Chapter 15

Rough Set Approach to Machine Learning

15.1 Introduction

This chapter is primarily devoted to a rough set methodology for *supervised machine learning*.

In supervised machine learning paradigm, a learning algorithm is given a training data set, usually in the form of a decision system $\mathcal{A} = \langle U, A, d \rangle$, prepared by an expert. Observe that each decision system classifies elements from U into decision classes. The purpose of the algorithm is to return a set of decision rules, called a classifier, which allow to classify unseen objects, i.e., objects that are not described in the original decision table.

In this chapter, we provide a number of rough set methods that can be used in constructing classifiers. The process of such construction is presented in Chapter 16.

Most of the techniques discussed below are based on computing prime implicants.[1] Accordingly, they are computationally hard. However, many heuristics have been developed which turned out to be very promising. The results of experiments on many data sets, reported in the literature (see Section 15.9) show a very good quality of classification of unseen objects using these heuristics.

The rest of this chapter is organized as follows. In Section 15.2, we discuss a concept of a reduct which play an important role in rough set approach to machine learning. Sections 15.3 and 15.4 are devoted to attribute selec-

[1] See Definition2.5.3.

tion and attribute extraction processes. The former, allows to reduce the number of attributes in the training decision table. The latter, allows to decrease the size of value sets of attributes. In Section 15.4.2, we discuss a certain heuristics which could be applied in the extraction process. The heuristics is specially useful if the considered decision table is stored in a relational database. In Section 15.6, we show how minimal decision rules can be obtained from a decision table and, in Section 15.7, we present how these rules can be used to learn concepts. The final section is devoted to association rules. These are the rules providing associations among attributes in an information system. Association rules play an important role in data mining.[2]

15.2 Reducts in Information and Decision Systems

The crucial notion in rough set approach to learning is that of a reduct. In fact, the term "reduct" corresponds to a wide class of concepts. What typifies all of them is that they allow to reduce an original information (decision) system by removing redundant attributes. In this section we discuss three kinds of reducts which play an important role in the rest of this chapter.

Definition 15.2.1 Given an information system $\mathcal{A} = \langle U, A\rangle$, a *reduct* is a minimal set (wrt inclusion) of attributes $B \subseteq A$ such that $IND_{\mathcal{A}}(B) = IND_{\mathcal{A}}(A)$. The intersection of all reducts is called *core*. ∎

Intuitively, a reduct is a minimal set of attributes from A that preserves the original classification defined by A. Finding a minimal reduct is NP-hard. One can also show that, for any m, there exists an information system with m attributes having an exponential (wrt m) number of reducts. Fortunately, there are good heuristics allowing one to compute sufficiently many reducts in an acceptable time.

To provide a general method for computing reducts, we need the following notions.

Definition 15.2.2 Let $\mathcal{A}$ be an information system with n objects. The *discernibility matrix* of $\mathcal{A}$ is a $n \times n$ matrix with elements c_{ij} consisting of the set of attributes upon which objects x_i and x_j differ, i.e.,

$$c_{ij} = \{a \in A \mid a(x_i) \neq a(x_j)\}, \text{ for } i, j = 1, ..., n. \tag{15.1}$$

[2]Association rules, slightly different than those presented here, may be used in the process of classifier construction, but this topic will no be discussed in this chapter.

Object	*Speed*	*Color*	*Humidity*
car1	medium	green	large
car2	medium	yellow	small
car3	large	blue	large

Table 15.1: Information table considered in Example 15.2.4.

A *discernibility function* $f_{\mathcal{A}}$ for $\mathcal{A}$ is a propositional formula of m Boolean variables, $a_1^*, ..., a_m^*$, corresponding to the attributes $a_1, ..., a_m$, defined by

$$f_{\mathcal{A}}(a_1^*, ..., a_m^*) = \bigwedge_{1 \leq j < i \leq n} \bigvee_{c \in c_{ij}^*} c, \tag{15.2}$$

where $c_{ij}^* = \{a^* \mid a \in c_{ij}\}$. In the sequel, we write a_i instead of a_i^*. ■

The discernibility function $f_{\mathcal{A}}$ describes constrains which should hold if one would like to preserve discernibility between all pairs of discernible objects from $\mathcal{A}$. It requires to keep at least one attribute from each non-empty element of the discernibility matrix corresponding to any pair of discernible objects.

Theorem 15.2.3 Let $\mathcal{A} = \langle U, A \rangle$ be an information system. The set of all prime implicants of $f_{\mathcal{A}}$ determines the set of all reducts of $\mathcal{A}$. ■

Example 15.2.4 Consider an information system $\mathcal{A}$ given in Table 15.1.

The discernibility matrix of $\mathcal{A}$ is presented in Table 15.2. (The letters s, c and h stand for *Speed*, *Color* and *Humidity*, respectively.)

$\mathcal{M}(\mathcal{A})$	car1	car2	car3
car1		c,h	s,c
car2	c,h		s,c,h
car3	s,c	s,c,h	

Table 15.2: The discernibility matrix corresponding to information system shown in Table 15.1.

The discernibility function for the system $\mathcal{A}$ is thus given by

$$\begin{aligned} f_{\mathcal{A}}(s,c,h) &\equiv (c \vee h) \wedge (s \vee c) \wedge (s \vee c \vee h) \\ &\equiv (c \vee h) \wedge (s \vee c) \\ &\equiv (c \wedge s) \vee c \vee (h \wedge s) \vee (h \wedge c) \\ &\equiv c \vee (h \wedge s). \end{aligned}$$

The prime implicants of $f_{\mathcal{A}}(s, c, h)$ are c and $h \wedge s$. Accordingly, there are two reducts of $\mathcal{A}$, namely $\{Color\}$ and $\{Humidity, Speed\}$. ■

The second type of reducts that are used in this chapter are decision-relative reducts in decision systems.

Definition 15.2.5 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system. The *generalized decision* in $\mathcal{A}$ is the function $\partial_A : U \longrightarrow \mathsf{Pow}(V_d)$ defined by

$$\partial_A(x) = \{i \mid \exists x' \in U.\ x' IND_{\mathcal{A}}(A) x \text{ and } d(x') = i\}.$$

$\mathcal{A}$ is called *consistent (deterministic)*, if $|\partial_A(x)| = 1$, for any $x \in U$. Otherwise $\mathcal{A}$ is said to be *inconsistent (non-deterministic)*. Any set consisting of all objects with the same generalized decision value is called a *generalized decision class*. ■

Definition 15.2.6 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system. A *decision-relative reduct* of $\mathcal{A}$ is a minimal (wrt inclusion) set of attributes $B \subseteq A$ such that $\partial_B = \partial_A$. ■

Intuitively, the definition states that B allows us to classify exactly the same objects, as belonging to equivalence classes $U/\{d\}$, as A.

To compute decision-relative reducts, we extend the definitions of discernibility matrix and discernibility function.

Definition 15.2.7 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system. The discernibility matrix of $\mathcal{A}$ is the discernibility matrix of the information system $\langle U, A\rangle$. ■

Definition 15.2.8 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system and suppose that $\mathcal{M}(\mathcal{A}) = (c_{ij})$ is a discernibility matrix of $\mathcal{A}$. We construct a new matrix, $\mathcal{M}'(\mathcal{A}) = (c'_{ij})$, where

$$c'_{ij} = \begin{cases} \emptyset & \text{iff } \ d(x_i) = d(x_j) \\ c_{ij} & \text{otherwise.} \end{cases}$$

The matrix $\mathcal{M}'(\mathcal{A})$ is called the *decision-relative discernibility matrix* of $\mathcal{A}$. The *decision-relative discernibility function* $f^r_{\mathcal{A}}$ for $\mathcal{A}$ is constructed from the decision-relative discernibility matrix for $\mathcal{A}$ in the same way as the discernibility function is constructed from the discernibility matrix. ■

Theorem 15.2.9 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system. The set of all prime implicants of $f^r_{\mathcal{A}}$ determines the set of all decision-relative reducts of $\mathcal{A}$. ■

Object	*Speed*	*Color*	*Humidity*	*Dangerous*
car1	medium	green	large	no
car2	medium	yellow	small	no
car3	large	blue	large	yes

Table 15.3: Decision table considered in Example 15.2.10.

Example 15.2.10 Consider a decision system $\mathcal{A}$ represented in Table 15.3.

We first construct discernibility matrix of $\mathcal{A}$ (Table 15.4).

$\mathcal{M}(\mathcal{A})$	car1	car2	car3
car1		c,h	s,c
car2	c,h		s,c,h
car3	s,c,	s,c,h	

Table 15.4: The discernibility matrix of a decision system from Table 15.3.

The decision-relative discernibility matrix for $\mathcal{A}$ is given in Table 15.5

$\mathcal{M}'(\mathcal{A})$	car1	car2	car3
car1			s,c
car2			s,c,h
car3	s,c	s,c,h	

Table 15.5: The decision-relative discernibility matrix corresponding to decision system shown in Table 15.3.

Using the decision-relative discernibility matrix, we compute the decision-relative discernibility function for $\mathcal{A}$.

$$\begin{aligned} f^r_{\mathcal{A}}(s,c,h) &\equiv (s \vee c) \wedge (s \vee c \vee h) \\ &\equiv (s \vee c). \end{aligned}$$

The set of all prime implicants of $f^r_{\mathcal{A}}(s,c,h)$ is $\{s,c\}$. Therefore, there are two decision-relative reducts of $\mathcal{A}$, namely $\{Speed\}$ and $\{Color\}$. ∎

To each decision-relative reduct B of a decision system $\mathcal{A}$, we assign a new decision system, called *B-reduction* of $\mathcal{A}$. The details follows.

Definition 15.2.11 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system and

Objects	*Speed*	*Dangerous*
car1, car2	medium	no
car3	large	yes

Table 15.6: $\{Speed\}$-reduction of the decision system $\mathcal{A}$

.

suppose that B is a decision-relative reduct of $\mathcal{A}$. A *B-reduction* of $\mathcal{A}$ is a decision system $\mathcal{A}^* = \langle V, B, d\rangle$, where[3]

- $V = \{[x]_B : x \in U\}$
- $a([x]_B) = a(x)$, for each $a \in B$ and each $[x]_B \in V$
- $d([x]_B) = d(x)$, for each $[x]_B \in V$.

■

Example 15.2.12 [Example 15.2.10 continued] A $\{Speed\}$-reduction of the decision system $\mathcal{A}$ is the decision system presented in Table 15.6.

■

In certain applications instead of reducts, we prefer to use their approximations, called α-reducts, where α is a real number from the interval $[0, 1]$.

Definition 15.2.13 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system and let $\mathcal{M}(\mathcal{A})$ be the discernibility matrix of $\mathcal{A}$. Assume further that n is the number of non-empty sets in $\mathcal{M}(\mathcal{A})$. A set of attributes $B \subseteq A$ is called an *α-reduct* iff $\frac{m}{n} \geq \alpha$, where m is the number of sets that have a non-empty intersection with B. ■

15.3 Attribute Selection

In supervised machine learning, a learning algorithm is presented with a training decision system, usually represented in the form of decision table. Since conditional attributes of the table are typically extracted from large sets of unstructured data, it is often the case that some of them are irrelevant for classification purpose. Accordingly, they should be removed from the table. The *attribute selection problem* is the problem of choosing a relevant subset of attributes, while removing the remaining ones.

[3]Recall that $[x]_B$, where $x \in U$, denotes an equivalence class of relation $IND_{\mathcal{A}}(B)$ containing x (see Chapter 5, Section 5.3).

A natural solution of the attribute selection problem is to consider the intersection of decision-relative reducts of the training decision table as the source of relevant attributes. Unfortunately, there are two problems with this simple solution. Firstly, the intersection can be empty. Secondly, the number of attributes contained in all decision-relative reducts is typically small. In consequence, although those attributes perfectly characterize the training decision table, they are in general insufficient to provide a satisfactory classification of unseen objects.

To deal with the attribute selection problem, it is often reasonable to use various approximations of decision-relative reducts.

Definition 15.3.1 $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision table. Any subset B of A is called an *approximate reduct* of $\mathcal{A}$. The number

$$\varepsilon_{(A,\{d\})}(B) = \frac{(\gamma(A,\{d\}) - \gamma(B,\{d\}))}{\gamma(A,\{d\})} = 1 - \frac{\gamma(B,\{d\})}{\gamma(A,\{d\})} \tag{15.3}$$

is called an *error of reduct approximation*.[4] ■

The error of reduct approximation expresses how exactly the set of attributes B approximates the set of condition attributes A with respect to determining d. Note that $\varepsilon_{(A,\{d\})}(B) \in [0,1]$, where 0 indicates no error and the closer it is to 1, the error is greater.

There are two general approaches to attribute selection: an *open-loop approach* and a *close-loop approach*. What characterizes the methods based on open-loop approach is that they do not use any feedback information about classifier quality for attribute selection. In contrast, the methods based on the closed-loop approach use feedback information as criteria for attribute selection.

A number of attribute selection algorithms have been proposed in the machine learning literature, but they will not be discussed here. Rough set algorithms solving this problem are typically based on the closed-loop approach and consists of the following basic steps.

Firstly, decision-relative reducts are extracted from the training decision table. The attributes contained in these reducts (or in their intersection) are viewed as potentially relevant. Secondly, using the learning algorithm, the classifier based on the chosen attributes is constructed. The classifier is then tested on a new training data. If its performance is unsatisfactory (wrt some measure), a new set of attributes is constructed by extracting approximate reducts of the initial training table, and the process is repeated.

[4] Recall that the coefficient $\gamma(X,Y)$ expresses the degree of dependency between sets of attributes X and Y (see Definition 5.8.1).

Reducts need not be the only source of information concerning selection of attributes. Rough set approach offers another interesting possibility here. The main idea is to generalize the notion of attribute reduction by introducing a concept of a *significance of attributes*. The new notion enables attributes to be evaluated not only in two-valued scale (dispensable-indispensable), but also in a multi-valued scale, by assigning a real number from the interval [0,1] to an attribute. This number, expressing the importance of an attribute in a decision system, is evaluated by measuring the effect of removing the attribute from the table.

Definition 15.3.2 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system and let $a \in A$. The significance of an attribute a in $\mathcal{A}$ is defined by

$$\sigma_{(A,\{d\})}(a) = \frac{(\gamma(A,\{d\}) - \gamma(A-\{a\},\{d\}))}{\gamma(A,\{d\})} = 1 - \frac{\gamma(A-\{a\},\{d\})}{\gamma(A,\{d\})}. \quad (15.4)$$

the coefficient $\sigma_{A,\{d\}}(a)$ can be understood as a classification error which occurs when attribute a is dropped from the decision system. Note that $\sigma_{A,\{d\}}(a) \in [0,1]$, where 0 indicates no change and the closer it is to 1, the greater the error.

The significance coefficient can be extended to sets of attributes by (below, we assume that $B \subseteq A$)

$$\sigma_{(A,\{d\})}(B) = \frac{(\gamma(A,\{d\}) - \gamma(A-B,\{d\}))}{\gamma(A,\{d\})} = 1 - \frac{\gamma(A-B,\{d\})}{\gamma(A,\{d\})}. \quad (15.5)$$

■

We conclude this section by remarking that there are public domain software packages, for instance RSES system (for references see Section 15.9), offering various algorithms helpful for solving attribute selection problem.

15.4 Attribute Extraction

If $\mathcal{A} = \langle U, A, d\rangle$ is a decision table with large number of attribute values (relative to the number of elements of U), then there is very low chance that a new object will be properly recognized by matching its attribute value vector with rows of this table. Accordingly, to construct a high quality classifier, it is often necessary to reduce the cardinality of value sets of attributes in a training decision table. The task of reducing cardinality of value sets is referred to as *attribute extraction problem*.

In this section, we discuss two methods of attribute extraction: *discretization process*, used for real value attributes, and *symbolic attribute value grouping*, employed for symbolic attributes.

$\mathcal{A}$	a	b	d
u_1	0.8	2.0	1
u_2	1.0	0.5	0
u_3	1.3	3.0	0
u_4	1.4	1.0	1
u_5	1.4	2.0	0
u_6	1.6	3.0	1
u_7	1.3	1.0	1

(a)

$\Rightarrow$

$\mathcal{A}^{\mathbf{C}}$	$a^{\mathbf{C}}$	$b^{\mathbf{C}}$	d
u_1	0	2	1
u_2	1	0	0
u_3	2	3	0
u_4	3	1	1
u_5	3	2	0
u_6	4	3	1
u_7	2	1	1

(b)

Table 15.7: The discretization process: (a) The original decision system $\mathcal{A}$ considered in Example 15.4.2. (b) The $\mathbf{C}$-discretization of $\mathcal{A}$ considered in Example 15.4.4.

15.4.1 Discretization Process

A discretization process replaces value sets of conditional real-valued attributes by intervals. The replacement ensures that a consistent decision system is obtained (from a given consistent decision system) by substituting any object's original value in decision table by the unique name of the interval in which it is contained. This substantially reduces the size of the value sets of real-valued attributes.

Discretization is not specific to the rough set approach. A majority of rule or tree induction algorithms require it for good performance.

Definition 15.4.1 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system. Assume $V_a = [l_a, r_a) \subset \Re$, for any $a \in A$, where $\Re$ denotes the set of real numbers. A pair $\langle a, c\rangle$, where $a \in A$ and $c \in V_a$, is called a *cut* on V_a.

Any set of the form $\{\langle a, c_1^a\rangle, \langle a, c_2^a\rangle, \ldots, \langle a, c_k^a\rangle\}$, where each $\langle a, c_i^a\rangle$, for $1 \leq i \leq k$, is a cut on V_a is called a *set of cuts* on V_a.

Note that any attribute $a \in A$ defines a sequence of real numbers $v_1^a < v_2^a < \cdots < v_{k_a}^a$, where $\{v_1^a, v_2^a, \ldots, v_{k_a}^a\} = \{a(x) : x \in U\}$. The *set of basic cuts* on a, written $\mathbf{C}_a$, is specified by

$$\mathbf{C}_a = \{\langle a, (v_1^a + v_2^a)/2\rangle, \langle a, (v_2^a + v_3^a)/2\rangle, \ldots, \langle a, (v_{k_a-1}^a + v_{k_a}^a)/2\rangle\}.$$

The set $\bigcup_{a\in A} \mathbf{C}_a$ is called the *set of basic cuts* on $\mathcal{A}$. ∎

Example 15.4.2 Consider a (consistent) decision system $\mathcal{A}$ presented in Table 15.7(a).

We assume that the initial value domains for attributes a and b are:

$$V_a = [0, 2)\,; V_b = [0, 4)\,.$$

The sets of values of a and b for objects from U are:

$$\begin{aligned} a(U) &= \{0.8, 1.0, 1.3, 1.4, 1.6\} \\ b(U) &= \{0.5, 1.0, 2.0, 3.0\}. \end{aligned}$$

Accordingly, the sets of basic cuts of a and b are

$$\begin{aligned} \mathbf{C}_a &= \{\langle a, 0.9\rangle,\ \langle a, 1.15\rangle,\ \langle a, 1.35\rangle,\ \langle a, 1.5\rangle\} \\ \mathbf{C}_b &= \{\langle b, 0.75\rangle;\ \langle b, 1.5\rangle;\ \langle b, 2.5\rangle\}. \end{aligned}$$

■

Definition 15.4.3 Let $\mathcal{A} = \langle U, A, d\rangle$ be as in Definition 15.4.1. For $a \in A$, let $\mathbf{C}_a = \{\langle a, c_1^a\rangle, \langle a, c_2^a\rangle, \ldots, \langle a, c_k^a\rangle\}$ be any set of cuts of a. Assume that $c_1^a < c_2^a < \cdots < c_k^a$. The set of cuts $\mathbf{C} = \bigcup_{a\in A} \mathbf{C}_a$ defines a new decision system $\mathcal{A}^{\mathbf{C}} = \langle U, A^{\mathbf{C}}, d\rangle$, called the $\mathbf{C}$-*discretization of* $\mathcal{A}$, where

$$A^{\mathbf{C}} = \{a^{\mathbf{C}} : a \in A\}$$

$$a^{\mathbf{C}}(x) = \begin{cases} 0 & \text{iff } a(x) < c_1^a \\ i & \text{iff } a(x) \in [c_i^a, c_{i+1}^a), \text{for } i \in \{1, \ldots, k-1\} \\ k+1 & \text{iff } a(x) > c_k^a \end{cases}$$

■

Example 15.4.4 [Example 15.4.2 continued] Let $\mathbf{C} = \mathbf{C}_a \cup \mathbf{C}_b$. It is easily checked that the $\mathbf{C}$-discretization of $\mathcal{A}$ is the decision system presented in Table 15.7 (b). ■

Since a decision system can be discretized in infinitely many ways, a natural question is how to evaluate various discretizations.

Definition 15.4.5 The set of cuts $\mathbf{C}$ is called $\mathcal{A}$-*consistent*, if $\partial_{\mathcal{A}} = \partial_{\mathcal{A}^{\mathbf{C}}}$, where $\partial_{\mathcal{A}}$ and $\partial_{\mathcal{A}^{\mathbf{C}}}$ are generalized decisions of $\mathcal{A}$ and $\mathcal{A}^{\mathbf{C}}$, respectively. An $\mathcal{A}$-consistent set of cuts $\mathbf{C}$ is $\mathcal{A}$-*irreducible* if $\mathbf{C}'$ is not $\mathcal{A}$-consistent for any $\mathbf{C}' \subset \mathbf{C}$. The $\mathcal{A}$-consistent set of cuts $\mathbf{C}$ is $\mathcal{A}$-*optimal*, if $|\mathbf{C}| \leq |\mathbf{C}'|$, for any $\mathcal{A}$-consistent set of cuts $\mathbf{C}'$. ■

It is easily seen that the set of cuts considered in Example 15.4.4 is $\mathcal{A}$-consistent. However, as we shall see, it is neither optimal nor irreducible.

Since the purpose of the discretization process is to reduce the size of value sets of attributes, we are primarily interested in optimal sets of cuts. These

are extracted from basic sets of cuts of a given decision system. The details follow.

Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system where $U = \{u_1, \ldots, u_n\}$. Recall that any attribute $a \in A$ defines a sequence $v_1^a < v_2^a < \cdots < v_{k_a}^a$, where $\{v_1^a, v_2^a, \ldots, v_{k_a}^a\} = \{a(x) : x \in U\}$. Let $ID(\mathcal{A})$ be a set of pairs $\langle i, j\rangle$ such that $i < j$ and $d(u_i) \neq d(j)$. We now construct a propositional formula, called the *discernibility formula* of $\mathcal{A}$, as follows.

1. To each interval of the form $\left[v_k^a, v_{k+1}^a\right)$, $a \in A$ and $k \in \{1, \ldots, n_a - 1\}$, we assign a Boolean variable denoted by p_k^a. The set of all these variables is denoted by $V(\mathcal{A})$.

2. We first construct a family of formulas

$$\{B(a, i, j) : a \in A \text{ and } \langle i, j\rangle \in ID(\mathcal{A})\},$$

where $B(a, i, j)$ is a disjunction of all elements from the set

$$\left\{p_k^a : \left[v_k^a, v_{k+1}^a\right) \subseteq [\min\{a(u_i), a(u_j)\}, \max\{a(u_i), a(u_j)\})\right\}.$$

3. Next, we construct a family of formulas

$$\{C(i, j) : i, j \in \{1, \ldots, n\}, i < j \text{ and } \langle i, j\rangle \in ID(\mathcal{A})\},$$

where $C(i, j) = \bigvee_{a \in A} B(a, i, j)$.

4. Finally, the discernibility formula for $\mathcal{A}$, $D(\mathcal{A})$, is constructed by

$$D(\mathcal{A}) = \bigwedge C(i, j),$$

where $i < j$ and $\langle i, j\rangle \in ID(\mathcal{A})$.

Any non empty set $S = \{p_{k_1}^{a_1}, \ldots, p_{k_r}^{a_r}\}$ of Boolean variables from $V(\mathcal{A})$ uniquely defines a set of cuts, $\mathbf{C}(S)$, given by

$$\mathbf{C}(S) = \{\langle a_1, (v_{k_1}^{a_1} + v_{k_1+1}^{a_1})/2\rangle, \cdots, \langle a_1, (v_{k_r}^{a_r} + v_{k_r+1}^{a_r})/2\rangle\}.$$

Theorem 15.4.6 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system. For any non empty set $S \subseteq V(\mathcal{A})$ of Boolean variables the following conditions are equivalent:

1. The conjunction of variables from S is a prime implicant of the discernibility formula for $\mathcal{A}$.
2. $\mathbf{C}(S)$ is an $\mathcal{A}$-irreducible set of cuts on $\mathcal{A}$. ∎

Corollary 15.4.7 Let $\mathcal{A} = \langle U, A, d\rangle$ be a consistent decision system. For any non empty set $S \subseteq V(\mathcal{A})$ of Boolean variables the following conditions are equivalent:

1. The conjunction of variables from S is minimal (wrt to length) prime implicant of the discernibility formula for $\mathcal{A}$.
2. $\mathbf{C}(S)$ is an $\mathcal{A}$-optimal set of cuts on $\mathcal{A}$. ∎

Example 15.4.8 [Example 15.4.4 continued]

$$\begin{aligned} ID(\mathcal{A}) = \quad & \{\langle 1,2\rangle,\ \langle 1,3\rangle,\ \langle 1,5\rangle,\ \langle 2,4\rangle,\ \langle 2,6\rangle,\ \langle 2,7\rangle \\ & \langle 3,4\rangle,\ \langle 3,6\rangle,\ \langle 3,7\rangle,\ \langle 4,5\rangle,\ \langle 5,6\rangle,\ \langle 5,7\rangle\}. \end{aligned}$$

1. We introduce four Boolean variables, p_1^a, p_2^a, p_3^a, p_4^a, corresponding respectively to the intervals

   $$[0.8, 1.0)\,,\ [1.0, 1.3)\,,\ [1.3, 1.4)\,,\ [1.4, 1.6)\,,$$

   of the attribute a, and three Boolean variables, p_1^b, p_2^b, p_3^b, corresponding respectively to the intervals

   $$[0.5, 1.0)\,,\ [1.0, 2.0)\,,\ [2, 3.0)\,,$$

   of the attribute b.

2. The formulas $B(a, i, j)$ and $B(b, i, j$, where $i < j$ and $\langle i, j\rangle \in ID(\mathcal{A})$, are the following:

$B(a,1,2) \equiv p_1^a$	$B(b,1,2) \equiv p_1^b \vee p_2^b$
$B(a,1,3) \equiv p_1^a \vee p_2^a$	$B(b,1,3) \equiv p_3^b$
$B(a,1,5) \equiv p_1^a \vee p_2^a \vee p_3^a$	$B(b,1,3) \equiv \mathsf{False}$
$B(a,2,4) \equiv p_2^a \vee p_3^a$	$B(b,2,4) \equiv p_1^b$
$B(a,2,6) \equiv p_2^a \vee p_3^a \vee p_4^a$	$B(b,2,6) \equiv p_1^b \vee p_2^b \vee p_3^b$
$B(a,2,7) \equiv p_2^a$	$B(b,2,7) \equiv p_1^b$
$B(a,3,4) \equiv p_3^a$	$B(b,3,4) \equiv p_2^b \vee p_3^b$
$B(a,3,6) \equiv p_3^a \vee p_4^a$	$B(b,3,6) \equiv \mathsf{False}$
$B(a,3,7) \equiv \mathsf{False}$	$B(b,3,7) \equiv p_2^b \vee p_3^b$
$B(a,4,5) \equiv \mathsf{False}$	$B(b,4,5) \equiv p_2^b$
$B(a,5,6) \equiv p_4^a$	$B(b,5,6) \equiv p_3^b$
$B(a,5,7) \equiv p_3^a$	$B(b,5,7) \equiv p_2^b$

3. The formulas $C(i, j)$, where $i < j$ and $\langle i, j\rangle \in ID(\mathcal{A})$, are the following:

$C(1,2) \equiv p_1^a \vee p_1^b \vee p_2^b$	$C(1,3) \equiv p_1^a \vee p_2^a \vee p_3^b$
$C(1,5) \equiv p_1^a \vee p_2^a \vee p_3^a$	$C(2,4) \equiv p_2^a \vee p_3^a \vee p_1^b$
$C(2,6) \equiv p_2^a \vee p_3^a \vee p_4^a \vee p_1^b \vee p_2^b \vee p_3^b$	$C(2,7) \equiv p_2^a \vee p_1^b$
$C(3,4) \equiv p_3^a \vee p_2^b \vee p_3^b$	$C(3,6) \equiv p_3^a \vee p_4^a$
$C(3,7) \equiv p_2^b \vee p_3^b$	$C(4,5) \equiv p_2^b$
$C(5,6) \equiv p_4^a \vee p_3^b$	$C(5,7) \equiv p_3^a \vee p_2^b$

$\mathcal{A}^{\mathbf{C}}$	$a^{\mathbf{C}}$	$b^{\mathbf{C}}$	d
u_1	0	2	1
u_2	1	0	0
u_3	1	2	0
u_4	1	1	1
u_5	1	2	0
u_6	2	2	1
u_7	1	1	1

Table 15.8: The **C**-discretization considered in Example 15.4.8

4. The discernibility formula for $\mathcal{A}$ is then given by

$$\begin{aligned} D(\mathcal{A}) \equiv\ & (p_1^a \vee p_1^b \vee p_2^b) \wedge (p_1^a \vee p_2^a \vee p_3^b) \wedge \\ & (p_1^a \vee p_2^a \vee p_3^a) \wedge (p_2^a \vee p_3^a \vee p_1^b) \wedge \\ & (p_2^a \vee p_3^a \vee p_4^a \vee p_1^b \vee p_2^b \vee p_3^b) \wedge (p_2^a \vee p_1^b) \wedge \\ & (p_3^a \vee p_2^b \vee p_3^b) \wedge (p_3^a \vee p_4^a) \wedge (p_2^b \vee p_3^b) \wedge \\ & p_2^b \wedge (p_4^a \vee p_3^b) \wedge (p_3^a \vee p_2^b). \end{aligned} \tag{15.6}$$

The prime implicants of the formula $D(\mathcal{A})$ are

$$\begin{aligned} & p_2^a \wedge p_4^a \wedge p_2^b, \quad p_2^a \wedge p_3^a \wedge p_2^b \wedge p_3^b, \\ & p_3^a \wedge p_1^b \wedge p_2^b \wedge p_3^b, \quad p_1^a \wedge p_4^a \wedge p_1^b \wedge p_2^b. \end{aligned} \tag{15.7}$$

Suppose we take the the prime implicant $p_1^a \wedge p_4^a \wedge p_1^b \wedge p_2^b$. Its corresponding set of cuts is

$$\mathbf{C} = \{\langle a, 0.9\rangle, \langle a, 1.5\rangle, \langle b, 0.75\rangle, \langle b, 1.5\rangle\}.$$

The **C**-discretization of $\mathcal{A}$ is presented in Table 15.8.

■

The problem of searching for an optimal set of cuts **P** in a given decision system $\mathcal{A}$ is NPTIME-hard. However, it is possible to devise efficient heuristics which, in general, return reasonable sets of cuts. One of them, called *MD-heuristics*, is presented below.

We say that a cut $\langle a, c\rangle$ discerns objects x and y iff $a(x) < c \leq a(y)$ or $a(y) < c \leq a(x)$.

Algorithm 15.4.9 [MD-heuristics]

INPUT: a decision system $\mathcal{A} = \langle U, A, d\rangle$
OUTPUT: a set of cuts $\mathcal{C}$.

$\mathcal{A}^*$	$\langle a, 0.9\rangle$	$\langle a, 1.15\rangle$	$\langle a, 1.35\rangle$	$\langle a, 1.5\rangle$	$\langle b, 0.75\rangle$	$\langle b, 1.5\rangle$	$\langle b, 2.5\rangle$
$\langle u_1, u_2\rangle$	1	0	0	0	1	1	0
$\langle u_1, u_3\rangle$	1	1	0	0	0	0	1
$\langle u_1, u_5\rangle$	1	1	1	0	0	0	0
$\langle u_2, u_4\rangle$	0	1	1	0	1	0	0
$\langle u_2, u_6\rangle$	0	1	1	1	1	1	1
$\langle u_2, u_7\rangle$	0	1	0	0	1	0	0
$\langle u_3, u_4\rangle$	0	0	1	0	0	1	1
$\langle u_3, u_6\rangle$	0	0	1	1	0	0	0
$\langle u_3, u_7\rangle$	0	0	0	0	0	1	1
$\langle u_4, u_5\rangle$	0	0	0	0	0	1	0
$\langle u_5, u_6\rangle$	0	0	0	1	0	0	1
$\langle u_5, u_7\rangle$	0	0	1	0	0	1	0

Table 15.9: The **C**-discretization considered in Example 15.4.8

1. Set $\mathcal{C}$ to $\emptyset$.
2. Let $\bigcup_{a\in A} \mathbf{C}_a$ be the set of basic cuts on $\mathcal{A}$ (see Definition 15.4.1).
3. Construct an information table $\mathcal{A}^* = \langle U^*, A^*\rangle$ such that
   - U^* is the set of pairs $\langle u_i, u_j\rangle$ of objects discerned by d (in $\mathcal{A}$) such that $i < j$;
   - $A^* = \bigcup_{a\in A} \mathbf{C}_a$, where, for each $c \in A^*$,
   $$c(\langle x, y\rangle) = \begin{cases} 1 \text{ iff } c \text{ discerns } x \text{ and } y \text{ (in } \mathcal{A}) \\ 0 \text{ otherwise} \end{cases}$$
4. Choose a column from $\mathcal{A}^*$ with the maximal number of occurrences of 1's; add the cut corresponding to this column to $\mathcal{C}$; delete the column from $\mathcal{A}^*$, together with all rows marked with 1 in it;
5. If A^* is non-empty, then go to step 4 else stop. ■

Let n be the number of objects and let k be the number of attributes of a decision system $\mathcal{A}$. The MD-heuristics computes a set of cuts in $O(kn)$ steps using $O(kn)$ space only.

Example 15.4.10 Consider the decision system $\mathcal{A}$ given in Table 15.7 from Example 15.4.2. The information system $\mathcal{A}^*$ is presented in Table 15.9. Under the assumption that columns with maximal number of 1's are chosen from left to right (if many such columns exist in a given step), the set of cuts returned by the algorithm is $\{\langle a, 1.35\rangle, \langle b, 1.5\rangle, \langle a, 1.15\rangle, \langle a, 1.5\rangle\}$. ■

15.4.2 Symbolic Attribute Value Grouping

The symbolic attribute value grouping is a process of reducing the cardinality of value sets of symbolic attributes.

Definition 15.4.11 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision table. Any function $c_a : V_a \to \{1, \ldots, m\}$ (where $m \leq |V_a|$) is called a *clustering function* for V_a. The *rank of* c_a, denoted by $rank\,(c_a)$, is the value $|c_a\,(V_a)\,|$.

For $B \subseteq A$, a family of clustering functions is $\{c_a\}_{a\in B}$ is B-*consistent* iff

$$\forall a \in B \;\; [c_a(a(u)) = c_a(a(u'))]$$

implies

$$\langle u, u'\rangle \in IND_{\mathcal{A}}(B) \cup IND_{\mathcal{A}}(\{d\}), \text{ for any pair } \langle u, u'\rangle \in U.$$

■

The notion of B-consistency has the following intuitive interpretation: if two objects are indiscernible wrt clustering functions for value sets of attributes from B, then they are indiscernible either by the attributes from B or by the decision attribute.

We consider the following problem, called the *symbolic value partition grouping problem*:

> given a decision table $\mathcal{A} = \langle U, A, d\rangle$ and a set of attributes $B \subseteq A$, search for a B-consistent family $\{c_a\}_{a\in B}$ of clustering functions such that $\sum_{a\in B} rank\,(c_a)$ is minimal.

In order to solve this problem, we apply the following steps.

1. For each attribute $a \in B$, construct new binary attributes $a_v^{v'}$ (for $v, v' \in V_a$ and $v \neq v'$) defined by[5]

$$a_v^{v'}(x, y) = \begin{cases} 1 \text{ iff } v = a(x) \text{ and } v' = a(y) \\ 0 \text{ otherwise.} \end{cases}$$

2. Construct the discernibility matrix (see Definition 15.2.2) for the new attributes. Reduce the discernibility matrix by removing all empty columns and all empty rows.

3. Using the obtained reduced discernibility matrix, construct the discernibility function, as in Definition 15.2.2.

[5]The introduced attributes serve to discern between pairs of objects wrt an attribute a.

4. Compute the shortest prime implicant I of the discernibility function.[6]

5. Using I, construct, for each attribute $a \in B$, an undirected graph $\Gamma_a = \langle V_a, E_a \rangle$, where

   - V_a, as usual, is the set of values of attribute a
   - $E_a = \{\langle a(x), a(y)\rangle : x, y \in U \text{ and } a(x) \neq a(y)\}$. Using I one can construct E_a as follows:

     $$E_a = \{\langle a(v), a(v)\rangle : a_v^{v'} \text{ occurs in } I\}.$$

6. Find a minimal coloring of vertices[7] of Γ_a. The coloring defines a partition of V_a by assuming that all vertices of the same color belong to the same partition and no partition contains vertices with different colors. Partitions are named using successive natural numbers. The clustering function for V_a is $c_a(v) = i$, provided that v is the member of the i-th partition.

Remark 15.4.12 In practical implementations one does not construct the discernibility matrix explicitly, as required in Steps (2)-(3) above. Actually, prime implicants are directly extracted from the original information system, but using the idea of discernibility matrices.

Let us emphasize that in the Step (4) above, there can be many different shortest prime implicants and in the Step (6) there can be many different colorings of the obtained graphs. Accordingly, one can obtain many substantially different families of clustering functions resulting in different classifiers. In practice, one often generates a number of families of clustering functions, tests them against data and chooses the best one. ∎

Using the constructed family of partitions, one can obtain a new, usually substantially smaller decision table, according to the following definition.

Definition 15.4.13 Let $\mathcal{A} = \langle U, A, d\rangle$ be be a decision system and $B \subseteq A$. Any family of clustering functions $\mathbf{c} = \{c_a\}_{a \in B}$ specifies a new decision system $\mathcal{A}^{\mathbf{c}} = \langle U, A^{\mathbf{c}}, d\rangle$ called $\mathbf{c}$-*reduction of* $\mathcal{A}$ wrt B, where $A^{\mathbf{c}} = \{a^{\mathbf{c}} : a \in B\}$ and $a^{\mathbf{c}}(x) = c_a(a(x))$. ∎

Example 15.4.14 Consider a decision table presented in Table 15.10. We want to solve the symbolic value partition problem for $B = A$.

One has then to perform the following steps:

[6] The problem of finding the shortest prime implicant is NPTIME-hard, thus in practice, one usually computes an implicant as close to the shortest prime implicant as possible.

[7] The colorability problem is solvable in polynomial time for $k = 2$, but remains NPTIME-complete for all $k \geq 3$. But, similarly to discretization, one can apply some efficient heuristics searching for a (sub-) optimal partition.

1. Introduce the new attributes a_v^u, b_x^w, for all $u, v \in V_a, u \neq v$ and $w, x \in V_b, w \neq x$.
2. The reduced discernibility matrix is presented in Table 15.11.
3. The discernibility function is given by

$$
\begin{aligned}
& b_{b_4}^{b_1} \wedge b_{b_4}^{b_2} \wedge (a_{a_2}^{a_1} \vee b_{b_4}^{b_3}) \wedge (a_{a_3}^{a_1} \vee b_{b_4}^{b_1}) \wedge \\
& (a_{a_2}^{a_1} \vee b_{b_2}^{b_1}) \wedge a_{a_2}^{a_1} \wedge b_{b_3}^{b_2} \wedge (a_{a_3}^{a_2} \vee b_{b_2}^{b_1}) \wedge \\
& a_{a_2}^{a_1} \wedge (a_{a_2}^{a_1} \vee b_{b_2}^{b_1}) \wedge b_{b_3}^{b_1} \wedge a_{a_3}^{a_2} \wedge \\
& (a_{a_4}^{a_1} \vee b_{b_2}^{b_1}) \wedge a_{a_4}^{a_1} \wedge (a_{a_4}^{a_2} \vee b_{b_3}^{b_2}) \wedge (a_{a_4}^{a_3} \vee b_{b_2}^{b_1}) \wedge \\
& (a_{a_3}^{a_1} \vee b_{b_4}^{b_1}) \wedge (a_{a_3}^{a_1} \vee b_{b_4}^{b_2}) \wedge (a_{a_3}^{a_2} \vee b_{b_4}^{b_3}) \wedge b_{b_4}^{b_1} \wedge \\
& (a_{a_2}^{a_1} \vee b_{b_5}^{b_1}) \wedge (a_{a_2}^{a_1} \vee b_{b_5}^{b_2}) \wedge b_{b_5}^{b_3} \wedge (a_{a_3}^{a_2} \vee b_{b_5}^{b_1}).
\end{aligned}
$$

4. The shortest prime implicant for the function is:

$$
I \equiv a_{a_2}^{a_1} \wedge a_{a_3}^{a_2} \wedge a_{a_4}^{a_1} \wedge a_{a_4}^{a_3} \wedge b_{b_4}^{b_1} \wedge b_{b_4}^{b_2} \wedge b_{b_3}^{b_2} \wedge b_{b_3}^{b_1} \wedge b_{b_5}^{b_3}
$$

5. The graphs corresponding to a and b are shown in Figure 15.1.
6. The graphs are 2-colored, as shown in Figure 15.1, where nodes marked by $\bigotimes$ represent the black color and the other nodes represent the white color. The colors gives rise to the following clustering functions:

$$
\begin{aligned}
c_a(a_1) &= c_a(a_3) = 1; \\
c_a(a_2) &= c_a(a_4) = 2 \\
c_b(b_1) &= c_b(b_2) = c_b(b_5) = 1; \\
c_b(b_3) &= c_b(b_4) = 2.
\end{aligned}
$$

Given the clustering functions, one can construct a new decision table (Table 15.12), according to Definition 15.4.13. ∎

We conclude this section by remarking that discretization and symbolic attribute value grouping can be simultaneously used in decision systems including both real-value and symbolic attributes.

15.5 Discretization of Large Data Sets Stored in Relational Databases

Searching algorithms for constructing optimal sets of cuts on real-valued attributes have been intensively studied. The main goal of such algorithms is

$\mathcal{A}$	a	b	d
u_1	a_1	b_1	0
u_2	a_1	b_2	0
u_3	a_2	b_3	0
u_4	a_3	b_1	0
u_5	a_1	b_4	1
u_6	a_2	b_2	1
u_7	a_2	b_1	1
u_8	a_4	b_2	1
u_9	a_3	b_4	1
u_{10}	a_2	b_5	1

Table 15.10: The decision table considered in Example 15.4.14

.

$\mathcal{M}(\mathcal{A})$	u_1	u_2	u_3	u_4
u_5	$b_{b_4}^{b_1}$	$b_{b_4}^{b_2}$	$a_{a_2}^{a_1}, b_{b_4}^{b_3}$	$a_{a_3}^{a_1}, b_{b_4}^{b_1}$
u_6	$a_{a_2}^{a_1}, b_{b_2}^{b_1}$	$a_{a_2}^{a_1}$	$b_{b_3}^{b_2}$	$a_{a_3}^{a_2}, b_{b_2}^{b_1}$
u_7	$a_{a_2}^{a_1}$	$a_{a_2}^{a_1}, b_{b_2}^{b_1}$	$b_{b_3}^{b_1}$	$a_{a_3}^{a_2}$
u_8	$a_{a_4}^{a_1}, b_{b_2}^{b_1}$	$a_{a_4}^{a_1}$	$a_{a_4}^{a_2}, b_{b_3}^{b_2}$	$a_{a_4}^{a_3}, b_{b_2}^{b_1}$
u_9	$a_{a_3}^{a_1}, b_{b_4}^{b_1}$	$a_{a_3}^{a_1}, b_{b_4}^{b_2}$	$a_{a_3}^{a_2}, b_{b_4}^{b_3}$	$b_{b_4}^{b_1}$
u_{10}	$a_{a_2}^{a_1}, b_{b_5}^{b_1}$	$a_{a_2}^{a_1}, b_{b_5}^{b_2}$	$b_{b_5}^{b_3}$	$a_{a_3}^{a_2}, b_{b_5}^{b_1}$

Table 15.11: The reduced discernibility matrix corresponding to decision table shown in Table 15.10.

$a^{\mathbf{c}}$	$b^{\mathbf{c}}$	d
1	1	0
2	2	0
1	2	1
2	1	1

Table 15.12: The reduced table corresponding to graphs shown in Figure 15.1.

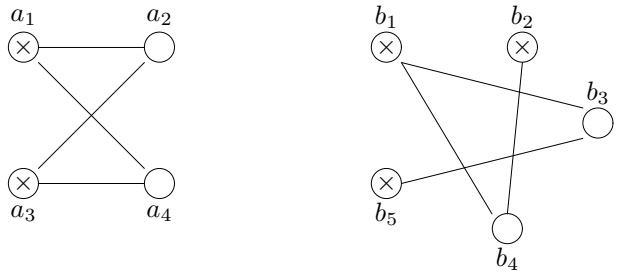

Figure 15.1: Coloring of attribute value graphs constructed in Example 15.4.14.

to discover cuts which can be used to synthesize decision trees or decision rules of high quality wrt some quality measures (e.g., quality of classification of new unseen objects, quality defined by the decision tree height, support and confidence of decision rules).

In general, all those problems are hard from computational point of view. Accordingly, numerous heuristics providing their approximate solutions have been developed. These heuristics are based on approximate measures estimating the quality of extracted cuts. Among such measures *discernibility measures* are relevant for rough set approach.

In this section, we outline an approximate approach to the problem of finding optimal sets of cuts, assuming that a large decision table is represented in a relational database. In such a case, even the linear time complexity wrt the number of cuts is inacceptable because of the time needed for one step. The critical factor for time complexity of algorithms solving that problem is the number of SQL queries of the form

```
SELECT COUNT
FROM aTable
WHERE (anAttribute BETWEEN value1 AND value2)
AND (additional condition)
```

necessary to construct cuts on real-valued attribute sets. We assume the answer time for such queries does not depend on the interval length[8]. Using a straightforward approach to optimal cuts selection (wrt a given measure), the number of necessary queries is of order $O(N)$, where N is the number of preassumed cuts. By introducing some optimization measures, it is possible to reduce the size of searching space. It turns out that $O(\log N)$ simple queries suffices to construct a set of cuts very close to the optimal one.

[8]This assumption is satisfied in some existing database management systems.

Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system with real-valued condition attributes. Any cut $\langle a, c\rangle$, where $a \in A$ and c is a real number, defines two disjoint sets given by

$$U_L(a,c) = \{x \in U : a(x) < c\}$$
$$U_R(a,c) = \{x \in U : a(x) \geq c\}.$$

Recall that a cut $\langle a, c\rangle$ discerns a pair of objects x, y if either $a(x) < c \leq a(y)$ or $a(y) < c \leq a(x)$.

Definition 15.5.1 Let $\mathcal{A} = \langle U, A, d\rangle$ be a decision system with real-valued condition attributes and decision classes X_i, for $i = 1, \ldots, r(d)$. A *discerinibility quality of a cut* $\langle a, c\rangle$, denoted by $W(a, c)$, is defined by

$$\begin{aligned} W(a,c) &= \sum_{i \neq j}^{r(d)} L_i(a,c) * R_j(a,c) \qquad (15.8) \\ &= \left(\sum_{i=1}^{r(d)} L_i(a,c)\right) * \left(\sum_{i=1}^{r(d)} R_i(a,c)\right) - \sum_{i=1}^{r(d)} L_i(a,c) * R_i(a,c), \end{aligned}$$

where $L_i(a,c) = |X_i \cap U_L(a,c)|$ and $R_i(a,c) = |X_i \cap U_R(a,c)|$, for $i = 1, \ldots, r(d)$. ∎

In the sequel we will be interested in finding cuts maximizing the function $W(a, c)$.

The following definition will be useful.

Definition 15.5.2 Let $\mathcal{C}_a = \{\langle a, c_1\rangle, \ldots, \langle a, c_N\rangle\}$ be a set of cuts on attribute a over a decision system $\mathcal{A}$ and assume $c_1 < c_2 ... < c_N$. By a *median of the* i^{th} *decision class*, denoted by $Median(i)$, we mean the minimal index j for which the cut $\langle a, c_j\rangle \in \mathcal{C}_a$ minimizes the value $|L_i(a, c_j) - R_i(a, c_j)|$,[9] where L_i and R_i are defined as in Definition 15.5.1. ∎

One can use only $O(r(d) * \log N)$ SQL queries to determine the medians of decision classes by using the well-known binary search algorithm.

Then one can show that the quality function $W_a(i) \stackrel{\text{def}}{=} W(a, c_i)$, for $i = 1, \ldots, N$, is increasing in $\{1, \ldots, min\}$ and decreasing in $\{max, \ldots, N\}$, where min and max are defined by

$$min = \min_{1 \leq i \leq N} Median(i)$$
$$max = \max_{1 \leq i \leq N} Median(i).$$

[9]The minimization means that $|L_i(a, c_j) - R_i(a, c_j)| = \min_{1 \leq k \leq N} |L_i(a, c_k) - R_i(a, c_k)|$.

In consequence, the search space for maximum of $W(a, c_i)$ is reduced to $i \in [min, max]$.

Now one can apply the divide and conquer strategy to determine the best cut, given by $c_{Best} \in [c_{min}, c_{max}]$, wrt the chosen quality function. First, we divide the interval containing all possible cuts into k intervals. One then predicts, using some heuristics, the interval, say i, which most probably contains the best cut. This process is recursively applied to interval i, until the considered interval consists of one cut. The problem which remains to be solved is how to define such approximate measures which could help us to predict the suitable interval.

Let us consider a simple probabilistic model. Let $\langle a, c_L\rangle$, $\langle a, c_R\rangle$, such that $c_L < c_R$, be two cuts. For any cut $\langle a, c\rangle$ satisfying $c_L < c < c_R$, we assume that $x_1, \ldots, x_{r(d)}$, where $x_i = |X_i \cap U_L(a, c)) \cap U_R(a, c_L)|$, for $i = 1, \ldots, r(d)$, are independent random variables with uniform distribution over sets $\{0, ..., M_1\}$, ..., $\{0, ..., M_{r(d)}\}$, respectively, where

$$M_i = M_i(a, c_L, c_R) = |X_i \cap U_L(a, c_R) \cap U_R(a, c_L)|,$$

for $i = 1, \ldots, r(d)$. Under these assumptions the following fact holds. For any cut $c \in [c_L, c_R]$, the mean $E(W(a, c))$ of quality $W(a, c)$, is given by

$$E(W(a, c)) = \frac{W(a, c_L) + W(a, c_R) + conflict(\langle a, c_L\rangle, \langle a, c_R\rangle)}{2} \tag{15.9}$$

where $conflict(\langle a, c_L\rangle, \langle a, c_R\rangle) = \sum_{i \neq j} M_i * M_j$.

In addition, the standard deviation of $W(a, c)$ is given by

$$D^2(W(a, c)) = \sum_{i=1}^{n} \left[\frac{M_i(M_i + 2)}{12} \left(\sum_{j \neq i} (R_j(a, c_R) - L_j(a, c_L)) \right)^2 \right] \tag{15.10}$$

Formulas (15.9) and (15.10) can be used to construct a predicting measure for the quality of the interval $[c_L, c_R]$:

$$Eval\,([c_L, c_R], \alpha) \quad = \quad E(W(a, c)) + \alpha \sqrt{D^2(W(a, c))} \tag{15.11}$$

where the real parameter α from $[0, 1]$ can be tuned in a learning process.

To determine the value $Eval\,([c_L, c_R], \alpha)$, we need to compute the numbers

$\langle L_1(a, c_L), \ldots, L_{r(d)}(a, c_L)\rangle, \langle M_1, \ldots, M_{r(d)}\rangle$
$\langle R_1(a, c_R), \ldots, R_{r(d)}(a, c_R)\rangle$.

This requires $O(r(d))$ SQL queries of the form

```
SELECT COUNT
FROM DecTable
WHERE (attribute_a BETWEEN value_1 AND value_2) AND (dec = i).
```

Accordingly, the number of queries required by this algorithm is $O(r(d)k \log_k N)$. In practice, we set $k = 3$, since the function $f(k) = r(d)k \log_k N$ over positive integers is taking minimum for $k = 3$.

Numerous experiments on different data sets have shown that the proposed solution allows one to find sets of cuts which is very close to the optimal ones. For more details the reader is referred to the literature (see Section 15.9).

15.6 Minimal Decision Rules

In this section, we shall show how to construct minimal rules for a given decision system.

Definition 15.6.1 Given a decision table $\mathcal{A}$, by a *minimal decision rule* (wrt $\mathcal{A}$) we mean a rule True in $\mathcal{A}$ which becomes False in $\mathcal{A}$ if any descriptor from the left-hand side of the rule is removed.[10] ∎

The minimal number of descriptors in the left-hand side of a minimal decision rule defines the largest subset of a decision class. Accordingly, information included in conditional part of any minimal rule is sufficient for prediction of the decision value for all objects satisfying this part. The conditional parts of minimal rules define largest object sets relevant for approximating decision classes. It turns out that the conditional parts of minimal rules can be computed using prime implicants.

To compute the set of all minimal rules wrt to a decision system $\mathcal{A} = \langle U, A, d \rangle$, we proceed as follows.

For any object $x \in U$:

1. Construct a discernibility function f_x by considering in the decision-relative discernibility matrix for $\mathcal{A}$ the row corresponding to object x, only.

2. Compute all prime implicants of f_x.

3. On the basis of prime implicants create minimal rules corresponding to x. To do this, for each prime implicant I, consider the set $A(I)$ of

[10] A decision rule $\varphi \Rightarrow \psi$ is True in $\mathcal{A}$ iff $\|\varphi\|_{\mathcal{A}} \subseteq \|\psi\|_{\mathcal{A}}$ (see Chapter 5, Section 5.6).

Object	L	W	C	S
1	7.0	large	green	no
2	7.0	large	blue	no
3	4.0	medium	green	yes
4	4.0	medium	red	yes
5	5.0	medium	blue	no
6	4.5	medium	green	no
7	4.0	large	red	no

Table 15.13: Decision table considered in Example 15.6.2

Objects	L	W	S
1, 2	7.0	large	no
3,4	4.0	medium	yes
5	5.0	medium	no
6	4.5	medium	no
7	4.0	large	no

Table 15.14: $\{L, W\}$-reduction considered in Example 15.6.2

attributes corresponding to propositional variables in I, and construct a rule:

$$\left(\bigwedge_{a \in A(I)} (a = a(x)) \right) \Rightarrow d(x).$$

The following example illustrates the idea.

Example 15.6.2 Consider a decision system $\mathcal{A}$ given in Table 15.13 containing values of conditional attributes of vehicles (L, W, C, standing for *Length, Width*, and *Color*, respectively), a decision attribute S standing for *Small* and allowing one to decide whether a given vehicle is small. This system has exactly one decision-relative reduct consisting of attributes L and W. The $\{L, W\}$-reduction of $\mathcal{A}$ is shown as Table 15.14.

Table 15.14 results in the following set of non-minimal decision rules:

$$(L = 7.0) \ \wedge \ (W = \mathsf{large}) \ \Rightarrow \ (S = \mathsf{no})$$
$$(L = 4.0) \ \wedge \ (W = \mathsf{medium}) \ \Rightarrow \ (S = \mathsf{yes})$$
$$(L = 5.0) \ \wedge \ (W = \mathsf{medium}) \ \Rightarrow \ (S = \mathsf{no})$$
$$(L = 4.5) \ \wedge \ (W = \mathsf{medium}) \ \Rightarrow \ (S = \mathsf{no})$$
$$(L = 4.0) \ \wedge \ (W = \mathsf{large}) \ \Rightarrow \ (S = \mathsf{no}).$$

	3	4
1	L, W	L, W, C
2	L, W, C	L, W, C
5	L, C	L, C
6	L	L, C
7	W, C	W

Table 15.15: Reduced decision-relative discernibility matrix from Example 15.6.2

To obtain minimal decision rules, we apply the construction provided above, for $x \in \{1, \ldots, 7\}$.

1. The discernibility functions $f_1, \ldots, f_7$ are constructed on the basis of the reduced discernibility matrix shown in Table 15.15:

$$\begin{aligned}
f_1 &\equiv (L \vee W) \wedge (L \vee W \vee C) \equiv (L \vee W) \\
f_2 &\equiv (L \vee W \vee C) \wedge (L \vee W \vee C) \equiv (L \vee W \vee C) \\
f_3 &\equiv (L \vee W) \wedge (L \vee W \vee C) \wedge (L \vee C) \wedge L \wedge (W \vee C) \\
&\equiv (L \wedge W) \vee (L \wedge C) \\
f_4 &\equiv (L \vee W \vee C) \wedge (L \vee W \vee C) \wedge (L \vee C) \wedge (L \vee C) \wedge W \\
&\equiv (L \wedge W) \vee (C \wedge W) \\
f_5 &\equiv (L \vee C) \wedge (L \vee C) \equiv (L \vee C) \\
f_6 &\equiv L \wedge (L \vee C) \equiv L \\
f_7 &\equiv (W \vee C) \wedge W \equiv W.
\end{aligned}$$

2. The prime implicants corresponding to formulas $f_1, \ldots, f_7$ are the following:

f_1: L, W
f_2: L, W, C
f_3: $L \wedge W$, $L \wedge C$
f_4: $L \wedge W$, $C \wedge W$
f_5: L, C
f_6: L
f_7: W.

3. On the basis of prime implicants we create minimal decision rules for objects $1, \ldots, 7$. For instance, from prime implicants L and W corresponding to f_1, we get the following rules based on object 1:

$$(L = 7.0) \Rightarrow (S = \mathsf{no})$$
$$(W = \mathsf{large}) \Rightarrow (S = \mathsf{no}).$$

On the basis of object 3 and prime implicants $L \wedge W$ and $L \wedge C$ for f_3 we obtain rules:

$$(L = 4.0) \wedge (W = \mathsf{medium}) \Rightarrow (S = \mathsf{yes})$$
$$(L = 4.0) \wedge (C = \mathsf{green}) \Rightarrow (S = \mathsf{yes}).$$

Similarly, the minimal rules can easily be obtained for all other formulas. ■

In practice, the number of minimal decision rules can be large. One thus usually considers subsets of those rules.

The main challenge in inducing rules from decision systems lies in determining which attributes should be included in the conditional part of the rule. Using the strategy outlined above, first the minimal rules are computed. Their conditional parts describe largest object sets with the same generalized decision value in a given decision system. Although such minimal decision rules can be computed, this approach can result in a set of rules of unsatisfactory classification quality. Such detailed rules might appear too strict to classify unseen objects. Shorter rules should rather be synthesized. Although they will not be perfect on the known cases, there is a good chance that they will be of high quality when classifying new objects. They can be constructed by computing approximations of reducts, obtained by dropping some descriptors from the conditional parts of minimal rules. This way one defines larger sets, not purely included in decision classes, but included in them in a satisfactory degree.

15.7 Example: Learning of Concepts

Consider the situation described in Example 7.6, where a concept of traffic congestion was approximated by a combination of rough and crisp knowledge. Approximations of some concepts, like *Speed* or *Distance*, were assumed to be given. Let us now show how classifiers of these notions can automatically be generated on the basis of training data.

Let us concentrate on the concept of *Distance*. Recall that rough relation $Distance(x, y, z)$ denoted the approximate distance between vehicles x and y, where $z \in \{\mathsf{small}, \mathsf{medium}, \mathsf{large}, \mathsf{unknown}\}$. Below we simplify it a little, and consider $Distance(x, z)$ meaning that the distance between x and the vehicle directly preceding x is z, where z is as before.[11] Assume sample training data[12] is gathered in a decision table shown in Table 15.16, where:

[11] In fact, here we consider a distance to be small if it causes a dangerous situation, and to be large if the situation is safe.

[12] Of course, a real-life sample data would consist of hundreds or thousands of examples.

Object	SL	VS	W	AD	$Distance$
1	70	60	rain	3.0	small
2	70	70	sun	5.0	medium
3	50	60	rain	5.0	small
4	50	60	sun	9.0	medium
5	30	15	rain	9.0	large
6	30	10	sun	5.0	large
7	70	60	rain	15.0	large
8	50	40	rain	15.0	large

Table 15.16: Training data considered in Example 15.7.

Object	1	3
2	VS, W, AD	SL, VS, W
4	SL, W, AD	W, AD
5	SL, VS, AD	SL, VS, AD
6	SL, VS, W, AD	SL, VS, W
7	AD	SL, AD
8	SL, VS, AD	VS, AD

Table 15.17: Discernibility matrix of Table 15.16 for decision small.

- SL stands for the "speed limit" on the considered road segment
- VS stands for the "vehicle speed"
- W stands for "weather conditions"
- AD stands for "actual distance" between a given vehicle and its predecessor on the road.

For the sake of simplicity we concentrate on generating rules to determine that the distance between two objects is small.

On the basis of training data one can compute the discernibility matrix. Since we are interested in rules for decision small only, it suffices to consider a simplified discernibility matrix with columns labelled by objects 1 and 3, as only for these objects the corresponding decision is small. The resulting discernibility matrix is shown in Table 15.17.

The discernibility matrix gives rise to the following discernibility functions:

$$\begin{aligned} f_1 &\equiv (VS \vee W \vee AD) \wedge (SL \vee W \vee AD) \wedge (SL \vee VS \vee AD) \\ &\quad \wedge (SL \vee VS \vee W \vee AD) \wedge AD \wedge (SL \vee VS \vee AD) \\ &\equiv AD \end{aligned}$$

$$\begin{aligned} f_3 &\equiv (SL \vee VS \vee W) \wedge (W \vee AD) \wedge (SL \vee VS \vee AD) \\ &\quad \wedge (SL \vee VS \vee W) \wedge (SL \vee AD) \wedge (VS \vee AD) \\ &\equiv (W \wedge AD) \vee (SL \wedge AD) \vee (VS \wedge AD) \vee (SL \wedge VS \wedge W). \end{aligned}$$

¿From the above one can easily find prime implicants and obtain the following rules for decision small:[13]

$$(AD = 3.0) \Rightarrow (Distance = \mathsf{small}) \tag{15.12}$$
$$(W = \mathsf{rain}) \wedge (AD = 5.0) \Rightarrow (Distance = \mathsf{small})$$
$$(SL = 50) \wedge (AD = 5.0) \Rightarrow (Distance = \mathsf{small})$$
$$(VS = 60) \wedge (AD = 5.0) \Rightarrow (Distance = \mathsf{small})$$
$$(SL = 50) \wedge (VS = 60) \wedge (W = \mathsf{rain}) \Rightarrow (Distance = \mathsf{small}).$$

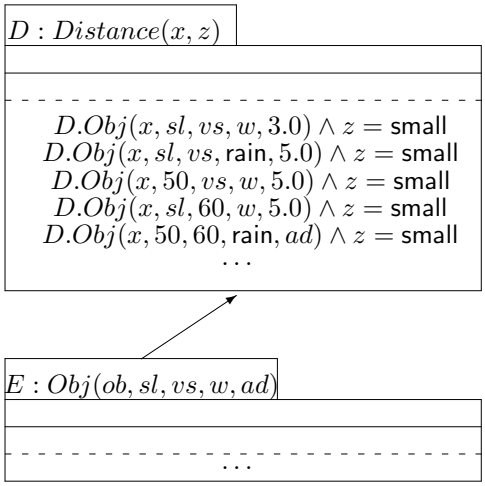

Figure 15.2: CAKE diagram for *Distance*.

On the basis of the generated rules one can construct a CAKE diagram corresponding to the described situation. Assume that incoming information of objects (vehicles) is stored in the extensional database as relation $Obj(ob, sl, vs, w, ad)$ of the structure similar to that of Table 15.16, except that no decision column is now given[14]. The resulting CAKE diagram is shown in Figure 15.2. Of course, the diagram should also contain rules for decisions medium and large obtained by learning process.

[13] In practical applications one would have to discretize AD before creating rules.
[14] Is to be decided on the basis of the obtained rules.

Customer	*Bread*	*Milk*	*Jam*	*Beer*
1	yes	yes	no	no
2	yes	yes	yes	yes
3	yes	yes	yes	no
4	no	yes	yes	no

Table 15.18: Information table considered in Example 15.8.1.

15.8 Association Rules

In this section, we show how rough set techniques can be used to extract association rules from information systems. Association rules, which play an important role in the field of data mining, provide associations among attributes. To each rule there is assigned a real number from the interval [0,1] measuring a confidence of the rule. The following example will help to illustrate this.

Example 15.8.1 Consider an information table given in Table 15.18. Each row in the table represents items bought by a customer. For instance, customer 1 bought bread and milk, whereas customer 4 bought milk and jam. An association rule that can be extracted from the above table is: a customer who bought bread also bought milk. We represent this by

$$(Bread = \mathsf{yes}) \Rightarrow (Milk = \mathsf{yes}).$$

Since all customers who bought bread actually bough milk also, the confidence of this rule is 1. Consider now the rule

$$(Bread = \mathsf{yes}) \wedge (Milk = \mathsf{yes}) \Rightarrow (jam = \mathsf{yes})$$

stating that a customer who bought bread and milk bought jam also. Since three customers bough both bread and milk and two of them bought jam, the confidence of the rule is 2/3. ∎

We now formalize the above idea.

Recall that by a template we mean a conjunction of descriptors, i.e., expressions of the form $a = v$, where a is an attribute and $v \in V_a$.

Definition 15.8.2 Let $\mathcal{A}$ be an information system and T be a template. By $support_{\mathcal{A}}(T)$ we denote the number of objects satisfying T. ∎

Definition 15.8.3 Let $\mathcal{A}$ be an information system and $T = D_1 \wedge \ldots \wedge D_m$ be a template. By an association rule generated from T, we mean any expression

of the form

$$\bigwedge_{D_i \in P} D_i \Rightarrow \bigwedge_{D_j \in Q} D_j,$$

where $\{P, Q\}$ is a partition of $\{D_1, \ldots, D_m\}$.

By a *confidence of association rule* $\phi \equiv \bigwedge_{D_i \in P} D_i \Rightarrow \bigwedge_{D_j \in Q} D_j$ we mean the coefficient

$$confidence_{\mathcal{A}}(\phi) = \frac{support_{\mathcal{A}}(D_1 \wedge \ldots \wedge D_m)}{support_{\mathcal{A}}(\bigwedge_{D_i \in P} D_i)}.$$

■

There are two basic steps in methods aimed at association rule generation. (Below s and c stand for support and confidence thresholds wrt a given information system $\mathcal{A}$).

1. Generate as many templates $T = D_1 \wedge \ldots \wedge D_k$ as possible, such that $support_{\mathcal{A}}(T) \geq s$ and $support_{\mathcal{A}}(T \wedge D_i) < s$, for any descriptor D_i different from all descriptors $D_1, \ldots, D_k$.

2. Search for a partition $\{P, Q\}$ of T, for each T generated in the previous step, satisfying

   (a) $support_{\mathcal{A}}(P) < \frac{support_{\mathcal{A}}(T)}{c}$

   (b) P has the shortest length among templates satisfying (a).

   Each such a partition leads to an association rule of the form $P \Rightarrow Q$ whose confidence is greater than c.

The second step, which is crucial in the process of extracting association rules, can be solved using rough set methods.

Definition 15.8.4 Let $T = D_1 \wedge D_2 \wedge \ldots \wedge D_m$ be a template with $support_{\mathcal{A}}(T) \geq s$. For a given confidence threshold $c \in [0, 1]$, the association rule $\phi \equiv P \Rightarrow Q$ is called *c-irreducible* if $confidence_{\mathcal{A}}(P \Rightarrow Q) \geq c$ and for any association rule $\phi' \equiv P' \Rightarrow Q'$ such that P' is a sub-formula of P, we have $confidence_{\mathcal{A}}(P' \Rightarrow Q') < c$. ■

The problem of searching for c-irreducible association rules from a given template is equivalent to the problem of searching for α-reducts (see Definition 15.2.13), for some α, in a decision table.

Definition 15.8.5 Let $\mathcal{A}$ be an information system and $T = D_1 \wedge D_2 \wedge \ldots \wedge D_m$ be a template. By a *characteristic table* for T wrt $\mathcal{A}$, we understand a decision system $\mathcal{A}|_T = \langle U, A|_T, d \rangle$, where

1. $A|_T = \{a_{D_1}, a_{D_2}, ..., a_{D_m}\}$ is a set of attributes corresponding to the descriptors of T such that

$$a_{D_i}(u) = \begin{cases} 1 & \text{if the object } u \text{ satisfies } D_i, \\ 0 & \text{otherwise.} \end{cases}$$

2. the decision attribute d determines if the object satisfies a template T, i.e.,

$$d(u) = \begin{cases} 1 & \text{if the object } u \text{ satisfies } T, \\ 0 & \text{otherwise.} \end{cases}$$

■

The following theorem provides the relationship between association rules and approximations of reducts.

Theorem 15.8.6 For a given information system $\mathcal{A} = \langle U, A \rangle$, a template $T = D_1 \wedge D_2 \wedge \ldots \wedge D_m$ and a set of descriptors $P \subseteq \{D_1, \ldots, D_m\}$, the association rule

$$\bigwedge_{D_i \in P} D_i \Rightarrow \bigwedge_{D_j \in \{D_1, \ldots, D_m\} - P} D_j$$

is

1. 1-irreducible association rule from T iff $\bigcup_{D_i \in P} \{a_{D_i}\}$ is a decision-relative reduct of $\mathcal{A}|_T$.
2. c-irreducible association rule from T iff $\bigcup_{D_i \in P} \{a_{D_i}\}$ is an α-reduct of $\mathcal{A}|_T$, where

$$\alpha = 1 - \left[\left(\frac{1}{c} - 1 \right) \Big/ \left(\frac{|U|}{support_{\mathcal{A}}(T)} - 1 \right) \right].$$

■

The problem of searching for the shortest association rules is NPTIME-hard.

The following example illustrates the main idea of the method for searching association rules.

Example 15.8.7 Consider the information table $\mathcal{A}$ with 18 objects and 9 attributes presented at the top part of Table 15.19.

Assume that the template

$$T = (a_1 = 0) \wedge (a_3 = 2) \wedge (a_4 = 1) \wedge (a_6 = 0) \wedge (a_8 = 1) \tag{15.13}$$

has been extracted from $\mathcal{A}$. It is easily seen that $support_{\mathcal{A}}(T) = 10$. The new constructed decision table $\mathcal{A}|_T$ is presented in the bottom part of Table 15.19.

The reduced discernibility matrix $\mathcal{A}|_T$ is given as Table 15.20, where for simplicity, the second column represents, in fact, ten columns with identical contents, labeled by $u_2, u_3, u_4, u_8, u_9, u_{10}, u_{13}, u_{15}, u_{16}, u_{17}$, respectively.

Given the discernibility matrix, one easily computes the discernibility function for $\mathcal{A}|_T$:

$$\begin{aligned} f_{\mathcal{A}_T}(a_{D_1}, a_{D_2}, a_{D_3}, a_{D_4}, a_{D_5}) \quad \equiv \quad & (a_{D_2} \vee a_{D_4} \vee a_{D_5}) \\ & \wedge (a_{D_1} \vee a_{D_3} \vee a_{D_4}) \\ & \wedge (a_{D_2} \vee a_{D_3} \vee a_{D_4}) \\ & \wedge (a_{D_1} \vee a_{D_2} \vee a_{D_3} \vee a_{D_4}) \\ & \wedge (a_{D_1} \vee a_{D_3} \vee a_{D_5}) \\ & \wedge (a_{D_2} \vee a_{D_3} \vee a_{D_5}) \\ & \wedge (a_{D_3} \vee a_{D_4} \vee a_{D_5}) \\ & \wedge (a_{D_1} \vee a_{D_5}), \end{aligned}$$

where D_i denotes the i-th conjunct of (15.13).

The discernibility function has the following prime implicants: $a_{D_3} \wedge a_{D_5}$, $a_{D_4} \wedge a_{D_5}$, $a_{D_1} \wedge a_{D_2} \wedge a_{D_3}$, $a_{D_1} \wedge a_{D_2} \wedge a_{D_4}$, $a_{D_1} \wedge a_{D_2} \wedge a_{D_5}$, $a_{D_1} \wedge a_{D_3} \wedge a_{D_4}$. This gives rise to the reducts: $\{a_{D_3}, a_{D_5}\}$, $\{a_{D_4}, a_{D_5}\}$, $\{a_{D_1}, a_{D_2}, a_{D_3}\}$, $\{a_{D_1}, a_{D_2}, a_{D_4}\}$, $\{a_{D_1}, a_{D_2}, a_{D_5}\}$, $\{a_{D_1}, a_{D_3}, a_{D_4}\}$. Thus, there are 6 association rules with confidence 1, i.e., 1-irreducible:

$$\begin{aligned} D_3 \wedge D_5 &\Rightarrow D_1 \wedge D_2 \wedge D_4 \\ D_4 \wedge D_5 &\Rightarrow D_1 \wedge D_2 \wedge D_3 \\ D_1 \wedge D_2 \wedge D_3 &\Rightarrow D_4 \wedge D_5 \\ D_1 \wedge D_2 \wedge D_4 &\Rightarrow D_3 \wedge D_5 \\ D_1 \wedge D_2 \wedge D_5 &\Rightarrow D_3 \wedge D_4 \\ D_1 \wedge D_3 \wedge D_4 &\Rightarrow D_2 \wedge D_5 \end{aligned}$$

For confidence 0.9, we look for α-reducts of the decision table $\mathcal{A}|_T$, where $\alpha = 1 - \left(\frac{1}{0.9} - 1\right) / \left(\frac{18}{10} - 1\right) \approx 0.86$. Hence, we look for a set of descriptors that covers at least $\lceil (18 - 10) * (\alpha) \rceil = \lceil 8 * 0.86 \rceil = 7$ elements

$\mathcal{A}$	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
u_1	0	1	1	1	80	2	2	2	3
u_2	0	1	2	1	81	0	aa	1	aa
u_3	0	2	2	1	82	0	aa	1	aa
u_4	0	1	2	1	80	0	aa	1	aa
u_5	1	1	2	2	81	1	aa	1	aa
u_6	0	2	1	2	81	1	aa	1	aa
u_7	1	2	1	2	83	1	aa	1	aa
u_8	0	2	2	1	81	0	aa	1	aa
u_9	0	1	2	1	82	0	aa	1	aa
u_{10}	0	3	2	1	84	0	aa	1	aa
u_{11}	0	1	3	1	80	0	aa	2	aa
u_{12}	0	2	2	2	82	0	aa	2	aa
u_{13}	0	2	2	1	81	0	aa	1	aa
u_{14}	0	3	2	2	81	2	aa	2	aa
u_{15}	0	4	2	1	82	0	aa	1	aa
u_{16}	0	3	2	1	83	0	aa	1	aa
u_{17}	0	1	2	1	84	0	aa	1	aa
u_{18}	1	2	2	1	82	0	aa	2	aa

$\mathcal{A}\|_T$	a_{D_1}	a_{D_2}	a_{D_3}	a_{D_4}	a_{D_5}	d
	$(a_1 = 0)$	$(a_3 = 2)$	$(a_4 = 1)$	$(a_6 = 0)$	$(a_8 = 1)$	
u_1	1	0	1	0	0	0
u_2	1	1	1	1	1	1
u_3	1	1	1	1	1	1
u_4	1	1	1	1	1	1
u_5	0	1	0	0	1	0
u_6	1	0	0	0	1	0
u_7	0	0	0	0	1	0
u_8	1	1	1	1	1	1
u_9	1	1	1	1	1	1
u_{10}	1	1	1	1	1	1
u_{11}	1	0	1	1	0	0
u_{12}	1	0	0	1	0	0
u_{13}	1	1	1	1	1	1
u_{14}	1	1	0	0	0	0
u_{15}	1	1	1	1	1	1
u_{16}	1	1	1	1	1	1
u_{17}	1	1	1	1	1	1
u_{18}	0	1	1	1	0	0

Table 15.19: Information table $\mathcal{A}$ and decision table $\mathcal{A}|_T$.

$\mathcal{M}(\mathcal{A}\|_T)$	u_2, u_3, u_4, u_8, u_9 $u_{10}, u_{13}, u_{15}, u_{16}, u_{17}$
u_1	$a_{D_2}, a_{D_4}, a_{D_5}$
u_5	$a_{D_1}, a_{D_3}, a_{D_4}$
u_6	$a_{D_2}, a_{D_3}, a_{D_4}$
u_7	$a_{D_1}, a_{D_2}, a_{D_3}, a_{D_4}$
u_{11}	$a_{D_1}, a_{D_3}, a_{D_5}$
u_{12}	$a_{D_2}, a_{D_3}, a_{D_5}$
u_{14}	$a_{D_3}, a_{D_4}, a_{D_5}$
u_{18}	a_{D_1}, a_{D_5}

Table 15.20: Reduced discernibility matrix for $\mathcal{A}|_{\mathbf{T}}$ form Example 15.8.7

of discernibility matrix $\mathcal{M}(\mathcal{A}|_{\mathbf{T}})$. One can see that the following sets of descriptors: $\{D_1, D_2\}$, $\{D_1, D_3\}$, $\{D_1, D_4\}$, $\{D_1, D_5\}$, $\{D_2, D_3\}$, $\{D_2, D_5\}$, $\{D_3, D_4\}$ have nonempty intersection with exactly 7 members of the discernibility matrix $\mathcal{M}(\mathcal{A}|_{\mathbf{T}})$. Consequently, the 0.9-irreducible association rules obtained from those sets are the following:

$$
\begin{aligned}
D_1 \wedge D_2 &\Rightarrow D_3 \wedge D_4 \wedge D_5 \\
D_1 \wedge D_3 &\Rightarrow D_2 \wedge D_4 \wedge D_5 \\
D_1 \wedge D_4 &\Rightarrow D_2 \wedge D_3 \wedge D_5 \\
D_1 \wedge D_5 &\Rightarrow D_2 \wedge D_3 \wedge D_4 \\
D_2 \wedge D_3 &\Rightarrow D_1 \wedge D_4 \wedge D_5 \\
D_2 \wedge D_5 &\Rightarrow D_1 \wedge D_3 \wedge D_4 \\
D_3 \wedge D_4 &\Rightarrow D_1 \wedge D_2 \wedge D_5.
\end{aligned}
$$

■

15.9 Bibliographic Notes

Machine learning has a long tradition in artificial intelligence. The books [99, 95] provide a good introduction to the subject.

A variety of methods for computing reducts and their applications can be found in [148, 83, 115, 125, 126, 121, 175, 154]). The fact that the problem of finding a minimal reduct of a given information system is NPTIME-hard has been proved in [148].

As we mentioned, there exists a number of good heuristics that compute sufficiently many reducts in an acceptable time. Moreover, a successful methodology, based on different reducts, has been developed for solution of many