Applications of Rough sets in Machine Learning and Data Mining

Part 1: Basic rough set methods for data analysis

Nguyen Hung Son

University of Warsaw, Poland

Milan, 26 July 2016

Outline

Outline

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
2. Building blocks: basic rough set methods
   - Decision rule extraction
   - Discretization
3. Different types of reducts
   - Core, Reductive and Redundant attributes
   - Complexity Results
4. Approximate Boolean Reasoning
5. Exercises

The Need for Approximate Reasoning

Many tasks in data mining can be formulated as an approximate reasoning problem.

Assume that there are

- Two agents A_1 and A_2;

The Need for Approximate Reasoning

Many tasks in data mining can be formulated as an approximate reasoning problem.

Assume that there are

- Two agents A_1 and A_2;
- They are talking about objects from a common universe $\mathcal{U}$;

The Need for Approximate Reasoning

Many tasks in data mining can be formulated as an approximate reasoning problem.

Assume that there are

- Two agents A_1 and A_2;
- They are talking about objects from a common universe $\mathcal{U}$;
- They use different languages $\mathcal{L}_1$ and $\mathcal{L}_2$;

The Need for Approximate Reasoning

Many tasks in data mining can be formulated as an approximate reasoning problem.

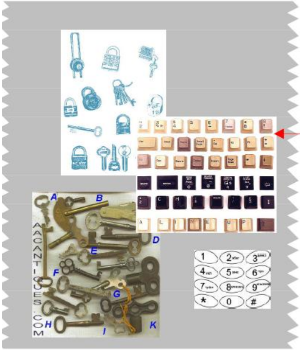
Assume that there are

- Two agents A_1 and A_2;
- They are talking about objects from a common universe $\mathcal{U}$;
- They use different languages $\mathcal{L}_1$ and $\mathcal{L}_2$;
- *Every formula ψ in $\mathcal{L}_1$ (and $\mathcal{L}_2$) describes a set C_ψ of objects from $\mathcal{U}$.*

Each agent, who wants to understand the other, should perform

- an approximation of concepts used by the other;
- an approximation of reasoning scheme, e.g., derivation laws;

Concept approximation problem



An universe of keys

TEACHER

$\mathcal{L}_1 = \{\text{keyboard, ...}\}$

LEARNER

$\mathcal{L}_2 = \{\text{black, brown, white, metal, plastic, ...}\}$

Each agent, who wants to understand the other, should perform

- an approximation of concepts used by the other;
- an approximation of reasoning scheme, e.g., derivation laws;

Classification Problem

Given

- A concept $C \subset \mathcal{U}$ used by teacher;
- A sample $U = U^+ \cup U^-$, where
  - $U^+ \subset C$: positive examples;
  - $U^- \subset \mathcal{U} \setminus C$: negative examples;
- Language $\mathcal{L}_2$ used by learner;

Goal

build an approximation of C in terms of $\mathcal{L}_2$

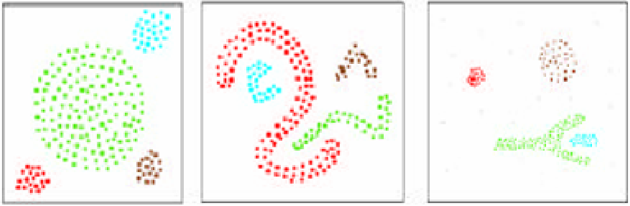
- with simple description;
- with high quality of approximation;
- using efficient algorithm.

Decision table
$\mathbb{S} = (U, A \cup \{dec\})$
describes training data set.

	a_1	a_2	...	dec
u_1	1	0	...	0
u_2	1	1	...	1
...	...	...	...	...
u_n	0	1	...	0

Clustering Problem

- **Original definition:** Division of data into groups of similar objects.



- **In terms of approximate reasoning:** Looking for approximation of a similarity relation (i.e., a concept of being similar):
    - Universe: the set of pairs of objects;
    - Teacher: a partial knowledge about similarity + optimization criteria;
    - Learner: describes the similarity relation using available features;

Association Discovery

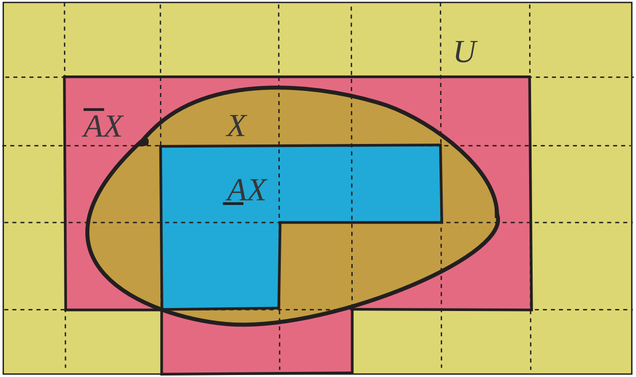
- **Basket data analysis:** looking for approximation of customer behavior in terms of association rules;
  - Universe: the set of transactions;
  - Teacher: hidden behaviors of individual customers;
  - Learner: uses association rules to describe some common trends;

Association Discovery

- **Basket data analysis:** looking for approximation of customer behavior in terms of association rules;
  - Universe: the set of transactions;
  - Teacher: hidden behaviors of individual customers;
  - Learner: uses association rules to describe some common trends;
- **Time series data analysis:**
  - Universe: Sub-sequences obtained by windowing with all possible frame sizes.
  - Teacher: the actual phenomenon behind the collection of timed measurements, e.g., stock market, earth movements.
  - Learner: trends, variations, frequent episodes, extrapolation.

Rough set approach to Concept approximations

- Lower approximation – we are sure that these objects are in the set.
- Upper approximation - it is possible (likely, feasible) that these objects belong to our set (concept). They *roughly* belong to the set.



Generalized definition

Rough approximation of the concept C (induced by a sample X):

any pair $\mathbb{P} = (\mathbf{L}, \mathbf{U})$ satisfying the following conditions:

1. $\mathbf{L} \subseteq \mathbf{U} \subseteq \mathcal{U}$;
2. $\mathbf{L}, \mathbf{U}$ are subsets of $\mathcal{U}$ expressible in the language $\mathcal{L}_2$;
3. $\mathbf{L} \cap X \quad \subseteq \quad C \cap X \quad \subseteq \quad \mathbf{U} \cap X$;
4. $^{(*)}$ the set $\mathbf{L}$ is maximal (and $\mathbf{U}$ is minimal) in the family of sets definable in $\mathcal{L}$ satisfying (3).

Generalized definition

Rough approximation of the concept C (induced by a sample X):

any pair $\mathbb{P} = (\mathbf{L}, \mathbf{U})$ satisfying the following conditions:

1. $\mathbf{L} \subseteq \mathbf{U} \subseteq \mathcal{U}$;
2. $\mathbf{L}, \mathbf{U}$ are subsets of $\mathcal{U}$ expressible in the language $\mathcal{L}_2$;
3. $\mathbf{L} \cap X \quad \subseteq \quad C \cap X \quad \subseteq \quad \mathbf{U} \cap X$;
4. $^{(*)}$ the set $\mathbf{L}$ is maximal (and $\mathbf{U}$ is minimal) in the family of sets definable in $\mathcal{L}$ satisfying (3).

Rough membership function of concept C:

any function $f : \mathcal{U} \to [0, 1]$ such that the pair $(\mathbf{L}_f, \mathbf{U}_f)$, where

- $\mathbf{L}_f = \{x \in \mathcal{U} : f(x) = 1\}$ and
- $\mathbf{U}_f = \{x \in \mathcal{U} : f(x) > 0\}$.

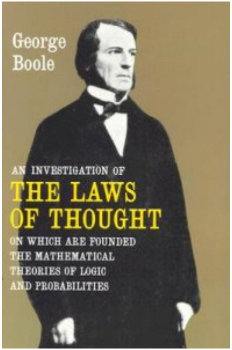
is a rough approximation of C (induced from sample U)

Example of Rough Set models

- **Standard rough sets defined by attributes:**
  - lower and upper approximation of X by attributes from B are defined by indiscernible classes.
- **Tolerance based rough sets:**
  - Using *tolerance* relation (also similarity relation) instead of indiscernibility relation.
- **Variable Precision Rough Sets (VPRS)**
  - allowing some admissible level $0 \leq \beta \leq 1$ of classification inaccuracy.
- **Generalized approximation space**

Outline

Boolean algebra in Computer Science



George Boole (1815-1864)

- George Boole was truly one of the founders of computer science;
- Boolean algebra was an attempt to use algebraic techniques to deal with expressions in the propositional calculus.
- Boolean algebras find many applications in electronic and computer design.
- They were first applied to switching by Claude Shannon in the 20th century.
- Boolean Algebra is also a convenient notation for representing Boolean functions.

Algebraic approach to problem solving

Word Problem:

Madison has a pocket full of nickels and dimes.

- She has 4 more dimes than nickels.
- The total value of the dimes and nickels is $1.15.

How many dimes and nickels does she have?

Algebraic approach to problem solving

Word Problem:

Madison has a pocket full of nickels and dimes.

- She has 4 more dimes than nickels.
- The total value of the dimes and nickels is $1.15.

How many dimes and nickels does she have?

- **Problem modeling:**

$$N = \text{number of nickels}$$
$$D = \text{number of dimes}$$
$$D = N + 4$$
$$10D + 5N = 115$$

Algebraic approach to problem solving

Word Problem:

Madison has a pocket full of nickels and dimes.

- She has 4 more dimes than nickels.
- The total value of the dimes and nickels is $1.15.

How many dimes and nickels does she have?

- **Problem modeling:**

$$N = \text{number of nickels}$$
$$D = \text{number of dimes}$$
$$D = N + 4$$
$$10D + 5N = 115$$

- **Solving algebraic problem:**

$$\ldots \Rightarrow D = 9; N = 5$$

Algebraic approach to problem solving

Word Problem:

Madison has a pocket full of nickels and dimes.

- She has 4 more dimes than nickels.
- The total value of the dimes and nickels is $1.15.

How many dimes and nickels does she have?

- **Problem modeling:**

$$N = \text{number of nickels}$$
$$D = \text{number of dimes}$$
$$D = N + 4$$
$$10D + 5N = 115$$

- **Solving algebraic problem:**

$$\ldots \Rightarrow D = 9; N = 5$$

- **Hura:** 9 dimes and 5 nickels!

Boolean Algebra:

a tuple

$$\mathcal{B} = (B, +, \cdot, 0, 1)$$

satisfying following axioms:

- **Commutative laws:**
  $(a + b) = (b + a)$ and
  $(a \cdot b) = (b \cdot a)$
- **Distributive laws:**
  $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
  $a + (b \cdot c) = (a + b) \cdot (a + c)$
- **Identity elements:**
  $a + 0 = a$ and $a \cdot 1 = a$
- **Complementary:**
  $a + \overline{a} = 1$ and $a \cdot \overline{a} = 0$

$$\mathcal{B}_2 = (\{0, 1\}, +, \cdot, 0, 1)$$

is the smallest, but the most important, model of general Boolean Algebra.

x	y	$x + y$	$x \cdot y$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

x	$\neg x$
0	1
1	0

Applications:

- circuit design;
- propositional calculus;

Boolean Algebra:

a tuple

$$\mathcal{B} = (B, +, \cdot, 0, 1)$$

satisfying following axioms:

- **Commutative laws:**
  $(a + b) = (b + a)$ and
  $(a \cdot b) = (b \cdot a)$
- **Distributive laws:**
  $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
  $a + (b \cdot c) = (a + b) \cdot (a + c)$
- **Identity elements:**
  $a + 0 = a$ and $a \cdot 1 = a$
- **Complementary:**
  $a + \overline{a} = 1$ and $a \cdot \overline{a} = 0$

Binary Boolean algebra

$$\mathcal{B}_2 = (\{0, 1\}, +, \cdot, 0, 1)$$

is the smallest, but the most important, model of general Boolean Algebra.

x	y	$x + y$	$x \cdot y$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

x	$\neg x$
0	1
1	0

Applications:

- circuit design;
- propositional calculus;

Boolean function

- Any function $\boxed{f : \{0,1\}^n \to \{0,1\}}$ is called *a Boolean function*;

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $f : \{0,1\}^n \rightarrow \{0,1\}$ is called *a Boolean function*;
- *An implicant* of function f is a term $t = x_1 ... x_m \overline{y_1} ... \overline{y_k}$ such that

$$\forall_{x_1,...,x_n} t(x_1,...,x_n) = 1 \Rightarrow f(x_1,...,x_n) = 1$$

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $\boxed{f : \{0,1\}^n \to \{0,1\}}$ is called *a Boolean function*;
- *An implicant* of function f is a term $\boxed{t = x_1 ... x_m \overline{y_1} ... \overline{y_k}}$ such that

$$\forall_{x_1,...,x_n} t(x_1, ..., x_n) = 1 \Rightarrow f(x_1, ..., x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $f : \{0,1\}^n \to \{0,1\}$ is called *a Boolean function*;
- *An implicant* of function f is a term $t = x_1 ... x_m \overline{y_1} ... \overline{y_k}$ such that

$$\forall_{x_1,...,x_n} t(x_1,...,x_n) = 1 \Rightarrow f(x_1,...,x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

A Boolean function can be represented by many Boolean formulas;

- $\phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz$

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $\boxed{f : \{0,1\}^n \to \{0,1\}}$ is called *a Boolean function*;
- *An implicant* of function f is a term $\boxed{t = x_1...x_m\overline{y_1}...\overline{y_k}}$ such that

$$\forall_{x_1,...,x_n} t(x_1,...,x_n) = 1 \Rightarrow f(x_1,...,x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

A Boolean function can be represented by many Boolean formulas;

- $\phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz$
- $\phi_2 = (x+y+z)(\overline{x}+y+z)(x+\overline{y}+z)(x+y+\overline{z})$

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $\boxed{f : \{0,1\}^n \to \{0,1\}}$ is called *a Boolean function*;
- *An implicant* of function f is a term $\boxed{t = x_1...x_m\overline{y_1}...\overline{y_k}}$ such that

$$\forall_{x_1,...,x_n} t(x_1,...,x_n) = 1 \Rightarrow f(x_1,...,x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

A Boolean function can be represented by many Boolean formulas;

- $\phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz$
- $\phi_2 = (x+y+z)(\overline{x}+y+z)(x+\overline{y}+z)(x+y+\overline{z})$
- $\phi_3 = xy + xz + yz$

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $\boxed{f : \{0,1\}^n \to \{0,1\}}$ is called *a Boolean function*;
- *An implicant* of function f is a term $\boxed{t = x_1...x_m\overline{y_1}...\overline{y_k}}$ such that

$$\forall_{x_1,...,x_n} t(x_1,...,x_n) = 1 \Rightarrow f(x_1,...,x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

A Boolean function can be represented by many Boolean formulas;

- $\phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz$
- $\phi_2 = (x+y+z)(\overline{x}+y+z)(x+\overline{y}+z)(x+y+\overline{z})$
- $\phi_3 = xy + xz + yz$
- $xy\overline{z}$ is an implicant

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean function

- Any function $f : \{0,1\}^n \to \{0,1\}$ is called *a Boolean function*;
- *An implicant* of function f is a term $t = x_1 ... x_m \overline{y_1} ... \overline{y_k}$ such that

$$\forall_{x_1,...,x_n} t(x_1, ..., x_n) = 1 \Rightarrow f(x_1, ..., x_n) = 1$$

- *Prime implicant:* an implicant that ceases to be so if any of its literal is removed.

A Boolean function can be represented by many Boolean formulas;

- $\phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz$
- $\phi_2 = (x+y+z)(\overline{x}+y+z)(x+\overline{y}+z)(x+y+\overline{z})$
- $\phi_3 = xy + xz + yz$
- $xy\overline{z}$ is an implicant
- xy is a prime implicant

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Boolean Reasoning Approach

Theorem (Blake Canonical Form)

A Boolean function can be represented as a disjunction of all of its prime implicants: $f = t_1 + t_2 + ... + t_k$

Boolean Reasoning Approach

Theorem (Blake Canonical Form)

A Boolean function can be represented as a disjunction of all of its prime implicants: $f = t_1 + t_2 + ... + t_k$

Boolean Reasoning Schema

1. **Modeling:** Represent the problem by a collection of Boolean equations
2. **Reduction:** Condense the equations into a single Boolean equation

$$f = 0 \quad \text{or} \quad f = 1$$

3. **Development:** Construct the Blake Canonical form, i.e., generate the prime implicants of f
4. **Reasoning:** Apply a sequence of reasoning to solve the problem

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$A \to \overline{B} \wedge C \leftrightsquigarrow A(B + \overline{C}) = 0$$

$$\ldots \leftrightsquigarrow BD(AC + \overline{AC}) = 0$$

$$\ldots \leftrightsquigarrow \overline{B}C(A + \overline{D}) = 0$$

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$
\begin{aligned}
A \to \overline{B} \wedge C \leftrightsquigarrow A(B + \overline{C}) &= 0 \\
\ldots \leftrightsquigarrow BD(AC + \overline{AC}) &= 0 \\
\ldots \leftrightsquigarrow \overline{B}C(A + \overline{D}) &= 0
\end{aligned}
$$

- **After reduction:** $f = A(B + \overline{C}) + BD(AC + \overline{AC}) + \overline{B}C(A + \overline{D}) = 0$

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$A \to \overline{B} \wedge C \leadsto A(B + \overline{C}) = 0$$

$$\ldots \leadsto BD(AC + \overline{AC}) = 0$$

$$\ldots \leadsto \overline{B}C(A + \overline{D}) = 0$$

- **After reduction:** $f = A(B + \overline{C}) + BD(AC + \overline{AC}) + \overline{B}C(A + \overline{D}) = 0$
- **Blake Canonical form:** $f = B\overline{C}D + \overline{B}C\overline{D} + A = 0$

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$
\begin{aligned}
A \to \overline{B} \wedge C \leadsto A(B + \overline{C}) &= 0 \\
\ldots \leadsto BD(AC + \overline{AC}) &= 0 \\
\ldots \leadsto \overline{B}C(A + \overline{D}) &= 0
\end{aligned}
$$

- **After reduction:** $f = A(B + \overline{C}) + BD(AC + \overline{AC}) + \overline{B}C(A + \overline{D}) = 0$
- **Blake Canonical form:** $f = B\overline{C}D + \overline{B}C\overline{D} + A = 0$
- **Facts:**

$$
\begin{aligned}
BD &\longrightarrow C \\
C &\longrightarrow B + D \\
A &\longrightarrow 0
\end{aligned}
$$

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$A \to \overline{B} \wedge C \leadsto A(B + \overline{C}) = 0$$
$$\ldots \leadsto BD(AC + \overline{AC}) = 0$$
$$\ldots \leadsto \overline{B}C(A + \overline{D}) = 0$$

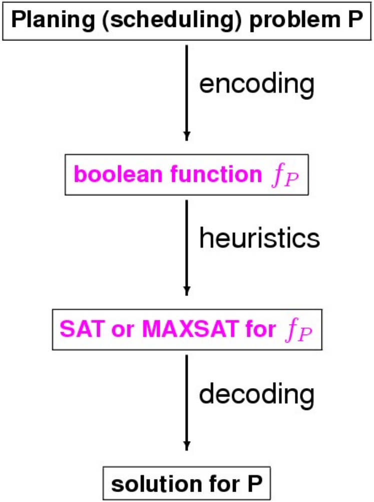
- **After reduction:** $f = A(B + \overline{C}) + BD(AC + \overline{AC}) + \overline{B}C(A + \overline{D}) = 0$
- **Blake Canonical form:** $f = B\overline{C}D + \overline{B}C\overline{D} + A = 0$
- **Facts:**

$$BD \longrightarrow C$$
$$C \longrightarrow B + D$$
$$A \longrightarrow 0$$

- **Reasoning:** (theorem proving) e.g., show that

  "C cannot go alone."

Boolean reasoning for decision problems

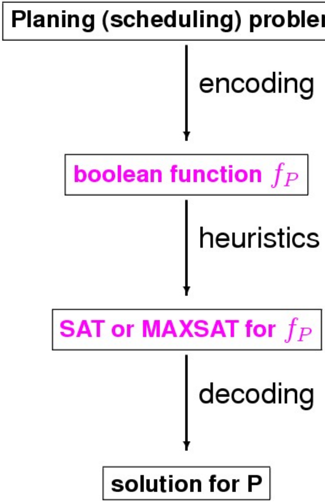


- SAT: whether an equation

$$f(x_1, ..., x_n) = 1$$

has a solution?

Boolean reasoning for decision problems

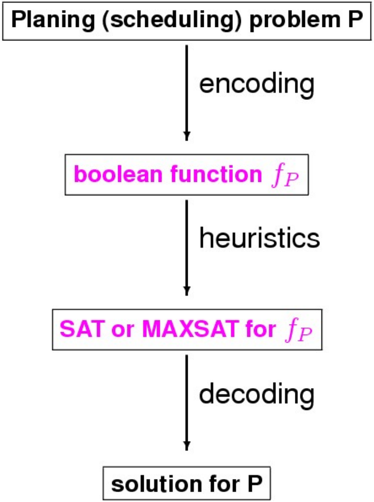


- SAT: whether an equation

$$f(x_1, ..., x_n) = 1$$

  has a solution?
- SAT is the first problem which has been proved to be NP-complete (the Cook's theorem).

Boolean reasoning for decision problems

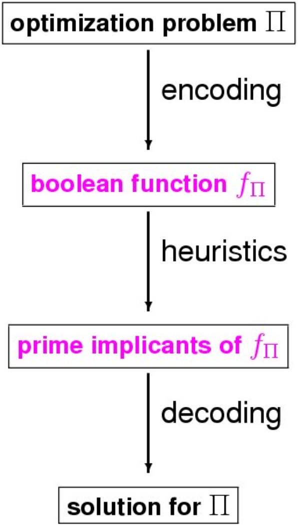


- SAT: whether an equation

$$f(x_1, ..., x_n) = 1$$

  has a solution?
- SAT is the first problem which has been proved to be NP-complete (the Cook's theorem).
- E.g., scheduling problem may be solved by SAT-solver.

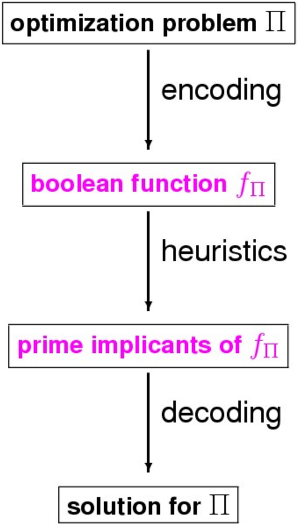
Boolean reasoning for optimization problems



- A function $\phi : \{0,1\}^n \rightarrow \{0,1\}$ is "*monotone*" if

$$\forall_{\mathbf{x},\mathbf{y}\in\{0,1\}^n} (\mathbf{x} \leqslant \mathbf{y}) \Rightarrow (\phi(\mathbf{x}) \leqslant \phi(\mathbf{y}))$$

Boolean reasoning for optimization problems

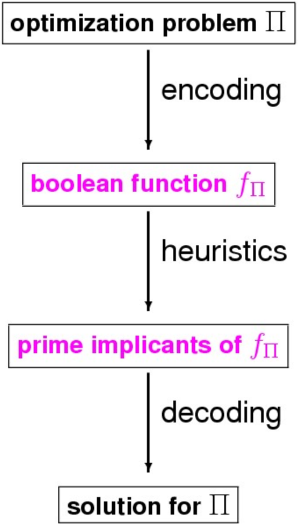


- A function $\phi : \{0,1\}^n \to \{0,1\}$ is "*monotone*" if

$$\forall_{\mathbf{x},\mathbf{y}\in\{0,1\}^n} (\mathbf{x} \leqslant \mathbf{y}) \Rightarrow (\phi(\mathbf{x}) \leqslant \phi(\mathbf{y}))$$

- Monotone functions can be represented by a boolean expression without negations.

Boolean reasoning for optimization problems



- A function $\phi : \{0,1\}^n \to \{0,1\}$ is "*monotone*" if

$$\forall_{\mathbf{x},\mathbf{y} \in \{0,1\}^n} (\mathbf{x} \leqslant \mathbf{y}) \Rightarrow (\phi(\mathbf{x}) \leqslant \phi(\mathbf{y}))$$

- Monotone functions can be represented by a boolean expression without negations.
- **Minimal Prime Implicant Problem:**

input: Monotone Boolean function f of n variables.

output: A prime implicant of f with the minimal length.

is NP-hard.

Heuristics for minimal prime implicants

Example

$f = (x_1 + x_2 + x_3)(x_2 + x_4)(x_1 + x_3 + x_5)(x_1 + x_5)(x_4 + x_6)$

The prime implicant can be treated as a set covering problem.

1. **Greedy algorithm:** In each step, select the variable that most frequently occurs within clauses
2. **Linear programming:** Convert the given function into a system of linear inequations and applying the Integer Linear Programming (ILP) approach to this system.
3. **Evolutionary algorithms:**
   The search space consists of all subsets of variables
   the cost function for a subset X of variables is defined by (1) the number of clauses that are uncovered by X, and (2) the size of X,

Boolean Reasoning Approach to Rough sets

- Reduct calculation;
- Decision rule generation;
- Real value attribute discretization;
- Symbolic value grouping;
- Hyperplanes and new direction creation;

Reduction

- **Do we need all attributes?**
- **Do we need to store the entire data?**
- **Is it possible to avoid a costly test?**

Reducts are subsets of attributes that preserve the same amount of information. They are, however, (NP-)hard to find.

- Efficient and robust heuristics exist for reduct construction task.
- Searching for reducts may be done efficiently with the use of evolutionary computation.
- Overfitting can be avoided by considering several reducts, pruning rules and lessening discernibility constraints.

Data reduction in Rough sets

What is a reduct?

Reducts are minimal subsets of attributes which contain a necessary portion of *information* of the set of all attributes.

- Given an information system $\mathbb{S} = (U, A)$ and a monotone evaluation function

$$\mu_{\mathbb{S}} : \mathcal{P}(A) \longrightarrow \Re^{+}$$

- The set $B \subset A$ is called *μ-reduct*, if
  - $\mu(B) = \mu(A)$,
  - for any proper subset $B' \subset B$ we have $\mu(B') < \mu(B)$;
- The set $B \subset A$ is called *approximated reduct*, if
  - $\mu(B) \geq \mu(A) - \varepsilon$,
  - for any proper subset ...

Example

- Consider the *playing tennis* decision table
- Let us try to predict the decision for last two objects
- RS methodology:
  - Reduct calculation
  - Rule calculation
  - Matching
  - Voting

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

Example: Decision reduct

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants $\Longrightarrow$ reducts

Example: Decision reduct

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants $\Longrightarrow$ reducts

Example: Decision reduct

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

Discernibility matrix;

$\mathbb{M}$	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants $\Longrightarrow$ reducts

Example: Decision reduct

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$
\begin{aligned}
f =&(\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \vee \alpha_2 + \alpha_3 + \alpha_4) \\
&(\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3) \\
&(\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)
\end{aligned}
$$

Example: Decision reduct

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \vee \alpha_2 + \alpha_3 + \alpha_4)$$
$$(\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3)$$
$$(\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)$$

- simplifying the function by *absorbtion law* (i.e. $p \wedge (p + q) \equiv p$):

$$f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3)$$

Example: Decision reduct

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \vee \alpha_2 + \alpha_3 + \alpha_4)$$
$$(\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3)$$
$$(\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)$$

- simplifying the function by *absorbtion law* (i.e. $p \wedge (p + q) \equiv p$):

$$f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3)$$

- Transformation from CNF to DNF: $f = \alpha_1\alpha_4\alpha_2 + \alpha_1\alpha_4\alpha_3$

Example: Decision reduct

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \vee \alpha_2 + \alpha_3 + \alpha_4)$$
$$(\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3)$$
$$(\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)$$

- simplifying the function by *absorbtion law* (i.e. $p \wedge (p+q) \equiv p$):

$$f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3)$$

- Transformation from CNF to DNF: $f = \alpha_1\alpha_4\alpha_2 + \alpha_1\alpha_4\alpha_3$
- Each component corresponds to a reduct: $R_1 = \{a_1, a_2, a_4\}$ and $R_2 = \{a_1, a_3, a_4\}$

Outline

Boolean reasoning approach

- Reducts
- Decision rules
- Discretization
- Feature selection and Feature extraction

Outline

Example: Decision Rule Extraction

$\mathbb{M}$	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$f_{u_3} = (\alpha_1)(\alpha_1 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2) = \alpha_1$$

Decision rule:

$$(a_1 = \text{overcast}) \implies dec = \text{no}$$

Example: Decision Rule Extraction

$\mathbb{M}$	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$\begin{aligned} f_{u_8} &= (\alpha_1 + \alpha_2)(\alpha_1)(\alpha_1 + \alpha_2 + \alpha_3)(\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4)(\alpha_2 + \alpha_3) \\ &\quad (\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)(\alpha_1 + \alpha_4) \\ &= \alpha_1(\alpha_2 + \alpha_3)(\alpha_3 \vee \alpha_4) = \alpha_1\alpha_3 + \alpha_1\alpha_2\alpha_4 \end{aligned}$$

Decision rules:

- $(a_1 = \text{sunny}) \wedge (a_3 = \text{high}) \implies dec = \text{no}$
- $(a_1 = \text{sunny}) \wedge (a_2 = \text{mild}) \wedge (a_4 = FALSE) \implies dec = \text{no}$

Example: all conssistent decision rules

Rid	Condition	⇒Decision	supp.
1	outlook(overcast)⇒	yes	4
2	humidity(normal) AND windy(FALSE)⇒	yes	4
3	outlook(sunny) AND humidity(high)⇒	no	3
4	outlook(rainy) AND windy(FALSE)⇒	yes	3
5	outlook(sunny) AND temp.(hot)⇒	no	2
6	outlook(rainy) AND windy(TRUE)⇒	no	2
7	outlook(sunny) AND humidity(normal)⇒	yes	2
8	temp.(cool) AND windy(FALSE)⇒	yes	2
9	temp.(mild) AND humidity(normal)⇒	yes	2
10	temp.(hot) AND windy(TRUE)⇒	no	1
11	outlook(sunny) AND temp.(mild) AND windy(FALSE)⇒	no	1
12	outlook(sunny) AND temp.(cool)⇒	yes	1
13	outlook(sunny) AND temp.(mild) AND windy(TRUE)⇒	yes	1
14	temp.(hot) AND humidity(normal)⇒	yes	1

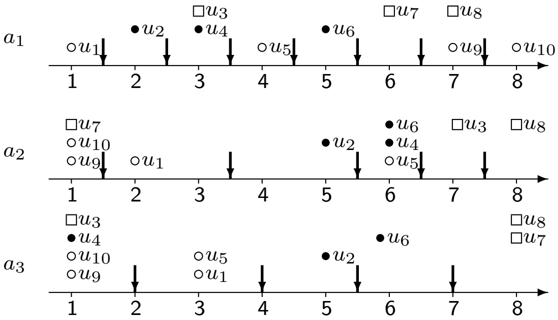
Outline

Discretization problem

Given a decision table $\mathbb{S} = (U, A \cup \{d\})$ where

$$U = \{x_1, \ldots, x_n\};\ A = \{a_1, ..., a_k : U \to \Re\} \text{ and } d : U \to \{1, ..., r(d)\}$$

A	a_1	a_2	a_3	d
u_1	1.0	2.0	3.0	0
u_2	2.0	5.0	5.0	1
u_3	3.0	7.0	1.0	2
u_4	3.0	6.0	1.0	1
u_5	4.0	6.0	3.0	0
u_6	5.0	6.0	5.0	1
u_7	6.0	1.0	8.0	2
u_8	7.0	8.0	8.0	2
u_9	7.0	1.0	1.0	0
u_{10}	8.0	1.0	1.0	0



Discretization problem

- A cut (a, c) on an attribute $a \in A$ discerns a pair of objects $x, y \in U$ if
$$(a(x) - c)(a(y) - c) < 0.$$
- A set of cuts $\mathbf{C}$ is consistent with $\mathbb{S}$ (or $\mathbb{S}$–consistent, for short) if and only if for any pair of objects $x, y \in U$ such that $dec(x) = dec(y)$, the following condition holds:

  IF x, y are discernible by $\mathbb{S}$ **THEN** x, y are discernible by $\mathbf{C}$.
- The consistent set of cuts $\mathbf{C}$ is called *irreducible* iff $\mathbf{Q}$ is not consistent for any proper subset $\mathbf{Q} \subset \mathbf{C}$.
- The consistent set of cuts $\mathbf{C}$ is called it optimal iff $card(C) \leq card(Q)$ for any consistent set of cuts $\mathbf{Q}$.

Discretization problem

OPTIDISC: optimal discretization problem
input: A decision table $\mathbb{S}$.
output: $\mathbb{S}$-optimal set of cuts.

The corresponding decision problem can be formulated as:

DISCSIZE: k-cuts discretization problem
input: A decision table $\mathbb{S}$ and an integer k.
question: Decide whether there exists a $\mathbb{S}$-irreducible set of cuts $\mathbf{P}$ such that $card(\mathbf{P}) < k$.

Theorem

Computational complexity of discretization problems

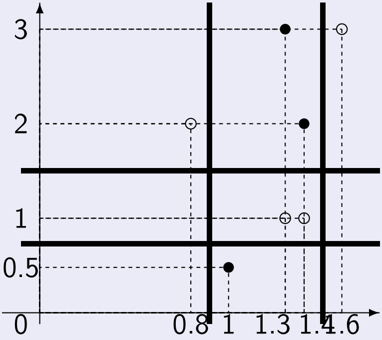
- *The problem* DiscSize *is NP–complete.*
- *The problem* OptiDisc *is NP–hard.*

Boolean reasoning method for discretization

Example of a consistent set of cuts

$\mathbb{S}$	a	b	d
u_1	0.8	2	1
u_2	1	0.5	0
u_3	1.3	3	0
u_4	1.4	1	1
u_5	1.4	2	0
u_6	1.6	3	1
u_7	1.3	1	1

$$\mathbf{C} = \{(a; 0.9), (a; 1.5), (b; 0.75), (b; 1.5)\}$$



The discernibility formulas $\psi_{i,j}$ for different pairs (u_i, u_j) of objects:

$$
\begin{array}{ll}
\psi_{2,1} = p_1^a + p_1^b + p_2^b; & \psi_{2,4} = p_2^a + p_3^a + p_1^b; \\
\psi_{2,6} = p_2^a + p_3^a + p_4^a + p_1^b + p_2^b + p_3^b; & \psi_{2,7} = p_2^a + p_1^b; \\
\psi_{3,1} = p_1^a + p_2^a + p_3^b; & \psi_{3,4} = p_2^a + p_2^b + p_3^b; \\
\psi_{3,6} = p_3^a + p_4^a; & \psi_{3,7} = p_2^b + p_3^b; \\
\psi_{5,1} = p_1^a + p_2^a + p_3^a; & \psi_{5,4} = p_2^b; \\
\psi_{5,6} = p_4^a + p_3^b; & \psi_{5,7} = p_3^a + p_2^b.
\end{array}
$$

The discernibility formula $\Phi_{\mathbb{S}}$ in CNF form is given by

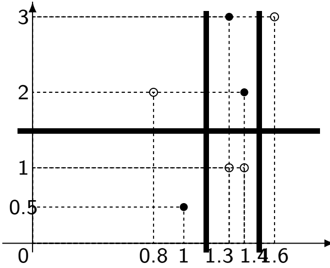
$$
\begin{aligned}
\Phi_{\mathbb{S}} = \quad & \left(p_1^a + p_1^b + p_2^b\right) \left(p_1^a + p_2^a + p_3^b\right) \left(p_1^a + p_2^a + p_3^a\right) \left(p_2^a + p_3^a + p_1^b\right) p_2^b \\
& \left(p_2^a + p_2^b + p_3^b\right) \left(p_2^a + p_3^a + p_4^a + p_1^b + p_2^b + p_3^b\right) \left(p_3^a + p_4^a\right) \left(p_4^a + p_3^b\right) \\
& \left(p_2^a + p_1^b\right) \left(p_2^b + p_3^b\right) \left(p_3^a + p_2^b\right).
\end{aligned}
$$

Transforming the formula $\Phi_{\mathbb{S}}$ into its DNF form we obtain four prime implicants:

$$
\Phi_{\mathbb{S}} = \quad p_2^a p_4^a p_2^b + p_2^a p_3^a p_2^b p_3^b + p_3^a p_1^b p_2^b p_3^b + p_1^a p_4^a p_1^b p_2^b.
$$

Discretization by reduct calculation

$\mathbb{S}^*$	p_1^a	p_2^a	p_3^a	p_4^a	p_1^b	p_2^b	p_3^b	d^*
(u_1, u_2)	1	0	0	0	1	1	0	1
(u_1, u_3)	1	1	0	0	0	0	1	1
(u_1, u_5)	1	1	1	0	0	0	0	1
(u_4, u_2)	0	1	1	0	1	0	0	1
(u_4, u_3)	0	0	1	0	0	1	1	1
(u_4, u_5)	0	0	0	0	0	1	0	1
(u_6, u_2)	0	1	1	1	1	1	1	1
(u_6, u_3)	0	0	1	1	0	0	0	1
(u_6, u_5)	0	0	0	1	0	0	1	1
(u_7, u_2)	0	1	0	0	1	0	0	1
(u_7, u_3)	0	0	0	0	0	1	1	1
(u_7, u_5)	0	0	1	0	0	1	0	1
new	0	0	0	0	0	0	0	0



Outline

Information systems and Decision tables

	Diploma	Experience	French	Reference	Decision
x_1	MBA	Medium	Yes	Excellent	Accept
x_2	MBA	Low	Yes	Neutral	Reject
x_3	MCE	Low	Yes	Good	Reject
x_4	MSc	High	Yes	Neutral	Accept
x_5	MSc	Medium	Yes	Neutral	Reject
x_6	MSc	High	Yes	Excellent	Accept
x_7	MBA	High	No	Good	Accept
x_8	MCE	Low	No	Excellent	Reject

$$\mathbb{D} = (U, A \cup \{d\})$$

Indiscernibility Relation

- For any $B \subset A$:

$$x \ IND(B) \ y \iff inf_B(x) = inf_B(y)$$

  $IND(B$ is a equivalent relation.
- $[u]_B = \{v : u \ IND(B) \ v\}$ – the equivalent class of $IND(B)$.
- $B \subseteq A$ defines a partition of U:

$$U|_B = \{[u]_B : u \in U\}$$

- For any subsets $P, Q \subseteq A$:

$$U|_P = U|_Q \iff \forall_{u \in U} [u]_P = [u]_Q \quad (1)$$

$$U|_P \preceq U|_Q \iff \forall_{u \in U} [u]_P \subseteq [u]_Q \quad (2)$$

- Properties:

$$P \subseteq Q \Longrightarrow U|_P \preceq U|_Q \quad (3)$$

$$\forall_{u \in U} \quad [u]_{P \cup Q} = [u]_P \cap [u]_Q \quad (4)$$

What are reducts?

Reducts are minimal subsets of attributes which contain a necessary portion of *information* of the set of all attributes.

- Given an information system $\mathbb{S} = (U, A)$ and a monotone evaluation function

$$\mu_{\mathbb{S}} : \mathcal{P}(A) \longrightarrow \Re^{+}$$

- The set $B \subset A$ is called μ-*reduct*, if
  - $\mu(B) = \mu(A)$,
  - for any proper subset $B' \subset B$ we have $\mu(B') < \mu(B)$;
- The set $B \subset A$ is called *approximated reduct*, if
  - $\mu(B) \geq \mu(A) - \varepsilon$,
  - for any proper subset ...

Definition (CORE and RED)

$$\mu\text{-RED} = \text{set off all } \mu\text{-reducts}; \quad \mu\text{-CORE} = \bigcap_{B \in \mu\text{-RED}} B$$

Positive Region Based Reducts

- For any $B \subseteq A$ and $X \subseteq U$:

$$\underline{B}(X) = \{u : [u]_B \subseteq X\}; \qquad \overline{B}(X) = \{u : [u]_B \cap X \neq \emptyset\}$$

- Let $\mathbb{S} = (U, A \cup \{dec\})$ be a decision table, let $B \subseteq A$, and let $U|_{dec} = \{X_1, ..., X_k\}$:

$$POS_B(dec) = \bigcup_{i=1}^{k} \underline{B}(X_i)$$

- If $R \subseteq A$ satisfies
  1. $POS_R(dec) = POS_A(dec)$
  2. For any $a \in R : POS_{R-\{a\}}(dec) \neq POS_A(dec)$

  then R is called the *reduct of A based on positive region*.
- $PRED(A)$ = set of reducts based on positive region;
- This is the μ-reduct, where $\mu(B) = |POS_B(dec)|$

Reducts

- Indiscernibility relation

$$(x,y) \in IND(B) \iff \forall_{a\in A} a(x) = a(y)$$
$$(x,y) \in IND_{dec}(B) \iff dec(x) = dec(y) \vee \forall_{a\in A} a(x) = a(y)$$

- A *decision-relative reduct* is a minimal set of attributes $R \subseteq A$ such that $IND_{dec}(R) = IND_{dec}(A)$.
- The set of all reducts is denoted by:

$$\mathcal{RED}(\mathbb{D}) = \{R \subseteq A : R \text{ is a reduct of } \mathbb{D}\}$$

Outline

The importance of attributes

$$\mathcal{RED}(\mathbb{D}) = \{R \subseteq A : R \text{ is a reduct of } \mathbb{D}\}$$

- Core attributes:
$$CORE(\mathbb{D}) = \bigcap_{R \in \mathcal{RED}(\mathbb{D})} R$$
- An attribute $a \in A$ is called **reduct attribute** if it occurs in at least one of reducts
$$REAT(\mathbb{D}) = \bigcup_{R \in \mathcal{RED}(\mathbb{D})} R$$
- The attribute is called *redundant attribute* if it is not a reductive attribute.
- An attribute b is redundant $\Leftrightarrow b \in A - REAT$

The problem setting

It is obvious that for any reduct $R \in \mathcal{RED}(\mathbb{D})$:

$$CORE(\mathbb{D}) \subseteq R \subseteq REAT(\mathbb{D})$$

The problem

For a given a decision table $\mathbb{S} = (U, A \cup \{dec\})$ calculate

$$CORE(\mathbb{D}) = \bigcap_{R \in \mathcal{RED}(\mathbb{D})} R \quad \text{and} \quad REAT(\mathbb{D}) = \bigcup_{R \in \mathcal{RED}(\mathbb{D})} R$$

Example

	a_1	a_2	a_3	a_4	Decision
x_1	MBA	Medium	Yes	Excellent	Accept
x_2	MBA	Low	Yes	Neutral	Reject
x_3	MCE	Low	Yes	Good	Reject
x_4	MSc	High	Yes	Neutral	Accept
x_5	MSc	Medium	Yes	Neutral	Reject
x_6	MSc	High	Yes	Excellent	Accept
x_7	MBA	High	No	Good	Accept
x_8	MCE	Low	No	Excellent	Reject

In this example:

- the set of all reducts $\mathcal{RED}(\mathbb{D}) = \{\{a_1, a_2\}, \{a_2, a_4\}\}$
- Thus

$$CORE(\mathbb{D}) = \{a_2\} \quad REAT(\mathbb{D}) = \{a_1, a_2, a_4\}$$

- the redundant attribute: a_3

Outline

Discernibility matrix

	a_1	a_2	a_3	a_4	Decision
x_1	MBA	Medium	Yes	Excellent	Accept
x_2	MBA	Low	Yes	Neutral	Reject
x_3	MCE	Low	Yes	Good	Reject
x_4	MSc	High	Yes	Neutral	Accept
x_5	MSc	Medium	Yes	Neutral	Reject
x_6	MSc	High	Yes	Excellent	Accept
x_7	MBA	High	No	Good	Accept
x_8	MCE	Low	No	Excellent	Reject

	x_1	x_4	x_6	x_7
x_2	a_2, a_4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4
x_3	a_1, a_2, a_4	a_1, a_2, a_4	a_1, a_2, a_4	a_1, a_2, a_3
x_5	a_1, a_4	a_2	a_2, a_4	a_1, a_2, a_3, a_4
x_8	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1, a_2, a_4

Boolean approach to reduct problem

- Boolean discernibility function:

$$\Delta_{\mathbb{D}}(a_1,...,a_4) = (a_2+a_4)(a_1+a_2)(a_1+a_2+a_4)(a_2+a_3+a_4)$$
$$(a_1+a_2+a_4)(a_1+a_2+a_4)(a_1+a_2+a_4)(a_1+a_2+a_3)$$
$$(a_1+a_4)(a_2)(a_2+a_4)(a_1+a_2+a_3+a_4)(a_1+a_2+a_3)$$
$$(a_1+a_2+a_3+a_4)(a_1+a_2+a_3)(a_1+a_2+a_4)$$

- In general: $R = \{a_{i_1}, ...a_{i_j}\}$ is a reduct in $\mathbb{D} \Leftrightarrow$ the monomial

$$m_R = a_{i_1} \cdot ... \cdot a_{i_j}$$

is a prime implicant of $\Delta_{\mathbb{D}}(a_1,...,a_k)$

Theorem

For any attribute $a \in A$, a is a core attribute if and only if a occurs in discernibility matrix as a singleton. As a consequence, the problem of searching for core attributes can be solved in polynomial time

Simplifying the discernibility function

- Absorption law:

$$x + (x \cdot y) = x \qquad x \cdot (x + y) = x$$

- In our example: irreducible CNF of the discernibility function is as follows:

$$\Delta_{\mathbb{D}}(a_1, ..., a_4) = a_2 \cdot (a_1 + a_4)$$

- Complexity of searching for irreducible CNF: $O(n^4 k)$ steps.

Calculation of reductive attribute

Theorem

For any decision table $\mathbb{D} = (U, A \cup \{d\})$. *If*

$$\Delta_{\mathbb{D}}(a_1, ..., a_k) = \left(\sum_{a \in C_1} a\right) \cdot \left(\sum_{a \in C_2} a\right) \dots \left(\sum_{a \in C_m} a\right)$$

is the irreducible CNF of discernibility function $\Delta_{\mathbb{D}}(a_1, ..., a_k)$, *then*

$$REAT(\mathbb{D}) = \bigcup_{i=1}^{m} C_i \tag{5}$$

Therefore the problem of calculation of all reductive attributes can be solved in $O(n^4 k)$ *steps.*

Outline

Boolean Reasoning Approach to Rough sets

Complexity of encoding functions

Given a decision table with n objects and m attributes

Problem	Nr of variables	Nr of clauses
minimal reduct	$O(m)$	$O(n^2)$
decision rules	$O(n)$ functions	
	$O(m)$	$O(n)$
discretization	$O(mn)$	$O(n^2)$
grouping	$O(\sum_{a \in A} 2^{\lvert V_a \rvert})$	$O(n^2)$
hyperplanes	$O(n^m)$	$O(n^2)$

Greedy algorithm:

time complexity of searching for the best variable:

$$O(\#variables \times \#clauses)$$

Data Mining

The iterative and interactive process of discovering *non-trivial, implicit, previously unknown* and *potentially useful (interesting) information or patterns* from large databases.

W. Frawley and G. Piatetsky-Shapiro and C. Matheus,(1992)

The science of extracting *useful information* from large data sets or databases.

D. Hand, H. Mannila, P. Smyth (2001)

Rough set algorithms based on BR reasoning:

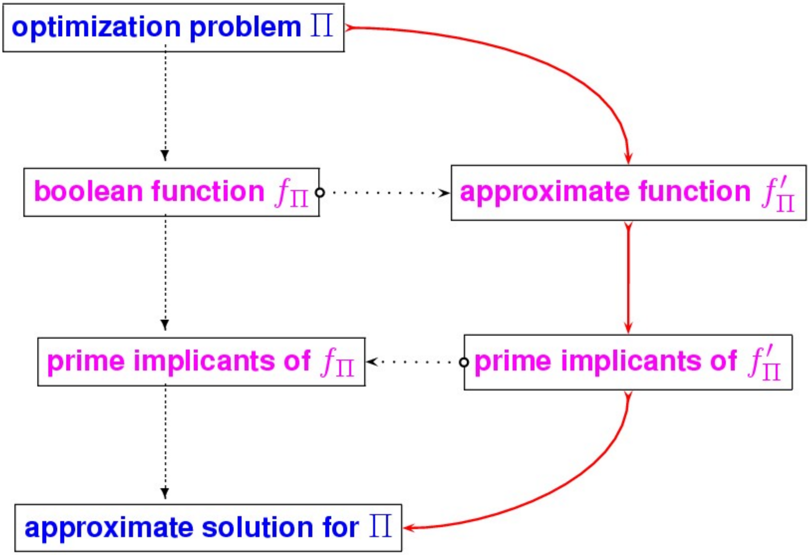
Advantages:

- accuracy: high;
- interpretability: high;
- adjustability: high;
- etc.

Disadvantages:

- Complexity: high;
- Scalability: low;
- Usability of domain knowledge: weak;

Approximate Boolean Reasoning



Example: Decision reduct

$\mathbb{A}$	a_1	a_2	a_3	a_4	*dec*
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

Discernibility matrix;

$\mathbb{M}$	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

The set R is a reduct if (1) it has nonempty intersection with each cell of the discernibility matrix and (2) it is minimal.

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants $\Longrightarrow$ reducts

MD heuristics

- First we have to calculate the number of occurrences of each attributes in the discernibility matrix:

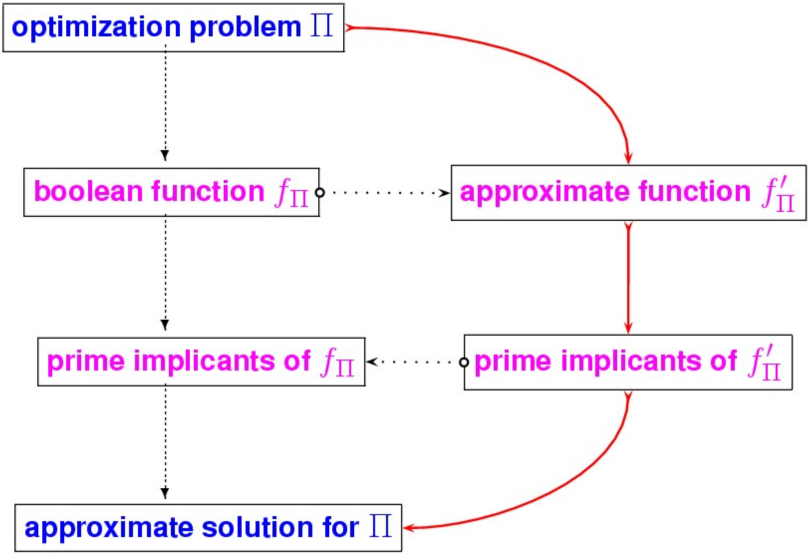
$$eval(a_1) = disc_{dec}(a_1) = 23 \qquad eval(a_2) = disc_{dec}(a_2) = 23$$
$$eval(a_3) = disc_{dec}(a_3) = 18 \qquad eval(a_4) = disc_{dec}(a_4) = 16$$

  Thus a_1 and a_2 are the two most preferred attributes.
- Assume that we select a_1. Now we remove those cells that contain a_1. Only 9 cells remain, and the number of occurrences are:

$$eval(a_2) = disc_{dec}(a_1, a_2) - disc_{dec}(a_1) = 7$$
$$eval(a_3) = disc_{dec}(a_1, a_3) - disc_{dec}(a_1) = 7$$
$$eval(a_4) = disc_{dec}(a_1, a_4) - disc_{dec}(a_1) = 6$$

- If this time we select a_2, then the are only 2 remaining cells, and, both are containing a_4;
- Therefore, the greedy algorithm returns the set $\{a_1, a_2, a_4\}$ as a reduct of sufficiently small size.

Approximate Boolean Reasoning



MD heuristics for reducts without discernibility matrix?

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	?
14	rainy	mild	high	TRUE	?

1. Number of occurences of attibutes in $\mathbb{M}$;
2. Number of occurences of a set of attibutes in $\mathbb{M}$;

- Contingence table for a_1:

a_1	$dec = no$	$dec = yes$	$total$
$sunny$	3	2	5
$overcast$	0	3	3
$rainy$	1	3	4
$total$	4	8	12

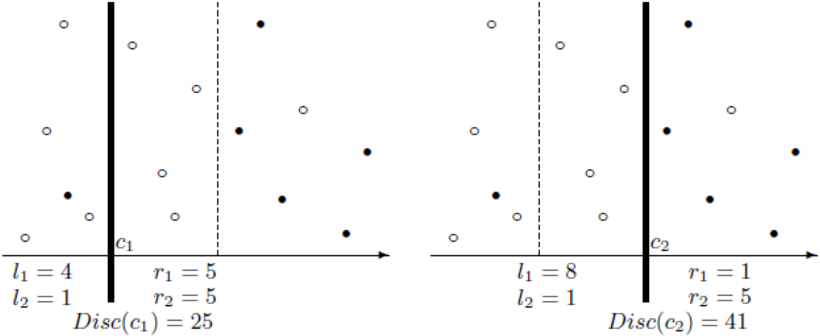
$$disc_{dec}(a_1) = 4 \cdot 8 - 3 \cdot 2 - 0 \cdot 3 - 1 \cdot 3 = 23$$

- Contingence table for $\{a_1, a_2\}$:

(a_1, a_2)	no	yes	$total$
$sunny, hot$	2	0	2
$sunny, mild$	1	1	2
$sunny, cool$	0	1	1
$overcast$	0	3	3
$rainy, mild$	0	2	2
$rainy, cool$	1	1	2
$total$	4	8	12

$$disc_{dec}(a_1, a_2) = 4 \cdot 8 - 2 \cdot 0 - \ldots = 30$$

Discernibility measure for discretization



- number of conflicts in a set of objects X: $conflict(X) = \sum_{i<j} N_i N_j$
- the discernibility of a cut (a, c):

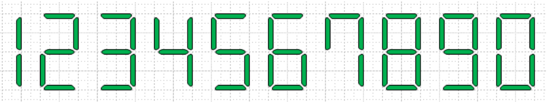
$$W(c) = conflict(U) - conflict(U_L) - conflict(U_R)$$

where $\{U_L, U_R\}$ is a partition of U defined by c.

Outline

Exercise 1: Digital Clock Font

Each digit in Digital Clock is made of a certain number of dashes, as shown in the image below. Each dash is displayed by a LED (light-emitting diode)



Propose a decision table to store the information about those digits and use the rough set methods to solve the following problems:

1. Assume that we want to switch off some LEDs to save the energy, but we still want to recognise the parity of the shown digit based on the remaining dashes. What is the minimal set of dashes you want to display?
2. The same question for the case we want to recognise all digits.

Exercise 2: Core attribute

Propose an algorithm of searching for all core attributes that does not use the discernibility matrix and has time complexity of $O(k \cdot n \log n)$.

Exercise 3: Decision table with maximal number of reducts

We know that the number of reducts for any decision table $\mathbb{S}$ with m attributes can not exceed the upper bound

$$N(m) = \binom{m}{\lfloor m/2 \rfloor}.$$

For any integer m construct a decision table with m attributes such that the number of reducts for this table equals to $N(m)$.

Applications of Rough sets in Machine Learning and Data Mining

Part II: Rough Sets and Machine Learning

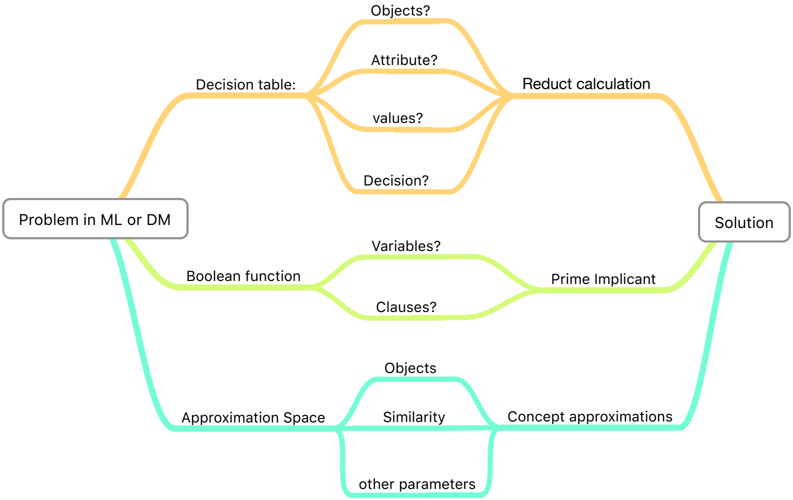
Nguyen Hung Son

University of Warsaw, Poland

Milan, 26 July 2016

Outline

Rough set approach to ML and Data Mining



Outline

1. Rule-base classifiers
   - Rule-based classifier
2. Rough sets and decision tree
   - MD-heuristics and decision tree
3. Concept Approximation with Layered learning
   - General idea
   - Applications
   - Differential Approach to Continuous Decision
4. Exercises

Decision description language

Let A be a set of attributes. The description language for A is a triple

$$\mathcal{L}(A) = (\mathbf{D}, \{\vee, \wedge, \neg\}, \mathbf{F})$$

where

- $\mathbf{D}$ is a called the set of *descriptors*

$$\mathbf{D} = \{(a = v) : a \in A \text{ and } v \in Val_a\}$$

- $\{\vee, \wedge, \neg\}$ is a set of standard Boolean operators
- $\mathbf{F}$ is a set of boolean expressions defined on $\mathbf{D}$ called *formulas*.
- For any $B \subseteq A$ we denote by $\mathbf{D}|_B$ the set of descriptors restricted to B where $\mathbf{D}|_B = \{(a = v) : a \in B \text{ and } v \in Val_a\}$ We also denote by $\mathbf{F}|_B$ the set of formulas build from $\mathbf{D}|_B$.

Semantics of formulas

The semantics

Let $\mathbb{S} = (U, A)$ be an information table describing a sample $U \subset \mathbb{X}$. The semantics of any formula $\phi \in \mathbf{F}$, denoted by $[[\phi]]_{\mathbb{S}}$, is defined by induction as follows:

$$[[(a = v)]]_{\mathbb{S}} = \{x \in U : a(x) = v\} \quad (1)$$

$$[[\phi_1 \vee \phi_2]]_{\mathbb{S}} = [[\phi_1]]_{\mathbb{S}} \cup [[\phi_2]]_{\mathbb{S}} \quad (2)$$

$$[[\phi_1 \wedge \phi_2]]_{\mathbb{S}} = [[\phi_1]]_{\mathbb{S}} \cap [[\phi_2]]_{\mathbb{S}} \quad (3)$$

$$[[\neg\phi]]_{\mathbb{S}} = U \setminus [[\phi]]_{\mathbb{S}} \quad (4)$$

We associate with every formula ϕ the following numeric features:

- $length(\phi)$ = the number of descriptors that occur in ϕ;
- $support(\phi) = |[[\phi]]_{\mathbb{S}}|$ = the number of objects that match the formula;

Decision rules

Definition of Decision Rules

Let $\mathbb{S} = \{U, A \cup \{dec\}\}$ be a decision table. Any implication of a form

$$\phi \Rightarrow \delta$$

where $\phi \in \mathbf{F}_A$ and $\delta \in \mathbf{F}_{dec}$, is called *the decision rule* in $\mathbb{S}$.
The formula ϕ is called *the premise* of the decision rule $\mathbf{r}$ and δ is called *the consequence* of $\mathbf{r}$. We denote the premise and the consequence of the decision rule $\mathbf{r}$ by $prev(\mathbf{r})$ and $cons(\mathbf{r})$, respectively.

Decision rules ...

Generic decision rule

The decision rule $\mathbf{r}$ whose the premise is a boolean monomial of descriptors, i.e.,

$$\mathbf{r} \equiv (a_{i_1} = v_1) \wedge \ldots \wedge (a_{i_m} = v_m) \Rightarrow (dec = k) \tag{5}$$

is called *the generic decision rule*.

We will consider generic decision rules only. For a simplification, we will talk about decision rules keeping in mind the generic ones.

Decision rules ...

Every decision rule $\mathbf{r}$ of the form (5) can be characterized by the following featured:

$$
\begin{aligned}
length(\mathbf{r}) &= \text{the number of descriptor on the assumption of } \mathbf{r} \\
&\quad \text{(i.e. the left hand side of implication)} \\
[\mathbf{r}] &= \text{the carrier of } \mathbf{r}\text{, i.e. the set of objects from } U \\
&\quad \text{satisfying the assumption of } \mathbf{r} \\
support(\mathbf{r}) &= \text{the number of objects satisfying the assumption of} \\
&\quad \mathbf{r}\text{: } support(\mathbf{r}) = card([\mathbf{r}]) \\
confidence(\mathbf{r}) &= \text{the confidence of } \mathbf{r}\text{: } confidence(\mathbf{r}) = \frac{|[\mathbf{r}] \cap DEC_k|}{|[\mathbf{r}]|}
\end{aligned}
$$

The decision rule $\mathbf{r}$ is called *consistent* with $\mathbb{A}$ if

$$confidence(\mathbf{r}) = 1$$

Minimal rules

minimal consistent rules

For a given decision table $\mathbb{S} = (U, A \cup \{dec\})$, the consistent rule:

$$\mathbf{r} = \phi \Rightarrow (dec = k)$$

is called the *minimal consistent decision rule* if any decision rule $\phi' \Rightarrow (dec = k)$ where ϕ' is a shortening of ϕ is not consistent with $\mathbb{S}$.

Outline

General approach

Any rule based classification method consists of three phases :

1. Learning phase: generates a set of decision rules $RULES(\mathbb{A})$ from a given decision table $\mathbb{A}$.
2. Rule selection phase: selects from $RULES(\mathbb{A})$ the set of such rules that can be supported by x. We denote this set by $MatchRules(\mathbb{A}, x)$.
3. Classifying phase: makes a decision for x using some voting algorithm for decision rules from $MatchRules(\mathbb{A}, x)$ with respect to the following cases:
    1. If $MatchRules(\mathbb{A}, x)$ is empty: the decision for x is "$UNKNOWN''$", i.e. we have no idea how to classify x;
    2. If $MatchRules(\mathbb{A}, x)$ consists of decision rules for the same decision class, say k^{th} decision class: in this case $dec(x) = k$;
    3. If $MatchRules(\mathbb{A}, x)$ consists of decision rules for the different decision classes: in this case the decision for x should be made using some voting algorithm for decision rules from $MatchRules(\mathbb{A}, x)$.

Rule filtering

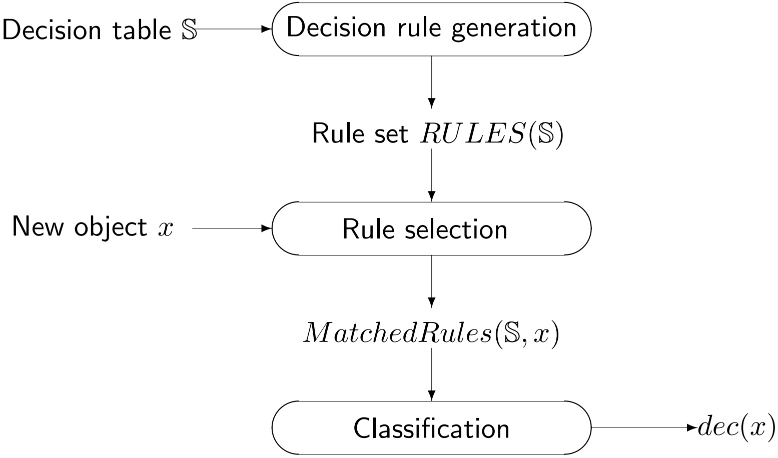
- Every set of rules determines a rough approximation of the given concept via the **conflict solver**;
- The quality of rules is estimated by training data set - a finite sample of the whole universe;
- Conflict solving = elimination of noisy and mistakes caused by "abnormal rules"!
- Not every rule, which is compatible with the training data set, is also compatible with the universe;
- It is better to eliminate abnormal rules according to the domain knowledge;

Filtering approach

supervised methods of filtering:

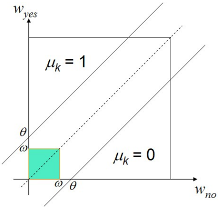
- according to rule support;
- according to the class coverage ratio of rules;
- according to rule length;
- by coverage algorithm: e.g., LEM2 method

Rule based classifier



Standard Rough set approach to rule based classifier

Rid	Condition ⇒	Decision	supp.	match
1	outlook(overcast)⇒	yes	4	0
2	humidity(normal) AND windy(FALSE)⇒	yes	4	0
3	outlook(sunny) AND humidity(high)⇒	no	-3	1
4	outlook(rainy) AND windy(FALSE)⇒	yes	3	0
5	outlook(sunny) AND temp.(hot)⇒	no	-2	1/2
6	outlook(rainy) AND windy(TRUE)⇒	no	-2	1/2
7	outlook(sunny) AND humidity(normal)⇒	yes	2	1/2
8	temp.(cool) AND windy(FALSE)⇒	yes	2	0
9	temp.(mild) AND humidity(normal)⇒	yes	2	1/2
10	temp.(hot) AND windy(TRUE)⇒	no	-1	1/2
11	outlook(sunny) AND temp.(mild) AND windy(FALSE)⇒	no	-1	2/3
12	outlook(sunny) AND temp.(cool)⇒	yes	1	1/2
13	outlook(sunny) AND temp.(mild) AND windy(TRUE)⇒	yes	1	1
14	temp.(hot) AND humidity(normal)⇒	yes	1	0



The testing object $x = \langle sunny, mild, high, TRUE \rangle$

is classified by the decision function:

$$Dec(x) = S\left(\sum_{i=1}^{n} w_i \cdot dec(R_i) \cdot Match(x, R_i)\right)$$

Classifier

Result of a concept approximation method.

It is also called the *classification algorithm* featured by

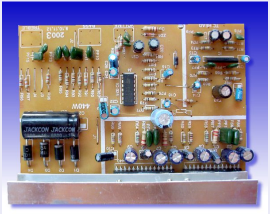
- **Input:** information vector of an object;
- **Output:** whether an object belong to the concept;
- **Parameters:** are necessary for tuning the quality of classifier;

Rough classifier

Outside look: 4 possible answers

- YES (lower approximation)
- POSSIBLY YES (boundary region)
- NO
- DON'T KNOW

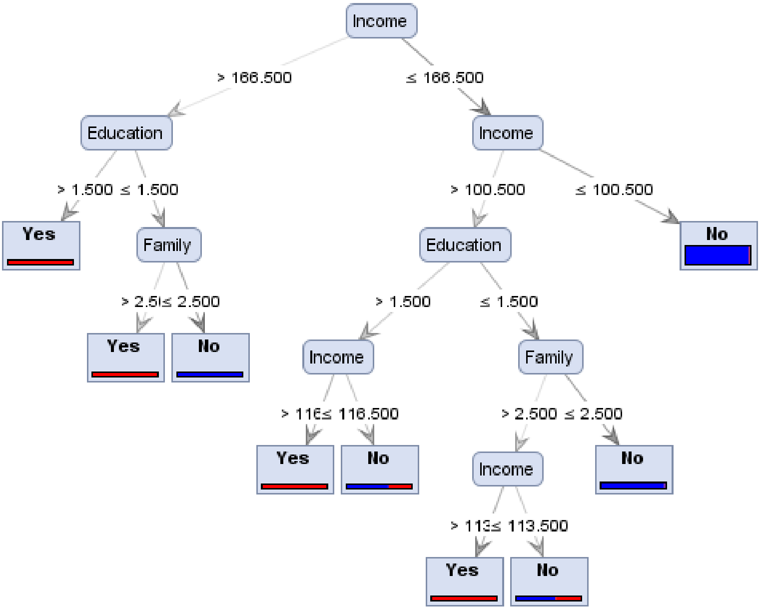
Inside:



- Feature selection/reduction;
- Feature extraction (discretization, value grouping, hyperplanes ...);
- Decision rule extraction;
- Data decomposition;
- Reasoning scheme approximation;

Outline

Outline


Income
> 166.500
≤ 166.500
Education
Income
> 1.500 ≤ 1.500
> 100.500
≤ 100.500
Yes
Family
Education
No
> 2.5 ≤ 2.500
> 1.500
≤ 1.500
Yes
No
Income
Family
> 116 ≤ 116.500
> 2.500 ≤ 2.500
Yes
No
Income
No
> 113 ≤ 113.500
Yes
No

Decision tree

Decision tree is a classification algorithm defined by a nested "IF–THEN–ELSE– of "CASE-SWITCH–" command.

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:
  - a form of pre-prunning technique that can prevent the overfitting problem.

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:
  - a form of pre-prunning technique that can prevent the overfitting problem.
  - Efficient method for soft cut calculation in large data sets.

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:
  - a form of pre-prunning technique that can prevent the overfitting problem.
  - Efficient method for soft cut calculation in large data sets.
- two types of soft trees:

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:
  - a form of pre-prunning technique that can prevent the overfitting problem.
  - Efficient method for soft cut calculation in large data sets.
- two types of soft trees:
  - *Rough decision tree*

Decision tree induction using Discernibility measure

MD-decision tree

- use the discernibility measure to evaluate the tests,
- binary decision using cuts for real value attributes and binary partitions for symbolic value attributes.

Soft decision trees

- advantages:
  - a form of pre-prunning technique that can prevent the overfitting problem.
  - Efficient method for soft cut calculation in large data sets.
- two types of soft trees:
  - *Rough decision tree*
  - *fuzzy decision tree*

Recursive function $build\_tree(U, dec, \mathbf{T})$:

if $(stop\_condition(U, dec) = \mathbf{true})$ **then**
    $\mathbf{T}.etykieta = category(U, dec)$;
    $\mathbf{return}$;
end if
$t := choose\_best\_test(U)$;
$\mathbf{T}.test := t$;
for $v \in R_t$ **do**
    $U_v := \{x \in U : t(x) = v\}$;
    create new trees $\mathbf{T}'$;
    $\mathbf{T}.$branch$(v) = \mathbf{T}'$;
    $build\_tree(U_v, dec, \mathbf{T}')$
end for

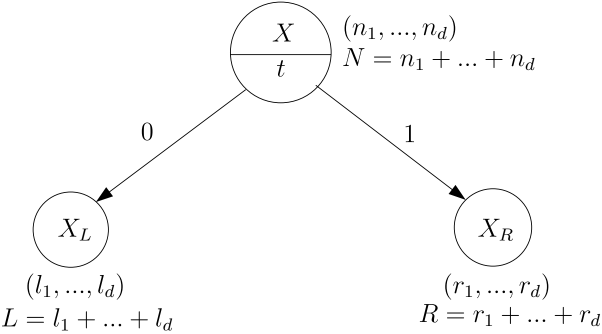


Figure: The partition of the set of objects U defined by a binary test

With those notations the discernibility measure for binary tests can be also computed as follows:

$$\begin{aligned} Disc(t, X) &= conflict(X) - conflict(X_1) - conflict(X_2) \\ &= \frac{1}{2}\sum_{i \neq j} n_i n_j - \frac{1}{2}\sum_{i \neq j} l_i l_j - \frac{1}{2}\sum_{i \neq j} r_i r_j \end{aligned}$$

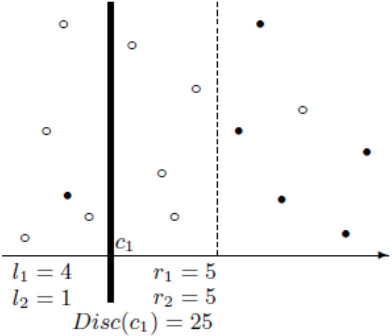
We can show that:

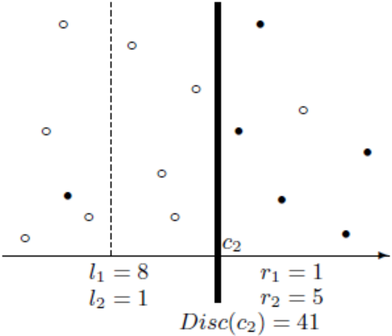
$$
\begin{aligned}
Disc(t,X) &= \frac{1}{2}\left(N^2 - \sum_{i=1}^{d} n_i^2\right) - \frac{1}{2}\left(L^2 - \sum_{i=1}^{d} l_i^2\right) - \frac{1}{2}\left(R^2 - \sum_{i=1}^{d} r_i^2\right) \\
&= \frac{1}{2}\left(N^2 - L^2 - R^2\right) - \frac{1}{2}\sum_{i=1}^{d}(n_i^2 - l_i^2 - r_i^2) \\
&= \frac{1}{2}\left[(L+R)^2 - L^2 - R^2\right] - \frac{1}{2}\sum_{i=1}^{d}[(l_i + r_i)^2 - l_i^2 - r_i^2] \\
&= LR - \sum_{i=1}^{d} l_i r_i
\end{aligned}
$$

Thus

$$
\begin{aligned}
Disc(t,X) = LR - \sum_{i=1}^{d} l_i r_i &= \sum_{i=1}^{d} l_i \sum_{i=1}^{d} r_i - \sum_{i=1}^{d} l_i r_i \\
&= \sum_{i \neq j} l_i r_j \qquad (6)
\end{aligned}
$$

Discernibility measure





- number of conflicts in a set of objects X: $conflict(X) = \sum_{i<j} N_i N_j$
- the discernibility of a cut (a, c):

$$W(c) = conflict(U) - conflict(U_L) - conflict(U_R)$$

where $\{U_L, U_R\}$ is a partition of U defined by c.

Outline

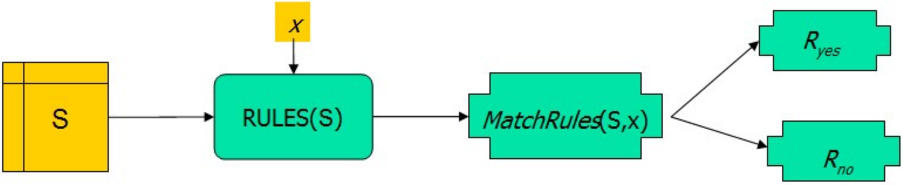
Outline

Hardness of Approximation

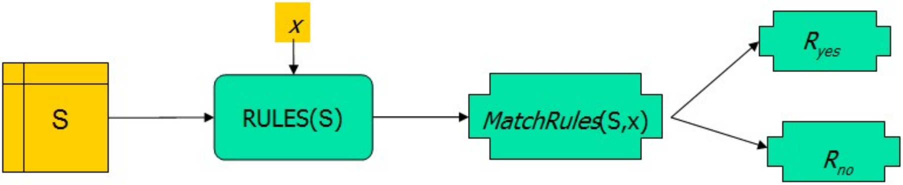
Why the concept approximation problem is hard?

- **Learnability of the target concept:** some concepts are too complex and cannot be approximated directly from feature value vectors.
  - PAC algorithms;
  - Effective learnability of some concept spaces;
  - VC dimension, ...
- **Time and space complexity:** Many problems related to optimal approximation are NP-hard.

Rough Classifier Defined by Rules

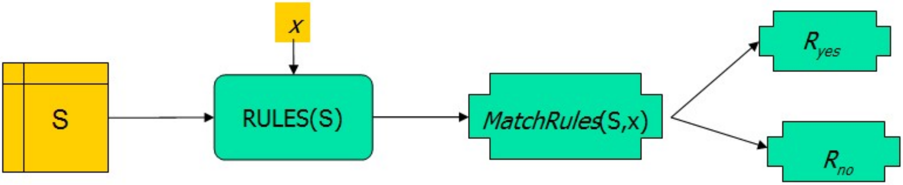


Rough Classifier Defined by Rules



$$w_{yes} = \sum_{\mathbf{r} \in \mathbf{R}_{yes}} strength(\mathbf{r}) \qquad w_{no} = \sum_{\mathbf{r} \in \mathbf{R}_{no}} strength(\mathbf{r})$$

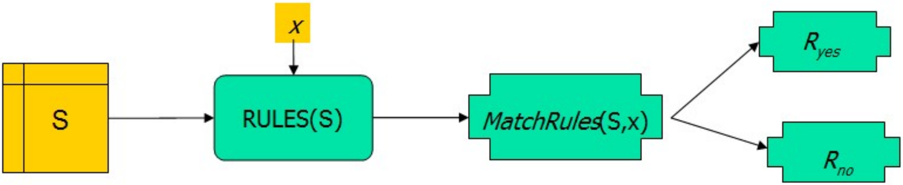
Rough Classifier Defined by Rules



$$w_{yes} = \sum_{\mathbf{r} \in \mathbf{R}_{yes}} strength(\mathbf{r}) \qquad w_{no} = \sum_{\mathbf{r} \in \mathbf{R}_{no}} strength(\mathbf{r})$$

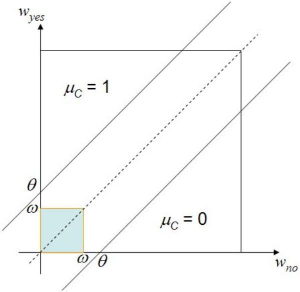
$$\mu_C(x) = \begin{cases} \text{undetermined} & \text{if } \max(w_{yes}, w_{no}) < \omega \\ 0 & \text{if } w_{no} - w_{yes} \geq \theta \text{ and } w_{no} > \omega \\ 1 & \text{if } w_{yes} - w_{no} \geq \theta \text{ and } w_{yes} > \omega \\ \frac{\theta + (w_{yes} - w_{no})}{2\theta} & \text{in other cases} \end{cases}$$

Rough Classifier Defined by Rules

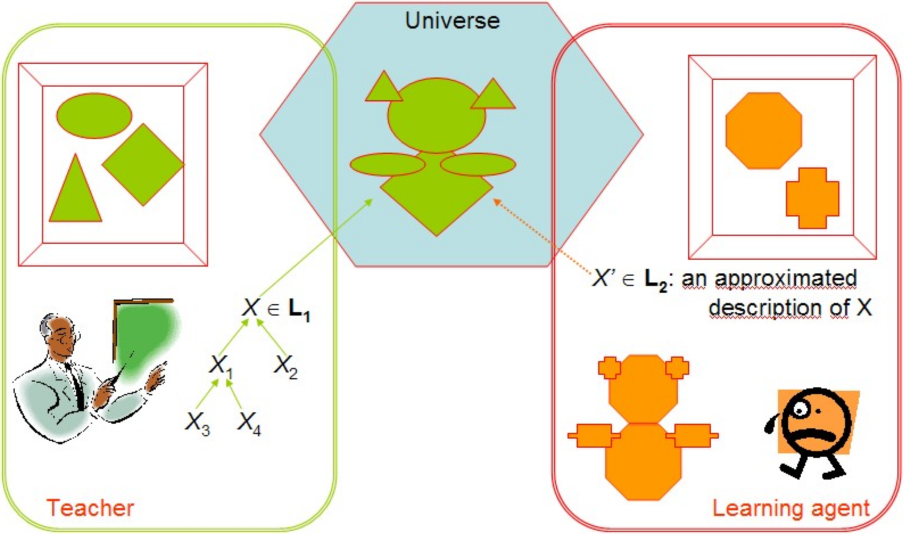


$$w_{yes} = \sum_{\mathbf{r} \in \mathbf{R}_{yes}} strength(\mathbf{r}) \qquad w_{no} = \sum_{\mathbf{r} \in \mathbf{R}_{no}} strength(\mathbf{r})$$

$$\mu_C(x) = \begin{cases} \text{undetermined} & \text{if } \max(w_{yes}, w_{no}) < \omega \\ 0 & \text{if } w_{no} - w_{yes} \geq \theta \text{ and } w_{r} \\ 1 & \text{if } w_{yes} - w_{no} \geq \theta \text{ and } w_{y} \\ \frac{\theta + (w_{yes} - w_{no})}{2\theta} & \text{in other cases} \end{cases}$$



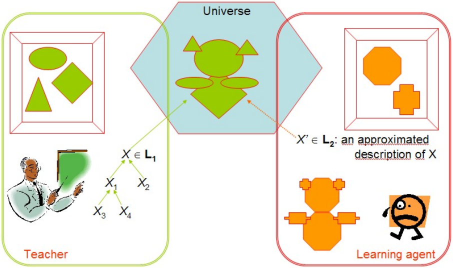
Reasoning via Layered Learning



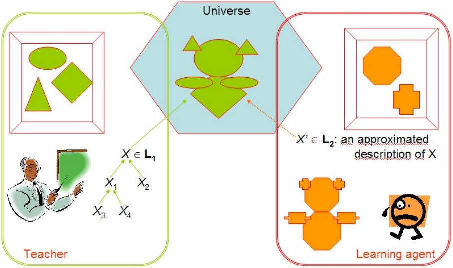
Reasoning via Layered Learning

Given:

- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$



Reasoning via Layered Learning



Given:

- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$

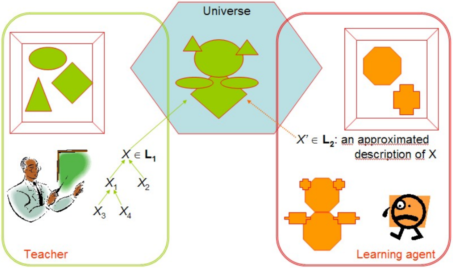
Goal: For each concept C in the hierarchy:

- construct a decision system $\mathbb{S}_C$;
- induce a rough approximation of C, i.e., a rough membership functions for C: $[\mu_{C^+}(x), \mu_{C^-}(x)]$

Reasoning via Layered Learning

Given:

- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$



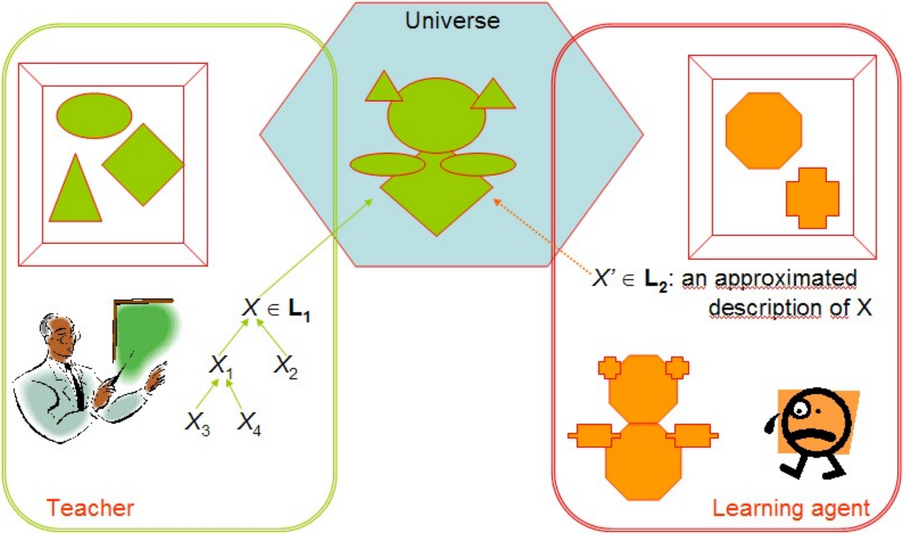
Goal: For each concept C in the hierarchy:

- construct a decision system $\mathbb{S}_C$;
- induce a rough approximation of C, i.e., a rough membership functions for C: $[\mu_{C^+}(x), \mu_{C^-}(x)]$

System control: The system can be tuned by

- uncertainty parameters: θ;
- learning parameters for each level.

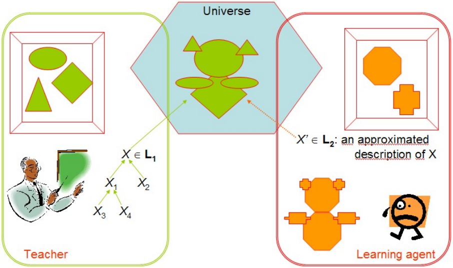
Reasoning via Layered Learning



Reasoning via Layered Learning

Given:

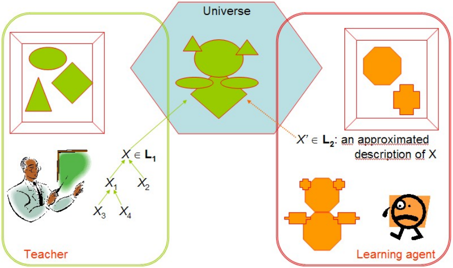
- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$



Reasoning via Layered Learning

Given:

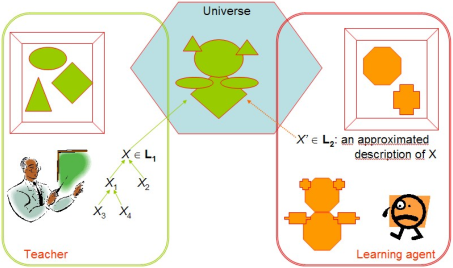
- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$



Goal: For each concept C in the hierarchy:

- construct a decision system $\mathbb{S}_C$;
- induce a rough approximation of C, i.e., a rough membership functions for C: $[\mu_{C^+}(x), \mu_{C^-}(x)]$

Reasoning via Layered Learning



Given:

- U: the set of examples;
- A: the set of attributes;
- H: concept decomposition diagram;
- $D = dec_{C_1}, dec_{C_2}, ...dec_C$

Goal: For each concept C in the hierarchy:

- construct a decision system $\mathbb{S}_C$;
- induce a rough approximation of C, i.e., a rough membership functions for C: $[\mu_{C^+}(x), \mu_{C^-}(x)]$

System control: The system can be tuned by

- uncertainty parameters: θ;
- learning parameters for each level.

Two-layered Approach to Concept Approximation

Typical KDD task:

Searching for patterns from data to describe a concept (sets of objects) or a relation.

Two-layered Approach to Concept Approximation

Typical KDD task:

Searching for patterns from data to describe a concept (sets of objects) or a relation.

Our proposition:

Decompose the concept approximation problem into:

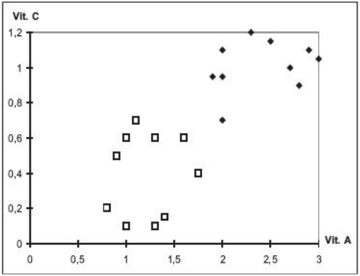
1. Searching for (rough) approximation of the relevant relation:

$$R \longmapsto \widetilde{R} = (\underline{R}, \overline{R})$$

2. inducing the approximation of the target concept using the partial knowledge about the relation R.

Pairwise Space

Vit.A	Vit.C	Fruit	Vit.A	Vit.C	Fruit
1.0	0.6	Apple	2.0	0.7	Pear
1.75	0.4	Apple	2.0	1.1	Pear
1.3	0.1	Apple	1.9	0.95	Pear
0.8	0.2	Apple	2.0	0.95	Pear
1.1	0.7	Apple	2.3	1.2	Pear
1.3	0.6	Apple	2.5	1.15	Pear
0.9	0.5	Apple	2.7	1.0	Pear
1.6	0.6	Apple	2.9	1.1	Pear
1.4	0.15	Apple	2.8	0.9	Pear
1.0	0.1	Apple	3.0	1.05	Pear



Given

Decision table

$$\mathbb{S} = (U, A \cup \{dec\})$$

δ_{a_i} – distance function on a_i

New decision table

- $U \times U$ – pairs of objects;
- $\delta_{a_i}(x, y)$ – continue attributes;

$$d(x, y) = \begin{cases} 0 & dec(x) = dec(y) \\ 1 & otherwise \end{cases}$$

Example of pairwise space

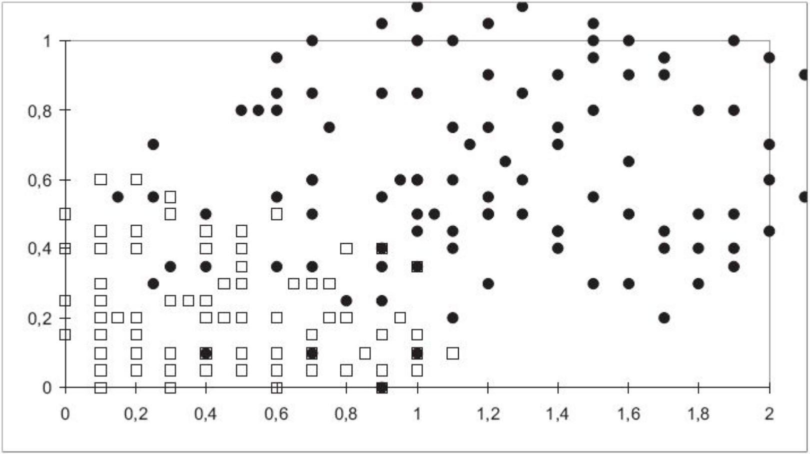
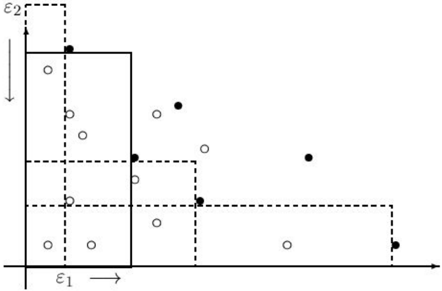
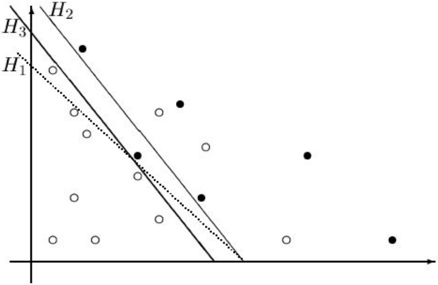


Illustration of some relations in the pairwise space





$\langle x, y\rangle \in \tau_2(\varepsilon_1, ..., \varepsilon_k) \Leftrightarrow$
$\delta_{a_i}(x, y) \le \varepsilon_i$ for any $a_i \in A$.

$\langle x, y\rangle \in \tau_3(w) \Leftrightarrow$
$\delta_{a_1}(x, y) + ... + w_k\delta_{a_k}(x, y) \le w$

Layered learning algorithm

for $l := 0$ to max_level **do**
  for (any concept C_k at the level l in H) **do**
    if $l = 0$ **then**
      $\mathbb{S}_{C_k} := (U, A_k, dec_{C_k});$
    else
      $A_k := \bigcup O_{k_i};$
      $\mathbb{S}_{C_k} := (U, A_k, dec_{C_k});$
    end if
    generate the rule set $RULES(\mathbb{S}_{C_k})$ for decision table $\mathbb{S}_{C_k}$;
    generate the output vector $O_k = \{w_{yes}^{C_k}, w_{no}^{C_k}\}$,
  end for
end for

Example: Nursery data set

- Creator: Vladislav Rajkovic et al. (13 experts)
- Donors: Marko Bohanec (marko.bohanec@ijs.si)
Blaz Zupan (blaz.zupan@ijs.si)
- Date: June, 1997
- Number of Instances: 12960 (instances completely cover the attribute space)
- Number of Attributes: 8

Attributes

NURSERY	not_recom, recommend, very_recom, priority, spec_prior
. EMPLOY	*Employment of parents and child's nursery*
. . parents	usual, pretentious, great_pret
. . has_nurs	proper, less_proper, improper, critical, very_crit
. STRUCT_FINAN	*Family structure and financial standings*
. . STRUCTURE	*Family structure*
. . . form	complete, completed, incomplete, foster
. . . children	1, 2, 3, more
. . housing	convenient, less_conv, critical
. . finance	convenient, inconv
. SOC_HEALTH	*Social and health picture of the family*
. . social	non-prob, slightly_prob, problematic
. . health	recommended, priority, not_recom

Method:

1. Use clustering algorithm to approximate intermediate concepts;
2. Use rule based algorithm (RSES system) to approximate the target concept;

Method:

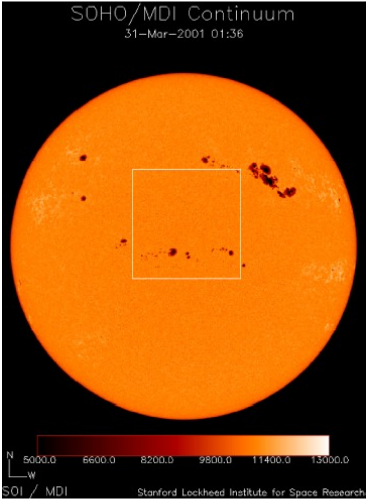
1. Use clustering algorithm to approximate intermediate concepts;
2. Use rule based algorithm (RSES system) to approximate the target concept;

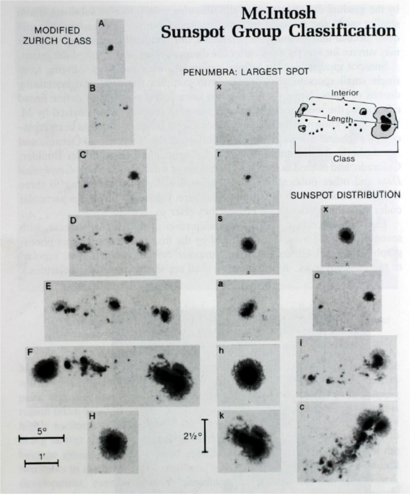
Results: (60% – training, 40% – testing)

	original attributes only	using intermediate concepts
Accuracy	83.4	99.9%
Coverage	85.3%	100%
Nr of rules	634	42 (for the target concept) 92 (for intermediate concepts)

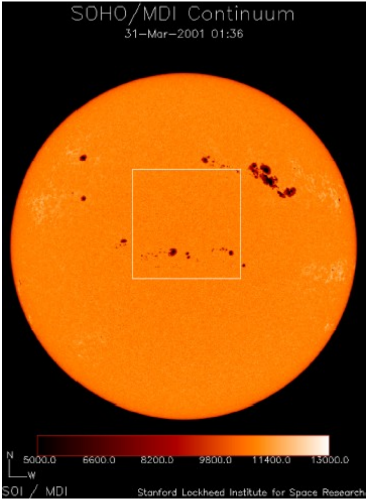
Outline

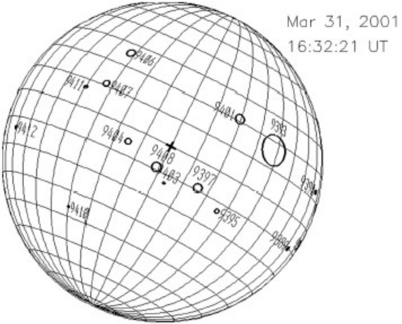
Sunspots Recognition and Classification





Sunspots Recognition and Classification

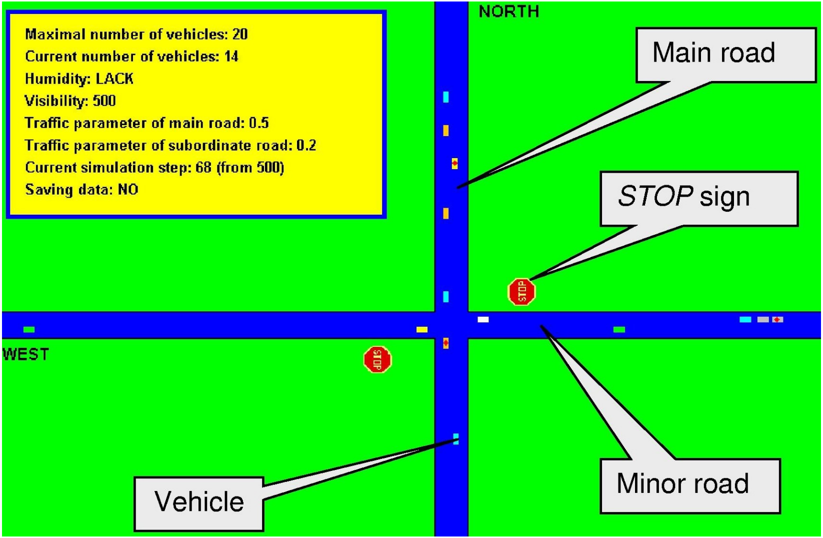




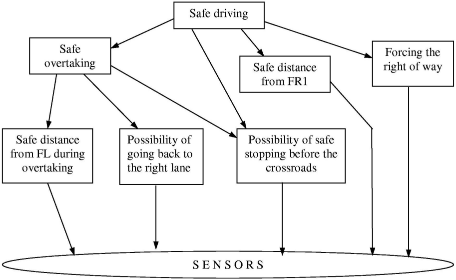
Joint USAF/NOAA Solar Region Summary (MAR 30, 2001 24:00:00 UT)

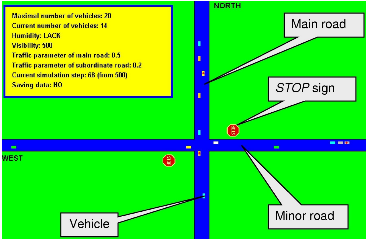
NMBR	LOCATI	LO	AREA	Mcl	LL	NN	MAG TYPE
9387	N08W92	216	0060	Hax	02	01	Alpha
9389	S10W63	187	0050	Bxo	08	08	Beta
9390	N13W65	189	0050	Hax	02	01	Alpha
9393	N17W30	154	2240	Fkc	19	63	Beta-Gamma-Delta
9395	S13W17	141	0050	Hax	02	02	Alpha
9396	S06W65	209	0140	Dao	10	05	Beta
9397	S09W06	130	0180	Eao	15	23	Beta-Gamma
9401	N21W11	135	0230	Eki	13	37	Beta-Gamma
9403	S13E06	118	0010	Bxo	05	02	Beta
9404	S05E23	101	0080	Cao	04	07	Beta
9406	N26E41	083	0170	Hax	03	01	Alpha

Road Situation Simulator

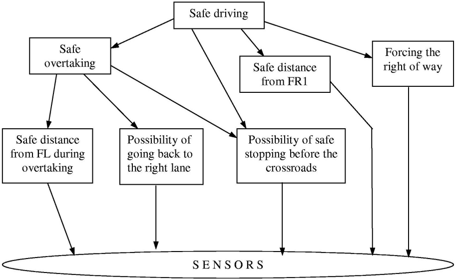


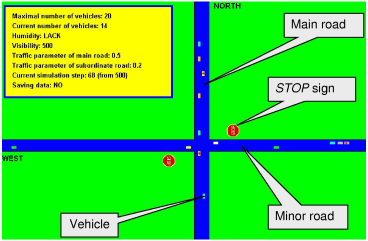
Road Situation Simulator



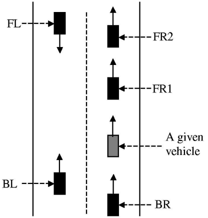


Road Situation Simulator





- **Universe** = set of vectors $s(c, t)$, where
  - c is a car;
  - t is a time instant;
- **Concept** = "Dangerous situation on the road";
- **Evaluation measures:**
  - True positive rate;
  - Coverage rate;
  - Computation time;
  - Rule sizes;



Outline

Ill-defined data

- The proteochemometrics can be seen as the search for possible combinations of ligand-receptor sites with optimal binding strength.
- The ability of the binding affinity prediction is crucial in this task
- the experimental method is very expensive both in terms of time and monetary value.
- This is the reason why data sets in this domain have small sizes.

Differential Calculus to Function Approximation

- **ill-defined data:** limited number of objects and large number of attributes;
- prediction of a **real decision variable** based on nominal attributes;
- the need for the knowledge about the **real mechanisms behind the data**;

No.	Combination	B-1	1-4	4-6	6-E	PB	PE	Binding affinity
1	A2B2C2D2a2b2	1	1	1	1	1	1	4.52526247
2	A1B2C1D1a2b2	-1	1	-1	-1	1	1	4.818066119
3	A1B2C2D1a2b2	-1	1	1	-1	1	1	5.036009902
...	...	...	...	...	...	...	...	
...	...	...	...	...	...	...	...	
39	A1B1C1D1a1b1	-1	-1	-1	-1	-1	-1	8.963821581
40	A1B1C1D1a2b1	-1	-1	-1	-1	1	-1	8.998482244

Existing solutions:

data and sizes	possible comb.	dec. domain
data set A : 40 × 6	64	$(0, 10)$
data set B : 60 × 8	384	$(0, 10)$
data set C : 130 × 55	$2^{41}3^{11}4^{2}6$	$(0, 10)$

- Regression tree, linear regression: ?
- Discretization of decision attribute: ?

Our propositions:

- 2-layered learning idea and decision rule techniques.
- we decompose this learning task into several subtasks:
  1. **Approximate the preference relation between objects;**
  2. **Use approximate preference relation to solve other subtasks:**
     - learning ranking order,
     - prediction of continuous decision value, or
     - searching for optimal combination.
- ...

Two-layer method

Input

1. A decision table

$\mathbb{S}$	a_1	a_2	...	dec
u_1	1	-1	...	4.23
u_2	1	1	...	4.31
...	...	...	...	...
u_n	-1	1	...	8.92

2. Domain knowledge

Two-layer method

Input

1. A decision table

$\mathbb{S}$	a_1	a_2	...	dec
u_1	1	-1	...	4.23
u_2	1	1	...	4.31
...	...	...	...	...
u_n	-1	1	...	8.92

2. Domain knowledge

First level

- Create comparing table

	Δ_{a_1}	Δ_{a_2}	...	change
u_1, u_2	$1 \to 1$	$-1 \to 1$	...	$\nearrow$
u_1, u_3	...	...	...	$\searrow$
...	...	...	...	...

- Learn the preference relation, i.e., decision rules of form

$$\Delta_{a_2} : -1 \to 1 \wedge a_6 = 1 ... \implies change = \searrow$$

Two-layer method

Input

1. A decision table

$\mathbb{S}$	a_1	a_2	...	dec
u_1	1	-1	...	4.23
u_2	1	1	...	4.31
...	...	...	...	...
u_n	-1	1	...	8.92

2. Domain knowledge

First level

- Create comparing table

	Δ_{a_1}	Δ_{a_2}	...	change
u_1, u_2	$1 \to 1$	$-1 \to 1$	...	$\nearrow$
u_1, u_3	...	...	...	$\searrow$
...	...	...	...	...

- Learn the preference relation, i.e., decision rules of form

$$\Delta_{a_2} : -1 \to 1 \wedge a_6 = 1 ... \implies change = \searrow$$

Second level

- Ranking prediction;
- Decision value prediction;
- Experiment design, action rules;

Mathematical analogy

Real function analysis

Searching for maximum of a real function $f : \mathbb{R}^k \to \mathbb{R}$

1. Get some information about its differential, e.g., gradient

$$\nabla f = \left\langle \frac{df}{dx_1}, \ldots, \frac{df}{dx_k} \right\rangle$$

2. Discover the properties of $f(\mathbf{x}_0)$ from its differential, e.g.,
   $\nabla f(\mathbf{x}_0)$ is the direction which promises maximum increase of f

Rough differential calculus

Mathematical analogy

Real function analysis

Searching for maximum of a real function $f : \mathbb{R}^k \to \mathbb{R}$

1. Get some information about its differential, e.g., gradient

$$\nabla f = \left\langle \frac{df}{dx_1}, \ldots, \frac{df}{dx_k} \right\rangle$$

2. Discover the properties of $f(\mathbf{x}_0)$ from its differential, e.g.,
$\nabla f(\mathbf{x}_0)$ is the direction which promises maximum increase of f

Rough differential calculus

- Assume $\mathcal{F}$ is the right function for target concept, i.e.,

$$C = \mathcal{F}(a_1, ..., a_k)$$

Mathematical analogy

Real function analysis

Searching for maximum of a real function $f : \mathbb{R}^k \to \mathbb{R}$

1. Get some information about its differential, e.g., gradient

$$\nabla f = \left\langle \frac{df}{dx_1}, \dots, \frac{df}{dx_k} \right\rangle$$

2. Discover the properties of $f(\mathbf{x}_0)$ from its differential, e.g.,
$\nabla f(\mathbf{x}_0)$ is the direction which promises maximum increase of f

Rough differential calculus

- Assume $\mathcal{F}$ is the right function for target concept, i.e.,

$$C = \mathcal{F}(a_1, ..., a_k)$$

- Decision rules for the comparing table indicate:
How the changes on attributes effect on the changes of decision

Mathematical analogy

Real function analysis

Searching for maximum of a real function $f : \mathbb{R}^k \to \mathbb{R}$

1. Get some information about its differential, e.g., gradient

$$\nabla f = \left\langle \frac{df}{dx_1}, \ldots, \frac{df}{dx_k} \right\rangle$$

2. Discover the properties of $f(\mathbf{x}_0)$ from its differential, e.g.,
   $\nabla f(\mathbf{x}_0)$ *is the direction which promises maximum increase of* f

Rough differential calculus

- Assume $\mathcal{F}$ is the right function for target concept, i.e.,

$$C = \mathcal{F}(a_1, ..., a_k)$$

- Decision rules for the comparing table indicate:
  How the changes on attributes effect on the changes of decision
- Such rules are discovered knowledge!

Mathematical analogy

Real function analysis

Searching for maximum of a real function $f : \mathbb{R}^k \to \mathbb{R}$

1. Get some information about its differential, e.g., gradient

$$\nabla f = \left\langle \frac{df}{dx_1}, \ldots, \frac{df}{dx_k} \right\rangle$$

2. Discover the properties of $f(\mathbf{x}_0)$ from its differential, e.g.,
   $\nabla f(\mathbf{x}_0)$ *is the direction which promises maximum increase of* f

Rough differential calculus

- Assume $\mathcal{F}$ is the right function for target concept, i.e.,

$$C = \mathcal{F}(a_1, ..., a_k)$$

- Decision rules for the comparing table indicate:
  How the changes on attributes effect on the changes of decision
- Such rules are discovered knowledge!
- Meaning of rough set rules: short, certain, possible

Ranking

- Ranking learning can be understood as a problem of reconstruction of the correct ranking list of a set of objects;
- Let $\mathbb{S} = (U, A \cup \{dec\})$ be a training data set and $(u_1, ..., u_n)$ is an ordered sequence of objects from U according to dec, i.e.,

$$dec(u_1) \leq dec(u_2) \leq ... \leq dec(u_n).$$

- The problem is to reconstruct the ranking list of objects from a test data set $\mathbb{S}' = (V, A \cup \{dec\})$ without using decision attribute dec.
- Our algorithm is based on the round robin tournament system which is carried out on the set of objects $U \cup V$.

Round robin algorithm for ranking

- Similarly to football leagues, every object from V – playing the tournament – obtains a total score summarizing its played matches.
- The objects from V are sorted with respect to their scores.
- The scoring method use $\pi_{\mathbf{L},U}(x,y)$as a referee:

$$Score(x) = \sum_{y \in U \cup V} w(y) \cdot \pi_{\mathbf{L},U}(x,y)$$

where $w(y)$ is a weighting parameter that measures the importance of the object y in our ranking algorithm. In our experiments:

$$w(y) = \begin{cases} 1 & \text{if } y \text{ is a test object, i.e., } y \in V; \\ 1 + \frac{i}{n} & \text{if } y = u_i \in U. \end{cases}$$

- The algorithm can be applied for all the objects from $U \cup V$ to embed V into the ordered sequence $(u_1, u_2, ..., u_n)$.

Evaluation of ranking algorithms

- There are several well known "compatibility tests" for this problem, e.g., Spearman R, Kendall τ, or Gamma coefficients.
- If the proper ranking list of V is denoted by $X = (x_1, x_2 ..., x_k)$, then the second ranking list is a permutation of elements of V, and represented by $Y = (x_{\sigma(1)}, x_{\sigma(2)}, ..., x_{\sigma(k)})$
- **The Spearman coefficient** for a permutation $\sigma : \{1, ..., k\} \rightarrow \{1, ..., k\}$ is computed by

$$R = 1 - \frac{6 \sum_{i=1}^{k} (\sigma(i) - i)^2}{k(k-1)(k+1)} \tag{7}$$

- The Spearman coefficient takes values from $[-1; 1]$.

Further applications

- Prediction of continuous decision:
  - Embed the object x into the sequence $(u_1, u_2, ..., u_n)$ by applying ranking algorithm for objects from $\{x\} \cup U$
  - Assuming that x is embedded between u_i and u_{i+1}, then

$$prediction(x) = \frac{dec(u_i) + dec(u_{i+1})}{2}$$

  is returned as a result of prediction.

Further applications

- Prediction of continuous decision:
  - Embed the object x into the sequence $(u_1, u_2, ..., u_n)$ by applying ranking algorithm for objects from $\{x\} \cup U$
  - Assuming that x is embedded between u_i and u_{i+1}, then

$$prediction(x) = \frac{dec(u_i) + dec(u_{i+1})}{2}$$

  is returned as a result of prediction.
- Experiment design:
  Point out the minimal number of changes that can improve the current combination;

Further applications

- Prediction of continuous decision:
  - Embed the object x into the sequence $(u_1, u_2, ..., u_n)$ by applying ranking algorithm for objects from $\{x\} \cup U$
  - Assuming that x is embedded between u_i and u_{i+1}, then

$$prediction(x) = \frac{dec(u_i) + dec(u_{i+1})}{2}$$

  is returned as a result of prediction.

- Experiment design:
  Point out the minimal number of changes that can improve the current combination;
- Optimization by dynamic learning;

The prediction algorithm

Let the training set of objects $U = \{u_1, ...u_n\}$ be given. The prediction algorithm computes the decision value of the test object $x \notin U$ as follows:

The algorithm:

Require: The set of labeled objects U and unlabeled object x;
parameters: learning algorithm $\mathbf{L}$;

Ensure: A predicted decision for x;

1: Embed the object x into the sequence $(u_1, u_2, ..., u_n)$ by applying ranking algorithm for objects from $\{x\} \cup U$ using $\mathbf{L}$ and decision table for U;

2: Let us assume that x is embedded between u_i and u_{i+1};

3: Return $prediction(x) = \frac{dec(u_i)+dec(u_{i+1})}{2}$ as a result of prediction.

The error rate on the set of testing objects V is measured by

$$error(V) = \frac{1}{card(V)} \sum_{x \in V} |dec(x) - prediction(x)|$$

Dynamic ranking

- The quality of ranking algorithm can be low due to the small number of objects.
- In many applications the number of training objects is increasing in time, but it is connected with certain cost of examination.
- We can treat a ranking problem as an optimization problem:
  - get the highest value element (combination)
  - require as low as possible the number of examples, i.e., to minimize the number of examinations and the cost of the whole process.

Dynamic ranking algorithm

The dynamic ranking algorithm

Require: The set of labeled objects U and unlabeled objects V;
parameters: learning algorithm $\mathbf{L}$ and positive integer $request\_size$;

Ensure: A list of objects to be requested; Ranking of elements in the U_2 in the $RankList$;

```
U_1 ← U; U_2 ← V;
RankList ← [ ];   //the empty list
while not STOP CONDITION do
    Rank elements of U_2 by using L and decision table for U_1; Let this
    ranking list be: (x_1, x_2, ...);
    for i = 1 to request_size do
        RankList.append(x_i)
        U_1 ← U_1 ∪ {x_i};   U_2 ← U_2 \ {x_i};
    end for
end while
```

Experiments - Data sets

- 4 tables:

data and sizes	possible comb.	dec. domain
data set A : 40 × 6	64	$(0, 10)$
data set B : 60 × 8	384	$(0, 10)$
data set C : 130 × 55	$2^{41}3^{11}4^{2}6$	$(0, 10)$
Artificial : 64 × 6	64	(5.7,33)

Artificial decision:

$$dec = e^{{a_1}^{a_2}} + (a_1 + a_2 + a_3 + a_4 + a_5) * a_6 / a_3 + \sin(a_4) + \ln(a_5) + noise$$

- 6 learning algorithms
- 7-fold cross validation

Results for real data

Learning algorithm	Table A		Table B		Table C	
	acc.(%)	pred.error	acc.(%)	pred.error	acc.(%)	pred.error
rough set	79.26	0.4843	81.63	0.3815	75.57	0.4328
naive bayes	72.7	0.849	74.22	0.5355	56.89	0.8925
nnge	76.75	0.5170	80.54	0.345	-	-
boost nnge	80.67	0.4383	83.76	0.3779	-	-
j48	75.8	0.6981	81.29	0.3821	76.2	0.4958
boost j48	80.17	0.4935	85.23	0.318	-	-

Results for artificial data

Learning algorithm	Ranking		Prediction		Dynamic Ranking	
	Spearman	acc.(%)	Pearson	pred.error	pos.	Spearman
Decision Rules	0.8930	83.28%	0.9653	1.4547	1.3	0.9501
Naive Bayes	0.7984	78.52%	0.5948	3.8336	1.3	0.8540
Nnge	0.7770	77.19%	0.9178	1.8245	2.5	0.9165
Boosting Nnge	0.8318	80.27%	0.9184	1.6244	1.6	0.9433
C45	0.7159	75.7%	0.8372	2.2108	2.7	0.8736
Boosting C45	0.8536	80.74%	0.9661	1.3483	1.6	0.9475

Outline

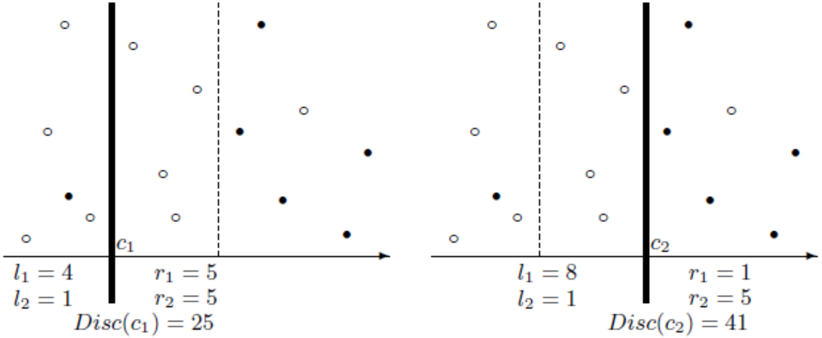
Exercise 1: decision rules vs. decision tree

Each path of decision tree can be interpreted as a decision rule. Thus decision tree can be treated as a set of decision rules.

1. True or false: "Each path of a minimal decision tree is a minimal consistent decision rule" ?
2. What are the main differences between
   1. the set of decision rules in rough classifiers; and
   2. the set of decision rules stored in a consistent decision tree?
3. Find the maximal possible number $M(k)$ of minimal and consistent decision rules for a decision table with k attributes?

Exercise 2: Boundary cuts

Prove that if c is the best cut for an atribute then c must be one of the boundary cut.



Exercise 3: Are the best cuts really good?

A real number $v_i \in a(U)$ *is called single value of an attribute* a *if there is exactly one object* $u \in U$ *such that* $a(u) = v_i$. *The cut* $(a; c)$ *is called the single cut if* c *is lying between two single values* v_i *and* v_{i+1}.

Prove the following properties related to single cuts:

Theorem

In case of decision tables with two decision classes, any single cut c_i, *which is a local maximum of the function* $Disc$, *resolves at least half of conflicts in the decision table, i.e.*

$$Disc(c_i) \geq \frac{1}{2} \cdot \mathit{conflict}(\mathbb{S}).$$

What can you say about the depth of decision tree build by MD-heuristics?

Applications of Rough sets in Machine Learning and Data Mining

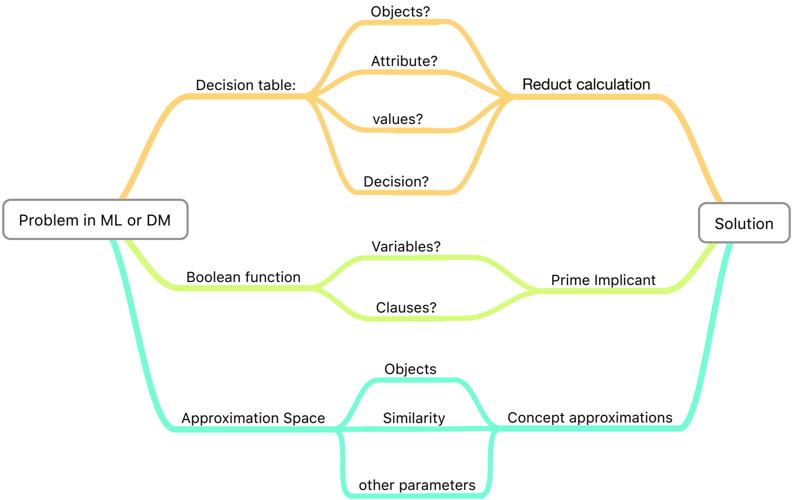
Part III: Rough sets and Data mining

Nguyen Hung Son

University of Warsaw, Poland

Milan, 26 July 2016

Rough set approach to ML and Data Mining



Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

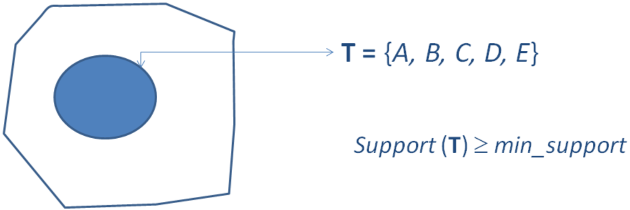
Association rule generation

Problem:

For a given information table A, an integer s, and a real number $c \in [0, 1]$, search for as much as possible association rules R such that $support(R) \geq s$ and $confidence(R) \geq c$;

Association rule generation methods consist of two steps:

1. Generate as much as possible frequent templates $T = D_1 \wedge ... D_k$



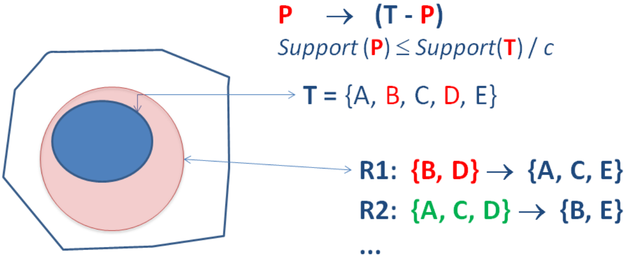
Association rule generation

Problem:

For a given information table A, an integer s, and a real number $c \in [0, 1]$, search for as much as possible association rules R such that $support(R) \geq s$ and $confidence(R) \geq c$;

Association rule generation methods consist of two steps:

1. Generate as much as possible frequent templates $T = D_1 \wedge ... D_k$
2. For any template T, search for a partition $T = P \wedge (T - P)$ s.t.:



Reduct approach to association rules

Surprise!: the second step can be solved by rough set methods (using α-reducts).

Reduct approach to association rules

Surprise!: the second step can be solved by rough set methods (using α-reducts).

Theorem

Given:

- $\mathbf{D}$ – *an information system,*
- T – *a template,*
- c – *minimal confidence level;*

An implication $P \implies (T - P)$ *is* c*-confident association rule if and only if* P *is an* α*-reduct of a decision table* $\mathbf{D}|_T$*, where*

$$\alpha = 1 - \frac{1/c - 1}{n/support(T) - 1}$$

$\mathbb{D}\vert_{A,B,C,D,E}$	
t_1	A C
t_2	A B C D E
t_3	A B C D E
t_4	A B C D E
t_5	B E
t_6	A E
t_7	E
t_8	A B C D E
t_9	A B C D E
t_{10}	A B C D E
t_{11}	A C D
t_{12}	A D
t_{13}	A B C D E
t_{14}	A B
t_{15}	A B C D E
t_{16}	A B C D E
t_{17}	A B C D E
t_{18}	B C D

$\mathbb{D}\|_{\mathbf{T}}$	A	B	C	D	E	dec
t_1	1	0	1	0	0	0
t_2	1	1	1	1	1	1
t_3	1	1	1	1	1	1
t_4	1	1	1	1	1	1
t_5	0	1	0	0	1	0
t_6	1	0	0	0	1	0
t_7	0	0	0	0	1	0
t_8	1	1	1	1	1	1
t_9	1	1	1	1	1	1
t_{10}	1	1	1	1	1	1
t_{11}	1	0	1	1	0	0
t_{12}	1	0	0	1	0	0
t_{13}	1	1	1	1	1	1
t_{14}	1	1	0	0	0	0
t_{15}	1	1	1	1	1	1
t_{16}	1	1	1	1	1	1
t_{17}	1	1	1	1	1	1
t_{18}	0	1	1	1	0	0

$\mathbb{D}\vert_{\mathbf{T}}$	A	B	C	D	E	dec
t_1	1	0	1	0	0	0
t_2	1	1	1	1	1	1
t_3	1	1	1	1	1	1
t_4	1	1	1	1	1	1
t_5	0	1	0	0	1	0
t_6	1	0	0	0	1	0
t_7	0	0	0	0	1	0
t_8	1	1	1	1	1	1
t_9	1	1	1	1	1	1
t_{10}	1	1	1	1	1	1
t_{11}	1	0	1	1	0	0
t_{12}	1	0	0	1	0	0
t_{13}	1	1	1	1	1	1
t_{14}	1	1	0	0	0	0
t_{15}	1	1	1	1	1	1
t_{16}	1	1	1	1	1	1
t_{17}	1	1	1	1	1	1
t_{18}	0	1	1	1	0	0

For $c = 100\%$ we have $\alpha = 100\%$

100% ass. rules

$C, E \Rightarrow A, B, D$
$D, E \Rightarrow A, B, C$
$A, B, C \Rightarrow D, E$
$A, B, D \Rightarrow C, E$
$A, B, E \Rightarrow C, D$
$A, C, D \Rightarrow B, E$

$\mathbb{D}\vert_{\mathbf{T}}$	A	B	C	D	E	dec
t_1	1	0	1	0	0	0
t_2	1	1	1	1	1	1
t_3	1	1	1	1	1	1
t_4	1	1	1	1	1	1
t_5	0	1	0	0	1	0
t_6	1	0	0	0	1	0
t_7	0	0	0	0	1	0
t_8	1	1	1	1	1	1
t_9	1	1	1	1	1	1
t_{10}	1	1	1	1	1	1
t_{11}	1	0	1	1	0	0
t_{12}	1	0	0	1	0	0
t_{13}	1	1	1	1	1	1
t_{14}	1	1	0	0	0	0
t_{15}	1	1	1	1	1	1
t_{16}	1	1	1	1	1	1
t_{17}	1	1	1	1	1	1
t_{18}	0	1	1	1	0	0

For $c = 100\%$ we have $\alpha = 100\%$

100% ass. rules

$C, E \Rightarrow A, B, D$
$D, E \Rightarrow A, B, C$
$A, B, C \Rightarrow D, E$
$A, B, D \Rightarrow C, E$
$A, B, E \Rightarrow C, D$
$A, C, D \Rightarrow B, E$

For $c = 90\%$ we have

$$\alpha = 1 - \frac{\frac{1}{c} - 1}{\frac{n}{s} - 1} = 0.86$$

90% ass. rules

$A, B \Rightarrow C, D, E$
$A, C \Rightarrow C, D, E$
$A, D \Rightarrow B, C, E$
$A, E \Rightarrow B, C, D$
$B, C \Rightarrow A, D, E$
$B, E \Rightarrow A, C, D$
$C, D \Rightarrow A, B, E$

$\mathbb{D}\|_{A,B,C,D,E}$	
t_1	A C
t_2	A B C D E
t_3	A B C D E
t_4	A B C D E
t_5	B E
t_6	A E
t_7	E
t_8	A B C D E
t_9	A B C D E
t_{10}	A B C D E
t_{11}	A C D
t_{12}	A D
t_{13}	A B C D E
t_{14}	A B
t_{15}	A B C D E
t_{16}	A B C D E
t_{17}	A B C D E
t_{18}	B C D

For $c = 100\%$ we have $\alpha = 100\%$

100% ass. rules

$C, E \Rightarrow A, B, D$
$D, E \Rightarrow A, B, C$
$A, B, C \Rightarrow D, E$
$A, B, D \Rightarrow C, E$
$A, B, E \Rightarrow C, D$
$A, C, D \Rightarrow B, E$

For $c = 90\%$ we have

$$\alpha = 1 - \frac{\frac{1}{c} - 1}{\frac{n}{s} - 1} = 0.86$$

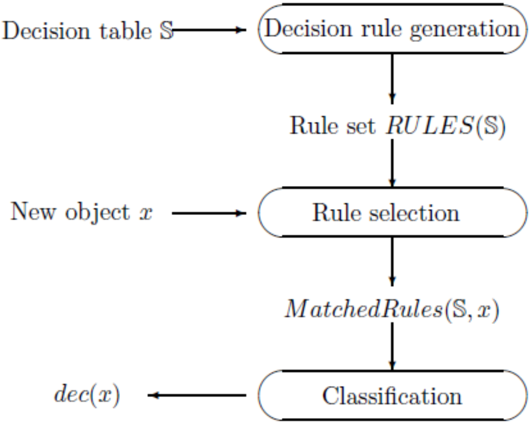
90% ass. rules

$A, B \Rightarrow C, D, E$
$A, C \Rightarrow C, D, E$
$A, D \Rightarrow B, C, E$
$A, E \Rightarrow B, C, D$
$B, C \Rightarrow A, D, E$
$B, E \Rightarrow A, C, D$
$C, D \Rightarrow A, B, E$

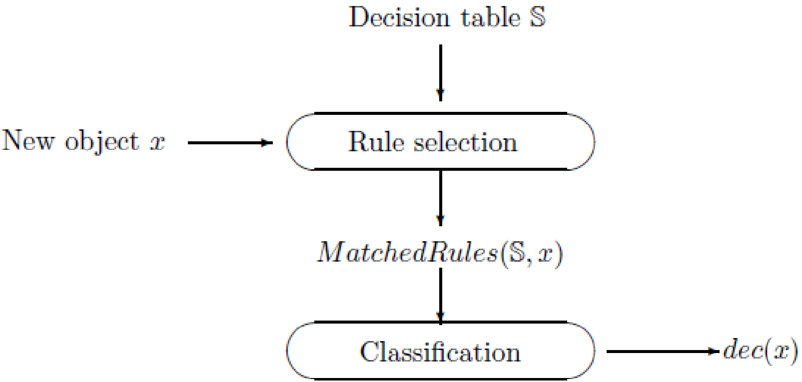
Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

Eager vs. lazy rough classifiers



Eager vs. lazy rough classifiers



Apriori-based reduct calculation

$\mathbb{A}$	a_1	a_2	a_3	a_4	dec
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no
x	sunny	mild	high	TRUE	?

$\Rightarrow$

$\mathbb{A}\|_x$	d_1	d_2	d_3	d_4	dec
ID	$a_1\|_x$	$a_2\|_x$	$a_3\|_x$	$a_4\|_x$	dec
1	1	0	1	0	no
2	1	0	1	1	no
3	0	0	1	0	yes
4	0	1	1	0	yes
5	0	0	0	0	yes
6	0	0	0	1	no
7	0	0	0	1	yes
8	1	1	1	0	no
9	1	0	0	0	yes
10	0	1	0	0	yes
11	1	1	0	1	yes
12	0	1	1	1	yes
13	0	0	0	0	yes
14	0	1	1	1	no

Standard (eager) method

rules	supp.
outlook(overcast)⇒play(yes)	4
humidity(normal) AND windy(FALSE)⇒play(yes)	4
outlook(sunny) AND humidity(high)⇒play(no)	3
outlook(rainy) AND windy(FALSE)⇒play(yes)	3
outlook(sunny) AND temperature(hot)⇒play(no)	2
outlook(rainy) AND windy(TRUE)⇒play(no)	2
outlook(sunny) AND humidity(normal)⇒play(yes)	2
temperature(cool) AND windy(FALSE)⇒play(yes)	2
temperature(mild) AND humidity(normal)⇒play(yes)	2
temperature(hot) AND windy(TRUE)⇒play(no)	1
outlook(sunny) AND temperature(mild) AND windy(FALSE)⇒play(no)	1
outlook(sunny) AND temperature(cool)⇒play(yes)	1
outlook(sunny) AND temperature(mild) AND windy(TRUE)⇒play(yes)	1
temperature(hot) AND humidity(normal)⇒play(yes)	1

The testing object

$$\langle sunny, mild, high, TRUE \rangle$$

is matched by two decision rules:

- (outlook = sunny) AND (humidity = high) $\Rightarrow play = no$ (rule nr 3)
- (outlook = sunny) AND (temperature = mild) AND (windy = TRUE) $\Rightarrow play = yes$ (rule nr 13)

Lazy algorithm on $\mathbb{A}|_x$

$\lambda_{max} = 3; \sigma_{\min} = 1; \alpha_{\min} = 1$

$i = 1$			
$\mathbf{C}_1$	check	$\mathbf{R}_1$	$\mathbf{F}_1$
$\{d_1\}$	(3,2)		$\{d_1\}$
$\{d_2\}$	(4,2)		$\{d_2\}$
$\{d_3\}$	(4,3)		$\{d_3\}$
$\{d_4\}$	(3,3)		$\{d_4\}$

$i = 2$			
$\mathbf{C}_2$	check	$\mathbf{R}_2$	$\mathbf{F}_2$
$\{d_1, d_2\}$	(1,1)		$\{d_1, d_2\}$
$\{d_1, d_3\}$	(3,0)	$\{d_1, d_3\}$	
$\{d_1, d_4\}$	(1,1)		$\{d_1, d_4\}$
$\{d_2, d_3\}$	(2,2)		$\{d_2, d_3\}$
$\{d_2, d_4\}$	(1,1)		$\{d_2, d_4\}$
$\{d_3, d_4\}$	(2,1)		$\{d_3, d_4\}$

$i = 3$			
$\mathbf{C}_3$	check	$\mathbf{R}_3$	$\mathbf{F}_3$
$\{d_1, d_2, d_4\}$	(0,1)	$\{d_1, d_2, d_4\}$	
$\{d_2, d_3, d_4\}$	(1,1)		$\{d_2, d_3, d_4\}$

$MatchRules(\mathbb{A}, x) = \mathbf{R}_2 \cup \mathbf{R}_3$:

(outlook = sunny) AND (humidity = high) $\Rightarrow play = no$

(outlook = sunny) AND (temperature = mild) AND (windy = TRUE) $\Rightarrow play = yes$

FDP(Frequent Decision Pattern)-tree

- The key concept, adopted from FP-growth algorithm for frequent pattern mining;
- FDP is the prefix tree for the collection of ordered list of descriptors;
- Each node in FDP tree has four fields:
  - *descriptor_name* is the name of descriptor,
  - *support* is the number of training objects that contain all descriptors on the path from the root to the current node,
  - *class_distribution* is the detail support for each decision class and
  - *node_link* are used to create list of nodes of the same descriptor

General scheme

- Construction of $FDP(x)$. This step requires only two data scanning passes:
  - First pass:
    - calculate the frequencies of descriptors from $inf_A(x)$
    - create $DESC(x)$ – the ordered list of frequent descriptors;
  - Second pass:
    - convert each training object u into a list $D(u)$ of frequent descriptors from $DESC(x)$ that occur in $inf_A(u)$;
    - insert the list $D(u)$ into the data structure $FDP(x)$.
- Generation of the set of frequent decision rules from $FDP(x)$ by a recursive procedure.
- Insert the obtained rules into a data structure called *the minimal rule tree* – denoted by $MRT(x)$ – to get the set of irreducible decision rules. This data structure can be used to perform different voting strategy.

Example: FDP-tree construction – I step

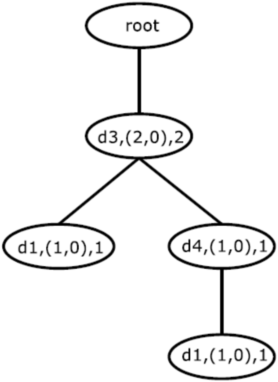
$\mathbb{A}$	a_1	a_2	a_3	a_4	*dec*
ID	outlook	temp.	hum.	windy	play
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no
x	sunny	mild	high	TRUE	?

$\Longrightarrow$

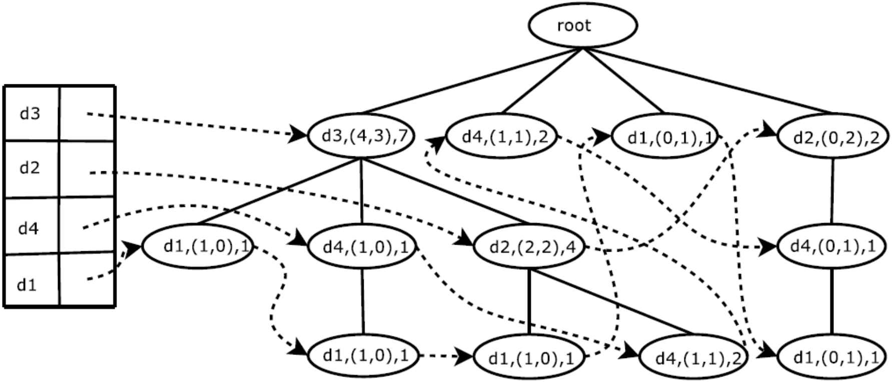
ID	descriptor lists	dec
1	d3, d1	[no]
2	d3, d4, d1	[no]
3	d3	[yes]
4	d3, d2	[yes]
5		[yes]
6	d4	[no]
7	d4	[yes]
8	d3, d2, d1	[no]
9	d1	[yes]
10	d2	[yes]
11	d2, d4, d1	[yes]
12	d3, d2, d4	[yes]
13		[yes]
14	d3, d2, d4	[no]

Descriptor:	(outlook=sunny)	(temp.=mild)	(hum.=high)	(windy=true)
Notation:	d1	d2	d3	d4
Frequency:	5	6	7	6

Example: FDP-tree construction – II step

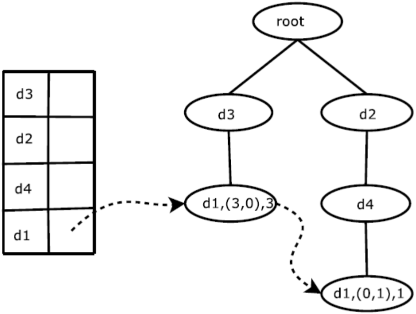


Example: FDP-tree construction – II step



Example: Rule extraction from FDP-tree

1	(outlook = sunny) $\wedge$ (hum. = high) $\Rightarrow play = no$
2	(outlook = sunny) $\wedge$ (temp. = mild) $\wedge$ (windy = TRUE) $\Rightarrow play = yes$
3	(outlook = sunny) $\wedge$ (temp. = mild) $\wedge$ (hum. = high) $\Rightarrow play = no$
4	(outlook = sunny) $\wedge$ (hum. = high) $\wedge$ (windy = TRUE) $\Rightarrow play = no$



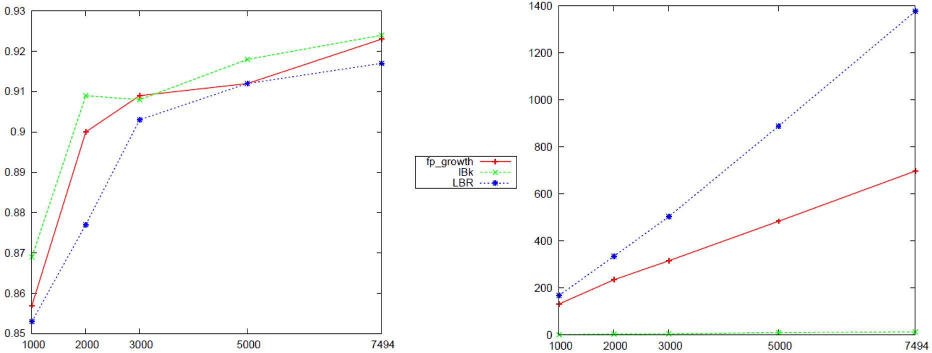
1	(outlook = sunny) $\wedge$ (hum. = high) $\Rightarrow play = no$
2	(outlook = sunny) $\wedge$ (temp. = mild) $\wedge$ (windy = TRUE) $\Rightarrow play = yes$

Data sets

- Data sets: *Poker Hand, Covertype, Pen-Based Recognition of Handwritten Digits*
- Source: UCI Machine Learning Repository (`http://archive.ics.uci.edu/ml/datasets`)
- Testing objective: performance, scalability, accuracy.

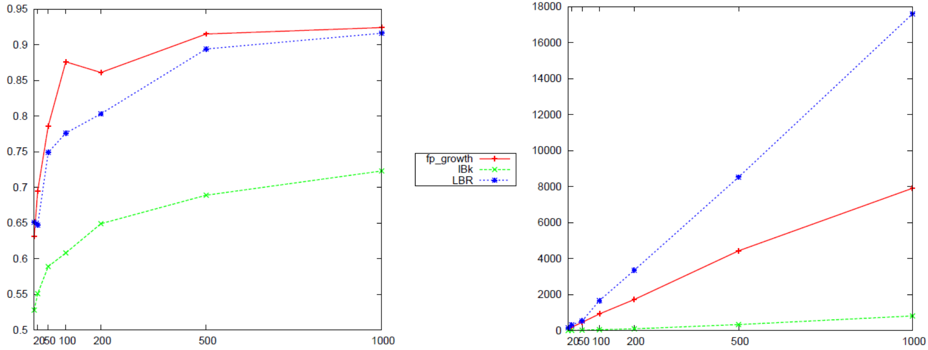
Pendigit

16 attributes, 10 decision classes, 7494 training objects, 3699 test objects;

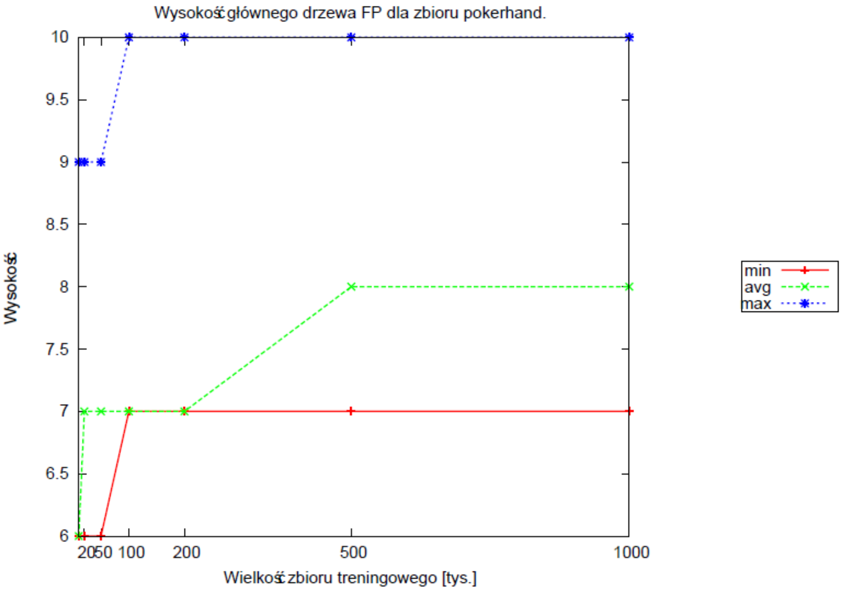


Poker Hand data

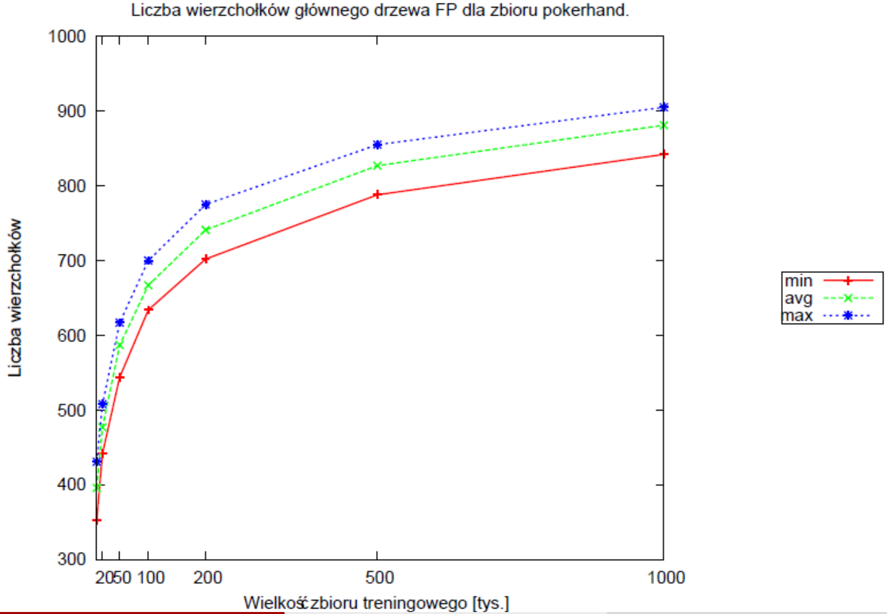
10 attributes, 10 decision classes, 1000000 training objects, 1000 test objects



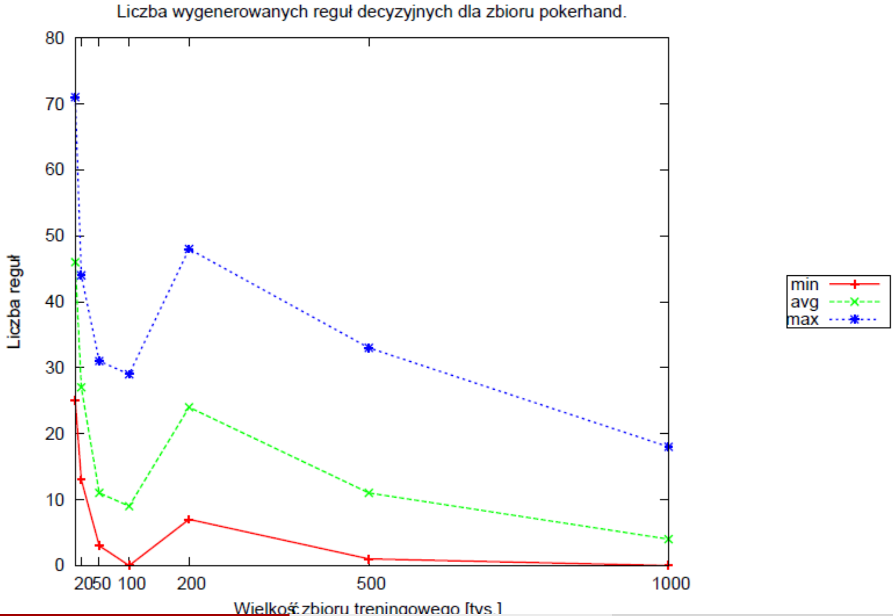
Poker Hand data – height of FDP-tree



Poker Hand data – nr of nodes

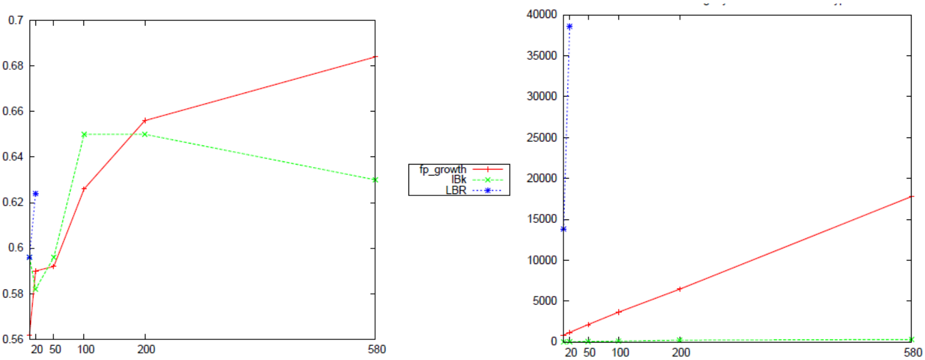


Poker Hand data – nr of rules

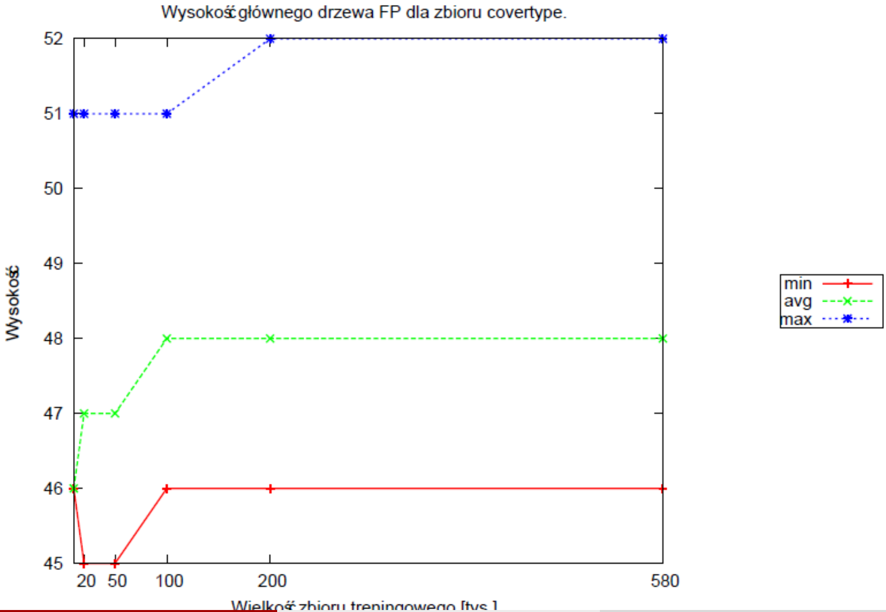


Covertype

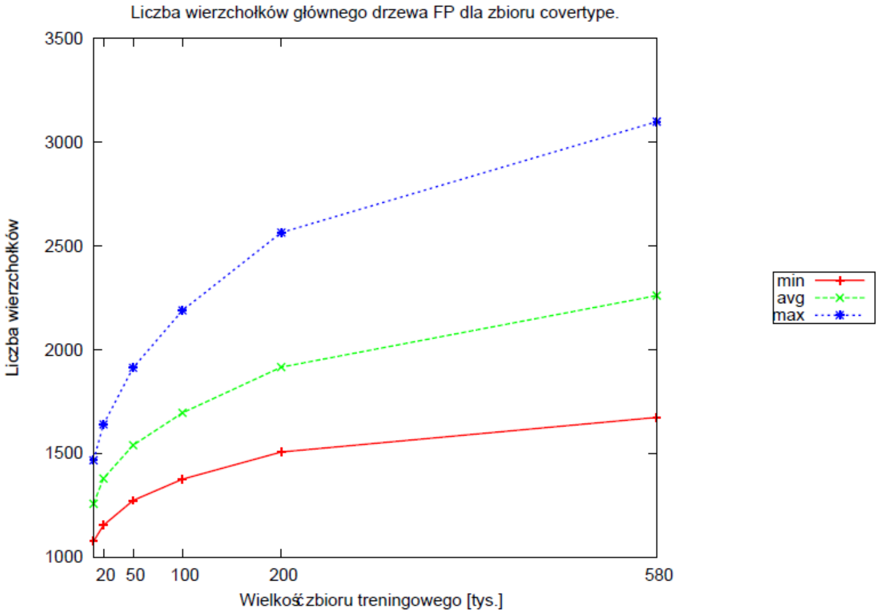
54 attributes, 7 decision classes, 580000 training objects, 500 test objects;



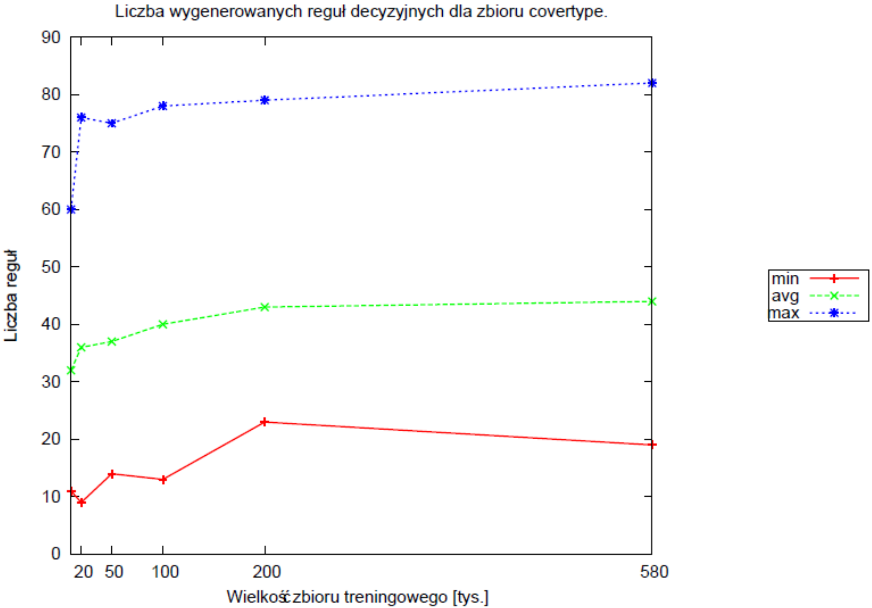
Covertype – height of FDP-tree



Covertype – nr of nodes



Covertype – nr of rules



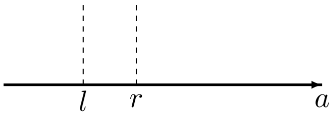
Outline

Outline

Soft cuts and soft DT

A soft cut is any triple $p = \langle a, l, r \rangle$, where

- $a \in A$ is an attribute,
- $l, r \in \Re$ are called the left and right bounds of p ;
- the value $\varepsilon = \frac{r-l}{2}$ is called the uncertain radius of p.
- We say that a soft cut p discerns a pair of objects x_1, x_2 if $a(x_1) < l$ and $a(x_2) > r$.



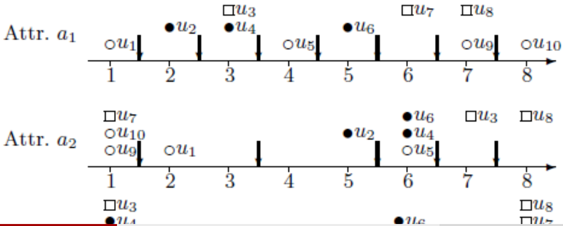
- The intuitive meaning of $p = \langle a, l, r \rangle$:
  - *there is a real cut somewhere between l and r.*
  - *for any value $v \in [l, r]$ we are not able to check if v is either on the left side or on the right side of the real cut.*
  - *$[l, r]$ is an uncertain interval of the soft cut p.*
  - *normal cut can be treated as soft cut of radius 0.*

Soft Decision Tree

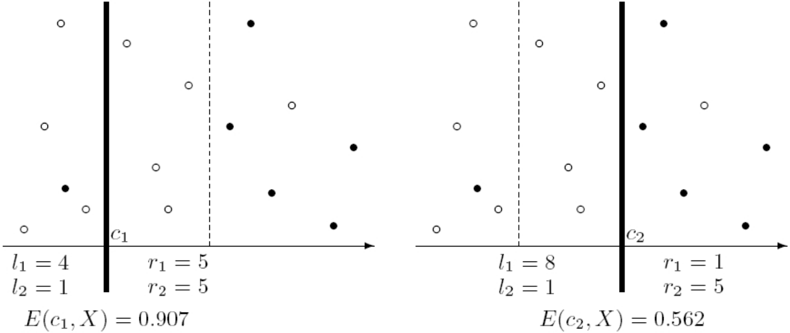
- The test functions can be defined by soft cuts
- Here we propose two strategies using described above soft cuts:
  - *fuzzy decision tree*: any new object u can be classified as follows:
    - For every internal node, compute the probability that u turns left and u turns right;
    - For every leave L compute the probability that u is reaching L;
    - The decision for u is equal to decision labeling the leaf with largest probability.
  - *rough decision tree*: in case of uncertainty
    - Use both left and right subtrees to classify the new object;
    - Put together their answer and return the answer vector;
    - Vote for the best decision class.

Searching for best cuts

A	a_1	a_2	a_3	d
u_1	1.0	2.0	3.0	0
u_2	2.0	5.0	5.0	1
u_3	3.0	7.0	1.0	2
u_4	3.0	6.0	1.0	1
u_5	4.0	6.0	3.0	0
u_6	5.0	6.0	5.0	1
u_7	6.0	1.0	8.0	2
u_8	7.0	8.0	8.0	2
u_9	7.0	1.0	1.0	0
u_{10}	8.0	1.0	1.0	0



Entropy measure



- entropy of an object set X: $Ent(X) = -\sum_{j=1}^{d} p_j \log p_j$
- the entropy of the partition induced by a cut (a, c):

$$E(a, c; U) = \frac{|U_L|}{|U|} Ent(U_L) + \frac{|U_R|}{|U|} Ent(U_R)$$

STANDARD ALGORITHM FOR BEST CUT

- For a given attribute a and a set of candidate cuts $\{c_1, ..., c_N\}$, the best cut (a, c_i) with respect to given heuristic measure

$$F : \{c_1, ..., c_N\} \to \mathbb{R}^+$$

can be founded in time $\Omega(N)$.

- The minimal number of simple SQL queries of form

```
SELECT COUNT
FROM datatable
WHERE (a BETWEEN c_L AND c_R) GROUPED BY d.
```

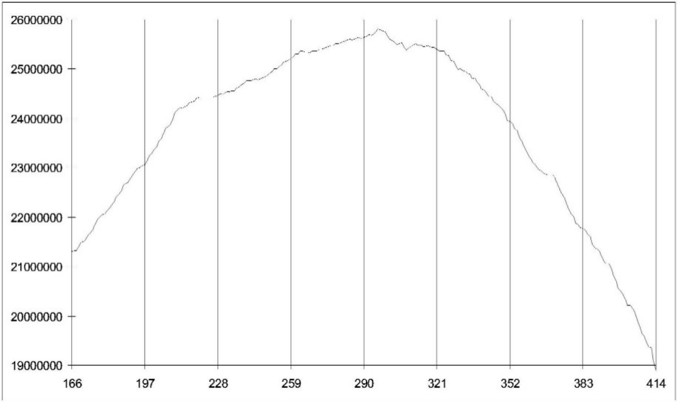
necessary to find out the best cut is $\Omega(dN)$

OUR PROPOSITIONS FOR SOFT CUTS

- Tail cuts can be eliminated
- Divide and Conquer Technique

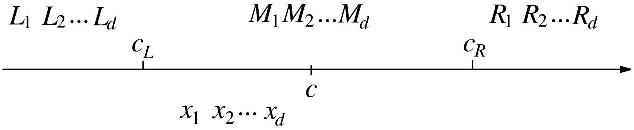
Divide and Conquer Technique:

1. *Divide the set of possible cuts into k intervals;*
2. *Select the interval that most probably contains the best cut;*
3. *If the considered interval is not STABLE enough then Go to Step 1*
4. *Return the current interval(cut) as a result.*



Divide and Conquer Technique:

- The number of SQL queries is $O(d \cdot k \log_k n)$ and is minimum for $k = 3$;
- How to define the measure evaluating the quality of the interval $[c_L; c_R]$?



Discernibility measure:

We construct estimation measures for intervals in four cases:

	Discernibility measure	Entropy Measure
Independency assumption	?	?
Dependency assumption	?	?

Under **dependency assumption**, i.e.

$$\frac{x_1}{M_1} \simeq \frac{x_2}{M_2} \simeq \ldots \simeq \frac{x_d}{M_d} \simeq \frac{x_1 + \ldots + x_d}{M_1 + \ldots + M_d} = \frac{x}{M} =: t \in [0, 1]$$

discernibility measure for $[c_L; c_R]$ can be estimated by:

$$\frac{W(c_L) + W(c_R) + conflict(c_L; c_R)}{2} + \frac{[W(c_R) - W(c_L)]^2}{conflict(c_L; x_R)}$$

Under **dependency assumption**, i.e. $x_1, ..., x_d$ are independent random variables with uniform distribution over sets $\{0, ..., M_1\}$, ..., $\{0, ..., M_d\}$, respectively.

- The mean $E(W(c))$ for any cut $c \in [c_L; c_R]$ satisfies

$$E(W(c)) = \frac{W(c_L) + W(c_R) + conflict(c_L; c_R)}{2}$$

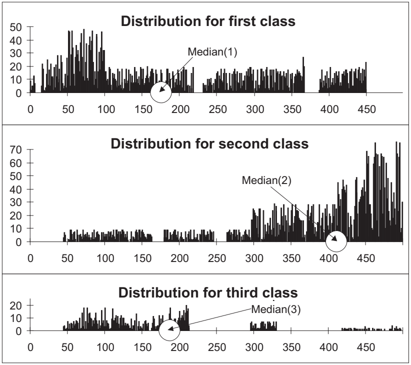
- and for the standard deviation of $W(c)$ we have

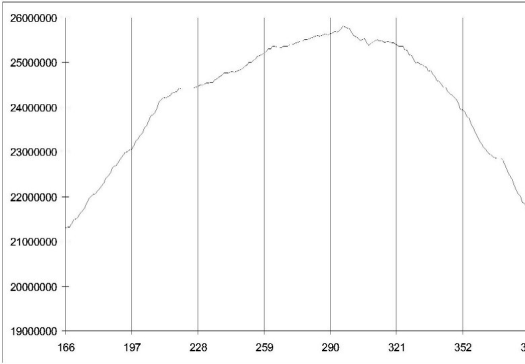
$$D^2(W(c)) = \sum_{i=1}^{n} \left[\frac{M_i(M_i+2)}{12} \left(\sum_{j \neq i} (R_j - L_j) \right)^2 \right]$$

- One can construct the measure estimating quality of the best cut in $[c_L; c_R]$ by

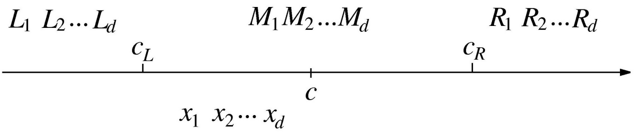
$$\boxed{Eval\left([c_L; c_R], \alpha\right) = E(W(c)) + \alpha \sqrt{D^2(W(c))}}$$

Example


Distribution for first class
Median(1)
50
40
30
20
10
0
0 50 100 150 200 250 300 350 400 450
Distribution for second class
Median(2)
70
60
50
40
30
20
10
0
0 50 100 150 200 250 300 350 400 450
Distribution for third class
Median(3)
20
10
0
0 50 100 150 200 250 300 350 400 450


26000000
25000000
24000000
23000000
22000000
21000000
20000000
19000000
166 197 228 259 290 321 352

Experimental results



$$Eval\left([c_L; c_R], \alpha\right) = E(W(c)) = \frac{W(c_L) + W(c_R) + conflict(c_L; c_R)}{2}$$

Accuracy

Data sets	#objects×#attr.	SLIQ	ENT	MD	MD*
Australian	690 × 14	84.9	85.2	86.2	86.2
German	1000 × 24	-	70	69.5	70.5
Heart	270 × 13	-	77.8	79.6	79.6
Letter	20000 × 16	84.6	86.1	85.4	83.4
SatImage	6435 × 36	86.3	84.6	82.6	83.9
Shuttle	57000 × 9	99.9	99.9	99.9	98.7

Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

TRSM- Tolerance Rough Sets Model

- Let $D = \{d_1, d_2, \dots, d_N\}$ be a set of documents and $T = \{t_1, t_2, \dots, t_M\}$ set of *index terms* for D
- TRSM is an *approximation space* $\mathcal{R} = (T, I_\theta, \nu, P)$ determined over the set of terms T as follows:
  - **Tolerance classes of terms:** (uncertain parameterized function by a threshold θ)

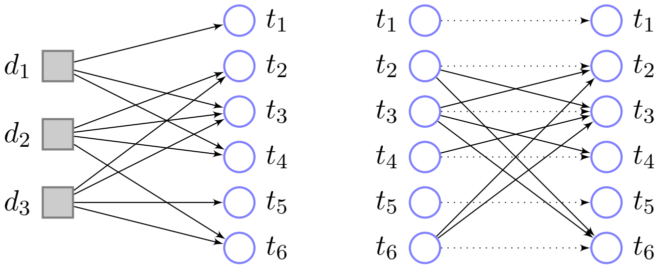
    $$I_\theta(t_i) = \{t_j \mid f_D(t_i, t_j) \geq \theta\} \cup \{t_i\}$$

    where $f_D(t_i, t_j) = |\{d \in D : d \text{ contains both } t_i \text{ and } t_j\}|$
  - **Vague inclusion function:** For $t_i \in T$, $X \subseteq T$:

    $$\mu(t_i, X) = \nu(I_\theta(t_i), X) = \frac{|I_\theta(t_i) \cap X|}{|I_\theta(t_i)|}$$

  - **Structural function:** all tolerance classes of terms are considered as structural subsets: $P(I_\theta(t_i)) = 1$ for all $t_i \in T$.

Tolerance classes



Example: tolerance classes

Term	Tolerance classes for a query "jaguar" using 200 results (returned by Google) and $\theta = 9$	Document frequency
Atari	**Atari, Jaguar**	**10**
Mac	**Mac, Jaguar, OS, X**	**12**
onca	onca, Jaguar, Panthera	9
Jaguar	Atari, Mac, onca, Jaguar, club, Panthera, new, information, OS, site, Welcome, X, Cars	185
club	Jaguar, club	27
Panthera	**onca, Jaguar, Panthera**	**9**
new	Jaguar, new	29
information	Jaguar, information	9
OS	Mac,Jaguar, OS, X	15
site	Jaguar, site	19
Welcome	Jaguar, Welcome	21
X	Mac, Jaguar, OS, X	14
Cars	**Jaguar, Cars**	**24**

- In context of Information Retrieval, a tolerance class represents a concept that is characterized by terms it contains.
- By varying the threshold θ (e.g., relatively to the size of document collection), one can control the degree of relatedness of words in tolerance classes (or the preciseness of the concept represented by a tolerance class).
- Finally, the lower and upper approximations of any subset $X \subseteq T$ can be determined — with the obtained tolerance $\mathcal{R} = (T, I_\theta, \nu, P)$ — respectively as

$$\mathbf{L}_\mathcal{R}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) = 1\};$$

$$\mathbf{U}_\mathcal{R}(X) = \{t_i \in T \mid \nu(I_\theta(t_i), X) > 0\}$$

Enriching document representation

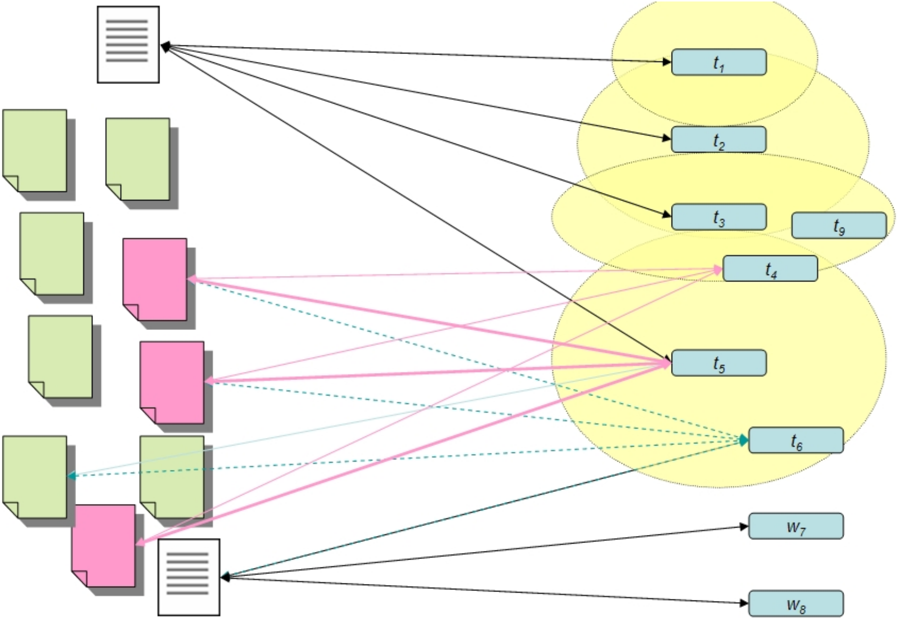
- Let $d_i = \{t_{i_1}, t_{i_2}, ..., t_{i_k}\}$ be a document in D.
- A "richer" representation of d_i can be achieved by its upper approximation in TRSM, i.e.,

$$\mathbf{U}_{\mathcal{R}}(d_i) = \{t_i \in T \mid \nu(I_\theta(t_i), d_i) > 0\}$$

- Extended TF*IDF weighting scheme:

$$w_{ij}^{new} = \begin{cases} (1 + log(f_{d_i}(t_j)) * \log \frac{N}{f_D(t_j)} & \text{if } t_j \in d_i \\ \min_{t_k \in d_i} w_{ik} * \frac{\log \frac{N}{f_D(t_j)}}{1 + \log \frac{N}{f_D(t_j)}} & \text{if } t_j \in \mathbf{U}_{\mathcal{R}}(d_i) \setminus d_i \\ 0 & \text{if } t_j \notin \mathbf{U}_{\mathcal{R}}(d_i) \end{cases}$$

where w_{ij} is the standard TF*IDF weight for term t_j in document d_i.


t_1
t_2
t_3
t_9
t_4
t_5
t_6
w_7
w_8

Title: EconPapers: Rough sets bankruptcy prediction models versus auditor
Description: Rough sets bankruptcy prediction models versus auditor signalling rates. Journal of Forecasting, 2003, vol. 22, issue 8, pages 569-586. Thomas E. McKee. ...

original vector		using upper approximation	
Term	Weight	Term	Weight
auditor	0.567	auditor	0.564
bankruptcy	0.4218	bankruptcy	0.4196
signalling	0.2835	signalling	0.282
EconPapers	0.2835	EconPapers	0.282
rates	0.2835	rates	0.282
versus	0.223	versus	0.2218
issue	0.223	issue	0.2218
Journal	0.223	Journal	0.2218
MODEL	0.223	MODEL	0.2218
prediction	0.1772	prediction	0.1762
Vol	0.1709	Vol	0.1699
		applications	0.0809
		Computing	0.0643

Outline

1. Rough sets and association analysis
   - Rough sets and association rules
   - Scalable Rule-based Classifier
2. Soft decision tree
   - Soft cuts
3. Rough sets and Text mining
   - Clustering of Web Search Results
   - Extended TRSM

Clustering web search results

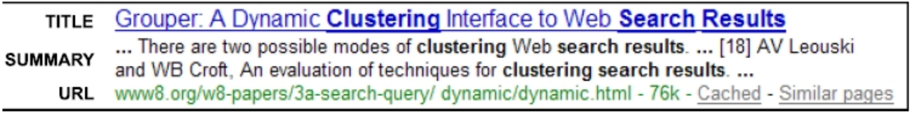
1. **Searching on the web is tedious and time-consuming:**
   - search engines can not index the huge and highly dynamic web contain,
   - the user's "intention behind the search" is not clearly expressed which results in too general, short queries;
2. **Results returned by search engine can count from hundreds to hundreds of thousands of documents.**

Clustering web search results

1. **Searching on the web is tedious and time-consuming:**
   - search engines can not index the huge and highly dynamic web contain,
   - the user's "intention behind the search" is not clearly expressed which results in too general, short queries;
2. **Results returned by search engine can count from hundreds to hundreds of thousands of documents.**
3. **Clustering of search results** = grouping similar snippets together:
   - facilitate presentation of results in more compact form
   - enable thematic browsing of the results set.


Google
Advanced Search Preferences Language Tools Search Tips
clustering search results
Google Search
Web Images Groups Directory News
Searched the web for clustering search results.
Results 1 - 10 of about 270,000. Search took 0.30 seconds.
Vivisimo Document Clustering - automatic categorization and meta ...
Vivisimo's document clustering and meta-search software automatically
categorizes search results on-the-fly into hierarchical clusters. ...
Description: Queries many major search engines, and then uses clustering technology to group the results by topic....
Category: Computers > Internet > Searching > Metasearch
vivisimo.com/ - 20k - Cached
Sponsored Links
results
A large range of health products
available at great prices.
www.healthchemist.com
TITLE
DESCRIPTION
URL
An Evaluation of Techniques for Clustering Search Results - ...
... In this paper, we compare classification methods from IR and Machine Learning
(ML) for clustering search results. Issues such as document... ...
citeseer.nj.nec.com/leouski96evaluation.html - 24k - Cached - Similar pages
An Evaluation of Techniques for Clustering Search Results - ...
... In this paper, we compare classification methods from IR and Machine Learning
(ML) for clustering search results. Issues such as document... ...
citeseer.nj.nec.com/leouski96evaluation.html - 24k - Cached - Similar pages
Citations: An evaluation of techniques for clustering search ...
AV Leouski and WB Croft. An evaluation of techniques for clustering search
results. ... An evaluation of techniques for clustering search results. ...
citeseer.nj.nec.com/context/43428/305473 - 20k - Cached - Similar pages
[More results from citeseer.nj.nec.com]
[PDF] An Evaluation of Techniques for Clustering Search Results
File Format: PDF/Adobe Acrobat - View as HTML
An Evaluation of Techniques for Clustering Search Results 1 Anton V. Leouski and
W. Bruce Croft Computer Science Department University of Massachusetts at ...
www-ciir.cs.umass.edu/~leouski/ publications/papers/ir-76.pdf - Similar pages
Adaptive Query Expansion Based on Clustering Search Results
... Adaptive Query Expansion Based on Clustering Search Results. EGUCHI KOJI ※1
※2 , ITO HIDETAKA ※1 , KUMAMOTO AKIRA ※1 , KANATA YAKICHI ※1. ...
www.ipsj.or.jp/members/Journal/ Eng/4005/article052.html - 4k - Cached - Similar pages

Snippet clustering problems



- Poor representation of snippets can result low correlation between documents and document clusters;
- Except good quality clusters, it is also required to produce meaningful, concise description for cluster;
- The algorithm must be fast to process results on-line.

Snippet clustering problems

TITLE	Grouper: A Dynamic **Clustering** Interface to Web **Search Results**
SUMMARY	... There are two possible modes of **clustering** Web **search results**. ... [18] AV Leouski and WB Croft, An evaluation of techniques for **clustering search results**. ...
URL	www8.org/w8-papers/3a-search-query/ dynamic/dynamic.html - 76k - Cached - Similar pages

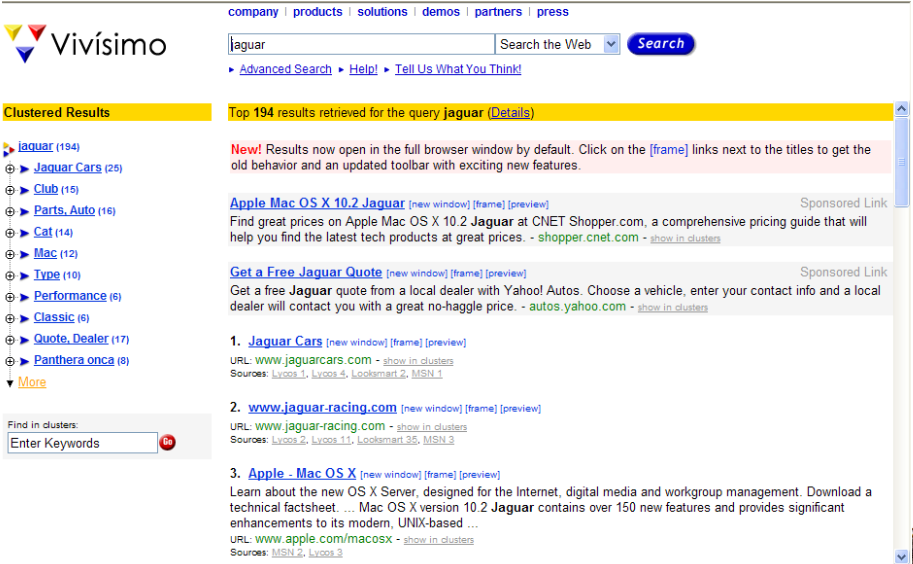
- Poor representation of snippets can result low correlation between documents and document clusters;
- Except good quality clusters, it is also required to produce meaningful, concise description for cluster;
- The algorithm must be fast to process results on-line.

Existing solutions:

use the domain knowledge likes thesaurus or ontology to correct the similarity relation between snippets.

- Global thesaurus, e.g., WordNet;
- Local and context relationships between terms;

Example: vivisimo screenshot

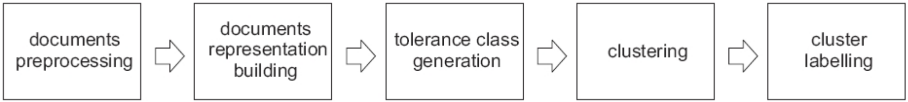


Rough set approach to snippet clustering

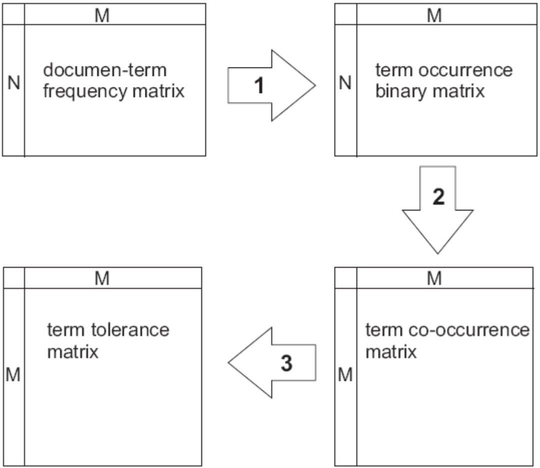
1. Approximation of similarity relation on the set of terms $\Rightarrow$ tolerance rough set model (TRSM);
2. Enriching document representation using upper approximation of snippets in TRSM;
3. Clustering the enriched representations of snippets

Tolerance Rough set Clustering algorithm:

1. **documents preprocessing:** In TRC, the following standard preprocessing steps are performed on snippets: *text cleansing, text stemming, and Stop-words elimination*.
2. **documents representation building:** two main procedures *index term selection and term weighting* are performed.
3. **tolerance class generation:** see next slide
4. **clustering**: k-mean clustering on the enriched document representations; use nearest-neighbor to assign unclassified documents to cluster.
5. **cluster labeling**: phrase labeling.



Step 3: Tolerance class generation



Step 4: Clustering

The set of index terms R_k representing cluster C_k is constructed so that:

- each document d_i in C_k share some or many terms with R_k
- terms in R_k occurs in most documents in C_k
- terms in R_k needs not to be contained by every document in C_k

The weighting for terms t_j in R_k is calculated as an averaged weight of all occurrences in documents of C_k:

$$w_j(R_k) = \frac{\sum_{d_i \in C_k} w_{ij}}{|\{d_i \in C_k \mid t_j \in d_i\}|}$$

Outline

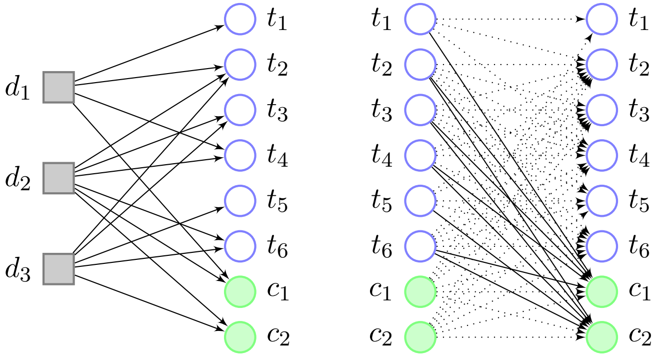
Extended TRSM using thesaurus

The extended TRSM is an approximation space $\mathcal{R}_C = (T \cup C, I_{\theta,\alpha}, \nu, P)$, where C is the mentioned above set of concepts.

- for each term $c_i \in C$ the set $I_{\theta,\alpha}(c_i)$ contains α top terms from the bag of terms of c_i calculated from the textual descriptions of concepts.
- for each term $t_i \in T$ the set $I_{\theta,\alpha}(t_i) = I_\theta(t_i) \cup C_\alpha(t_i)$ consists of the tolerance class of t_i from the standard TRSM and the set of concepts, whose description contains the term t_i as the one of the top α terms.

In extended TRSM, the document $d_i \in D$ is represented by

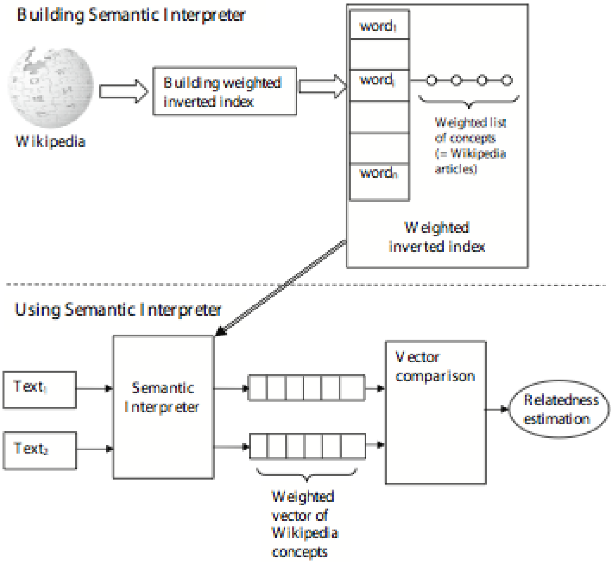
$$\mathbf{U}_{\mathcal{R}_C}(d_i) = \mathbf{U}_{\mathcal{R}}(d_i) \cup \{c_j \in C \mid \nu(I_{\theta,\alpha}(c_j), d_i) > 0\} = \bigcup_{t_j \in d_i} I_{\theta,\alpha}(t_i)$$



Challenge:

How to define the weighting schema?

Example: Explicit Semantic Analysis



Semantic indexing of Medical documents

	$term_0$	$term_1$	...	$term_M$
doc_0	w_{00}	w_{01}	...	w_{0M}
doc_1	w_{10}	...	...	...
...	...	...	w_{ij}	...
doc_N	w_{N0}	...	...	w_{NM}

Representation of system data

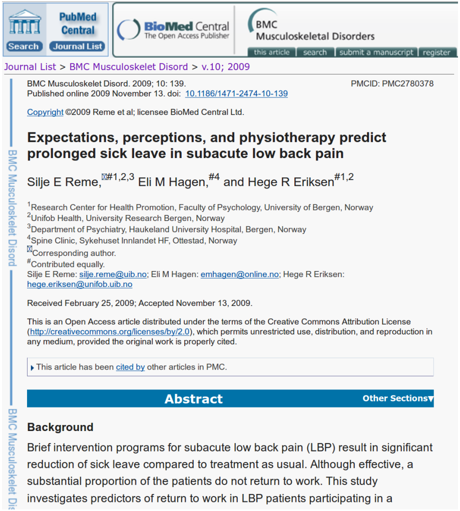
	$concept_0$	$concept_1$	...	$concept_K$
$term_0$	c_{00}	c_{01}	...	c_{0K}
$term_1$	c_{10}	...	...	...
...	...	...	c_{jk}	...
$term_M$	c_{M0}	...	...	c_{MK}

Representation of knowledge base

$$u_{ik} = \sum_{t_j \in T} w_{ij} \times c_{jk}$$

	$concept_0$	$concept_1$	...	$concept_K$
doc_0	u_{00}	u_{01}	...	u_{0K}
doc_1	u_{10}	...	...	...
...	...	...	u_{ik}	...
doc_N	u_{N0}	...	...	u_{NK}

New representation of system data



BMC Musculoskelet Disord. 2009; 10: 139.
Published online 2009 November 13. doi: 10.1186/1471-2474-10-139
PMCID: PMC2780378

Expectations, perceptions, and physiotherapy predict prolonged sick leave in subacute low back pain

Silje E Reme,[#1,2,3] Eli M Hagen,[#4] and Hege R Eriksen[#1,2]

[1]Research Center for Health Promotion, Faculty of Psychology, University of Bergen, Norway
[2]Unifob Health, University Research Bergen, Norway
[3]Department of Psychiatry, Haukeland University Hospital, Bergen, Norway
[4]Spine Clinic, Sykehuset Innlandet HF, Ottestad, Norway
Corresponding author.
[#]Contributed equally.
Silje E Reme: silje.reme@uib.no; Eli M Hagen: emhagen@online.no; Hege R Eriksen: hege.eriksen@unifob.uib.no

Received February 25, 2009; Accepted November 13, 2009.

This article has been cited by other articles in PMC.

Abstract Other Sections▼

Background

Brief intervention programs for subacute low back pain (LBP) result in significant reduction of sick leave compared to treatment as usual. Although effective, a substantial proportion of the patients do not return to work. This study investigates predictors of return to work in LBP patients participating in a

Top 20 concepts:

“Low Back Pain", “Pain Clinics", “Pain Perception", “Treatment Outcome", “Sick Leave", “Outcome Assessment (Health Care)", “Controlled Clinical Trials as Topic", “Controlled Clinical Trial", “Lost to Follow-Up", “Rehabilitation, Vocational", “Pain Measurement", “Pain, Intractable", “Cohort Studies", “Randomized Controlled Trials as Topic", “Neck Pain", “Sickness Impact Profile", “Chronic Disease", “Comparative Effectiveness Research", “Pain, Postoperative"

...

Experiment results

- Ontology: Medical Subject Headings (MeSH)
- Data Set: Pubmed Central
- Expert tags: documents in Pubmed Central are tagged by human experts using headings and (optionally) accompanying subheadings (qualifiers).
- A single document is typically tagged by 10 to 18 heading-subheading pairs.
- Quality Measure: Rand Index