národní
úložiště
šedé
literatury

A Relational Framework for Data Mining

Holeňa, Martin
2004

Institute of Computer Science
Academy of Sciences of the Czech Republic

A Relational Framework for Data Mining

Martin Holeňa

Technical report No. 909

May 2004

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 053 111, fax: (+4202) 86 585 789,
e-mail:ics@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

A Relational Framework for Data Mining[1]

Martin Holeňa

Technical report No. 909

May 2004

Abstract:

The last decade has witnessed a fast increase of the repertoir of available sophisticated data mining methods, based on a broad spectrum of quite diverse paradigms. Besides a number of traditional and recent statistical methods, increasingly frequent are various kinds of graphical dependence models, classification and regression trees, as well as methods based on nonstatistical paradigms, such as artificial neural networks, inductive logic programming, fuzzy sets theory or rough sets theory. Such a diversity leads to problems when interpreting, comparing and consolidating results obtained with different methods. Therefore, a unifying framework of view for different data mining methods would be very useful. Several frameworks of that kind have indeed been proposed in recent years, and also the present paper is a contribution in that direction. It proposes a framework based on the theory of relations, conceived not only in the classical sense of set-theoretical relations, but in the broader sense of fuzzy sets. Underlying assumptions and basic principles of the framework are explained, and a survey of the main classes of data mining methods to which it is applicable is given. The survey points a specificity of applying the framework to the extraction of rules from data by means of artificial neural networks. The applicability of the framework is documented through elaborating it for two particular data mining methods. One of them is the method GUHA, a classical method of exploratory data analysis, the other is a method for rule extraction by means of piecewise-linear multilayer perceptrons.

Keywords:
Data mining frameworks, rules extraction from data, method Guha, observational calculus, neural-networks based rules extraction

[1]Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod vodárenskou věží 2, CZ-182 07 Praha 8, e-mail: martin@cs.cas.cz

[1]The research reported in this paper has been supported by the COST Action 274, "Theory and Application of Relational Structures as Knowledge Instruments (TARSKI)"

1 Introduction

The term *data mining* emerged some 20 years ago in statistics. Originally, it was used rather derogatorily, to denote a search for the best fitting model or the most significant hypothesis by trying a large number of models or hypotheses on the same body of data, not worrying about the interpretability and reproducibility of the obtained results [47, 13]. Nowadays, data mining belongs to the most prominent information technologies, experiencing a boom of interest from users and software producers. During the late 90s, companies in commerce, finance and other sectors started to introduce specialized data mining systems, and many important software producers launched their own systems of that kind. Meanwhile, the meaning of the term has shifted and it is nowadays mostly understood as *searching large amounts of data for some formalized patterns of interest* [20, 18].

In connection with data mining, another term, originating in artificial intelligence, is very often used – *knowledge discovery.* Differently to data mining, knowledge discovery is usually understood as the overall process of extracting from data what is deemed to be the inherently present knowledge [7, 18]. In addition to the data mining step, that process covers a number of further, interactively interleaved steps, most importantly:

- data selection;
- data cleaning;
- data reduction;
- data projection and feature selection;
- incorporating prior knowledge;
- evaluation of the found patterns;
- interpreting the obtained results;
- consolidating the results with previous knowledge.

Needless to say, data mining is the core step of the knowledge discovery process. However, to be able to really discover new valid knowledge, data mining needs to be appropriately combined with the remaining steps. Otherwise, it degrades to its original derogatory meaning.

Traditionally, analysing data and extracting useful knowledge from them has been a domain of statisticians. Therefore, many methods used in data mining are actually statistical data analysis methods, most of them known for decades before data mining emerged (see [16] or [30] for a survey), though several statistical methods important in data mining have been elaborated only during the last 20–30 years [23, 17, 22]. In addition, a statistical component is crucially important in graphical dependency models, belonging primarily to the area of combinatorial graph theory [56, 66], and in various kinds of classification and regression trees, which belong primarily to machine learning [8, 58]. Moreover, since data mining sometimes needs to be performed in situations when rigorous statistical methods are not applicable, it has encouraged the development of alternative data analysis approaches, relying on artificial neural networks, fuzzy logic and other comparatively recent approaches [46, 63, 64], cf. Figure 1.1. The diversity of existing data mining methods inevitably entails problems when interpreting, comparing and consolidating results obtained with different of them. To face those problems, a unifying framework of view for different data mining methods can be very useful, and several frameworks of that kind have been indeed proposed in recent years [3, 44, 69]. However, none of them has found a broader acceptance, indicating that further research in this area is still needed. In this paper, a framework based on the theory of relations is proposed, which is simple yet covers the broad spectrum of all those methods whose results can be expressed in the language of some formal logic. The basic principles of the approach are explained in the next section and illustrated on a classical method of exploratory data analysis, the method GUHA, in Section 3. Then Section 4 shows that the framework implies a specific position of methods for knowledge extraction from data by means of artificial neural networks because they use an intermediate representation at an intermediate stage between data and rules. That specificity is illustrated on a method for rule extraction by means of piecewise-linear multilayer perceptrons.

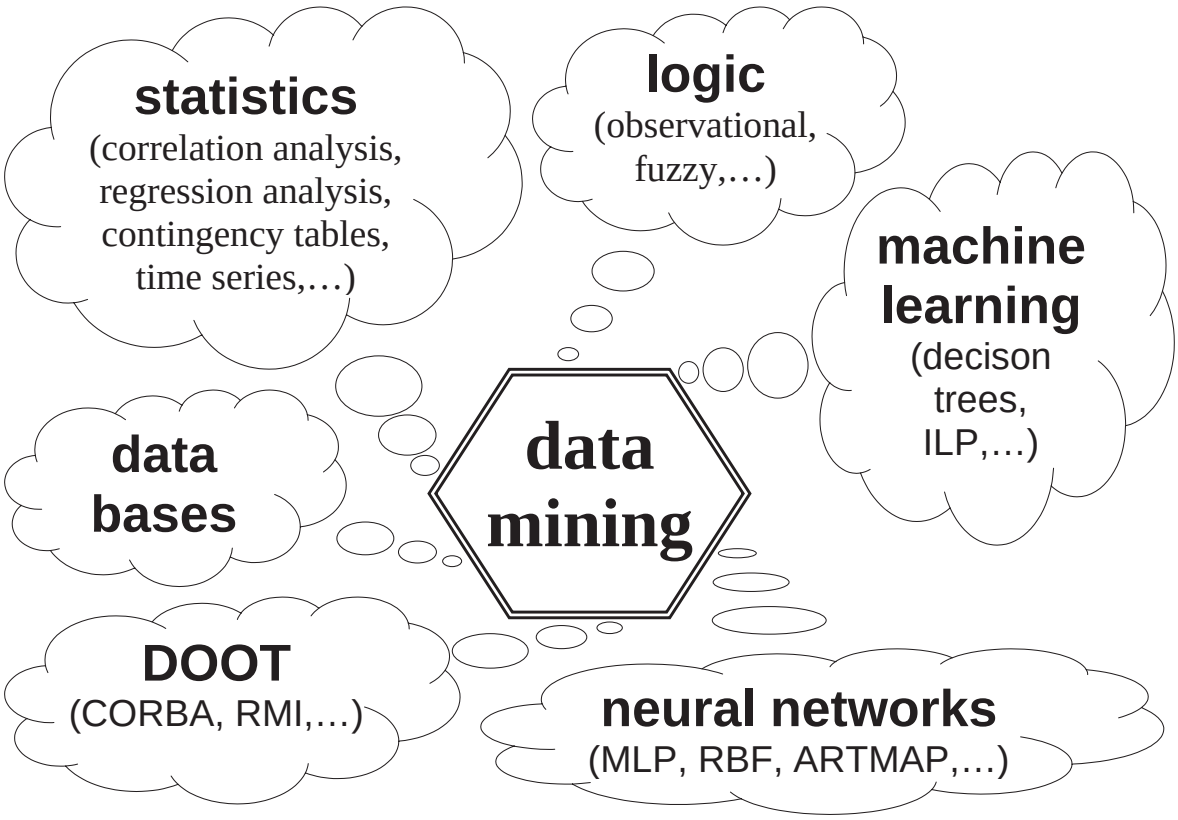

Figure 1.1: Main data mining approaches and supporting technologies

2 Data mining viewed as a transformation of relations

The proposed framework is based on the following assumptions:

(i) The *input* to each data mining method consists of *tuples of data values*. Each component of each tuple corresponds to a particular attribute, and is possibly known only with some incomplete certainty. The same input may occur several times in the method. Finally, the number of components can vary between tuples, to accomodate for missing values and not applicable attributes.

(ii) The output from a data mining method is assumed to be expressible as sentences of some formal logic, which in the context of data mining are usually called *logical rules*. Although this assumption does not hold for all data mining methods, it still covers many methods based on quite diverse paradigms. Examples will be given in the sequel. Moreover, every covered method is assumed to be coherent, in the sense that for any input, the set of sentences expressing the output is not contradictory. Notice that the expressibility in the language of some logic does not imply that existing implementations of the method really output results expressed in that way. For comprehensibility reasons, other representations may be preferred, most often various graphical representations. The probably best known example of such situation are classification trees [8, 58, 5].

The input tuples are nowadays usually obtained from a relational database – being either directly stored there as rows of a particular table, or derived from stored tables through the application of relational operators. In the past, they were typically obtained from data matrices stored in flat files, or through navigational operators from records stored in hierarchical and network databases. As far as the output is concerned, in all existing methods whose outputs are expressible using a formal logic, that logic is always either the classical Boolean logic or its generalization, such as the observational

calculus or some fuzzy logic. To simplify the discussion in the sequel, let us add this restriction to the assumption (ii) above.

The fact that each component of each input tuple corresponds to a particular attribute implies that each input tuple is an element of the product space of value sets of particular attributes. To record possible multiple occurrences of each tuple, an auxiliary attribute can be added, and the product space can be extended with the value set of this attribute – the set of natural numbers. Consequently, all input tuples whose components correspond to the same set of attributes together with their number of occurrences form an extensionally described relation in such an extended product space. For example, think of mining data from a patient database. Then the components of the input values record, e.g., results of laboratory tests, subjective complaints of the patient, medical diagnoses. Especially the last two kinds of values may be known only with incomplete certainty. Different sets of data are due to different tables of the database (such as "Family History", "Laboratory Tests", "Treatments"), and due to values that are either missing (a particular laboratory test was not performed) or not applicable (e.g., attributes concerning pregnancy if sex=male). In all cases, tuples with components corresponing to the same attributes together with the number of their occurrences form a particular input relation.

On the other hand, the interpretation of an n-ary predicate used to express the output from a data mining method is always an intensionally described relation in the product space of some n sets serving as domains of individual object variables. The kind of that relation depends on the considered logic – for the classical Boolean logic, it is a traditional set-theoretic relation ("crisp" relation), whereas for a fuzzy logic, it is a fuzzy set on the product space of the sets interpreting the individual variables. As far as the semantics of the individual components of that space is concerned, i.e., the choice of the domains of individual object variables, it depends exclusively on the nature and purpose of the considered data mining method. Often, they are again the value sets of some variables, maybe even value sets of the input attributes. However, if the purpose of the data mining method is to state some probabilistic properties of the input data, then sets of probability distributions are involved as components of the product space in which the output relation lies (cf. Section 3.1).

These fundamental observations already suggest that relations can serve as a means to abstractly describe arbitrary data mining methods. Moreover, they suggest that each data mining method can be viewed, basically, as a transformation of extensionally described input data relations into intensionally described interpretations of output predicates. Stated in a formal way, if V_A denotes the value set of an attribute A, D_x denotes the domain of an object variable x, and $\mathcal{P}(S)$ denotes the power set of a set S, then each data mining method can be viewed, for an appropriate choice of attributes $A_1, \ldots, A_m$ and object variables $x_1, \ldots, x_n$, as a transformation

$$\mathcal{M} : \mathcal{R}_{\mathcal{M}} \to \mathcal{P}(\bigotimes_{j=1}^{n} D_{x_j}), \tag{2.1}$$

where $\mathcal{R}_{\mathcal{M}} \subset \mathcal{P}(\bigotimes_{i=1}^{m} V_{A_i} \times \mathcal{N})$ denotes the set of input data relations acceptable by the method.

Nonetheless, this basic scheme needs some details to be clarified:

- According to assumption (i), different input tuples of data values may have different numbers of components. Moreover, also tuples with equal number of components may correspond to different sets of attributes. Similarly, different output sentences may contain different sets of object variables. The basic transformation scheme (2.1) can not accomodate variability in sets of attributes and sets of object variables. However, a standard remedy helps – to form an extended product space from the value sets of all attributes corresponding to any input data tuple, and another extended product space from the domains of all object variables involved in any output sentence, and to establish a correspondence between the relations occurring in (2.1) and relations in those product spaces by means of cylindric extensions and projections. Only after the input data relations and interpretations of output sentences have been projected to the

extended product spaces, the transformation (2.1) is applied:

$$\begin{array}{ccc} \mathcal{R} & \xrightarrow[\text{transformation}]{\text{final}} & \mathcal{P}(\bigotimes_{j=1}^{N} D_{x_j}), \\ {\scriptstyle\text{cylindric extensions}}\uparrow & & \downarrow{\scriptstyle\text{projections}} \\ \mathcal{R}_{\mathcal{M}} & \xrightarrow[\text{transformation}]{\text{original}} & \mathcal{P}(\bigotimes_{j=1}^{n}(D_{x_j}) \end{array} \tag{2.2}$$

where M denotes the number of all attributes corresponding to any input data tuple, N denotes the number of all object variables involved in any output sentence, and $\mathcal{R} \subset \mathcal{P}(\bigotimes_{i=1}^{M} V_{A_i} \times \mathcal{N})$.

- The assumption that each component of each input tuple may be known only with some certainty implies that the extensional input relation is actually a fuzzy relation on the value sets of all attributes corresponding to any input data tuple. Similarly, as we already know, the intensional output relation can be a fuzzy relation on the domains of object variables involved in any output sentence. Needless to say, fuzzy relations already cover, as their special cases, also set-theoretic relations. Denoting $\mathcal{F}(S)$ the set of all fuzzy subsets of a set S, (2.2) finally turns to

$$\begin{array}{ccc} \mathcal{R} & \xrightarrow[\text{transformation}]{\text{final}} & \mathcal{F}(\bigotimes_{j=1}^{n} D_{x_j}), \\ {\scriptstyle\text{cylindric}\atop\scriptstyle\text{extensions}}\uparrow & & \downarrow{\scriptstyle\text{projections}} \\ \mathcal{R}_{\mathcal{M}} & \xrightarrow[\text{transformation}]{\text{original}} & \mathcal{F}(\bigotimes_{j=1}^{n} D_{x_j}) \end{array} \tag{2.3}$$

where the sets $\mathcal{R}_{\mathcal{M}}$ and $\mathcal{R}$ of input data relations are now sets of fuzzy relations, i.e., $\mathcal{R}_{\mathcal{M}} \subset \mathcal{F}(\bigotimes_{i=1}^{m} V_{A_i} \times \mathcal{N})$, $\mathcal{R} \subset \mathcal{F}(\bigotimes_{i=1}^{M} V_{A_i} \times \mathcal{N})$.

3 Direct methods

For most data mining methods to which the above proposed relational framework is applicable, i.e., for most data mining methods that fulfill the assumption (ii) stated at the beginning of the previous section, the scheme (2.2) or its fuzzy extension (2.3) are already sufficient to completely describe the application of the method to data from a relational point of view. In those methods, no other relations or transformations between relations can be identified, the extensionally described input data relations are transformed directly into the intensionally described interpretations of output predicates. Therefore, the term *direct methods* will be used in connection with this majority situation. A brief survey of those methods will be given below in Subsection 3.2. Before in Subsection 3.1, one particular method will be presented in some detail and used to illustrate the applicability of the proposed approach.

3.1 Example 1 – the method GUHA

The example to be presented here is the method *GUHA (General unary hypotheses automaton)*. That example has not been chosen arbitrarily – GUHA is presumably the oldest method for automated extraction of sentences of a formal logic from data. It originated in the mid-sixties [27], and its development has been basically finished in the late seventies, witnessed with the monograph [28], with two issues of the International Journal of Man-Machine Studies that focused on GUHA [39, 40], and with an elaborate mainframe implementation. But even nowadays, several PC implementations of the method exist and are used in real-world applications (cf. the survey papers [31, 35, 29]).

As input, GUHA takes only binary data matrices. However, each implementation of the method incorporates some preprocessing routines for discretization of continuous attributes and dichotomization of nominal or discretized attributes. As output, GUHA extracts sentences of a two-valued predicate logic. Those sentences concern the (finite or infinite) population from which the data originated, more precisely the probability distributions of various properties of that population. But to deduce such sentences, GUHA makes use also of sentences concerning only the observed data. To differentiate

between both kinds of sentences, sentences concerning the whole population are called *theoretical sentences* in GUHA, whereas sentences concerning the observed data are called *observational sentences*. To combine both kinds of sentences, GUHA uses the observational calculus, which extends the traditional Boolean logic in two respects [28]:

(i) The classical quantifiers $\forall$ and $\exists$, which are used to form sentences concerning the whole considered population, are extended with *generalized quantifiers*, used to form sentences concerning the observed data. Like the classical quantifiers, also generalized quantifiers bind free variables in open formulae. Since they concern finite data sets, sentences of the observational calculus are interpreted in finite Boolean structures, or equivalently, in sets of binary matrices. To this end, a specific binary function $\mathrm{Tf}_\sim$ on the set of all binary matrices with m columns is related to each m-ary generalized quantifier $\sim$, i.e., $\mathrm{Tf} : \bigcup_{n\in\mathcal{N}}\{0,1\}^{n,m} \to \{0,1\}$, and this function is called *truth function* of $\sim$. Using $\mathrm{Tf}_\sim$, the observational sentence $(\sim x)\ (\psi_1(x),\ldots,\psi_m(x))$, built from the quantifier $\sim$ and formulae $\psi_1,\ldots,\psi_m$ free in x evaluates in the available data on n objects as

$$\|(\sim x)\ (\psi_1(x),\ldots,\psi_m(x))\| = \mathrm{Tf}_\sim(\|\psi_1\|,\ldots,\|\psi_m\|), \tag{3.1}$$

where for $j = 1,\ldots,m$, $\|\psi_j\|$ denotes the evaluation of $\psi_j(x)$ in the data, i.e., a column vector of length n consisting of the evaluations of $\psi_j(x)$ on the n individual objects.

(ii) In addition to the classical *deduction rule* modus ponens

"from Φ and $\Phi \to \Psi$, deduce Ψ",

an arbitrary number of additional inference rules may be employed, which provide the possibility to deduce, from theoretical sentences concerning the probability distributions of properties of the population underlying the observed data, and from observational sentences concerning the data themselves, another theoretical sentence concerning those probability distributions. Hence, those inference rules always have the form

$$\text{"from theoretical assumptions } \bigwedge\mathcal{A} \text{ and observations/data } \bigwedge\mathcal{D},\quad \text{deduce a theoretical sentence } TS\text{"}, \tag{3.2}$$

where $\bigwedge\mathcal{A} = TS_1 \wedge \cdots \wedge TS_a$ is the conjunction of a set of theoretical sentences expressing the considered theoretical assumptions, $\mathcal{A} = \{TS_1,\ldots,TS_a\}$, whereas $\bigwedge\mathcal{D} = OS_1 \wedge \cdots \wedge OS_d$ is the conjunction of a set of observational sentences, $\mathcal{D} = \{OS_1,\ldots,OS_a\}$, found valid in the observed data. A particular inference rule is typically related to a particular generalized quantifier or a small number of related quantifiers. Examples will be given below.

To apply the framework presented in Section 2 to the GUHA method is straightforward, provided the following conditions are met:

(i) the theoretical sentence TS is a statement about relationships between probability distributions (it is immaterial whether also any of the theoretical sentences $TS_1 \ldots TS_a$ is such a statement) ;

(ii) at least one of those probability distributions is on the value set V_A of some input data attribute $A \in \{A_1,\ldots,A_m\}$ or on the cartesian product $V_{A_{i_1}} \times \cdots \times V_{A_{i_k}}$ of several distinct input attributes $V_{A_{i_1}} \ldots V_{A_{i_k}} \in \{A_1,\ldots,A_m\}$;

(iii) the employed inference rule connects the validity of the sentences $OS_1,\ldots,OS_d$ with the probability distribution on V_A, respectively with the probability distribution on $V_{A_{i_1}} \times \cdots \times V_{A_{i_k}}$ (such a connection can be of various kinds, examples will be given below).

Indeed, in virtue of the conditions (i)–(iii), TS is interpreted by means of a relation on sets of probability distributions, and there is a connection between that relation and the input data relation. It is this connection that determines the transformation of the extensionally described input relation to the intensionally described interpretation of TS in the GUHA method.

Although the generalized quantifiers and additional inference rules of GUHA can be defined in an arbitrarily abstract way compatible with (3.1) and (3.2), and they do not necessarily have to fulfill the above conditions [59], in an overwhelming majority of practical applications of the method they fulfill them. The reason is that the quantifiers and inference rules encountered in applications nearly

always rely on statistical fundamentals, and statistics leads in a natural way to the validity of such conditions. In the remainder of this section, main statististics-based generalized quantifiers of GUHA and their related inference rules will be reviewed. All of them fulfill the above conditions (i)–(iii), hence they allow a straightforward application of the relational framework of Section 2.

First, all generalized quantifiers encountered in the existing implementations of GUHA are binary. Moreover, their truth functions have the specific property of being the composition of a Boolean function on quadruples of nonnegative integers with the four-fold table of evaluations of the formulae to which the quantifier is applied. Using a simplified notation $\varphi \sim \psi$ instead of the rigorous $(\sim x)\,(\varphi(x), \psi(x))$ for the observational sentence built from a binary generalized quantifier $\sim$ and formulae φ, ψ, and calling the function on quadruples of nonnegative integers corresponding to $\sim$ for simplicity again truth function of $\sim$, $\mathrm{Tf}_\sim$ (i.e., now $\mathrm{Tf}_\sim : \mathcal{N}_0^4 \to \{0,1\}$), the evaluation (3.1) reads

$$\|\varphi\| = \mathrm{Tf}_\sim(a, b, c, d), \tag{3.3}$$

where

$$a = a_{\varphi,\psi} = \#\{i : \|\varphi\|_i = \|\psi\|_i = 1\} \tag{3.4}$$
$$b = b_{\varphi,\psi} = \#\{i : \|\varphi\|_i = 1 \;\&\; \|\psi\|_i = 0\} \tag{3.5}$$
$$c = c_{\varphi,\psi} = \#\{i : \|\varphi\|_i = 0 \;\&\; \|\psi\|_i = 1\} \tag{3.6}$$
$$d = d_{\varphi,\psi} = \#\{i : \|\varphi\|_i = \|\psi\|_i = 0\}, \tag{3.7}$$

using the notation $\#S$ for the number of elements of a set S.

Second, to connect the validity of the observational sentences with the probability distribution on the value sets of the involved attributes, the additional inference rules employed in GUHA always start with the usual statistical assumption that the input data are a random sample, i.e., a sequence of independent identically distributed random vectors. Notice that then also the sequence of evaluations $(\|\varphi\|_i, \|\psi\|_i)_{i=1}^n$ is a random sample, more precisely a two-dimensional binary random sample, whereas $a = a_{\varphi,\psi}$, $b = b_{\varphi,\psi}$, $c = c_{\varphi,\psi}$, $d = d_{\varphi,\psi}$ and, consequently, also $\mathrm{Tf}_\sim(a, b, c, d)$ are random variables.

Using the notation $\Pi(S)$ for the set of Borel probability distributions on a space S, these two fundamental features of GUHA entail the possibility to express the application of a binary quantifier $\sim$ to a data matrix IR serving as an extensional input relation as the value $\mathcal{M}(IR)$ of a relational transformation $\mathcal{M} : \mathcal{P}(\bigotimes_{i=1}^m V_{A_i} \times \mathcal{N}) \to \Pi(\bigotimes_{i=1}^m V_{A_i}) \otimes \Pi(\{0,1\}^2)$, such that

$$\mathcal{M}(IR) \subset \{(P_{\otimes V_{A_i}}, P_{\{0,1\}^2}) : P_{\otimes V_{A_i}} \in \Pi(\bigotimes_{i=1}^m V_{A_i}) \;\&\; P_{\{0,1\}^2} \in \Pi(\{0,1\}^2) \;\&\; P_{\{0,1\}^2} \text{ is induced from } P_{\otimes V_{A_i}} \text{ by } (\|\varphi\|_i, \|\psi\|_i)\}. \tag{3.8}$$

For individual generalized quantifiers, (3.8) is always complemented with some specific additional condition, reflecting the definition of the quantifier's truth function and the related inference rule.

The most simple among the generalized quantifiers employed in GUHA is the *founded implication* with threshold $\theta \in (0,1)$, in symbols $\to_\theta$. Its truth function is defined:

$$\mathrm{Tf}_{\to_\theta}(a, b, c, d) = \begin{cases} 1 & \text{if } a \geq 1 \;\&\; \frac{a}{a+b} > \theta, \\ 0 & \text{else.} \end{cases} \tag{3.9}$$

To that quantifier, the following inference rule is related:

"If a consistent unbiased estimate of the conditional probability of $\|\psi\|_i$ evaluating as 1 conditioned on $\|\varphi\|_i$ evaluating as 1 is greater than θ, then the probability distribution $P_{(\varphi,\psi)}$ of $(\|\varphi\|_i, \|\psi\|_i)$ is such that $P_{(\varphi,\psi)}(\|\psi\|_i = 1 | \|\varphi\|_i = 1) > \theta$". (3.10)

Due to the above inference rule and due to the fact that $\frac{a}{a+b}$ indeed is a consistent unbiased estimate of $\|\psi\|_i$ evaluating as 1 conditioned on $\|\varphi\|_i$ evaluating as 1, the condition (3.8) concerning the value

that the relational transformation $\mathcal{M}$ takes for the input relation IR is complemented with

$\mathrm{Tf}_{\rightarrow_\theta}(a,b,c,d) = 1$ iff the conditional probability distribution
$P_{\{0,1\}|\{0,1\}}(1|1)$ corresponding to the second component
of $\mathcal{M}(IR) = (P_{\bigotimes V_{A_i}}, P_{\{0,1\}^2})$ fulfills $P_{\{0,1\}|\{0,1\}}(1|1) \geq \theta$. (3.11)

Another quantifier related to estimation is the *simple quantifier* with threshold $\delta > 0$, denoted $\sim_\delta$. Its truth function is

$$\mathrm{Tf}_{\sim_\delta}(a,b,c,d) = \begin{cases} 1 & \text{if } \min\{a,b,c,d\} \geq 1 \ \& \\ & \ \& \ \frac{ad}{bc} > \exp(\delta), \\ 0 & \text{else,} \end{cases} \tag{3.12}$$

and the related inference rule is

"If a consistent estimate of the logarithmic interaction of the vector
$(\|\varphi\|_i, \|\psi\|_i)$ is greater than δ, then the probability distribution $P_{(\varphi,\psi)}$
of $(\|\varphi\|_i, \|\psi\|_i)$ is such that the logarithmic interaction of $(\|\varphi\|_i, \|\psi\|_i)$
corresponding to $P_{(\varphi,\psi)}$ is greater than δ. " (3.13)

Taking into consideration the fact that $\ln \frac{ad}{bc}$ is a consistent estimate of the logarithmic interaction of the vector $(\|\varphi\|_i, \|\psi\|_i)$, the condition (3.8) for the value $\mathcal{M}(IR) = (P_{\bigotimes V_{A_i}}, P_{\{0,1\}^2})$ of the relational transformation $\mathcal{M}$ is complemented with

$$\mathrm{Tf}_{\rightarrow_\theta}(a,b,c,d) = 1 \text{ iff } \ln \frac{P_{\{0,1\}^2}(1,1)P_{\{0,1\}^2}(0,0)}{P_{\{0,1\}^2}(1,0)P_{\{0,1\}^2}(0,1)} > \delta. \tag{3.14}$$

Several further important quantifiers are based on statistical hypotheses testing. All of them share the related inference rule, which could be paraphrased *reject distributions making the obtained observations unlikely:*

"If the realization of the random sample $(\|\varphi\|_i, \|\psi\|_i)_{i=1}^n$
corresponding to the extensionally described
input relation IR lies within a given set $C_n \subset \{0,1\}^{n,2}$,
then $P_{\{0,1\}^2}$ is not such that $P^n_{\{0,1\}^2}(C_n)$ would be small". (3.15)

The most simple among those quantifiers is the *lower critical implication* (someitmes called also *likely implication*) with threshold $\theta \in (0,1)$ and significance level $\alpha \in (0,1)$, in symbols $\rightarrow^!_{\theta,\alpha}$. It has the truth function

$$\mathrm{Tf}_{\rightarrow^!_{\theta,\alpha}}(a,b,c,d) = \begin{cases} 1 & \text{if } \sum_{i=a}^{a+b} \binom{a+b}{i} \theta^i (1-\theta)^{a+b-i} \leq \alpha, \\ 0 & \text{else.} \end{cases} \tag{3.16}$$

Taking into account (3.3), the definition (3.16) of the truth function $\mathrm{Tf}_{\rightarrow^!_{\theta,\alpha}}$ can be reformulated in statistical terms as

$\mathrm{Tf}_{\rightarrow^!_{\theta,\alpha}}(a,b,c,d) =$

$$= \begin{cases} 1 & \text{if the binomial test rejects at the significance level } \alpha \text{ the} \\ & \text{null hypothesis } P_{\psi|\varphi} \leq \theta \text{ against the alternative } P_{\psi|\varphi} > \theta \\ & \text{for conditional probability } P_{\psi|\varphi} \text{ of } \|\psi\|_i \text{ evaluating as 1} \\ & \text{conditioned on } \|\varphi\|_i \text{ evaluating as 1,} \\ 0 & \text{else.} \end{cases} \tag{3.17}$$

Provided the chosen significance level (e.g., 0.05, 0.01) can be considered small, combining this definition with the inference rule (3.15) leads to the following additional condition for the value that the relational transformation $\mathcal{M}$ takes for the data matrix IR:

$$\begin{aligned}\mathrm{Tf}_{\rightarrow^{!}_{\theta,\alpha}}(a,b,c,d) = 1 \text{ iff}\\ \text{the conditional probability distribution } P_{\{0,1\}|\{0,1\}}(1|1)\\ \text{corresponding to the second component of}\\ \mathcal{M}(IR) = (P_{\bigotimes V_{A_i}}, P_{\{0,1\}^2}) \text{ fulfills } P_{\{0,1\}|\{0,1\}}(1|1) > \theta. \quad (3.18)\end{aligned}$$

Another often used quantifier based on statistical hypotheses testing is the *Fisher quantifier* with significance level $\alpha \in (0,1)$, denoted $\sim^{F}_{\alpha}$. Its truth function is defined

$$\mathrm{Tf}_{\sim^{F}_{\alpha}}(a,b,c,d) = \\ = \begin{cases} 1 & \text{if } ad > bc \;\&\\ & \sum_{i=a}^{\min(a+b,a+c)} \frac{(a+b)!(a+c)!(b+d)!(c+d)!}{(a+b+c+d)!i!(a+b-i)!(a+c-i)!(d+i-a)!} \leq \alpha, \\ 0 & \text{else,}\end{cases} \quad (3.19)$$

which in statistical terms means

$$\mathrm{Tf}_{\sim^{F}_{\alpha}}(a,b,c,d) = \\ = \begin{cases} 1 & \text{if the one-sided Fisher test rejects at the significance level } \alpha\\ & \text{the null hypothesis of independence of marginal probability}\\ & \text{distributions } P_{\varphi}, P_{\psi} \text{ of } \|\varphi\|_i \text{ and } \|\psi\|_i, \text{ respectively, against the}\\ & \text{alternative of a positive logarithmic interaction of } (\|\varphi\|_i, \|\psi\|_i),\\ 0 & \text{else.}\end{cases} \quad (3.20)$$

If the chosen significance level can be considered small, then combining (3.20) with (3.15) yields the additional condition for the value $clM(IR)$ of the relational transformation $\mathcal{M}$:

$$\mathrm{Tf}_{\rightarrow^{F}_{\alpha}}(a,b,c,d) = 1 \text{ iff } \ln \frac{P_{\{0,1\}^2}(1,1)P_{\{0,1\}^2}(0,0)}{P_{\{0,1\}^2}(1,0)P_{\{0,1\}^2}(0,1)} > 0. \quad (3.21)$$

Finally, the quantifier $\sim^{\chi^2}_{\alpha}$, $\alpha \in (0,1)$, called χ^2-*quantifier* with significance level α, has the truth function

$$\mathrm{Tf}_{\sim^{\chi^2}_{\alpha}}(a,b,c,d) = \begin{cases} 1 & \text{if } ad > bc \;\&\\ & \frac{(a+b+c+d)(ad-bc)^2}{(a+b)(a+c)(b+d)(c+d)} \geq \chi^2(1-2\alpha), \\ 0 & \text{else,}\end{cases} \quad (3.22)$$

where $\chi^2(1-2\alpha)$ is the $(1-2\alpha)$-quantile of the χ^2 distribution with one degree of freedom. Again, this definition can be rewritten in statistical terms as

$$\mathrm{Tf}_{\sim^{\chi^2}_{\alpha}}(a,b,c,d) = \\ = \begin{cases} 1 & \text{if the } \chi^2 \text{ test asymptotically rejects at the significance level } \alpha\\ & \text{the null hypothesis of independence of marginal probability}\\ & \text{distributions } P_{\varphi}, P_{\psi} \text{ of } \|\varphi\|_i \text{ and } \|\psi\|_i, \text{ respectively, against the}\\ & \text{alternative of a positive logarithmic interaction of } (\|\varphi\|_i, \|\psi\|_i),\\ 0 & \text{else.}\end{cases} \quad (3.23)$$

In this case, combining (3.23) with (3.15) complements the condition (3.8) for the relational transformation $\mathcal{M}$ with

$$\mathrm{Tf}_{\rightarrow^{\chi^2}_{\alpha}}(a,b,c,d) = 1 \text{ iff asymptotically } \ln \frac{P_{\{0,1\}^2}(1,1)P_{\{0,1\}^2}(0,0)}{P_{\{0,1\}^2}(1,0)P_{\{0,1\}^2}(0,1)} > 0. \tag{3.24}$$

Figure 3.1 shows a typical output from the GUHA method. In that output, the Fisher quantifier has been used.

DB "carab" analysis: non-dichotomously present species + 1-5 ecological factors

shortest hypotheses with support 5 by Fisher quantifier, significance level 0.01 %

Ecological factors			Species	Fisher quantifier
G_1_1 = yes			Pseudoophonus_rufipes = 1 - 2	1.9e-006
distance WL = < 40	G_1_1 = yes		Bembidion_semipunctatum = 3 - 5	7.9e-005
distance WL = < 40	G_1_1 = yes		Loricera_pilicornis = at least 2	9.1e-005
distance WL = < 40	G_1_2 = no		Elaphrus_riparius = 1 - 2	8.8e-005
distance WL = < 40	height herbs = < 20		Bembidion_argenteolum = 2 - 10	2.2e-005
distance WL = < 40	height herbs = < 70		Bembidion_femoratum = 1 - 2	6.6e-005
distance WL = < 40	height herbs = > 40		Pseudoophonus_rufipes = 1 - 2	4.5e-005
distance WL = < 40	cover herbs = < 50		Bembidion_femoratum = 1 - 2	3.9e-005
distance WL = < 40	cover herbs = < 60		Bembidion_femoratum = 1 - 2	3.9e-005
distance WL = < 60	height herbs = < 20		Bembidion_argenteolum = 2 - 10	8.8e-005
distance WL = < 60	height herbs = > 40		Pseudoophonus_rufipes = 1 - 2	3.1e-005
distance WL = < 60	height herbs = > 40		Pseudoophonus_rufipes = 2 - 6	1.9e-005
distance WL = < 70	height herbs = > 40		Pseudoophonus_rufipes = 2 - 6	5.9e-005
distance WL = > 70	cover herbs = < 50		Formicidae = yes	3.9e-006
distance WL = > 70	cover herbs = < 60		Formicidae = yes	1.6e-005
distance WL = > 70	cover litter = < 20		Formicidae = yes	1.3e-005
distance WL = > 70	cover litter = < 40		Formicidae = yes	1.3e-005
distance WL = > 70	cover litter = < 50		Formicidae = yes	1.3e-005
distance WL = > 70	cover litter = < 60		Formicidae = yes	1.3e-005
distance FG = -50 - 30	G_1_1 = yes		Loricera_pilicornis = at least 2	4.9e-005
distance FG = -50 - 30	G_1_1 = yes		Pseudoophonus_rufipes = 1 - 2	5.7e-007
distance FG = -50 - 70	G_1_1 = yes		Loricera_pilicornis = at least 2	4.9e-005
distance FG = -50 - 70	G_1_1 = yes		Pseudoophonus_rufipes = 1 - 2	5.7e-007
G_1_1 = yes	cover litter = < 20		Bembidion_semipunctatum = 3 - 5	7.9e-005
G_1_1 = yes	cover litter = < 20		Loricera_pilicornis = at least 2	9.1e-005
G_1_1 = yes	cover litter = < 40		Bembidion_semipunctatum = 3 - 5	7.9e-005
G_1_1 = yes	cover litter = < 40		Loricera_pilicornis = at least 2	9.1e-005
G_5 = no	height herbs = > 40		Pseudoophonus_rufipes = 2 - 6	4.6e-005
sand = 1	height herbs = > 40		Pseudoophonus_rufipes = 1 - 2	1.2e-005
height herbs = > 40	cover litter = < 20		Pseudoophonus_rufipes = 1 - 2	4.9e-005
height herbs = > 40	cover litter = < 20		Pseudoophonus_rufipes = 2 - 6	2.6e-005
height herbs = > 40	cover litter = < 40		Pseudoophonus_rufipes = 2 - 6	7.7e-005
height herbs = > 40	cover litter = < 50		Pseudoophonus_rufipes = 2 - 6	7.7e-005
height herbs = > 40	cover litter = < 60		Pseudoophonus_rufipes = 2 - 6	7.7e-005
distance WL = < 40	distance FG = < 30	G_1_2 = no	Elaphrus_riparius = 1 - 2	7.4e-005

Figure 3.1: An example html-output from one of the recent implementations of the GUHA method

3.2 Survey of main direct methods

Among modern data mining methods, the closest relatives of GUHA are methods for mining *association rules*, i.e., Boolean sentences valid with at least a prescribed confidence and supported with at least a prescribed proportion of the available data [1, 68, 45]. Admittedly, the two decades that separate the advent of these modern methods from that of GUHA brought various shifts in the related terminology. Already the term "association rules" evokes a correspondence to a class of generalized quantifiers of GUHA called *associational quantifiers*. However, modern association rules are actually always Boolean implications, due to which they correspond to the much narrower class of *implicational quantifiers*. Nevertheless, terminological differences only obscure a surprisingly strong similarity between methods for association rules mining and the method GUHA, as a recent comparative analysis of both approaches has shown [29].

Inductive logic programming (ILP) consists basically in the induction of an intensional description of a relation from the extensional descriptions of the intersection of that relation with the data and of the intersection of its complement with the data, i.e., from the positive and negative examples that are for the relation available in the data [11, 53]. When inducing the intensional description of a new relation, use may be made of intensional descriptions of already existing relations. Nowadays, already numerous implemented ILP-systems exist, for the induction of relations corresponding to a single concept as well as of relations corresponding to a relationship between several concepts, for both the batch-mode and the incremental induction, for both interactive and noninteractive induction.

Decision trees are a machine learning method for the extraction of classification rules from data [8, 58, 54]. Due to the hierarchical structure of systems of such rules, they can be easily visualized as tree graphs. It is this visualizability that accounts for the name of the method, and more importantly, for its great popularity - provided the height of the tree is small, the obtained classification rules are well comprehensible. In addition, decision trees are very robust against outliers because the borders between validity regions of individual antecedents are piecewise constant, and usually do not depend on distant data.

Differently to inductive logic programming, the induction of intensionally described relations by means of *rough sets* relies not on positive and negative examples, but instead on lower and upper approximations [43, 50, 57]. The former are examples that are completely covered with the induced relation, the latter are examples that have an nonempty intersection with that relation. Hence, the uncertainty of the extracted rules is captured by means of a pair of set-theoretic relations in the case of the rough-set approach, rather than with a fuzzy relation like in the extraction of fuzzy rules.

Rule extraction by means of genetic algorithms is nowadays probably the most elaborate application of evolutionary methods to knowledge extraction from data [21]. To use genetic algorithms for the extraction of rules from data is possible due to the fact that they are an optimization method that requires only the function values of the objective function. As an objective function serves in the case of rule extraction some quantitative property of the resulting set of obtained rules, e.g., its confidence, accuracy, completeness, some interestingness measure, or a combination of several such properties. As any other other optimization method, also a genetic algorithm needs to start from some initial element of a sequence of points intended to converge to the sought optimum of the objective function. Differently to other optimization methods, however, genetic algorithms seek the optimum using a whole population of such sequences, hence they need to start from a whole population of initial points, forming together the first generation of the algorithm. To obtain that first generation is an independent problem, intensively studied both in general, and in the specific context of using genetic algorithms for rule extraction from data [67, 21]. As far as the optimization procedure itself is concerned, i.e., the application of the genetic operators selection, crossover and mutation to the individuals forming the population, there exist two principally different approaches. In the *Pittsburgh approach*, the optimized individuals are whole sets of rules, each of which attempts to completely describe the data. In each generation, a population of such rulesets is obtained. In the *Michigan approach*, on the other hand, individual rules are optimized, and the population of a generation is a set of rules. Hence, the Michigan approach is computationally simpler, but it has to externally solve the problems of inconsistence, redundancy, and incompleteness of data description, the solution of which can in the Pittsburgh approach be directly built into the genetic operators.

4 Knowledge discovery with neural networks

However broad the spectrum of existing direct methods is, those methods still do not cover all situations when the above relational framework can be used. More precisely, there exists an important class of data mining methods that do not belong to direct methods as defined above, yet admit applying the relational framework - methods for knowledge discovery in data by means of *artificial neural networks (ANNs)*.

Actually, already the mapping computed by the network incorporates knowledge transferred to the ANN during its training, knowledge about implications that certain values of network inputs have for the values of its outputs. That knowledge is captured in the *ANN architecture*, and especially in a multidimensional *parameter vector*, which together with the architecture uniquely determines the computed mapping. As an example, parameters of the most common kind of neural networks, the multilayer perceptron, are the *connection weights* of all connections between subsequent layers, as well as the *activation thresholds* of all hidden and output neurons. It is this distributed knowledge representation that accounts for the excellent approximation properties of multilayer perceptrons (cf., e.g., [36, 42, 37]). However, it is not easily human-comprehensible (in terms that data mining borrowed from the cognitive science, this representation has a high "data fit", but a low "mental fit"). That is why methods attempting to transform the ANN-inherent knowledge into the form of *logical rules* have been developed since the late eighties ([14, 2, 19, 6, 9, 15, 41, 62, 10], references to methods published before 1998 can be found in the survey papers [4, 61, 52]).

Up to now, already several dozens ANN-based rule extraction methods exist. Individual methods differ from each other in at least one of the following aspects:

expressive power of the extracted rules, given by the meaning they are able to convey (since it depends mainly on the set of possible truth values, it divides the existing rule extraction methods basically into two broad classes - methods extracting *Boolean rules*, and those extracting *fuzzy rules*);

relationship between the extracted rules and network architecture (*decompositional methods* and *black-box methods*);

computational complexity of the method;

universality of the method (to which kind or kinds of ANNs it is applicable, and whether its applicability depends on the training algorithm that has been used for the network;

soundness, completeness, accuracy and *fidelity* of the extracted rules.

Nevertheless, all the existing ANN-based rule extraction methods share one common feature – they employ not only those input-output pairs that have been employed already for network training, but at least in a certain extent also additional pairs from the product of the network input space and its output space, obtained through the mapping computed by the network. Some rule extraction methods actually employ only pairs obtained through the computed mapping, and do not need the original training pairs any more. Hence, the mapping computed by the network is always inserted between the data and the extracted rules in ANN-based rule extraction methods, and the distributed knowledge representation by means of the network architecture and the parameter vector is an intermediate knowledge representation, a specific feature of this kind of rule extraction methods.

To capture the ANN-based rule extraction and its specificity within the relational framework of Section 2 is straightforward, due to the fact that the mapping computed by a neural network is a special kind of a relation, and that this mapping is uniquely determined by the network architecture and the parameters. Indeed, consider an ANN with a particular architecture containing n_I input neurons and n_O output neurons, in which the computed mapping $F : \Re^{n_I} \rightarrow \Re^{n_O}$ is parametrized with a k-dimensional parameter vector $\mathbf{v}$. For example, if the considered neural network is a multilayer perceptron with one hidden layer consisting of n_H neurons, then $k = n_H(n_I + 1) + n_O(n_H + 1)$ and

$$\mathbf{v} = (w_{1,1}, \ldots, w_{1,n_H}, w_{2,1}, \ldots, w_{2,n_O}, \theta_{1,1}, \ldots, \theta_{1,n_H}, \theta_{2,1}, \ldots, \theta_{2,n_O}), \tag{4.1}$$

where $w_{1,h} = (w^1_{1,h}, \ldots, w^{n_I}_{1,h})$ for $h = 1, \ldots, n_H$ are weights of connections between the input and the hidden layer, $w_{2,o} = (w^1_{2,o}, \ldots, w^{n_H}_{2,o})$ for $o = 1, \ldots, n_O$ are weights of connections between the hidden

and the output layer, whereas $\theta_{1,1}, \theta_{1,2}, \ldots, \theta_{1,n_H}$ and $\theta_{2,1}, \theta_{2,2}, \ldots, \theta_{2,n_O}$ are scalar activity thresholds of, respectively, the hidden and the output neurons. The parametrizability of F means that there exists an architecture-specific parametrizing mapping π such that $F = \pi(\mathbf{v})$. In the above example of a multilayer perceptron with one hidden layer, if the specification of the architecture is completed with the requirements that all hidden neurons share the same activation function f and no activation function is assigned to output neurons, the parametrizing mapping π is defined by:

$$(\forall \mathbf{v} = (w_{1,1}, \ldots, w_{1,n_H}, w_{2,1}, \ldots, w_{2,n_O}, \theta_{1,1}, \ldots, \theta_{1,n_H}, \theta_{2,1}, \ldots, \theta_{2,n_O}) \in \Re^k)$$
$$(\forall x \in \Re^{n_I})\ \pi(\mathbf{v})(x) = F(x) =$$
$$= (\sum_{h=1}^{n_H} w_{2,1}^h f(w_{1,h} \cdot x - \theta_{1,h}) - \theta_{2,1}, \ldots, \sum_{h=1}^{n_H} w_{2,n_O}^h f(w_{1,h} \cdot x - \theta_{1,h}) - \theta_{2,n_O}). \quad (4.2)$$

Then F is a relation with the following properties:

(i) it is a relation between the input space and the output space of the neural network, $F \in \mathcal{P}(\Re^{n_I} \times \Re^{n_O})$;

(ii) differently to the extensional input data relation, F is intensional;

(iii) the intensional definition of F does not rely on the output logical rules, but on the parametrizing mapping π:

$$F = \{(x, y) \in \Re^{n_I} \times \Re^{n_O} : y = \pi(\mathbf{v})(x)\}, \quad (4.3)$$

e.g., for the parametrizing mapping (4.2):

$$F = \{(x, y) \in \Re^{n_I} \times \Re^{n_O} : y =$$
$$= (\sum_{h=1}^{n_H} w_{2,1}^h f(w_{1,h} \cdot x - \theta_{1,h}) - \theta_{2,1}, \ldots, \sum_{h=1}^{n_H} w_{2,n_O}^h f(w_{1,h} \cdot x - \theta_{1,h}) - \theta_{2,n_O}). \quad (4.4)$$

Taking into account this additional, intermediate relation, the relational transformation $\mathcal{M}$ in (2.1) needs to be split as follows:

$$\mathcal{M} : \mathcal{R}_{\mathcal{M}} \to \mathcal{P}(\Re^{n_I} \times \Re^{n_O}) \to \mathcal{P}(\bigotimes_{j=1}^{n} D_{x_j}), \quad (4.5)$$

or equivalently,

$$\mathcal{M} = \mathcal{M}_1 \circ \mathcal{M}_2. \quad (4.6)$$

where

$$\mathcal{M}_1 : \mathcal{R} \to \mathcal{P}(\Re^{n_I} \times \Re^{n_O}), \mathcal{M}_2 : \mathcal{P}(\Re^{n_I} \times \Re^{n_O}) \to \mathcal{P}(\bigotimes_{j=1}^{n} D_{x_j}). \quad (4.7)$$

Here, the transformation $\mathcal{M}_1$ transforms an input data relation into a mapping computed by the neural network. Hence, like the overall transformation $\mathcal{M}_1$, it transforms an extensionally described relation into an intensionally described one. The transformation is performed in course of network training. On the other hand, the transformation $\mathcal{M}_2$ transforms a computed mapping into the interpretation of a set of extracted logical rules, thus being a transformation between two intensionally described relations. This transformation is performed in course of the rule extraction from a trained neural network.

This general characterization of ANN-based rule extraction will now be illustrated on one particular method.

4.1 Example 2 – rule extraction by means of piecewise-linear neural networks

Most of the existing ANN-based rule extraction methods rely mainly on heuristics, and their underlying theoretical principles are not very deep. There are only few methods with solid mathematical

foundations. This subsection outlines one of them – a method for the extraction of Boolean rules from multilayer perceptrons, based on switching from a sigmoidal activation function to a piecewise-linear activation function.

Multilayer perceptrons (MLPs) are nowadays the most popular kind of artificial neural networks – both in general [65, 25, 32], and in the specific context of rule extraction [4, 55, 51, 14, 38, 62]. Basically, all multilayer perceptrons are very similar:

(i) Topologically, their set of neurons is partitioned into a finite linearly ordered set of *layers*, while the set of connections consists of the union of Cartesian products of each two neighbors in that linear ordering.

(ii) All hidden neurons (those that belong neither to the first, nor to the last layer) share the same *activation function*, and so do also all output neurons (those from the last layer).

(iii) An activation function is only required to be nonconstant and Borel measurable.

The considered rule extraction method focuses on multilayer perceptrons with the following additional properties:

- only one layer of hidden neurons;
- the activation function of the hidden neurons is *continuous sigmoidal*, more precisely continuous nondecreasing with two different finite limits in $-\infty$ and ∞;
- the activation function of the output neurons is the identity.

Moreover, the ultimate objective of network training is in this rule extraction method a mapping computed by an even much more specific multilayer perceptron, namely by a MLP in which the activation function of the hidden neurons is a *piecewise-linear sigmoidal*. Given a sequence of input-output training pairs, $(x_1, y_1), \ldots, (x_p, y_p) \in \Re^{n_I} \times \Re^{n_O}$, such a computed mapping can be obtained in two principally different ways:

(i) Using some special training method developed specifically for that kind of neural networks [24]. A disadvantage of this approach is that special training methods for ANNs with piecewise-linear activation functions are not available in common neural networks software.

(ii) Training a neural network with a general continuous sigmoidal activation function f (e.g., logistic or arctangens), and then switching from the mapping computed by that network to another mapping, obtained from the original one through replacing f with a piecewise-linear sigmoidal f_{PL} close enough to f in the metric space of continuous sigmoidal functions [49]. Then any training method available for the original network can be used, such as backpropagation, conjugate gradient methods, or the Levenberg-Marquardt method [12, 60, 26].

Both approaches are equivalent from the point of view of the optimization task to find the vector of parameters that minimizes with respect to the available training data a prescribed error function. That equivalence is established in [33], here only the main reasons will be recalled: Since all algorithms for the optimization of general functions are iterative, after a finite number of iterations they in general find only some suboptimal solution, i.e., a solution that does not guarantee to yield the minimal value of the error function, but still guarantees not to exceed that minimal value more than a prescribed tolerance ε. And instead of finding such an ε-suboptimal solution directly in a MLP with a piecewise-linear sigmoidal activation function, we can find it in the following two steps:

1. An $\frac{\varepsilon}{2}$-suboptimal solution is found in a MLP with a general continuous sigmoidal activation function.
2. For the found solution, the activation function of the MLP is everywhere replaced with a piecewise-linear sigmoidal activation function, sufficiently close to the original function in the metric space of continuous sigmoidals for the error function not to increase more than $\varepsilon/2$. The feasibility of this step is due to the density of the space of piecewise-linear sigmoidals in the space of continuous sigmoidals.

However, the approaches are not equivalent from the computational complexity point of view, since MLPs with piecewise-linear sigmoidal activation functions have a lower Vapnik-Chervonenkis dimension than MLPs with general continuous sigmoidal activation functions [48].

No matter in which way it is obtained, the final result is always the piecewise-linear computed mapping $F : \Re^{n_I} \to \Re^{n_O}$. From the point of view of the relational framework, $F \in \operatorname{val} \mathcal{M}_1 \cap \operatorname{dom} \mathcal{M}_2$, with the additional property

$$(\exists r \in \mathcal{N})(\exists P_1, \ldots, P_r \text{ – polyhedra in } \Re^{n_I}) \bigcup_{i=1}^{r} P_i = \Re^{n_I} \;\&$$
$$(\forall j = i, \ldots, r)\ F|P_i \text{ is a linear operator.} \quad (4.8)$$

Consequently, to any polyhedron in the output space of the network, $Q \subset \Re^{n_O}$, there exist $r' < r$ polyhedra in its input space, $P_1, \ldots, P_{r'} \subset \Re^{n_I}$ such that

$$Q = F(\bigcup_{i=1}^{r'} P_i). \quad (4.9)$$

Suppose that in any pair $(x_j, y_j) = ((x_j^1, \ldots, x_j^{n_I}), (y_j^1, \ldots, y_j^{n_O}))$, $j = 1, \ldots, p$, the numbers $x_j^1, \ldots, x_j^{n_I}$ and $y_j^1, \ldots, y_j^{n_O}$ are values of object variables $X^1, \ldots, X^{n_I}$ and $Y^1, \ldots, Y^{n_O}$ capturing quantifiable properties of objects in the application domain. Then (4.9) directly yields the following Boolean rule

$$\bigvee_{i=1}^{r'} (X^1, \ldots, X^{n_I}) \in P_i \to (Y^1, \ldots, Y^{n_O}) \in Q, \quad (4.10)$$

which is equivalent to the conjunction of Boolean rules

$$(X^1, \ldots, X^{n_I}) \in P_i \to (Y^1, \ldots, Y^{n_O}) \in Q \quad (4.11)$$

for $i = 1, \ldots, r'$. If the polyhedron $Q \subset \Re^{n_O}$ in (4.9) is in addition chosen to be a hyperrectangle with projections $O^1, \ldots, O^{n_O}$, then (4.10) and (4.11) turn to

$$\bigvee_{i=1}^{r'} (X^1, \ldots, X^{n_I}) \in P_i \to \bigwedge_{j \in \mathcal{O}} Y^j \in O^j, \quad (4.12)$$

and

$$(X^1, \ldots, X^{n_I}) \in P_i \to \bigwedge_{j \in \mathcal{O}} Y^j \in O^j, \quad (4.13)$$

respectively, where $\mathcal{O} = \{j : O^j \neq \Re\}$.

Since the interpretation of any of the rules (4.10)–(4.13) in $\Re^{n_I + n_O}$ is a relation in $\Re^{n_I + n_O}$, their extraction directly fits into the proposed relational framework, with $n = n_I + n_O$, and the object variables $X^1, \ldots, X^{n_I}$ and $Y^1, \ldots, Y^{n_O}$. Moreover, that relation has the specific property that its cut through the first n_I components always coincides with its projection into those components, namely in (4.11) and (4.13) with the polyhedron P_i, $i = 1, \ldots, r'$, whereas in (4.10) and (4.12) with the union of polyhedra $\bigcup_{i=1}^{r'} P_i$ (Figure 4.1). Similarly, its cut through the last n_O components always coincides with the respective projection, i.e., the polyhedron Q in (4.10)–(4.11), and the hyperrectangle with projections $O^1, \ldots, O^{n_O}$ in (4.12)–(4.13). Due to those properties, the projection into the first n_I components can be seen as an interpretation of the antecedent of the considered rule, whereas the projection into the last n_O components can be seen as an interpretation of its consequent.

Due to the linearity of polyhedra, rules of the kinds (4.10)–(4.13) are easy to store and manipulate in a computer. However, their comprehensibility is hindered by the difficult interpretation of the antecedent $(X^1, \ldots, X^{n_I}) \in P$ for a general polyhedron P, especially if the dimension n_I is high. Therefore, an additional step usually follows the extraction of rules of the kind (4.12)–(4.13), namely, *replacing some of the polyhedra $P_1, \ldots, P_{r'}$ in (4.12) with hyperrectangles.*

The decision whether to replace a polyhedron P with a hyperrectangle H depends on our dissatisfaction with that part of P that does not belong to H and that part of H that does not belong to P,

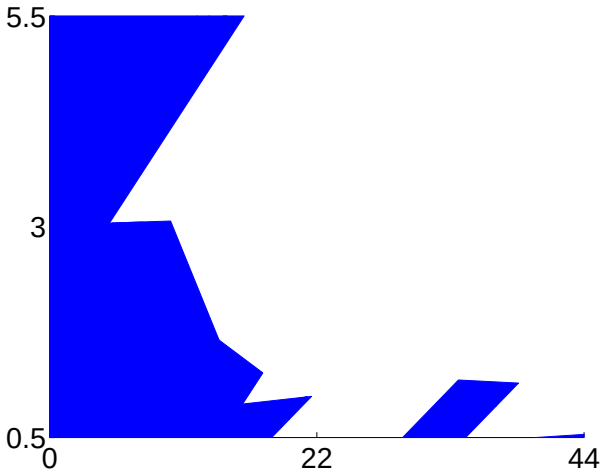

Figure 4.1: A two-dimensional cut through the union of polyhedra interpreting the antecedent of a particular rule of the kind (4.12), i.e., through the projection of the relation interpreting that rule into its first n_I components

i.e., on our dissatisfaction with the symmetric difference $P\Delta H$ of P and H. Let us denote that dissatisfaction $\mu_P(P\Delta H)$ (to express its possible dependence on P) and make the following assumptions about the way how it determines the replacement decision:

(i) the dissatisfaction is nonnegative ($\mu_P(P\Delta H) \geq 0$);

(ii) increasing the area $P\Delta H$ increases the dissatisfaction $\mu_P(P\Delta H)$;

(iii) the dissatisfaction $\mu_P(P\Delta H)$ is minimal among the dissatisfactions $\mu_P(P\Delta H')$ for hyperrectangles H' in the considered space;

(iv) for P to be replaceable with H, the dissatisfaction $\mu_P(P\Delta H)$ must not exceed some prescribed limit $\varepsilon > 0$;

(v) to be eligible for the replacement, P has to cover at least one point of the available data.

The assumptions (i)–(ii) imply that μ_P is a nonnegative monotone measure on the considered space, such that its domain contains $P\Delta H$ for any polyhedron P and any hyperrectangle H in that space, e.g., a *nonnegative Borel measure* on the space. If the considered space is the input space of a neural network, two measures are particularly attractive:

A. The empirical distribution of $x_1, \ldots, x_p$, i.e., the empirical distribution of the input components of the sequence $(x_1, y_1), \ldots, (x_p, y_p)$ of training pairs (observe that this measure does not depend on P).

B. The conditional empirical distribution of the input components of the training sequence, conditioned (hence, also dependent) on P.

An important property of the measures A. and B., not holding for general nonnegative Borel measures, is that for any polyhedron P in the input space of the network, a hyperrectangle H_P in that space can be found such that the assumption (iii.) is fulfilled, i.e.,

$$\mu_P(P\Delta H_P) = \min\{\mu_P(P\Delta H') : H' \text{ is a hyperrectangle in the input space of the network }\}. \quad (4.14)$$

Denote the projections of a hyperrectangle H in the input space of the network $I_H^1, \ldots, I_H^{n_I}$, introduce the notation $\mathcal{I}_H = \{j : I_H^j \neq \Re\}$, and consider only polyhedra compatible with the assumptions (iv.)–(v.), i.e., polyhedra from the set

$$\mathcal{C} = \{P_i : i = 1, \ldots, r' \;\&\; \mu_P(P\Delta H_P) \leq \varepsilon \;\&\; P_i \text{ covers at least one point of the available data}\}. \quad (4.15)$$

Then (4.13) turns for $P \in \mathcal{C}$ into

$$\bigwedge_{j \in \mathcal{I}_{H_P}} X^j \in I^j_{H_P} \rightarrow \bigwedge_{j \in \mathcal{O}} Y^j \in O^j, \tag{4.16}$$

whereas (4.12) implies

$$\bigvee_{P \in \mathcal{C}} \bigwedge_{j \in \mathcal{I}_{H_P}} X^j \in I^j_{H_P} \rightarrow \bigwedge_{j \in \mathcal{O}} Y^j \in O^j. \tag{4.17}$$

From the point of view of the proposed relational framework, the specificity of the relations (4.16) and (4.17) is that their projection into the first n_I components is a hyperrectangle and a union of hyperrectangles, respectively (Figure 4.2).

To illustrate the difference between a rule (4.12) and the rule (4.17) obtained from it in the above described way, Figures 4.1–4.2 show mutually corresponding 2-dimensional cuts through antecedents of those rules, i.e., through their projections into the first n_I components, for a particular rule (4.12) extracted from data in an ecological application [34].

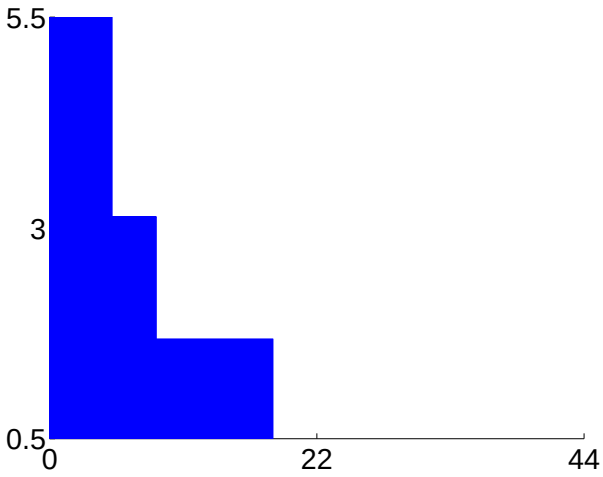

Figure 4.2: A two-dimensional cut through the union of hyperrectangles replacing those polyhedra from Figure 4.1 that are replaceable in accordance with the above assumptions (i)–(iv)

5 Conclusion

In this paper, a unifying relational framework for a broad class of data mining methods has been proposed. Its underlying assumptions and basic principles have been explained, and a survey of areas for which the assumptions make the framework appropriate has been given. To exemplify its usability, the framework has then been elaborated for two particular data mining methods of very different nature – the classical method of exploratory data analysis GUHA, and an ANN-based rule extraction method.

Needless to say, the framework needs to be elaborated for other relevant data mining methods before it can be routinely used. This will be a matter of further research, as well as elaborating the applicability of the framework for fuzzy data and for methods extracting fuzzy rules, which has been indicated in Section 2. Therefore, it would be premature to already attempt to seriously compare it with other data mining frameworks, such as those proposed in [3, 44, 69]. Much more important is to realize the fact that the new framework shares with the previous ones the final objective why they have been proposed, namely that a unifying framework of view is a prerequisite for the synergy of different data mining methods, for the comparison and consolidation, and consequently for a more

appropriate interpretation, of their results. Moreover, it shares with them the starting assumption that such a unifying framework should be abstract enough to be able to handle methods relying on different paradigms.

An interesting feature of the new approach is the fact that it assigns a specific position to methods for the extraction of rules from data by means of artificial neural networks, compared to other classes of methods to which the framework is applicable. Such a specificity of ANN-based rule extraction methods have not been indicated by any other data mining framework. That specificity might be worth investigation in the context of the many hopes that ANN-based rule extraction methods have raised in the early 90s, but then mostly failed to fulfill (cf. [4, 61] and references therein).

Bibliography

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, 1996.

[2] J.A. Alexander and M.C. Mozer. Template-based procedures for neural network interpretation. *Neural Networks*, 12:479–498, 1999.

[3] S.S. Anand, D.A. Bell, and J.G. Hughes. EDM: a general framework for data mining based on evidence theory. *Data and Knowledge Engineering*, 18:189–223, 1996.

[4] R. Andrews, J. Diederich, and A.B. Tickle. Survey and critique of techniques for extracting rules from trained artificical neural networks. *Knowledge Based Systems*, 8:378–389, 1995.

[5] C. Apte and S. Weiss. Data mining with decision trees and decision rules. *Future Generation Computer Systems*, 13:197–210, 1997.

[6] G. Bologna. Symbolic rule extraction from the DIMPL neural network. In S. Wermter and R. Sun, editors, *Hybrid Neural Systems*, pages 241–255. Springer Verlag, Heidelberg, 2000.

[7] R.J. Brachman and T. Anand. The process of knowledge dicovery in databases: A human-centered approach. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 37–57. AAAI Press, Menlo Park, 1996.

[8] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984.

[9] J. Chen and J. Liu. Using mixture principal component analysis networks to extract fuzzy rules from data. *Industrial and Engineering Chemistry Research*, 39:2355–2367, 2000.

[10] A.S. d'Avila Garcez, K. Broda, and D.M. Gabbay. Symbolic knowledge extraction from artificial neural networks: A sound approach. *Artificial Intelligence*, 125:155–207, 2001.

[11] L. De Raedt. *Interactive Theory Revision: An Inductive Logic Programming Approach*. Academic Press, London, 1992.

[12] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice Hall, Englewood Cliffs, 1983.

[13] F.T. Denton. Data mining as an industry. *The Review of Economics and Statistics*, 67:124–127, 1985.

[14] W. Duch, R. Adamczak, and K. Grabczewski. Extraction of logical rules from neural networks. *Neural Processing Letters*, 7:211–219, 1998.

[15] W. Duch, R. Adamczak, and K. Grabczewski. A new methodology of extraction, optimization and application of crisp and fuzzy logical rules. *IEEE Transactions on Neural Networks*, 11:277–306, 2000.

[16] J.F. Elder and D. Pregibon. A statistical perspective on knowledge discovery in databases. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 83–116. AAAI Press, Menlo Park, 1996.

[17] S.J. Farlow, editor. *Self-Organizing Methods in Modeling: GMDH Type Algorithms.* Marcel Dekker, New York, 1984.

[18] U.M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–36. AAAI Press, Menlo Park, 1996.

[19] G.D. Finn. Learning fuzzy rules from data. *Neural Computing & Applications*, 8:9–24, 1999.

[20] W.J. Frawley, G. Piatetsky-Shapiro, and C.J. Matheus. Knowledge discovery in databases: An overview. In G. Piatetsky-Shapiro and W. Frawley, editors, *Knowledge Discovery in Databases*, pages 1–27. AAAI Press, Menlo Park, 1991.

[21] A.A. Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms.* Springer Verlag, Berlin, 2002.

[22] J.H. Friedman. Multiple adaptive regression splines (with discussion). *Annals of Statistics*, 19:1–141, 1991.

[23] J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23:881–889, 1974.

[24] E.F. Gad, A.F. Atiya, S. Shaheen, and A. El-Dessouki. A new algorithm for learning in piecewise-linear neural networks. *Neural Networks*, 13:485–505, 2000.

[25] M.T. Hagan, H.B. Demuth, and M.H. Beale. *Neural Network Design.* PWS Publishing, Boston, 1996.

[26] M.T. Hagan and M. Menhaj. Training feedforward networks with the Marquadt algorithm. *IEEE Transactions on Neural Networks*, 5:989–993, 1994.

[27] P. Hájek, I. Havel, and M. Chytil. The GUHA method of automatic hypotheses determination. *Computing*, 1:293–308, 1966.

[28] P. Hájek and T. Havránek. *Mechanizing Hypothesis Formation.* Springer Verlag, Berlin, 1978.

[29] P. Hájek and M. Holeňa. Formal logics of discovery and hypothesis formation by machine. *Theoretical Computer Science*, 292:345–357, 2003.

[30] D. Hand. Data-mining – reaching beyond statistics. In M. Noirhomme-Fraiture, editor, *Procedings of KESDA'98 – International Conference on Knowledge Extraction from Statistical Data*, pages 91–101, 1998.

[31] D. Harmancová, M. Holeňa, and A. Sochorová. Overview of the Guha method for automating knowledge discovery in statistical data sets. In M. Noirhomme-Fraiture, editor, *Knowledge Extraction and Symbolic Data Analysis*, pages 65–77. Eurostat, Luxembourg, 1999.

[32] S. Haykin. *Neural Networks. A Comprehensive Foundation.* IEEE, New York, 1999.

[33] M. Holeňa. Extraction of logical rules from data by means of piecewise-linear neural networks. In *Proceedings of the 5th International Conference on Discovery Science*, pages 192–205. Springer Verlag, Berlin, 2002.

[34] M. Holeňa. Mining rules from empirical data with an ecological application. Technical report, Brandenburg University of Technology, Cottbus, 2002. ISBN 3-934934-07-2, 62 pages.

[35] M. Holeňa, A. Sochorová, and J. Zvárová. Increasing the diversity of medical data mining through distributed object technology. In P. Kokol, B. Zupan, J. Stare, M. Premik, and R. Engelbrecht, editors, *Medical Informatics Europe '99*, pages 442–447. IOS Press, Amsterdam, 1999.

[36] K. Hornik. Approximation capabilities of multilayer neural networks. *Neural Networks*, 4:251–257, 1991.

[37] K. Hornik, M. Stinchcombe, H. White, and P. Auer. Degree of approximation results for feedforward networks approximating unknown mappings and their derivatives. *Neural Computation*, 6:1262–1275, 1994.

[38] P. Howes and N. Crook. Using input parameter influences to support the decisions of feedforward neural networks. *Neurocomputing*, 24:191–206, 1999.

[39] *International Journal of Man-Machine Studies*, volume 10, number 1, 1978.

[40] *International Journal of Man-Machine Studies*, volume 15, number 7, 1981.

[41] M. Ishikawa. Rule extraction by successive regularization. *Neural Networks*, 13:1171–1183, 2000.

[42] V. Kůrková. Kolmogorov's theorem and multilayer neural networks. *Neural Networks*, 5:501–506, 1992.

[43] L.P. Khoo, Tor. S.B., and L.Y. Zhai. A rough-set approach for classification and rule induction. *Internationa Journal of Advanced Manufacturing Technology*, 15:438–444, 1999.

[44] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 249–272. AAAI Press, Menlo Park, 1996.

[45] F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Quantifiable data mining using ration rules. *VLDB Journal*, 8:254–266, 2000.

[46] R. Kruse and K.D. Meyer. *Statistics with Vague Data*. Reidel, Dordrecht, 1987.

[47] M.C. Lovell. Data mining. *The Review of Economics and Statistics*, 65:1–12, 1983.

[48] W. Maass. Bounds for the computational power and learning complexity of analog neural nets. *SIAM Journal on Computing*, 26:708–732, 1997.

[49] F. Maire. Rule-extraction by backpropagation of polyhedra. *Neural Networks*, 12:717–725, 1999.

[50] B. Mak and T. Munakata. Rule extraction from expert heuristics: A comparative study of rough sets with neural networks and id3. *European Journal of Operational Research*, 136:212–229, 2002.

[51] S. Mitra, R.K. De, and S.K. Pal. Knowledge-based fuzzy MLP for classification and rule generation. *IEEE Transactions on Neural Networks*, 8:1338–1350, 1997.

[52] S. Mitra and Y. Hayashi. Neuro-fuzzy rule generation: Survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11:748–768, 2000.

[53] S. Muggleton. *Inductive Logic Programming*. Academic Press, London, 1992.

[54] W. Müller and E. Wiederhold. Applying decision tree methodology for rules extraction under cognitive comstraints. *European Journal of Operational Research*, 136:212–229, 2002.

[55] D. Nauck, U. Nauck, and R. Kruse. Generating classification rules with the neuro-fuzzy system NEFCLASS. In *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society NAFIPS'96*, pages 466–470, 1996.

[56] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, San Francisco, 1988.

[57] L. Polkowski. *Rough Sets. Mathematical Foundations.* Physica-Verlag, Heidelberg, 2002.

[58] J. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, San Francisco, 1992.

[59] J. Rauch. Logical calculi for knowledge discovery in databases. In J. Komorowski and J.M. Żytkov, editors, *Principles of Data Mining and Knowledge Discovery*, pages 47–57. Springer Verlag, Berlin, 1997.

[60] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error back-propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Experiments in the Microstructure of Cognition*, pages 318–362, 1986.

[61] A.B. Tickle, R. Andrews, M. Golea, and J. Diederich. The truth will come to light: Directions and challenges in extracting rules from trained artificial neural networks. *IEEE Transactions on Neural Networks*, 9:1057–1068, 1998.

[62] H. Tsukimoto. Extracting rules from trained neural networks. *IEEE Transactions on Neural Networks*, 11:333–389, 2000.

[63] V.R. Vemuri and R.D. Rogers, editors. *Artificial Neural Networks: Forecasting Time Series.* IEEE Computer Society Press, Washington, 1993.

[64] R. Viertl. *Statistics Methods fo Non-Precise Data.* CRC Press, Boca Raton, 1995.

[65] H. White. *Artificial Neural Networks: Approximation and Learning Theory.* Blackwell Publishers, Cambridge, 1992.

[66] J Whittaker. *Graphical Models in Applied Multivariate Statistics.* John Wiley and Sons, New York, 1990.

[67] M.L. Wong and K.S. Leung. *Data Mining Using Grammar Based Genetic Programming and Applications.* Kluwer Academic Publishers, Dordrecht, 2000.

[68] M.J. Zaki, S. Parathasarathy, M. Ogihara, and W. Li. New parallel algorithms for fast discovery of association rules. *Data Mining and Knowledge Discovery*, 1:343–373, 1997.

[69] J.M. Żytkov and R. Zembowicz. Contingency tables as the foundation for concepts, concept hierarchies and rules: The 49er system approach. *Fundamenta Informaticae*, 30:383–399, 1997.