Rough sets in Data Science
Part 1: Basic rough set methods for data analysis

Nguyen Hung Son

University of Warsaw, Poland

2 grudnia 2016
Outline

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
Outline

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
     - Boolean Reasoning Methodology
     - Reducts

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
Figure 2-2. The data science process

Data Science Process

Real World

Raw Data is Collected

Data is Processed

Clean Data

Exploratory Data Analysis

Machine Learning Algorithms
Statistical Models

Make Decisions

Build Data Product

Communicate
Visualizations
Report Findings
MODERN DATA SCIENTIST

Data Scientist, the sexiest job of the 21st century, requires a mixture of multidisciplinary skills ranging from an intersection of mathematics, statistics, computer science, communication and business. Finding a data scientist is hard. Finding people who understand who a data scientist is, is equally hard. So here is a little cheat sheet on who the modern data scientist really is.

MATH & STATISTICS

- Machine learning
- Statistical modeling
- Experiment design
- Bayesian inference
- Supervised learning: decision trees, random forests, logistic regression
- Unsupervised learning: clustering, dimensionality reduction
- Optimization: gradient descent and variants

PROGRAMMING & DATABASE

- Computer science fundamentals
- Scripting language e.g. Python
- Statistical computing packages, e.g., R
- Databases: SQL and NoSQL
- Relational algebra
- Parallel databases and parallel query processing
- MapReduce concepts
- Hadoop and Hive/Pig
- Custom reducers
- Experience with xaaS like AWS

DOMAIN KNOWLEDGE & SOFT SKILLS

- Passionate about the business
- Curious about data
- Influence without authority
- Hacker mindset
- Problem solver
- Strategic, proactive, creative, innovative and collaborative

COMMUNICATION & VISUALIZATION

- Able to engage with senior management
- Story telling skills
- Translate data driven insights into decisions and actions
- Visual art design
- R packages like ggplot or lattice
- Knowledge of any of visualization tools e.g. Tableau
Many tasks in data mining can be formulated as an approximate reasoning problem.

Assume that there are

- Two agents $A_1$ and $A_2$;
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 $U$;
- They use different languages $\mathcal{L}_1$ and $\mathcal{L}_2$;
- Every formula $\psi$ in $\mathcal{L}_1$ (and $\mathcal{L}_2$) describes a set $C_\psi$ of objects from $U$.

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

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 $C \subset U$ used by teacher;
- A sample $U = U^+ \cup U^-$, where
  - $U^+ \subset C$: positive examples;
  - $U^- \subset U \setminus C$: negative examples;
- Language $L_2$ used by learner;

Goal
- build an approximation of $C$ in terms of $L_2$
  - with simple description;
  - with high quality of approximation;
  - using efficient algorithm.

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

<table>
<thead>
<tr>
<th>$u_1$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>...</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$u_n$</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>
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;
**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;
**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 $P = (L, U)$ satisfying the following conditions:

1. $L \subseteq U \subseteq U$;
2. $L, U$ are subsets of $U$ expressible in the language $L_2$;
3. $L \cap X \subseteq C \cap X \subseteq U \cap X$;
4. (*) the set $L$ is maximal (and $U$ is minimal) in the family of sets definable in $L$ satisfying (3).
Generalized definition

Rough approximation of the concept $C$ (induced by a sample $X$): any pair $\mathcal{P} = (L, U)$ satisfying the following conditions:

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

Rough membership function of concept $C$:
any function $f : \mathcal{U} \rightarrow [0, 1]$ such that the pair $(L_f, U_f)$, where

- $L_f = \{ x \in \mathcal{U} : f(x) = 1 \}$ and
- $U_f = \{ x \in \mathcal{U} : f(x) > 0 \}$.

is a rough approximation of $C$ (induced from sample $\mathcal{U}$)
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

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
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.
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?
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:

- \( N \) = number of nickels
- \( D \) = number of dimes

\[
\begin{align*}
D &= N + 4 \\
10D + 5N &= 115
\end{align*}
\]
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:

\[ \Rightarrow D = 9; \quad N = 5 \]
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:

... ⇒ \( D = 9; N = 5 \)

Hura: 9 dimes and 5 nickels!
Boolean Algebra:

A tuple

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

satisfying the 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:**
  \[ a + \bar{a} = 1 \] and \[ a \cdot \bar{a} = 0 \]

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

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

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Applications:

- circuit design;
- propositional calculus;
Boolean Algebra:

a tuple

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

satisfying following axioms:

- **Commutative laws:**
  \[ (a + b) = (b + a) \text{ and } (a \cdot b) = (b \cdot a) \]

- **Distributive laws:**
  \[ a \cdot (b + c) = (a \cdot b) + (a \cdot c) \]
  \[ a + (b \cdot c) = (a + b) \cdot (a + c) \]

- **Identity elements:**
  \[ a + 0 = a \text{ and } a \cdot 1 = a \]

- **Complementary:**
  \[ a + \overline{a} = 1 \text{ and } a \cdot \overline{a} = 0 \]

Binary Boolean algebra

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

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

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>x + y</th>
<th>x \cdot y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>\overline{x}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Applications:

- circuit design;
- propositional calculus;
Boolean function

Any function \( f : \{0, 1\}^n \to \{0, 1\} \) is called a \textit{Boolean function};
Any function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) is called a Boolean function;

An implicant of function \( f \) is a term \( t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k \) such that

\[
\forall x_1, \ldots, x_n \; t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1
\]
Boolean function

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

- An **implicant** of function \( f \) is a term \( t = x_1 \cdots x_m \bar{y}_1 \cdots \bar{y}_k \) such that

\[
\forall x_1, \ldots, x_n \ t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1
\]

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

\[
\begin{array}{ccc|c}
  x & y & z & f \\
  \hline
  0 & 0 & 0 & 0 \\
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  1 & 1 & 0 & 1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 1 & 1 \\
  0 & 1 & 1 & 1 \\
  1 & 1 & 1 & 1 \\
\end{array}
\]
Any function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) is called a *Boolean function*;

*An implicant* of function \( f \) is a term \( t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k \) such that

\[
\forall x_1, \ldots, x_n \ t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1
\]

*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;

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

- Any function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) is called a Boolean function;
- An implicant of function \( f \) is a term \( t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k \) such that
  \[ \forall x_1, \ldots, x_n \ t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1 \]
- 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:
- \( \phi_1 = xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz \)
- \( \phi_2 = (x + y + z)(\overline{x} + y + z)(x + \overline{y} + z)(x + y + \overline{z}) \)

\[
\begin{array}{ccc|c}
 x & y & z & f \\
 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 \\
 0 & 1 & 0 & 0 \\
 1 & 1 & 0 & 1 \\
 0 & 0 & 1 & 0 \\
 1 & 0 & 1 & 1 \\
 0 & 1 & 1 & 1 \\
 1 & 1 & 1 & 1 \\
\end{array}
\]
Boolean function

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

- An implicant of function $f$ is a term $t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k$ such that

$$\forall x_1, \ldots, x_n \, t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1$$

- 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;

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

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$z$</th>
<th>$f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Any function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) is called a *Boolean function*;

An *implicant* of function \( f \) is a term \( t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k \) such that

\[
\forall x_1, \ldots, x_n \ t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1
\]

*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;

\[
\begin{align*}
\phi_1 &= xy\overline{z} + x\overline{y}z + \overline{x}yz + xyz \\
\phi_2 &= (x + y + z)(\overline{x} + y + z)(x + \overline{y} + z)(x + y + \overline{z}) \\
\phi_3 &= xy + xz + yz \\
x\overline{y} & \text{ is an implicant}
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Boolean function

- Any function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) is called a **Boolean function**;
- An *implicant* of function \( f \) is a term \( t = x_1 \ldots x_m \overline{y}_1 \ldots \overline{y}_k \) such that
  \[ \forall x_1, \ldots, x_n t(x_1, \ldots, x_n) = 1 \Rightarrow f(x_1, \ldots, x_n) = 1 \]
- **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
Theorem (Blake Canonical Form)

A Boolean function can be represented as a disjunction of all of its prime implicants: 

\[ f = t_1 + t_2 + \ldots + t_k \]
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 then 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:

\[ A \rightarrow \overline{B} \land C \iff A(B + \overline{C}) = 0 \]
\[ \ldots \iff BD(AC + \overline{AC}) = 0 \]
\[ \ldots \iff \overline{BC}(A + \overline{D}) = 0 \]
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:

\[
A \rightarrow \overline{B} \land C \quad \iff \quad A(B + \overline{C}) = 0
\]

\[
\ldots \iff \quad BD(AC + \overline{AC}) = 0
\]

\[
\ldots \iff \quad BC(A + \overline{D}) = 0
\]

After reduction:

\[
f = A(B + \overline{C}) + BD(AC + \overline{AC}) + BC(A + \overline{D}) = 0
\]
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:

\[ A \rightarrow \overline{B} \land C \iff A(B + \overline{C}) = 0 \]
\[ \quad \ldots \iff BD(AC + \overline{AC}) = 0 \]
\[ \quad \ldots \iff BC(A + \overline{D}) = 0 \]

After reduction:
\[ f = A(B + \overline{C}) + BD(AC + \overline{AC}) + BC(A + \overline{D}) = 0 \]

Blake Canonical form:
\[ f = B\overline{C}D + \overline{B}CD + A = 0 \]

Reasoning:
(theorem proving)
e.g., show that “C cannot go alone.”
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:

\[ A \rightarrow \overline{B} \land C \iff A(B + \overline{C}) = 0 \]
\[ \ldots \iff BD(AC + \overline{AC}) = 0 \]
\[ \ldots \iff BC'(A + \overline{D}) = 0 \]

After reduction:

\[ f = A(B + \overline{C}) + BD(AC + \overline{AC}) + BC(A + \overline{D}) = 0 \]

Blake Canonical form:

\[ f = BCD + BCD + A = 0 \]

Facts:

\[ BD \rightarrow C \]
\[ C \rightarrow B + D \]
\[ A \rightarrow 0 \]
**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:**

<table>
<thead>
<tr>
<th>Boolean Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow \overline{B} \land C$</td>
<td>$A(B + \overline{C}) = 0$</td>
</tr>
<tr>
<td>$\ldots \leftrightarrow BD(AC + \overline{AC}) = 0$</td>
<td></td>
</tr>
<tr>
<td>$\ldots \leftrightarrow BC(A + \overline{D}) = 0$</td>
<td></td>
</tr>
</tbody>
</table>

**After reduction:**

$$f = A(B + \overline{C}) + BD(AC + \overline{AC}) + BC(A + \overline{D}) = 0$$

**Blake Canonical form:**

$$f = B\overline{C}D + \overline{B}C\overline{D} + A = 0$$

**Facts:**

- $BD \rightarrow C$
- $C \rightarrow B + D$
- $A \rightarrow 0$

**Reasoning:** (theorem proving)

- e.g., show that “C cannot go alone.”
Boolean reasoning for decision problems

Planing (scheduling) problem $\mathbf{P}$

encoding

boolean function $f_P$

heuristics

SAT or MAXSAT for $f_P$

decoding

solution for $\mathbf{P}$

- SAT: whether an equation
  \[ f(x_1, \ldots, x_n) = 1 \]
  has a solution?
Boolean reasoning for decision problems

Planing (scheduling) problem $P$

- encoding
- boolean function $f_P$
- heuristics
- SAT or MAXSAT for $f_P$
- decoding
- solution for $P$

- SAT: whether an equation
  
  $f(x_1, \ldots, x_n) = 1$

  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

Planing (scheduling) problem \( P \)

- encoding
- boolean function \( f_P \)
- heuristics
- SAT or MAXSAT for \( f_P \)
- decoding
- solution for \( P \)

- SAT: whether an equation
  \[ f(x_1, \ldots, x_n) = 1 \]
  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.
A function $\phi : \{0, 1\}^n \rightarrow \{0, 1\}$ is "monotone" if

$$\forall x, y \in \{0, 1\}^n (x \leq y) \Rightarrow (\phi(x) \leq \phi(y))$$
A function \( \phi : \{0, 1\}^n \rightarrow \{0, 1\} \) is "monotone" if

\[
\forall x, y \in \{0, 1\}^n (x \leq y) \Rightarrow (\phi(x) \leq \phi(y))
\]

Monotone functions can be represented by a boolean expression without negations.
A function $\phi : \{0, 1\}^n \rightarrow \{0, 1\}$ is "monotone" if

$$\forall x, y \in \{0, 1\}^n (x \leq y) \Rightarrow (\phi(x) \leq \phi(y))$$

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 apply 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;
Outline

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
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.
Introduction to the Special Theme

Modern Machine Learning: More with Less, Cheaper and Better

by Sander Bohte and Hung Son Nguyen

While the discipline of machine learning is often conflated with the general field of AI, machine learning specifically is concerned with the question of how to program computers to automatically recognise complex patterns and make intelligent decisions based on data. This includes such diverse approaches as probability theory, logic, combinatorial optimisation, search, statistics, reinforcement learning and control theory. In this day and age with an abundance of sensors and computers, applications are ubiquitous, ranging from vision to language processing, forecasting, pattern recognition, games, data mining, expert systems and robotics.
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 $\mathcal{S} = (U, A)$ and a monotone evaluation function
  \[ \mu_{\mathcal{S}} : \mathcal{P}(A) \rightarrow \mathbb{R}^+ \]
  The set $B \subset A$ is called \( \mu \)-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 ...
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

<table>
<thead>
<tr>
<th>ID</th>
<th>outlook</th>
<th>temp.</th>
<th>hum.</th>
<th>windy</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>FALSE</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>?</td>
</tr>
</tbody>
</table>
Example: Decision reduct

<table>
<thead>
<tr>
<th>A</th>
<th>a₁</th>
<th>a₂</th>
<th>a₃</th>
<th>a₄</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>outlook</td>
<td>temp.</td>
<td>hum.</td>
<td>windy</td>
<td>play</td>
</tr>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>FALSE</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>?</td>
</tr>
</tbody>
</table>

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants \(\rightarrow\) reducts
### Example: Decision reduct

<table>
<thead>
<tr>
<th>ID</th>
<th>outlook</th>
<th>temp.</th>
<th>hum.</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>FALSE</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>?</td>
</tr>
</tbody>
</table>

### Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants \(\Rightarrow\) reducts
Example: Decision reduct

<table>
<thead>
<tr>
<th>A</th>
<th>(a_1)</th>
<th>(a_2)</th>
<th>(a_3)</th>
<th>(a_4)</th>
<th>(\text{dec})</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>outlook</td>
<td>temp.</td>
<td>hum.</td>
<td>windy</td>
<td>play</td>
</tr>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>FALSE</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>?</td>
</tr>
</tbody>
</table>

Discernibility matrix:

\[ M \]

<table>
<thead>
<tr>
<th>(M)</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>(a_1, a_4)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_2)</td>
<td>(a_1, a_2)</td>
</tr>
<tr>
<td>4</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_2, a_3)</td>
<td>(a_1)</td>
</tr>
<tr>
<td></td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>5</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_4)</td>
<td>(a_1, a_2)</td>
</tr>
<tr>
<td></td>
<td>(a_3)</td>
<td>(a_3, a_4)</td>
<td>(a_3)</td>
<td>(a_3)</td>
</tr>
<tr>
<td>7</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_1)</td>
<td>(a_1, a_2)</td>
</tr>
<tr>
<td></td>
<td>(a_3, a_4)</td>
<td>(a_3)</td>
<td>(a_3, a_4)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td>9</td>
<td>(a_2, a_3)</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_4)</td>
<td>(a_2, a_3)</td>
</tr>
<tr>
<td></td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>10</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_2, a_4)</td>
<td>(a_1, a_3)</td>
</tr>
<tr>
<td></td>
<td>(a_3)</td>
<td>(a_3, a_4)</td>
<td>(a_3, a_4)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td>11</td>
<td>(a_2, a_3)</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td></td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>12</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_4)</td>
</tr>
<tr>
<td></td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
<td>(a_4)</td>
</tr>
</tbody>
</table>

Methodology

1. Discernibility matrix;
2. Discernibility Boolean function
3. Prime implicants \[\rightarrow\] reducts
Example: Decision reduct

\[ f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \lor \alpha_2 + \alpha_3 + \alpha_4) \]
\[ (\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3) \]
\[ (\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4) \]

<table>
<thead>
<tr>
<th>M</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>(a_1)</td>
<td>(a_1, a_4)</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2, a_3, a_4)</td>
</tr>
<tr>
<td>4</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_2)</td>
</tr>
<tr>
<td>5</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_4)</td>
<td>(a_1, a_2, a_3, a_4)</td>
</tr>
<tr>
<td>7</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
<td>(a_3, a_3)</td>
</tr>
<tr>
<td>9</td>
<td>(a_2, a_3)</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_4)</td>
<td>(a_2, a_3)</td>
</tr>
<tr>
<td>10</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_2, a_4)</td>
<td>(a_1, a_3)</td>
</tr>
<tr>
<td>11</td>
<td>(a_2, a_3)</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td>12</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_4)</td>
<td>(a_3)</td>
</tr>
</tbody>
</table>
Example: Decision reduct

\[
f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \lor \alpha_2 + \alpha_3 + \alpha_4)
\]
\[
\quad = (\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3)
\]
\[
\quad = (\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)
\]

simplifying the function by *absorption* law (i.e. \( p \land (p + q) \equiv p \)):

\[
f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3)
\]
Example: Decision reduct

\[ f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \lor \alpha_2 + \alpha_3 + \alpha_4) \]
\[ (\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3) \]
\[ (\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4) \]

- Simplifying the function by absorption law (i.e. \( p \land (p + q) \equiv p \)):

\[ f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3) \]

- Transformation from CNF to DNF: \( f = \alpha_1 \alpha_4 \alpha_2 + \alpha_1 \alpha_4 \alpha_3 \)
Example: Decision reduct

<table>
<thead>
<tr>
<th>M</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>(a_1)</td>
<td>(a_1, a_4)</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2, a_3, a_4)</td>
</tr>
<tr>
<td>4</td>
<td>(a_1, a_2)</td>
<td>(a_1, a_2, a_3, a_1)</td>
<td>(a_4)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>5</td>
<td>(a_1, a_2, a_4)</td>
<td>(a_1, a_2, a_3, a_4)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td>7</td>
<td>(a_2, a_3)</td>
<td>(a_1, a_2, a_3, a_1)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
</tr>
<tr>
<td>9</td>
<td>(a_1, a_2, a_1, a_2)</td>
<td>(a_1, a_2, a_3, a_4)</td>
<td>(a_1, a_3)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>11</td>
<td>(a_2, a_3, a_4)</td>
<td>(a_1, a_2)</td>
<td>(a_3, a_4)</td>
<td>(a_4)</td>
</tr>
<tr>
<td>12</td>
<td>(a_1, a_2, a_1, a_2)</td>
<td>(a_1, a_2, a_1, a_4)</td>
<td>(a_1, a_2, a_3, a_4)</td>
<td>(a_4, a_3)</td>
</tr>
</tbody>
</table>

\[ f = (\alpha_1)(\alpha_1 + \alpha_4)(\alpha_1 + \alpha_2)(\alpha_1 \lor \alpha_2 + \alpha_3 + \alpha_4) \]
\[ (\alpha_1 + \alpha_2 + \alpha_4)(\alpha_2 + \alpha_3 + \alpha_4)(\alpha_1 + \alpha_2 + \alpha_3) \]
\[ (\alpha_4)(\alpha_2 + \alpha_3)(\alpha_2 + \alpha_4)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4) \]

- Simplifying the function by *absorption* law (i.e. \( p \land (p + q) \equiv p \)):

\[ f = (\alpha_1)(\alpha_4)(\alpha_2 + \alpha_3) \]

- 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

1 Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
Boolean reasoning approach

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

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
Example: Decision Rule Extraction

<table>
<thead>
<tr>
<th>M</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$a_1$</td>
<td>$a_1, a_4$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_1, a_2$</td>
</tr>
<tr>
<td>4</td>
<td>$a_1, a_2$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_1$</td>
</tr>
<tr>
<td>5</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_4$</td>
<td>$a_1, a_2, a_3$</td>
</tr>
<tr>
<td>7</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1$</td>
<td>$a_1, a_2, a_3, a_4$</td>
</tr>
<tr>
<td>9</td>
<td>$a_2, a_3$</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_1, a_4$</td>
<td>$a_2, a_3$</td>
</tr>
<tr>
<td>10</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_2, a_4$</td>
<td>$a_1, a_3$</td>
</tr>
<tr>
<td>11</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_2, a_3$</td>
<td>$a_1, a_2$</td>
<td>$a_3, a_4$</td>
</tr>
<tr>
<td>12</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_1, a_2$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_4$</td>
</tr>
</tbody>
</table>

\[ f_{u_3} = (\alpha_1)(\alpha_1 \lor \alpha_4)(\alpha_1 \lor \alpha_2 \lor \alpha_3 \lor \alpha_4)(\alpha_1 \lor \alpha_2) = \alpha_1 \]

Decision rule:

\[(a_1 = \text{overcast}) \implies dec = \text{no} \]
Example: Decision Rule Extraction

<table>
<thead>
<tr>
<th>M</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$a_1$</td>
<td>$a_1, a_4$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_1, a_2$</td>
</tr>
<tr>
<td>4</td>
<td>$a_1, a_2$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_1$</td>
</tr>
<tr>
<td>5</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_4$</td>
<td>$a_1, a_2, a_3$</td>
</tr>
<tr>
<td>7</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1$</td>
<td>$a_1, a_2, a_3, a_4$</td>
</tr>
<tr>
<td>9</td>
<td>$a_2, a_3$</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_1, a_4$</td>
<td>$a_2, a_3$</td>
</tr>
<tr>
<td>10</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_2, a_4$</td>
<td>$a_1, a_3$</td>
</tr>
<tr>
<td>11</td>
<td>$a_2, a_3, a_4$</td>
<td>$a_2, a_3$</td>
<td>$a_1, a_2$</td>
<td>$a_3, a_4$</td>
</tr>
<tr>
<td>12</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_1, a_2$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_4$</td>
</tr>
</tbody>
</table>

$$f_{u_8} = (\alpha_1 + \alpha_2)(\alpha_1)(\alpha_1 + \alpha_2 + \alpha_3)(\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4)(\alpha_2 + \alpha_3)(\alpha_1 + \alpha_3)(\alpha_3 + \alpha_4)(\alpha_1 + \alpha_4)$$

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

Decision rules:

- $(a_1 = \text{sunny}) \land (a_3 = \text{high}) \implies dec = \text{no}$
- $(a_1 = \text{sunny}) \land (a_2 = \text{mild}) \land (a_4 = \text{FALSE}) \implies dec = \text{no}$
Example: all consistent decision rules

<table>
<thead>
<tr>
<th>Rid</th>
<th>Condition</th>
<th>⇒Decision</th>
<th>supp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>outlook(overcast)⇒ yes</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>humidity(normal) AND windy(FALSE)⇒ yes</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>outlook(sunny) AND humidity(high)⇒ no</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>outlook(rainy) AND windy(FALSE)⇒ yes</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>outlook(sunny) AND temp.(hot)⇒ no</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>outlook(rainy) AND windy(TRUE)⇒ no</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>outlook(sunny) AND humidity(normal)⇒ yes</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>temp.(cool) AND windy(FALSE)⇒ yes</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>temp.(mild) AND humidity(normal)⇒ yes</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>temp.(hot) AND windy(TRUE)⇒ no</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>outlook(sunny) AND temp.(mild) AND windy(FALSE)⇒ no</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>outlook(sunny) AND temp.(cool)⇒ yes</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>outlook(sunny) AND temp.(mild) AND windy(TRUE)⇒ yes</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>temp.(hot) AND humidity(normal)⇒ yes</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Outline

1 Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
Discretization problem

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

$$U = \{x_1, \ldots, x_n\}; \quad A = \{a_1, \ldots, a_k : U \rightarrow \mathbb{R}\} \quad \text{and} \quad d : U \rightarrow \{1, \ldots, r(d)\}$$

<table>
<thead>
<tr>
<th></th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>0</td>
</tr>
<tr>
<td>$u_2$</td>
<td>2.0</td>
<td>5.0</td>
<td>5.0</td>
<td>1</td>
</tr>
<tr>
<td>$u_3$</td>
<td>3.0</td>
<td>7.0</td>
<td>1.0</td>
<td>2</td>
</tr>
<tr>
<td>$u_4$</td>
<td>3.0</td>
<td>6.0</td>
<td>1.0</td>
<td>1</td>
</tr>
<tr>
<td>$u_5$</td>
<td>4.0</td>
<td>6.0</td>
<td>3.0</td>
<td>0</td>
</tr>
<tr>
<td>$u_6$</td>
<td>5.0</td>
<td>6.0</td>
<td>5.0</td>
<td>1</td>
</tr>
<tr>
<td>$u_7$</td>
<td>6.0</td>
<td>1.0</td>
<td>8.0</td>
<td>2</td>
</tr>
<tr>
<td>$u_8$</td>
<td>7.0</td>
<td>8.0</td>
<td>8.0</td>
<td>2</td>
</tr>
<tr>
<td>$u_9$</td>
<td>7.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0</td>
</tr>
<tr>
<td>$u_{10}$</td>
<td>8.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0</td>
</tr>
</tbody>
</table>
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 \(C\) is consistent with \(S\) (or \(S\)-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:
  \[
  \text{IF } x, y \text{ are discernible by } S \text{ THEN } x, y \text{ are discernible by } C.
  \]

- The consistent set of cuts \(C\) is called \textit{irreducible} iff \(Q\) is not consistent for any proper subset \(Q \subset C\).

- The consistent set of cuts \(C\) is called \textit{optimal} iff
  \[
  card(C) \leq card(Q)
  \]
  for any consistent set of cuts \(Q\).
Discretization problem

**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

<table>
<thead>
<tr>
<th>$S$</th>
<th>$a$</th>
<th>$b$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>0.8</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$u_2$</td>
<td>1</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>$u_3$</td>
<td>1.3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>$u_4$</td>
<td>1.4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$u_5$</td>
<td>1.4</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$u_6$</td>
<td>1.6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>$u_7$</td>
<td>1.3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

$C = \{(a; 0.9), (a; 1.5), (b; 0.75), (b; 1.5)\}$
The discernibility formulas $\psi_{i,j}$ for different pairs $(u_i, u_j)$ of objects:

\[
\begin{align*}
\psi_{2,1} &= p_1^a + p_1^b + p_2^b; \\
\psi_{2,6} &= p_2^a + p_3^a + p_4^a + p_1^b + p_2^b + p_3^b; \\
\psi_{3,1} &= p_1^a + p_2^a + p_3^b; \\
\psi_{3,6} &= p_3^a + p_4^a; \\
\psi_{5,1} &= p_1^a + p_2^a + p_3^a; \\
\psi_{5,6} &= p_4^a + p_3^b; \\
\psi_{2,4} &= p_2^a + p_3^a + p_1^b; \\
\psi_{2,7} &= p_2^a + p_1^b; \\
\psi_{3,4} &= p_2^a + p_2^b + p_3^b; \\
\psi_{3,7} &= p_2^b + p_3^b; \\
\psi_{5,4} &= p_2^b; \\
\psi_{5,7} &= p_3^a + p_2^b.
\end{align*}
\]

The discernibility formula $\Phi_S$ in $CNF$ form is given by

\[
\Phi_S = \left( p_1^a + p_1^b + p_2^b \right) \left( p_1^a + p_2^a + p_3^b \right) \left( p_1^a + p_2^a + p_3^a \right) \left( p_2^a + p_3^a + p_1^b \right) p_2^b \left( p_2^a + p_2^b + p_3^b \right) \left( p_2^a + p_2^a + p_4^a + p_1^b + p_2^b + p_3^b \right) \left( p_3^a + p_4^a \right) \left( p_4^a + p_3^b \right) \left( p_2^a + p_1^b \right) \left( p_2^b + p_3^b \right) \left( p_3^a + p_2^b \right).
\]

Transforming the formula $\Phi_S$ into its $DNF$ form we obtain four prime implicants:

\[
\Phi_S = p_2^a p_2^a p_4^b p_2^b + p_2^a p_3^a p_2^b p_3^b + p_3^a p_1^b p_3^a p_2^b + p_1^a p_4^a p_1^b p_2^b.
\]
Discretization by reduct calculation

<table>
<thead>
<tr>
<th>$S^*$</th>
<th>$p_1^a$</th>
<th>$p_2^a$</th>
<th>$p_3^a$</th>
<th>$p_4^a$</th>
<th>$p_1^b$</th>
<th>$p_2^b$</th>
<th>$p_3^b$</th>
<th>$d^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(u_1, u_2)$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_1, u_3)$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(u_1, u_5)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_4, u_2)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_4, u_3)$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(u_4, u_5)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_6, u_2)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(u_6, u_3)$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_6, u_5)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(u_7, u_2)$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$(u_7, u_3)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$(u_7, u_5)$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>new</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Outline

1 Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
## Information systems and Decision tables

<table>
<thead>
<tr>
<th></th>
<th>Diploma</th>
<th>Experience</th>
<th>French</th>
<th>Reference</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>MBA</td>
<td>Medium</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_2$</td>
<td>MBA</td>
<td>Low</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_3$</td>
<td>MCE</td>
<td>Low</td>
<td>Yes</td>
<td>Good</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_4$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Neutral</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_5$</td>
<td>MSc</td>
<td>Medium</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_6$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_7$</td>
<td>MBA</td>
<td>High</td>
<td>No</td>
<td>Good</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_8$</td>
<td>MCE</td>
<td>Low</td>
<td>No</td>
<td>Excellent</td>
<td>Reject</td>
</tr>
</tbody>
</table>

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

- For any $B \subseteq A$:

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

$IND(B)$ is an 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:

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

$$\forall u \in U \quad [u]_{P \cup Q} = [u]_P \cap [u]_Q \quad (4)$$
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 $S = (U, A)$ and a monotone evaluation function
  \[ \mu_S : \mathcal{P}(A) \rightarrow \mathbb{R}^+ \]
- The set $B \subset A$ is called $\mu$-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)

\[ \mu\text{-RED} = \text{set of all } \mu\text{-reducts; } \mu\text{-CORE} = \bigcap_{B \in \mu\text{-RED}} B \]
Positive Region Based Reducts

- For any \( B \subseteq A \) and \( X \subseteq U \):
  
  \[
  \overline{B}(X) = \{ u : [u]_B \subseteq X \}; \quad \overline{B}(X) = \{ u : [u]_B \cap X \neq \emptyset \}
  \]

- Let \( S = (U, A \cup \{ dec \}) \) be a decision table, let \( B \subseteq A \), and let \( U|_{dec} = \{X_1, \ldots, X_k\} \):
  
  \[
  POS_B(dec) = \bigcup_{i=1}^{k} B(X_i)
  \]

- 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 \( \mu \)-reduct, where \( \mu(B) = |POS_B(dec)| \)
Reducts

• Indiscernibility relation

\[(x, y) \in \text{IND}(B) \iff \forall a \in A a(x) = a(y)\]
\[(x, y) \in \text{IND}_{\text{dec}}(B) \iff \text{dec}(x) = \text{dec}(y) \lor \forall a \in A a(x) = a(y)\]

• A decision-relative reduct is a minimal set of attributes \( R \subseteq A \) such that \( \text{IND}_{\text{dec}}(R) = \text{IND}_{\text{dec}}(A) \).

• The set of all reducts is denoted by:

\[\mathcal{RED}(D) = \{ R \subseteq A : R \text{ is a reduct of } D \}\]
Outline

1 Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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 importance of attributes

\[ \text{RED}(D) = \{ R \subseteq A : R \text{ is a reduct of } D \} \]

- Core attributes:

\[ \text{CORE}(D) = \bigcap_{R \in \text{RED}(D)} R \]

- An attribute \( a \in A \) is called **reduct attribute** if it occurs in at least one of reducts

\[ \text{REAT}(D) = \bigcup_{R \in \text{RED}(D)} R \]

- The attribute is called **redundant attribute** if it is not a reductive attribute.

- An attribute \( b \) is redundant \( \iff b \in A - \text{REAT} \)
The problem setting

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

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

The problem

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

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

and

$$\text{REAT}(\mathbb{D}) = \bigcup_{R \in \mathcal{RED}(\mathbb{D})} R$$
Example

<table>
<thead>
<tr>
<th></th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>MBA</td>
<td>Medium</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_2$</td>
<td>MBA</td>
<td>Low</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_3$</td>
<td>MCE</td>
<td>Low</td>
<td>Yes</td>
<td>Good</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_4$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Neutral</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_5$</td>
<td>MSc</td>
<td>Medium</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_6$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_7$</td>
<td>MBA</td>
<td>High</td>
<td>No</td>
<td>Good</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_8$</td>
<td>MCE</td>
<td>Low</td>
<td>No</td>
<td>Excellent</td>
<td>Reject</td>
</tr>
</tbody>
</table>

In this example:

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

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

- the redundant attribute: $a_3$
Outline

1 Introduction
   • Rough Set Approach to Machine Learning and Data Mining
   • Boolean Reasoning Methodology
   • Reducts

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
## Discernibility matrix

<table>
<thead>
<tr>
<th></th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>MBA</td>
<td>Medium</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_2$</td>
<td>MBA</td>
<td>Low</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_3$</td>
<td>MCE</td>
<td>Low</td>
<td>Yes</td>
<td>Good</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_4$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Neutral</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_5$</td>
<td>MSc</td>
<td>Medium</td>
<td>Yes</td>
<td>Neutral</td>
<td>Reject</td>
</tr>
<tr>
<td>$x_6$</td>
<td>MSc</td>
<td>High</td>
<td>Yes</td>
<td>Excellent</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_7$</td>
<td>MBA</td>
<td>High</td>
<td>No</td>
<td>Good</td>
<td>Accept</td>
</tr>
<tr>
<td>$x_8$</td>
<td>MCE</td>
<td>Low</td>
<td>No</td>
<td>Excellent</td>
<td>Reject</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_4$</th>
<th>$x_6$</th>
<th>$x_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_2$</td>
<td>$a_2, a_4$</td>
<td>$a_1, a_2$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_2, a_3, a_4$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_1, a_2, a_4$</td>
<td>$a_1, a_2, a_3$</td>
</tr>
<tr>
<td>$x_5$</td>
<td>$a_1, a_4$</td>
<td>$a_2$</td>
<td>$a_2, a_4$</td>
<td>$a_1, a_2, a_3, a_4$</td>
</tr>
<tr>
<td>$x_8$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_3, a_4$</td>
<td>$a_1, a_2, a_3$</td>
<td>$a_1, a_2, a_4$</td>
</tr>
</tbody>
</table>
Boolean approach to reduct problem

- Boolean discernibility function:

\[ \Delta_D(a_1, ..., a_4) = (a_2 + a_4)(a_1 + a_2)(a_1 + a_2 + a_4)(a_2 + a_3 + a_4) \]
\[ (a_1 + a_2 + a_4)(a_1 + a_2 + a_4)(a_1 + a_2 + a_4)(a_1 + a_2 + a_3) \]
\[ (a_1 + a_4)(a_2 + a_4)(a_1 + a_2 + a_3 + a_4)(a_1 + a_2 + a_3) \]
\[ (a_1 + a_2 + a_3 + a_4)(a_1 + a_2 + a_3)(a_1 + a_2 + a_4) \]

- In general: \( R = \{a_{i_1}, ..., a_{i_j}\} \) is a reduct in \( D \) \( \iff \) the monomial

\[ m_R = a_{i_1} \cdot \ldots \cdot a_{i_j} \]

is a prime implicant of \( \Delta_D(a_1, ..., a_k) \)

**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:
  \[ x + (x \cdot y) = x \quad \text{and} \quad x \cdot (x + y) = x \]

- In our example: irreducible CNF of the discernibility function is as follows:
  \[ \Delta_D(a_1, \ldots, a_4) = a_2 \cdot (a_1 + a_4) \]

- 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

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

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

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

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

1 Introduction
- Rough Set Approach to Machine Learning and Data Mining
- Boolean Reasoning Methodology
- Reducts

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
### Complexity of encoding functions

Given a decision table with $n$ objects and $m$ attributes

<table>
<thead>
<tr>
<th>Problem</th>
<th>Nr of variables</th>
<th>Nr of clauses</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimal reduct</td>
<td>$O(m)$</td>
<td>$O(n^2)$</td>
</tr>
<tr>
<td>decision rules</td>
<td>$O(n)$ functions</td>
<td>$O(m)$</td>
</tr>
<tr>
<td>discretization</td>
<td>$O(mn)$</td>
<td>$O(n^2)$</td>
</tr>
<tr>
<td>grouping</td>
<td>$O(\sum_{a \in A} 2^{</td>
<td>V_a</td>
</tr>
<tr>
<td>hyperplanes</td>
<td>$O(n^m)$</td>
<td>$O(n^2)$</td>
</tr>
</tbody>
</table>

**Greedy algorithm:**

time complexity of searching for the best variable:

$$O(\#\text{variables} \times \#\text{clauses})$$
The iterative and interactive process of discovering non-trivial, implicit, previously unknown and potentially useful (interesting) information or patterns from large databases.


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

Optimization problem $\Pi$

Boolean function $f_\Pi$

Prime implicants of $f_\Pi$

Approximate function $f'_\Pi$

Prime implicants of $f'_\Pi$

Approximate solution for $\Pi$
Example: Decision reduct

<table>
<thead>
<tr>
<th>A</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>outlook</td>
<td>temp.</td>
<td>hum.</td>
<td>windy</td>
<td>play</td>
</tr>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
</tbody>
</table>

Discernibility matrix:

<table>
<thead>
<tr>
<th>M</th>
<th>1</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>$a_1$</td>
<td>$a_1$, $a_4$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_2$</td>
</tr>
<tr>
<td>4</td>
<td>$a_1$, $a_2$</td>
<td>$a_1$, $a_2$, $a_4$</td>
<td>$a_2$, $a_3$, $a_4$</td>
<td>$a_1$</td>
</tr>
<tr>
<td>5</td>
<td>$a_1$, $a_2$, $a_3$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_4$</td>
<td>$a_1$, $a_2$, $a_3$</td>
</tr>
<tr>
<td>6</td>
<td>$a_2$, $a_3$</td>
<td>$a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_4$</td>
<td>$a_2$, $a_3$</td>
</tr>
<tr>
<td>7</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_1$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
</tr>
<tr>
<td>8</td>
<td>$a_1$, $a_2$, $a_3$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_2$, $a_4$</td>
<td>$a_1$, $a_3$</td>
</tr>
<tr>
<td>9</td>
<td>$a_2$, $a_3$, $a_4$</td>
<td>$a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_2$</td>
<td>$a_3$, $a_4$</td>
</tr>
<tr>
<td>10</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_2$, $a_3$, $a_4$</td>
<td>$a_1$, $a_2$, $a_3$</td>
<td>$a_1$, $a_4$</td>
</tr>
</tbody>
</table>

Methodology

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

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

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

\[
\begin{align*}
\text{eval}(a_1) &= \text{disc}_{dec}(a_1) = 23 \\
\text{eval}(a_2) &= \text{disc}_{dec}(a_2) = 23 \\
\text{eval}(a_3) &= \text{disc}_{dec}(a_3) = 18 \\
\text{eval}(a_4) &= \text{disc}_{dec}(a_4) = 16
\end{align*}
\]

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:

\[
\begin{align*}
\text{eval}(a_2) &= \text{disc}_{dec}(a_1, a_2) - \text{disc}_{dec}(a_1) = 7 \\
\text{eval}(a_3) &= \text{disc}_{dec}(a_1, a_3) - \text{disc}_{dec}(a_1) = 7 \\
\text{eval}(a_4) &= \text{disc}_{dec}(a_1, a_4) - \text{disc}_{dec}(a_1) = 6
\end{align*}
\]

- If this time we select \( a_2 \), then there 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

optimization problem $\Pi$

boolean function $f_\Pi$

$\rightarrow$

approximate function $f'_\Pi$

prime implicants of $f_\Pi$

prime implicants of $f'_\Pi$

$\rightarrow$

approximate solution for $\Pi$
MD heuristics for reducts without discernibility matrix?

<table>
<thead>
<tr>
<th>A</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$a_4$</th>
<th>dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>outlook</td>
<td>temp. hum.</td>
<td>windy</td>
<td>play</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>TRUE</td>
<td>no</td>
</tr>
<tr>
<td>8</td>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>9</td>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>FALSE</td>
<td>no</td>
</tr>
<tr>
<td>10</td>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>FALSE</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>FALSE</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>TRUE</td>
<td>?</td>
</tr>
</tbody>
</table>

**Contingence table for $a_1$:**

<table>
<thead>
<tr>
<th>$a_1$</th>
<th>dec = no</th>
<th>dec = yes</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>3</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>overcast</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>rainy</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>total</td>
<td>4</td>
<td>8</td>
<td>12</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>$(a_1, a_2)$</th>
<th>no</th>
<th>yes</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny, hot</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>sunny, mild</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>sunny, cool</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>overcast</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>rainy, mild</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>rainy, cool</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>total</td>
<td>4</td>
<td>8</td>
<td>12</td>
</tr>
</tbody>
</table>

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

1. Number of occurrences of attributes in $M$;
2. Number of occurrences of a set of attributes in $M$;
Discernibility measure for discretization

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

\[
W(c) = \text{conflict}(U) - \text{conflict}(U_L) - \text{conflict}(U_R)
\]

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

1. Introduction
   - Rough Set Approach to Machine Learning and Data Mining
   - Boolean Reasoning Methodology
   - Reducts

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
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.
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 $\mathcal{S}$ 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 $N(m)$. 