CN2 algorithm

• Clark and Niblett 1989; Clark and Boswell 1991

• Combine ideas AQ with TDIDT (search as in AQ, additional evaluation
criteria or pruning as for TDIDT).

• AQ depends on a seed example
• Basic AQ has difficulties with noise handling
  • Latter solved by rule truncation (pos-pruning)

• Principles:
  • Covering approach (but stopping criteria relaxed).
  • Learning one rule – not so much example-seed driven.
  • Two options:
    • Generating an unordered set of rules (First Class, then
      conditions).
    • Generating an ordered list of rules (find first the best condition
      part than determine Class).
General schema of inducing minimal set of rules

- The procedure conducts a general to specific (greedy) search for the best rules (learn-one-rule) guided by the evaluation measures.
- At each stage add to the current condition part next elementary tests that optimize possible rule’s evaluation (no backtracking).

Procedure Sequential covering \((K, \text{Class}; A \text{ attributes}; E \text{ examples}, \tau - \text{ acceptance threshold})\):

\[
\begin{align*}
\text{begin} \\
R &:= \emptyset; \quad \{\text{set of induced rules}\} \\
r &:= \text{learn-one-rule}(K, \text{Class}; A \text{ attributes}; E \text{ examples}) \\
\text{while } \text{evaluate}(r, E) > \tau \text{ do} \\
\text{begin} \\
R &:= R \cup r; \\
E &:= E \setminus R; \quad \{\text{remove positive examples covered by } R\} \\
r &:= \text{learn-one-rule}(K, \text{Class}; A \text{ attributes}; E \text{ examples}); \\
\text{end;} \\
\text{return } R \\
\text{end.}
\end{align*}
\]

The contact lenses data

<table>
<thead>
<tr>
<th>Age</th>
<th>Spectacle prescription</th>
<th>Astigmatism</th>
<th>Tear production rate</th>
<th>Recommended lenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Myope</td>
<td>No</td>
<td>Normal</td>
<td>Soft</td>
</tr>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Myope</td>
<td>No</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>No</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Soft</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>
Example: contact lens data 2

• Rule we seek:
  If ?
  then recommendation = hard

• Possible conditions:

  Age = Young 
  Age = Pre-presbyopic 
  Age = Presbyopic 
  Spectacle prescription = Myope 
  Spectacle prescription = Hypermetrope 
  Astigmatism = no 
  Astigmatism = yes 
  Tear production rate = Reduced 
  Tear production rate = Normal 

ACK: slides coming from witten&eibe WEKA

Modified rule and covered data

• Condition part of the rule with the best elementary condition added:
  If astigmatism = yes
  then recommendation = hard

• Examples covered by condition part:

<table>
<thead>
<tr>
<th>Age</th>
<th>Spectacle prescription</th>
<th>Astigmatism</th>
<th>Tear production rate</th>
<th>Recommended lenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Prestbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Prestbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Prestbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Prestbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Prestbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
</tbody>
</table>
Further specialization, 2

- Current state: If astigmatism = yes
  and ?
  then recommendation = hard

- Possible conditions:

  Age = Young  2/4
  Age = Pre-presbyopic  1/4
  Age = Presbyopic  1/4
  Spectacle prescription = Myope  3/6
  Spectacle prescription = Hypermetrope  1/6
  Tear production rate = Reduced  0/6
  Tear production rate = Normal  4/6

Two conditions in the rule

- The rule with the next best condition added:

  If astigmatism = yes
  and tear production rate = normal
  then recommendation = hard

- Examples covered by modified rule:

<table>
<thead>
<tr>
<th>Age</th>
<th>Spectacle prescription</th>
<th>Astigmatism</th>
<th>Tear production rate</th>
<th>Recommended lenses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>
Further refinement, 4

- Current state:

| If astigmatism = yes and tear production rate = normal and ? then recommendation = hard |
|---|---|---|---|---|
| Age = Young | 2/2 |
| Age = Pre-presbyopic | 1/2 |
| Age = Presbyopic | 1/2 |
| Spectacle prescription = Myope | 3/3 |
| Spectacle prescription = Hypermetrope | 1/3 |

- Possible conditions:

- Tie between the first and the fourth test
  - We choose the one with greater coverage

The result

- Final rule:

| If astigmatism = yes and tear production rate = normal and spectacle prescription = myope then recommendation = hard |

- Second rule for recommending “hard lenses”:
  (built from instances not covered by first rule)

| If age = young and astigmatism = yes and tear production rate = normal then recommendation = hard |

- These two rules cover all “hard lenses”:
  - Process is repeated with other two classes

Thanks to witten&eibe
A simple covering algorithm

- Generates a rule by adding tests that maximize rule's accuracy
- Similar to situation in decision trees: problem of selecting an attribute to split on
  - But: decision tree inducer maximizes overall purity
- Each new term reduces rule's coverage:

Evaluation of candidates in Learning One Rule

- When is a candidate for a rule \( R \) treated as “good”? 
  - High accuracy \( P(K|R) \);
  - High coverage \( |[P]| = n \).
- Possible evaluation functions: \( \frac{n_K(R)}{n(R)} \)
  - Relative frequency:
    - where \( n_K \) is the number of correctly classified examples from class \( K \), and \( n \) is the number of examples covered by the rule → problems with small samples;
  - Laplace estimate: \( \frac{n_K(R) + 1}{n(R) + 1} \)
    - Good for uniform prior distribution of \( k \) classes
  - \( m \)-estimate of accuracy: \( \frac{(n_K(R) + mp)(n(R) + m)}{(n(R) + m)} \)
    - where \( n_K \) is the number of correctly classified examples, \( n \) is the number of examples covered by the rule, \( p \) is the prior probability of the class predicted by the rule, and \( m \) is the weight of \( p \) (domain dependent – more noise / larger \( m \)).
Other evaluation functions of rule R and class K

Assume rule R specialized to rule R’

- Entropy (Information gain and others versions).
- Accuracy gain (increase in expected accuracy)
  \[ P(K|R') - P(K|R) \]
- Many others
- Also weighted functions, e.g.

\[
WAG(R',R) = \frac{n_K(R')}{n_K(R)} \cdot (P(K | R') - P(K | R))
\]

\[
WIG(R',R) = \frac{n_K(R')}{n_K(R)} \cdot (\log_2(K | R') - \log_2(K | R))
\]

MODLEM – Algorithm for rule induction

- MODLEM [Stefanowski 98] generates a minimal set of rules.
- Its extra specificity – handling directly numerical attributes during rule induction; elementary conditions, e.g. \((a \geq v), (a < v), (a \in [v_1,v_2])\) or \((a = v)\).
- Elementary condition evaluated by one of three measures: class entropy, Laplace accuracy or Grzymala 2-LEF.

<table>
<thead>
<tr>
<th>obj</th>
<th>a1</th>
<th>a2</th>
<th>a3</th>
<th>a4</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>m</td>
<td>2.0</td>
<td>1</td>
<td>a</td>
<td>C1</td>
</tr>
<tr>
<td></td>
<td>if</td>
<td>(a1 = m) \text{ and } (a2 \leq 2.6) \text{ then } (D = C1) {x1,x3,x7}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>f</td>
<td>2.5</td>
<td>1</td>
<td>b</td>
<td>C2</td>
</tr>
<tr>
<td></td>
<td>if</td>
<td>(a2 \in [1.45, 2.4]) \text{ and } (a3 \leq 2) \text{ then } (D = C1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>m</td>
<td>1.5</td>
<td>3</td>
<td>c</td>
<td>C1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>f</td>
<td>2.3</td>
<td>2</td>
<td>c</td>
<td>C1</td>
</tr>
<tr>
<td></td>
<td>if</td>
<td>(a2 \geq 2.4) \text{ then } (D = C2) {x2,x6}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x5</td>
<td>f</td>
<td>1.4</td>
<td>2</td>
<td>a</td>
<td>C2</td>
</tr>
<tr>
<td></td>
<td>if</td>
<td>(a1 = f) \text{ and } (a2 \leq 2.15) \text{ then } (D = C2) {x5,x8}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x6</td>
<td>m</td>
<td>3.2</td>
<td>2</td>
<td>c</td>
<td>C2</td>
</tr>
<tr>
<td>x7</td>
<td>m</td>
<td>1.9</td>
<td>2</td>
<td>b</td>
<td>C1</td>
</tr>
<tr>
<td>x8</td>
<td>f</td>
<td>2.0</td>
<td>3</td>
<td>a</td>
<td>C2</td>
</tr>
</tbody>
</table>
### Procedure Madlem

**Procedure MODLEM**

*(input $B$ - a set of positive examples from a given decision concept;*  
*criteria - an evaluation measure;*  
*output $T$ - a single rule covering of $B$, treated here as rule-condition parts)*

**begin**

$G := B$; *(a temporary set of rules covered by generated rules)*

$T := \emptyset$; *(a candidate for a rule condition part)*

$S := G$; *(a set of objects currently covered by $T$)*

**while** $G \neq \emptyset$ *(look for rules until some examples remain uncovered)* **do** *(search)*

**begin**

$T := \emptyset$; *(a candidate for a rule condition part)*

$S := G$; *(a set of objects currently covered by $T$)*

**while** $(T \neq \emptyset)$ or $(\text{not}[T] \subseteq B)$ **do** *(stop condition for accepting a rule)*

**end**

for such attribute $q \in C$ do *(looking for the best elementary condition)*

**end**

$new.T := \text{Find_best_condition}(q, S)$;

**if** Better$(new.T, T, \text{criterion})$ **then** $T := new.T$;

**evaluate** if a new condition is better than previous one according to the chosen evaluation measure;

**end**

**end**

**end**

$T := T \cup \{t\}$; *(add the best condition to the candidate rule)*

$S := S \cup \{t\}$ *(from on examples covered by the condition)*

**end**

**end**

**end**

**end**

**end**

**end**

$[\text{boolean}] [T] \cap B$ **then** $T := T \cap \{t\}$ *(test a rule minimality)*

**end**

$[\text{boolean}] [T] \cap B$ **then** $T := T \cap \{t\}$ *(test a rule minimality)*

**end**

**end**

for each elementary condition $t \in T$ **do**

for such $T \in T$ **do**

$[\text{boolean}] [T] \cap B$ **then** $T := T \cap \{t\}$ *(test a rule minimality)*

**end**

**end**

**end**

**end**

**end**

**function** *Find_best_condition*

*(input $c$ - a given attribute; $S$ - a set of examples; output $best.J$ - best condition)*

**begin**

$best.J := \emptyset$;

**if** $c$ is a numerical attribute **then**

**begin**

$H := \text{list of sorted values for attribute $c$ and objects from $S$}$;

*| $H(i)$ - ith unique value in the list |

for $i := 1$ to $\text{length}(H)/2$ **do** *(search)*

**if** object class assignments for $H(i)$ and $H(i + 1)$ are different **then**

**begin**

$v := (H(i) + H(i + 1))/2$;

create a new $J$ as either $[c < v]$ or $[c \geq v]$;


**end**

**end**

**else** *(attribute is nominal)*

**begin**

for each value $v$ of attribute $c$ **do**

**if** Better$(c = v, best.J, \text{criterion})$ **then** $best.J := (c = v)$;

**end**

**end**

**function**.

---

**Set of positive examples**

Looking for the best rule

Testing conjunction

Finding the most discriminatory single condition

Extending the conjunction

Testing minimality

Removing covered examples

---

**Find best condition**

**function Find_best_condition**

*(input $c$ - a given attribute; $S$ - a set of examples; output $best.J$ - best condition)*

**begin**

$best.J := \emptyset$;

**if** $c$ is a numerical attribute **then**

**begin**

$H := \text{list of sorted values for attribute $c$ and objects from $S$}$;

*| $H(i)$ - ith unique value in the list |

for $i := 1$ to $\text{length}(H)/2$ **do** *(search)*

**if** object class assignments for $H(i)$ and $H(i + 1)$ are different **then**

**begin**

$v := (H(i) + H(i + 1))/2$;

create a new $J$ as either $[c < v]$ or $[c \geq v]$;


**end**

**end**

**end**

**end**

**else** *(attribute is nominal)*

**begin**

for each value $v$ of attribute $c$ **do**

**if** Better$(c = v, best.J, \text{criterion})$ **then** $best.J := (c = v)$;

**end**

**end**

**function**.
An Example (1)

Class (Decision = r)

\[ E = \{1, 2, 6, 7, 12, 14, 17\} \]

List of candidates

(Age=m) \{1,6,12,14,17+; 3,8,11,16-\}
(Age=sr) \{2,7+, 5,9,13-\}
(Job=u) \{1,6+; 11-\}
(Job=p) \{2+, 3,4,8,9,13,15,16-\}
(Job=b) \{7,12,14,17+;外科\}
(Pur=K) \{1,17+;外科\}
(Pur=S) \{2+, 13,15-\}
(Pur=W) \{6+, 9-\}
(Pur=M) \{7,14+, 4,8,10,11-\}

An Example (2)

- Numerical attributes: Income

<table>
<thead>
<tr>
<th>Income</th>
<th>500</th>
<th>600</th>
<th>700</th>
<th>800</th>
<th>1000</th>
<th>1100</th>
<th>1400</th>
<th>1500</th>
<th>1600</th>
<th>2300</th>
<th>2500</th>
<th>2600</th>
<th>3700</th>
<th>5000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1+</td>
<td>7+</td>
<td>6+</td>
<td>17+</td>
<td>12+</td>
<td>10-</td>
<td>2+</td>
<td>8-</td>
<td>11-</td>
<td>9-</td>
<td>5-</td>
<td>13-</td>
<td>3-</td>
<td>10-</td>
</tr>
</tbody>
</table>

(Income < 1050) \{1,6,7,12,14,17+;外科\}
(Income < 1250) \{1,6,7,12,14,17+;10-\}
(Income < 1450) \{1,2,6,7,12,14,17+;8,10-\}

Period

(Period < 1) \{1,6,7,14,17+;10,11-\}
(Period < 2.5) \{1,2,6,7,12,14,17+;10,11-\}
Example (3) - the minimal set of induced rule

1. if (Income<1050) then (Dec=r) [6]
2. if (Age=sr) and (Period<2.5) then (Dec=r) [2]
3. if (Period∈[3.5,12.5]) then (Dec=d) [2]
4. if (Age=st) and (Job=p) then (Dec=d) [3]
5. if (Age=m) and (Income∈[1050,2550)) then (Dec=p) [2]
6. if (Job=e) then (Dec=p) [1]
7. if (Age=sr) and (Period≥12.5) then (Dec=p) [2]

For inconsistent data:

- Approximations of decision classes (rough sets)
- Rule post-processing (a kind of post-pruning) or extra testing and earlier acceptance of rules.

Mushroom data (UCI Repository)

- This data set includes descriptions of hypothetical samples corresponding to 23 species of mushrooms in the Agaricus and Lepiota Family. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility.
- Number of examples: 8124.
- Number of attributes: 22 (all nominally valued)
- Missing attribute values: 2480 of them.
- Class Distribution:
  -- edible: 4208 (51.8%)
  -- poisonous: 3916 (48.2%)
MOLDEM rule set (Implemented in WEKA)

=== Classifier model (full training set) ===

Rule 1. (odor is in: {n, a, l})&(spore-print-color is in: {n, k, b, o, u, y, w})&(gill-size = b) => (class = e); [3920, 3920, 93.16%, 100%]
Rule 2. (odor is in: {n, a, l})&(spore-print-color is in: {n, h, k, u}) => (class = e); [3488, 3488, 82.89%, 100%]
Rule 3. (gill-spacing = w)&(cap-color is in: {c, n}) => (class = e); [304, 304, 7.22%, 100%]
Rule 4. (spore-print-color = r) => (class = p); [72, 72, 1.84%, 100%]
Rule 5. (stalk-surface-below-ring = y)&(gill-size = n) => (class = p); [40, 40, 1.02%, 100%]
Rule 6. (odor = n)&(gill-size = n)&(bruises? = t) => (class = p); [8, 8, 0.2%, 100%]
Rule 7. (odor is in: {f, s, y, p, c, m}) => (class = p); [3796, 3796, 96.94%, 100%]

Number of rules: 7
Number of conditions: 14

Approaches to Avoiding Overfitting

- **Pre-pruning**: stop learning the decision rules before they reach the point where they perfectly classify the training data

- **Post-pruning**: allow the decision rules to overfit the training data, and then post-prune the rules.
Pre-Pruning

The criteria for stopping learning rules can be:

• **minimum purity** criterion requires a certain percentage of the instances covered by the rule to be positive;

• **significance test** determines if there is a significant difference between the distribution of the instances covered by the rule and the distribution of the instances in the training sets.

Post-Pruning

1. Split instances into Growing Set and Pruning Set;
2. Learn set SR of rules using Growing Set;
3. Find the best simplification BSR of SR.
4. **while** (Accuracy(BSR, Pruning Set) > Accuracy(SR, Pruning Set) ) **do**
   4.1  SR = BSR;
   4.2  Find the best simplification BSR of SR.
5. return BSR;
Applying rule set to classify objects

- **Matching** a new object description $x$ to condition parts of rules.
  - Either object’s description satisfies all elementary conditions in a rule, or not.

  $\text{IF } (a_1=L) \text{ and } (a_3 \geq 3) \text{ THEN Class +}$
  $x \rightarrow (a_1=L),(a_2=s),(a_3=7),(a_4=1)$

- Two ways of assigning $x$ to class $K$ depending on the set of rules:
  - Unordered set of rules (AQ, CN2, PRISM, LEM)
  - Ordered list of rules (CN2, c4.5rules)

Applying rule set to classify objects

- The rules are ordered into priority decision list!
  Another way of rule induction – rules are learned by first determining Conditions and then Class (CN2)

**Notice**: mixed sequence of classes $K_1, \ldots, K$ in a rule list

**But**: ordered execution when classifying a new instance: rules are sequentially tried and the first rule that ‘fires’ (covers the example) is used for final decision

**Decision list \{R1, R2, R3, \ldots, D\:** rules $R_i$ are interpreted as *if-then-else* rules

If no rule fires, then DefaultClass (majority class in input data)
Priority decision list (C4.5 rules)

Specific list of rules - RIPPER (Mushroom data)
Learning ordered set of rules

- RuleList := empty; E\textsubscript{cur} := E

- repeat
  - learn-one-rule R
  - RuleList := RuleList ++ R
  - E\textsubscript{cur} := E\textsubscript{cur} - \{all examples covered by R\}
    (Not only positive examples !)
  - until performance(R, E\textsubscript{cur}) < Threshold\textsubscript{R}
  - RuleList := sort RuleList by performance(R,E)
  - RuleList := RuleList ++ DefaultRule(E\textsubscript{cur})

CN2 – unordered rule set
Applying unordered rule set to classify objects

- An unordered set of rules → three situations:
  - Matching to rules indicating the same class.
  - Multiple matching to rules from different classes.
  - No matching to any rule.
- An example:
  - e1={(Age=m), (Job=p),(Period=6),(Income=3000),(Purpose=K)}
    - rule 3: if (Period ∈ [3.5,12.5)) then (Dec=d) \[2\]
    - Exact matching to rule 3. → Class (Dec=d)
  - e2={(Age=m), (Job=p),(Period=2),(Income=2600),(Purpose=M)}
    - No matching!

Solving conflict situations

- LERS classification strategy (Grzymala 94)
  - Multiple matching
    - Two factors: \(\text{Strength}(R)\) – number of learning examples correctly classified by \(R\) and final class \(\text{Support}(Y_i)\):
      \[
      \sum_{\text{matching rules } R \text{ for } Y_i} \text{Strength}(R)
      \]
  - Partial matching
    - Matching factor \(\text{MF}(R)\) and
      \[
      \sum_{\text{partially match. rules } R \text{ for } Y_i} \text{MF}(R) \cdot \text{Strength}(R)
      \]
  - e2={(Age=m), (Job=p), (Period=2),(Income=2600),(Purpose=M)}
    - Partial matching to rules 2, 4 and 5 for all with \(\text{MF} = 0.5\)
    - \(\text{Support}(r) = 0.5 \cdot 2 = 1\); \(\text{Support}(d) = 0.5 \cdot 2 + 0.5 \cdot 2 = 2\)
  - Alternative approaches – e.g. nearest rules (Stefanowski 95)
    - Instead of \(\text{MF}\) use a kind of normalized distance \(x\) to conditions of \(r\)
Some experiments

- Analysing strategies (total accuracy in [%]):

<table>
<thead>
<tr>
<th>data set</th>
<th>all</th>
<th>multiple</th>
<th>exact</th>
</tr>
</thead>
<tbody>
<tr>
<td>large soybean</td>
<td>87.9</td>
<td>85.7</td>
<td>79.2</td>
</tr>
<tr>
<td>election</td>
<td>89.4</td>
<td>79.5</td>
<td>71.8</td>
</tr>
<tr>
<td>hsv2</td>
<td>77.1</td>
<td>70.5</td>
<td>59.8</td>
</tr>
<tr>
<td>concretes</td>
<td>88.9</td>
<td>82.8</td>
<td>81.0</td>
</tr>
<tr>
<td>breast cancer</td>
<td>87.1</td>
<td>59.3</td>
<td>51.2</td>
</tr>
<tr>
<td>imidazolium</td>
<td>53.3</td>
<td>44.8</td>
<td>34.4</td>
</tr>
<tr>
<td>lymphography</td>
<td>85.2</td>
<td>73.6</td>
<td>67.6</td>
</tr>
<tr>
<td>oncology</td>
<td>83.8</td>
<td>82.4</td>
<td>74.1</td>
</tr>
<tr>
<td>buses</td>
<td>98.0</td>
<td>93.5</td>
<td>90.8</td>
</tr>
<tr>
<td>bearings</td>
<td>96.4</td>
<td>90.9</td>
<td>87.3</td>
</tr>
</tbody>
</table>

- Comparing to other classification approaches
  - Depends on the data
  - Generally → similar to decision trees

Variations of inducing minimal sets of rules

- Sequential vs. simultaneous covering of data.
- General-to-specific vs. specific-to-general; begin search from single most general vs. many most specific starting hypotheses.
- Generate-and-test vs. example driven (as in AQ).
- Pre-pruning vs. post-pruning of rules
- What evaluation functions to use?
- …
Different perspectives of rule application

- In a descriptive perspective
  - To present, analyse the relationships between values of attributes, to explain and understand classification patterns

- In a prediction/classification perspective,
  - To predict value of decision class for new (unseen) object

Perspectives are different:
Moreover rules are evaluated in a different ways!

Evaluating single rules

- rule \( r \) (if \( P \) then \( Q \)) derived from \( DT \), examples \( U \).

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( Q )</td>
<td>( \neg Q )</td>
</tr>
<tr>
<td>( P )</td>
<td>( n_{PQ} )</td>
<td>( n_{PQ} )</td>
</tr>
<tr>
<td>( \neg P )</td>
<td>( n_{P\neg Q} )</td>
<td>( n_{P\neg Q} )</td>
</tr>
<tr>
<td>( n_Q )</td>
<td>( n_{PQ} )</td>
<td>( n )</td>
</tr>
</tbody>
</table>

- Reviews of measures, e.g.

- Support of rule \( r \)
  \[
  G(P \wedge Q) = \frac{n_{PQ}}{n}
  \]

- Coverage
  \[
  AS(P | Q) = \frac{n_{PQ}}{n_Q}
  \]

- Confidence of rule \( r \)
  \[
  AS(Q | P) = \frac{n_{PQ}}{n_P}
  \]

and others …
Other descriptive measures

Change of support – confirmation of supporting Q by a premise P (Piatetsky-Shapiro)
\[ CS(Q \mid P) = AS(Q \mid P) - G(Q) \]
where \[ G(Q) = \frac{n_Q}{n} \]

Interpretation: Zakres wartości od -1 do +1; Różnica między prawdopodobienstwami a prior i a posterior; dodatnie wartości wystąpienie przesłanki P powoduje konkluzję Q; ujemna wartość wskazuje że nie ma wpływu.

Degree of independence:
\[ IND(Q, P) = \frac{G(P \land Q)}{G(P) \cdot G(Q)} \]

Aggregated measures

Based on previous measures:
Significance of a rule (propozycja Yao i Liu)
\[ S(Q \mid P) = AS(Q \mid P) \cdot IND(Q, P) \]

Klosgen’s measure of interest
\[ K(Q \mid P) = G(P)^\alpha \cdot (AS(Q \mid P) - G(Q)) \]

Michalski’s weighted sum
\[ WSC(Q \mid P) = w_1 \cdot AS(Q \mid P) + w_2 \cdot AS(P \mid Q) \]

The relative risk (Ali, Srikant):
\[ r(Q \mid P) = \frac{AS(Q \mid P)}{AS(Q \mid \neg P)} \]
Descriptive requirements to single rules

- In descriptive perspective users may prefer to discover rules which should be:
  - strong / general – high enough rule coverage $AS(P|Q)$ or support.
  - accurate – sufficient accuracy $AS(Q|P)$.
  - simple (e.g. which are in a limited number and have short condition parts).
  - Number of rules should not be too high.
- Covering algorithms biased towards minimum set of rules - containing only a limited part of potentially 'interesting' rules.
- We need another kind of rule induction algorithms!

Explore algorithm (Stefanowski, Vanderpooten)

- Another aim of rule induction
  - to extract from data set inducing all rules that satisfy some user's requirements connected with his interest (regarding, e.g. the strength of the rule, level of confidence, length, sometimes also emphasis on the syntax of rules).
- Special technique of exploration the space of possible rules:
  - Progressively generation rules of increasing size using in the most efficient way some 'good' pruning and stopping condition that reject unnecessary candidates for rules.
  - Similar to adaptations of Apriori principle for looking frequent itemsets [AIS94]; Brute [Etzioni]
Explore – some algorithmic details

**procedure** Explore (LS: list of conditions; SC: stopping conditions; var R: set_of_rules);

begin
R ← ∅;
Good_Candidates(LS, R); (LS - ordered list of c1,c2,...,cn)
Q ← LS; (create a queue Q)
while Q ≠ ∅ do
begin
select the first conjunction C from Q ;
Q ← Q\{C\};
Extend(C, LC); (LC - list of extended conjunctions)
Good_Candidates(LC, R);
Q ← Q ∪ C; (place all conjunctions from LC at the end of Q)
end
end.

**procedure** Extend(C: conjunction, var L: list of conjunctions);

(This procedure puts in list L extensions of conjunction C that are possible candidates for rules)
begin
Let k be the size of C and h be the highest index of elementary conditions involved in C;
L←{C∧c*h+i where ch+i∈LS and such that all the k-subconjunctions of C ∧c*h+i of size k and involving c*h+i belong to Q ; i=1,...,n-h}
end

**procedure** Good_Candidates(LC: list of conjunctions, var R: set of rules );

(This procedure prunes list LC discarding:
- conjunctions whose extension cannot give rise to rules due to SC,
- conjunctions corresponding to rules which are already stored in R)

---

**Various sets of rules (Stefanowski and Vanderpooten 1994)**

- A minimal set of rules (LEM2):

<table>
<thead>
<tr>
<th>Rule</th>
<th>Conditions</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(q1 = 2) ∧ (q3 = 1) then (d = 1)</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>2</td>
<td>(q1 = 1) then (d = 1)</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>3</td>
<td>(q1 = 2) ∧ (q3 = 2) then (d = 1)</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>4</td>
<td>if (q1 = 3) then (d = 2)</td>
<td>9,10,11,13,14</td>
</tr>
<tr>
<td>5</td>
<td>if (q1 = 3) then (d = 2)</td>
<td>9,10,11,13,14</td>
</tr>
<tr>
<td>6</td>
<td>if (q1 = 2) ∧ (q1 = 1) ∧ (q3 = 1) then (d = 2)</td>
<td>12</td>
</tr>
</tbody>
</table>

- A „satisfactory” set of rules (Explore):

Let us assume that the user’s level of interest to the possible strength of a rule by assigning a value f = 50% in SC.

**Explore** gives the following decision rules:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Conditions</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>if (q1 = 3) then (d = 1)</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>2</td>
<td>if (q1 = 2) ∧ (q1 = 1) then (d = 1)</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>3</td>
<td>if (q1 = 3) then (d = 2)</td>
<td>9,10,11,13,14</td>
</tr>
<tr>
<td>4</td>
<td>if (q1 = 2) then (d = 2)</td>
<td>10,13,14,15</td>
</tr>
</tbody>
</table>
A diagnostic case study

- A fleet of homogeneous 76 buses (AutoSan H9-21) operating in an inter-city and local transportation system.
- The following symptoms characterize these buses:
  - $s_1$ – maximum speed [km/h],
  - $s_2$ – compression pressure [Mpa],
  - $s_3$ – blacking components in exhaust gas [%],
  - $s_4$ – torque [Nm],
  - $s_5$ – summer fuel consumption [l/100km],
  - $s_6$ – winter fuel consumption [l/100km],
  - $s_7$ – oil consumption [l/1000km],
  - $s_8$ – maximum horsepower of the engine [km].

Experts’ classification of busses:
1. Buses with engines in a good technical state – further use (46 buses),
2. Buses with engines in a bad technical state – requiring repair (30 buses).

LEM2 algorithm – (sequential covering)

- A minimal set of discriminating decision rules
  1. if ($s_2$≥2.4 MPa) & ($s_7$<2.1 l/1000km) then (technical state=good) [46]
  2. if ($s_2$<2.4 MPa) then (technical state=bad) [29]
  3. if ($s_7$≥2.1 l/1000km) then (technical state=bad) [24]

- The prediction accuracy (‘leaving-one-out’ reclassification test) is equal to 98.7%.
Another set of rules (EXPLORE)

All decision rules with min. SC1 threshold (rule coverage > 50%):

1. if \((s1 > 85 \text{ km/h})\) then (technical state=good) [34]
2. if \((s8 > 134 \text{ kM})\) then (technical state=good) [26]
3. if \((s2 > 2.4 \text{ MPa}) \& (s3 < 61 \%)\) then (technical state=good) [44]
4. if \((s2 > 2.4 \text{ MPa}) \& (s4 > 444 \text{ Nm})\) then (technical state=good) [44]
5. if \((s2 > 2.4 \text{ MPa}) \& (s7 < 2.1 \text{ /}1000\text{km})\) then (technical state=good) [46]
6. if \((s3 < 61 \%) \& (s4 > 444 \text{ Nm})\) then (technical state=good) [42]
7. if \((s1 < 77 \text{ km/h})\) then (technical state=bad) [25]
8. if \((s2 < 2.4 \text{ MPa})\) then (technical state=bad) [29]
9. if \((s7 < 2.1 \text{ /}1000\text{km})\) then (technical state=bad) [24]
10. if \((s3 < 61 \%) \& (s4 < 444 \text{ Nm})\) then (technical state=bad) [28]
11. if \((s3 < 61 \%) \& (s8 < 120 \text{ kM})\) then (technical state=bad) [27]

The prediction accuracy - 98.7%

Descriptive vs. classification properties (Explore)

<table>
<thead>
<tr>
<th>Rule count</th>
<th>Tuning conditions</th>
<th>Average rule length</th>
<th>Average rule strength</th>
<th>Average rule accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>All rules</td>
<td>1</td>
<td>2.1</td>
<td>4.03</td>
<td>92.97</td>
</tr>
<tr>
<td>5%</td>
<td>2</td>
<td>1.99</td>
<td>13.21</td>
<td>92.67</td>
</tr>
<tr>
<td>10%</td>
<td>22</td>
<td>1.84</td>
<td>17.27</td>
<td>93.9</td>
</tr>
<tr>
<td>15%</td>
<td>28</td>
<td>1.84</td>
<td>16.4</td>
<td>94</td>
</tr>
<tr>
<td>20%</td>
<td>17</td>
<td>1.8</td>
<td>21.5</td>
<td>83.15</td>
</tr>
<tr>
<td>25%</td>
<td>24</td>
<td>1.39</td>
<td>22.35</td>
<td>84.57</td>
</tr>
<tr>
<td>50%</td>
<td>9</td>
<td>1.85</td>
<td>31.81</td>
<td>97.97</td>
</tr>
</tbody>
</table>

The prediction accuracy - 98.7%

- Tuning a proper value of stopping condition SC (rule coverage) leads to sets of rules which are „satisfactory” with respect to a number of rules, average rule length and average rule strength without decreasing too much the classification accuracy.
Preference ordered data

• MCDA vs. traditional classification (ML & Stat):
  • Attributes with preference ordered domains \( \rightarrow \) criteria.
  • Ordinal classes rather than nominal labels.
  • “Semantic correlation” between values of criteria, and classes.
  • For objects \( x, y \) if \( a(x) \preceq a(y) \) then their labels \( \lambda(x) \preceq \lambda(y) \)

• Possible inconsistency

<table>
<thead>
<tr>
<th>Client</th>
<th>Month salary</th>
<th>Account status</th>
<th>Credit risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>9000</td>
<td>high</td>
<td>low</td>
</tr>
<tr>
<td>B</td>
<td>4000</td>
<td>medium</td>
<td>medium</td>
</tr>
<tr>
<td>C</td>
<td>5500</td>
<td>medium</td>
<td>high</td>
</tr>
</tbody>
</table>

• Dominance based rough set approach to handle it
  • Greco S., Matarazzo B., Slowinski R.

Dominance based decision rules

• Induced from rough approximations of unions of classes (upward and downward):

  • **certain** \( D \geq \)-decision rules, supported by objects \( \in C_{\leq}^{T_i} \) without ambiguity:

    \[
    \text{if } q_1(x) \preceq_{q_1} r_{q_1} \text{ and } q_2(x) \preceq_{q_2} r_{q_2} \text{ and } \ldots \text{ and } q_p(x) \preceq_{q_p} r_{q_p} \text{ then } x \in C_{\leq}^{T_i}
    \]

  • **possible** \( D \geq \)-decision rules, supported by objects \( \in C_{\leq}^{T_i} \) and ambiguous ones from its upper approximation:

    \[
    \text{if } q_1(x) \preceq_{q_1} r_{q_1} \text{ and } q_2(x) \preceq_{q_2} r_{q_2} \text{ and } \ldots \text{ and } q_p(x) \preceq_{q_p} r_{q_p}, \text{ then } x \text{ possibly } \in C_{\leq}^{T_i}
    \]

  • **certain** \( D \leq \)-decision rules, supported by objects \( \in C_{\geq}^{T_i} \) without ambiguity:

    \[
    \text{if } q_1(x) \succeq_{q_1} r_{q_1} \text{ and } q_2(x) \succeq_{q_2} r_{q_2} \text{ and } \ldots \text{ and } q_p(x) \succeq_{q_p} r_{q_p}, \text{ then } x \in C_{\geq}^{T_i}
    \]
Algorithms for inducing dominance based rules

- Greco, Slowinski, Stefanowski, Blaszczyński, Dembczyński and others – a number of proposals
- Minimal sets of rules:
  - DOMLEM → adaptation of ideas behind MODLEM.
  - DOMApriori → richer set of rules
- Robust rules → syntax based on an object from data table.
  - All rules → modifications of boolean reasoning
  - Glance → incremental learning.

Software from PUT
Learning First Order Rules

- Is object/attribute table sufficient data representation?
- Some limitations:
  - Representation expressiveness – unable to express relations between objects or object elements.
  - background knowledge sometimes is quite complicated.
- Can learn sets of rules such as
  - \( \text{Parent}(x,y) \rightarrow \text{Ancestor}(x,y) \)
  - \( \text{Parent}(x,z) \) and \( \text{Ancestor}(z,y) \rightarrow \text{Ancestor}(x,y) \)
- Research field of Inductive Logic Programming.

Why ILP? (slide of S.Matwin)

- expressiveness of logic as representation (Quinlan)

- can’t represent this graph as a fixed length vector of attributes
- can’t represent a “transition” rule:
  - A can-reach B if A link C, and C can-reach B
  - without variables
FINITE ELEMENT MESH DESIGN

Given a geometric structure and loadings/boundary conditions
Find an appropriate resolution for a finite element mesh

Examples: ten structures with appropriate meshes (cca. 650 edges)

Background knowledge

- Properties of edges (short, loaded, two-side-fixed, ...)
- Relations between edges (neighbor, opposite, equal)

ILP systems applied: GOLEM, CLAUDIEN

Many interesting rules discovered (according to expert evaluation)

Finite element mesh design (ctd.)

Example structure with an appropriate mesh

Example rules

\[ \text{mesh}(\text{Edge}, 7) \leftarrow \text{usual.length}(\text{Edge}), \]
\[ \text{neighbour.xy}(\text{Edge}, \text{Edge}Y), \text{two.side.fixed}(\text{Edge}Y), \]
\[ \text{neighbour.zx}(\text{Edge}Z, \text{Edge}), \text{not.loaded}(\text{Edge}Z) \]
\[ \text{mesh}(\text{Edge}, N) \leftarrow \text{equal}(\text{Edge}, \text{Edge}2), \text{mesh}(\text{Edge}2, N) \]
Application areas

- Medicine
- Economy, Finance
- Environmental cases
- Engineering
  - Control engineering and robotics
  - Technical diagnostics
  - Signal processing and image analysis
- Information sciences
- Social Sciences
- Molecular Biology
- Chemistry and Pharmacy
- …

Where to find more?

- J.W. Grzymala-Busse, LERS-A system for learning from example-s based on rough sets, In Intelligent Decision Support: Handbook of Applications and Advances of Rough Sets Theory, (Edited by R.Slowinski), pp. 3-18
Where to find more - 2


Any questions, remarks?