Computing and Informatics, Vol. 35, 2016, 528–554

PRACTICAL ASPECTS OF DATA MINING USING LISP-MINER

Petr Berka

University of Economics
W. Churchill Sq. 4,
130 67 Praha, Czech Republic
e-mail: berka@vse.cz

Abstract. The paper describes some practical aspects of using LISp-Miner for data mining. LISp-Miner is a software tool that is under development at the University of Economics, Prague. We will review the different types of knowledge patterns discovered by the system, and discuss their applicability for various data mining tasks. We also compare LISp-Miner 18.16 with Weka 3.6.9 and Rapid Miner 5.3.

Keywords: Data mining, GUHA method, LISp-Miner

Mathematics Subject Classification 2010: 68T05, 68T30

1 INTRODUCTION

Knowledge Management (KM) comprises a range of practices used in an organization to identify, create, represent, distribute and enable adoption of insights and experiences. Such insights and experiences comprise knowledge, either embodied in individuals or embedded in organizational processes or practice. Knowledge management thus aims at generation, representation, storage, transfer, transformation, application, embedding and protecting of (organizational) knowledge to "bring the right knowledge to the right people at the right time".

KM efforts typically focus on organizational objectives such as improved performance, competitive advantage, innovation, the sharing of lessons learned, and continuous improvement of the organization. Knowledge management can be studied from different perspectives:

- Techno-centric with a focus on technology, ideally those that enhance knowledge sharing and creation.
- Organizational with a focus on how an organization can be designed to facilitate knowledge processes best.
- Ecological with a focus on the interaction of people, identity, knowledge, and environmental factors as a complex adaptive system akin to a natural ecosystem.

The techno-centric perspective defines knowledge management systems as "IT systems developed to support and enhance the organizational process of knowledge creation, storage/retrieval, transfer and application" [1]. As more data is gathered in an organization, with the amount of data doubling every three years, data mining becomes an increasingly important tool to transform this data into knowledge. There is a number of different formalisms how to represent knowledge inferred (learned) from data: decision trees, association rules, decision rules, neural networks, Bayesian networks. There is also a number of data mining systems that implement the respective data mining methods. Let us mention here Weka and Rapid Miner as representatives of freely available tools or IBM SPSS Modeler and SAS Enterprise Miner as representatives of commerical systems.

In this paper we present the freely available data mining system LISp-Miner, that is under development at the University of Economics, Prague. We briefly review the data mining procedures implemented in the LISp-Miner system and discuss their usability for different data mining tasks. The paper thus summarizes the experience with LISp-Miner and its practical applications in a broader context of data mining tasks and systems.

The paper is organized as follows: Section 2 introduces the GUHA method, a formal framework of association rule mining, that inspired the work on LISp-Miner, Section 3 reviews the procedures that are implemented in the LISp-Miner systems and gives some details concerning the work with LISp-Miner, Section 4 mentions how LISp-Miner supports the CRISP-DM methodology, Section 5 shows data mining tasks, that can be solved using LISp-Miner, Section 6 presents some results when analyzing real data from medical domain and Section 7 contains a comparison of Lisp-Miner with Weka and Rapid Miner.

2 THE GUHA METHOD

GUHA is an original Czech method of exploratory data analysis which originates in 1960s. Its principle is to offer all interesting facts following from the given data to the given problem. The book [12] introduces the main principles of this method, a general theory of mechanized hypothesis formation based on mathematical logic and statistics. Hypotheses defined and studied in this book are relations between two conjunctions derived from values of attributes (columns) of an analyzed data table. Various types of relations between these conjunctions are used including relations corresponding to statistical hypothesis tests.

Within the GUHA procedure ASSOC, we understand the knowledge pattern (called hypothesis) as the expression

$$\phi \approx \psi / \gamma \tag{1}$$

where ϕ (called antecedent), ψ (called succedent) and γ (called condition) are conjunctions of literals and $\approx$ (called quantifier) denotes a relation between ϕ and ψ for the examples from the analyzed data table, that fulfill the condition γ. The rule $\phi \approx \psi / \gamma$ is true in the analyzed data table, if the relation associated with $\approx$ is satisfied for the frequencies a, b, c, d of the corresponding contingency table (Table 1). We can denote this fact by

$$\approx (a, b, c, d) = 1 \tag{2}$$

γ	ψ	$\neg\psi$	$\sum$
ϕ	a	b	r
$\neg\phi$	c	d	s
$\sum$	k	l	n

Table 1. 2×2 contingency table created for examples satisfying condition γ

A literal is defined as $A(coef)$ or $\neg A(coef)$ where A is an attribute and *coef* (coefficient) is a subset of possible values. So a category (attribute-value pair) is a literal as well. The relation $\approx$ needs not to be only the so called founded implication defined using the confidence measure $a/(a + b)$ of a "standard" association rule proposed by Agrawal (e.g. [2]), but also a statistical association evaluated using the χ^2 test.

When looking at the main differences between the GUHA method and standard association rule mining methods we can see, that:

- GUHA method offers more types of relations between ϕ and ψ; we can search not only for implications but also for equivalences or statistically based relations,
- GUHA method offers more expressive syntax of ϕ and ψ; ϕ and ψ are conjunctions of literals, not only conjunctions of categories,
- GUHA looks for relations between triples of conjunctions ϕ, ψ and γ, not only between pairs of ϕ and ψ as is the case of standard association rules.

The basic idea of the GUHA method is to find all hypotheses (knowledge patterns), that are true in the analyzed data in the above sense. A GUHA procedure generates (in a top-down way) each particular pattern and tests if it is true in the analyzed data. The output of the procedure consists of all such patterns, that do not immediately follow from other more simple output patterns.

Since 1960s GUHA method has been implemented several times on different computer platforms. We will describe the GUHA procedures, which are currently implemented in the LISp-Miner system, in the next section. These procedures differ in the types of knowledge patterns that are produced as the output.

3 LISP-MINER SYSTEM

3.1 General Information

LISp-Miner, a freely available system that is developed since 1996 at the University of Economics, Prague, implements various GUHA procedures that mine for different types of (mostly rule-like) knowledge patterns. LISp-Miner is oriented on work with categorial attributes. We can distinguish between binary attributes (that can have only two values, e.g. the attribute sex), nominal attributes (that can have more than two values and the values are not ordered, e.g. the attribute color) and ordinal attributes (that can have more than two values but their values are ordered, e.g. the attributes education or rank in the army). Numeric attributes must be discretized (turned into intervals) in advance, before running any of the mining procedure. The data preprocessing module LMDataSource can be used to this – the module offers equidistant discretization (the created intervals have the same width), equifrequent discretization (the created intervals contain roughly the same number of examples) or discretization, where the intervals are defined by user.

The values of attributes (i.e. categories) can be encoded either by symbolic codes (e.g. income(low)) or by numbers (e.g. income(1)). When using numbers to encode categories, various LISp-Miner procedures can handle them in some numeric operations (e.g. merging values to create broader intervals or compute distances between examples) – this will be detailed in subsequent subsections. Such numeric operation makes of course sense only for binary or ordinal attributes.

The analyzed data can contain missing values and the LISp-Miner procedures can handle these missings without any imputation. Unlike other data mining systems, where missings are handled on the level of analyzed data by replacing the missing code with an admissible value of the respective attribute, LISp-Miner handles missings on the level of knowledge patterns. If some data is missing, then in 4ft-Miner we obtain a 3×3 contingency table as shown in Table 2 instead of the standard 2×2 contingency table as shown in Table 1. This 3×3 table must then be transformed into the 2×2 table to compute the quantifiers that represent the relation between ϕ and ψ. The interested readers should refer to [21] for more details.

γ	ψ	ψ_X	$\neg\psi$
ϕ	a_{11}	a_{1X}	a_{10}
ϕ_X	a_{X1}	a_{XX}	a_{X0}
$\neg\phi$	a_{01}	a_{0X}	a_{00}

Table 2. 3×3 contingency table

Client	Income	Balance	Sex	Unemployed	Loan
c1	high	high	female	no	yes
c2	high	high	male	no	yes
c3	low	low	male	no	no
c4	low	high	female	yes	yes
c5	low	high	male	yes	yes
c6	low	low	female	yes	no
c7	high	low	male	no	yes
c8	high	low	female	yes	yes
c9	low	medium	male	yes	no
c10	high	medium	female	no	yes
c11	low	medium	female	yes	no
c12	low	medium	male	no	yes

Table 3. Running example data

3.2 LISp-Miner Procedures

By now, LISp-Miner consists of 10 data mining procedures: 4ft-Miner, SD4ft-Miner, AC4ft-Miner, KL-Miner, CF-Miner, SDKL-Miner, SDCF-Miner, KEX, ETree-Miner and MCluster-Miner. The 4ft-Miner is based on the original GUHA procedure ASSOC from the 60s, the other procedures have been designed during the development of the LISp-Miner system (see e.g. [25]). Most of the procedures mine for various types of rule-like patterns – this makes LISp-Miner more focused on particular type of models than standard data mining tools. We will use the running example data shown in Table 3 to explain each of these procedures.

Recall from Section 2, that *antecedent* ϕ corresponds to the left hand side of a rule, *succedent* ψ corresponds to the right hand of a rule, *coef* corresponds to the list of possible values of an attribute, and *quantifier* corresponds to an interestingness measure defined for a contingency table. While in GUHA procedure ASSOC, ϕ, ψ and γ are conjunctions of literals, in LISp-Miner procedures, ϕ, ψ, γ, α and β are more complex formulae called Boolean attributes or cedents. We will use the second notion as it is closer related to the LISp-Miner implementation. A cedent is a conjunction of partial cedents, each partial cedent is a conjunction or disjunction of literals and literals are defined as $A(coef)$ or $\neg A(coef)$.

The **4ft-Miner procedure** (authors J. Rauch and M. Šimůnek, for details see e.g. [25, 21]) mines for knowledge patterns, that can be understood as 4ft-association rules of the form

$$\phi \approx \psi / \gamma \tag{3}$$

where ϕ (antecedent), ψ (succedent) and γ (condition) are cedents and $\approx$ is a quantifier which is evaluated on the subset of examples, that satisfy the condition γ. If the condition is empty then the procedure analyzes the whole data table. The quantifier $\approx$ is defined using frequencies a, b, c, and d of the corresponding 2×2 contingency table. Table 4 shows some examples of the implemented quantifiers (types

of 4ft-association rules). The table also shows the relation, that the quantifiers must fulfill to consider the 4ft-association rule to be true. Here $p \in [0,1]$, $Base \in [1,n]$ and $\alpha \in [0,0.5]$ are user given parameters for probability, number of examples, and significance level respectively.

4ft quantifier $\approx$		
Name	Symbol	$\approx(a,b,c,d) = 1$ iff
Founded implication	$\Rightarrow_{p,Base}$	$\frac{a}{a+b} \geq p \wedge a \geq Base$
Founded double implication	$\Leftrightarrow_{p,Base}$	$\frac{a}{a+b+c} \geq p \wedge a \geq Base$
Founded equivalence	$\equiv_{p,Base}$	$\frac{a+d}{a+b+c+d} \geq p \wedge a \geq Base$
Fisher quantifier	$\approx_{\alpha,Base}$	$\sum_{i=a}^{\min(r,k)} \frac{\binom{k}{i}\binom{n-k}{r-i}}{\binom{r}{n}} \leq \alpha \wedge a \geq Base$
χ^2 quantifier	$\sim^2_{\alpha,Base}$	$\frac{(ad-bc)^2}{rkls} n \geq \chi^2_\alpha \wedge a \geq Base$
Above average dependence	$\sim^+_{q,Base}$	$\frac{a}{a+b} \geq (1+q)\frac{a+c}{a+b+c+d} \wedge a \geq Base$

Table 4. Examples of 4ft-quantifiers

As shown in Section 2, literals are in the form $A(coef)$ or $\neg A(coef)$. In 4ft-Miner, *coef*, the list of possible values of an attribute A can be:

- one category, this is simply a single value of an attribute A (e.g. city(London) or age(10–20)),
- a subset of given length (e.g., city(London, Paris) is a literal that contains a subset of length 2),
- an interval of given length (e.g., age(10–20, 20–30) or age(0–10, 10–20) are intervals of length 2),
- cyclical interval (e.g., if values of age are 0–10, 10–20, 20–30 and 30–40, then age(30–40, 0–10, 10–20) is a cyclical interval),
- a cut, i.e., an interval of given length, that contains the first or the last value (e.g., if values of age are 0–10, 10–20, 20–30 and 30–40, then age(0–10, 10–20) is a cut of length 2 but age(10–20, 20–30) is not a cut),
- left cut, a cut that contains the first value (e.g. age(0–10, 10–20)),
- right cut, a cut that contains the last value (e.g. age(30–40) or age(20–30, 30–40)).

While subsets can be created for all attributes, to create intervals and cuts make sense only for ordinal attributes.

So an example of founded implication created from the data shown in Table 3 can be not only the rule

$$income(high) \wedge balance(high) \Rightarrow loan(yes) \tag{4}$$

which corresponds to a standard association rule, but also the rule

$$balance(high, medium) \wedge \neg unemployed(yes) \Rightarrow loan(yes)/sex(male) \qquad (5)$$

which says that in the group of men, if a person has high or medium balance on his account and is not unemployed, then he will get the loan.

It is worth to mention, that more quantifiers can be set to restrict the list of resulting rules. Beside these quantifiers the results can also be restricted using a threshold (min or max) for any of the frequencies from the 2×2 contingency table. A special role within these parameters play *Base* and *Ceil*; *Base* is the lower bound for the frequency a, *Ceil* is the upper bound for a.

Let us consider our data from Table 3. When looking for founded implications with parameters $p = 0.9$ and $Base = 1$, where all literals occurring in the rule have as their coefficient one category, we will find 206 rules, part of them shown in Table 5. This table also illustrates the depth-first way of generating the cedents.

$balance(low) \wedge unemployed(yes) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge sex(female) \wedge income(low) \Rightarrow loan(no)$
$balance(low) \wedge unemployed(yes) \wedge sex(female) \wedge income(high) \Rightarrow loan(yes)$
$balance(low) \wedge unemployed(yes) \wedge sex(female) \wedge loan(yes) \Rightarrow income(high)$
$balance(low) \wedge unemployed(yes) \wedge sex(female) \wedge loan(no) \Rightarrow income(low)$
$balance(low) \wedge unemployed(yes) \wedge income(low) \Rightarrow loan(no)$
$balance(low) \wedge unemployed(yes) \wedge income(low) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge income(low) \wedge loan(no) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge income(high) \Rightarrow loan(yes)$
$balance(low) \wedge unemployed(yes) \wedge income(high) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge income(high) \wedge loan(yes) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge loan(yes) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge loan(yes) \Rightarrow income(high)$
$balance(low) \wedge unemployed(yes) \wedge loan(no) \Rightarrow sex(female)$
$balance(low) \wedge unemployed(yes) \wedge loan(no) \Rightarrow income(low)$
$balance(low) \wedge unemployed(no) \Rightarrow sex(male)$
$balance(low) \wedge unemployed(no) \wedge sex(male) \wedge income(low) \Rightarrow loan(no)$
$balance(low) \wedge unemployed(no) \wedge sex(male) \wedge income(high) \Rightarrow loan(yes)$
$balance(low) \wedge unemployed(no) \wedge sex(male) \wedge loan(yes) \Rightarrow income(high)$
$balance(low) \wedge unemployed(no) \wedge sex(male) \wedge loan(no) \Rightarrow income(low)$
$balance(low) \wedge unemployed(no) \wedge income(low) \Rightarrow loan(no)$
$balance(low) \wedge unemployed(no) \wedge income(low) \Rightarrow sex(male)$
$balance(low) \wedge unemployed(no) \wedge income(low) \wedge loan(no) \Rightarrow sex(male)$
$balance(low) \wedge unemployed(no) \wedge income(high) \Rightarrow loan(yes)$
$balance(low) \wedge unemployed(no) \wedge income(high) \Rightarrow sex(male)$
$balance(low) \wedge unemployed(no) \wedge income(high) \wedge loan(yes) \Rightarrow sex(male)$

Table 5. Result of 4ft-Miner example run

The **SD4ft-Miner procedure** (authors J. Rauch and M. Šimůnek, see [13]) mines for patterns of the form

$$\alpha \bowtie \beta : \varphi \approx \psi / \gamma \tag{6}$$

A true SD4ft pattern refers to a situation when a 4ft-association rule $\phi \approx \psi$ found for a subset of the analyzed data defined by the cedent α differs (according to the values of a quantifier $\approx$) from the (same) 4ft-association rule found for a subset of the analyzed data defined by the cedent β. Again, the analyzed data may or may not be restricted by a condition γ.

There is a number of SD4ft relations that measure the difference between the two 4ft-association rules in question, where each of the rules is characterized by a 4ft quantifier (founded implication, founded double implication, founded equivalence or above average). An example of such a measure is the absolute value of difference, so an example relation for two founded implications that must be fulfilled is shown in Equation (7).

$$\left| \frac{a_\alpha}{a_\alpha + b_\alpha} - \frac{a_\beta}{a_\beta + b_\beta} \right| \geq p. \tag{7}$$

Here the absolute value of the difference between the confidence of founded implication $\varphi \approx \psi$ computed for examples that satisfy $\alpha \wedge \gamma$ and the confidence of this founded implication computed for examples that satisfy $\beta \wedge \gamma$ is at least p.

When considering our running example, a true SD4ft pattern defined using the relation from Equation (7), where $p = 0.4$ is

$$\mathit{unemployed}(\mathit{yes}) \bowtie \mathit{unemployed}(\mathit{no}) : \mathit{balance}(\mathit{medium}) \Rightarrow \mathit{income}(\mathit{low}) \tag{8}$$

because in the group of unemployed, the confidence of the rule $\mathit{balance}(\mathit{medium}) \Rightarrow \mathit{income}(\mathit{low})$ is equal to 1 while in the group of employed, the confidence of the rule $\mathit{balance}(\mathit{medium}) \Rightarrow \mathit{income}(\mathit{low})$ is equal to 0.5.

The **AC4ft-Miner procedure** (authors J. Rauch and M. Šimůnek, see [24, 21]) mines for G-action rules of the form

$$\phi_{st} \wedge \Phi_{Chg} \approx \psi_{st} \wedge \Psi_{Chg} / \gamma \tag{9}$$

where ϕ_{st} is stable antecedent, Φ_{Chg} refers to change of antecedent, ψ_{st} is stable succedent, Ψ_{Chg} refers to change of succedent, quantifier $\approx$ corresponds to quantifiers used also in SD4ft-Miner and γ is a condition. The G-action rules are inspired by action rules defined in [20]. The basic idea of action rules is that we can find strong (w.r.t. support or confidence) pairs of rules that have same attributes in flexible parts of antecedent and succedent but these attributes have different values. In another words, if we change a value of a flexible attribute occurring in antecedent, then we will observe a change in a value of flexible attribute occurring in succedent. The so called stable antecedent and stable succedent cannot be affected.

For our running example let all attributes except sex be flexible. An example of a G-action rule can thus be

$$balance(low) \wedge income(low \rightarrow high) \Rightarrow loan(no \rightarrow yes) \tag{10}$$

This rule actually encodes two 4ft-association rules:

$$balance(low) \wedge income(low) \Rightarrow loan(no) \tag{11}$$

$$balance(low) \wedge income(high) \Rightarrow loan(yes) \tag{12}$$

The example G-action rule expresses that if a person with low balance will increase his income from low to high, his loan approval status will change from "no" to "yes". So this rule describes the actions that must be carried out (increase the income) to obtain the desired change (get loan approval).

The **KL-Miner procedure** (authors J. Rauch and M. Šimůnek, for more info see [16, 26]) mines for patterns of the form

$$R \sim C/\gamma, \tag{13}$$

where R and C are two categorial attributes, γ (condition) is a cedent and $\sim$ is a quantifier (describing the relationship between R and C) evaluated on the subset of examples, that satisfy the condition γ. There is a number of different formulas that define the quantifier $\sim$ using the frequencies from the K-L contingency table (Table 6). Some of them (e.g. chi-square test, entropy or mutual information) can be used for R and C being nominal, others (like e.g. Kendall's rank correlation coefficient) can be used for R and C being ordinal.

γ	c_1	$\ldots$	c_L	$\sum$
r_1	a_{11}	$\ldots$	a_{1L}	$n_{1.}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	
r_K	a_{K1}	$\ldots$	a_{KL}	$n_{K.}$
$\sum$	$n_{.1}$	$\ldots$	$n_{.L}$	n

Table 6. KxL contingency table

When looking at our sample data, we can evaluate 20 different KL patterns (pairs of attributes). One of the true KL patterns (for chi-square test with required value to be greater than 6) is

$$balance \sim loan \tag{14}$$

The **CF-Miner procedure** (authors J. Rauch and M.AŠimůnek, see [13, 26]) mines for patterns of the form

$$\sim C/\gamma, \tag{15}$$

where C is a categorial (nominal or ordinal) attribute, γ (condition) is a cedent and $\sim$ is a quantifier evaluated for frequencies of values of C on the subset of examples that satisfy the condition γ.

γ	c_1	...	c_L
	n_1	...	n_L

Table 7. CF frequency table

The possible quantifiers are related to descriptive statistics like count, average, minimum, maximum, variance or standard deviation. The CF pattern is true if the frequencies of categories of the attribute C satisfy some restrictions given on the value of $\sim$. In our sample data, a true CF pattern (for the relation min(frequency) ≥ 4) has been found for the income attribute.

The **SDKL-Miner procedure** (authors J. Rauch and M. Šimůnek, [13, 26]) mines for patterns of the form

$$\alpha \bowtie \beta : R \sim C/\gamma, \tag{16}$$

where R and C are two categorial attributes, α and β are two cedents that define two subsets of the analyzed data, γ (condition) is a cedent and $\sim$ is a quantifier that denotes a relation between R and C. This pattern is true if for examples that fulfill the condition γ, the subsets defined by α and β differ with respect to the quantifier $\sim$. The quantifiers $\sim$ are the same as defined for the procedure KL-Miner, and the difference can either be difference between frequencies from both contingency tables or difference between the values of $\sim$. For our sample data, one of the true SDKL patterns is the relation between income and balance based on entropy, once evaluated for the group of employed (the group α) and once evaluated for the group of unemployed (the group β),

$$\textit{unemployed}(\textit{no}) \bowtie \textit{unemployed}(\textit{yes}) : \textit{income} \sim \textit{balance}. \tag{17}$$

The **SDCF-Miner procedure** (authors J. Rauch and M. Šimůnek, [13, 26]) mines for patterns of the form

$$\alpha \bowtie \beta :\sim C/\gamma, \tag{18}$$

where C is a categorial attribute, α and β are two cedents that define two subsets of the data, for which the frequencies of C, characterized by the quantifier $\sim$ have some required difference. So in our data, a true SDCF pattern can be found for attribute income, subsets of unemployed and employed persons and the quantifier defined by absolute difference of frequencies of corresponding values computed within each subset and the restriction that the minimum absolute difference should be greater than 3,

$$\textit{unemployed}(\textit{no}) \bowtie \textit{unemployed}(\textit{yes}) :\sim \textit{income}. \tag{19}$$

The **KEX procedure** (authors M. Šimůnek and P. Berka, for details see [3]) mines for compositional decision rules, that can be used for classification. These rules have the form

$$Ant \Rightarrow C(v)[w] \tag{20}$$

where *Ant* is a conjunction of categories (KEX does not work with more structured literals like the other procedures implemented in LISp-Miner), C is the target attribute and v is a value of this attribute (so $C(v)$ is a class), and w (called weight) expresses the uncertainty of the rule.

The KEX algorithm searches the space of candidate rules in a heuristic top-down way. Unlike set covering algorithms where examples covered by a rule created during the learning process are removed from the training data (see e.g. [17, 10]), KEX repeatedly scans the whole data set, so an example can be covered by more rules. Thus more rules can be used during classification each contributing to the final decision about the class of an example. KEX uses a pseudo-Bayesian combination function borrowed from the expert system PROSPECTOR [11] to combine contributions of more rules:

$$w_1 \oplus w_2 = \frac{w_1 \cdot w_2}{w_1 \cdot w_2 + (1 - w_1) \cdot (1 - w_2)}. \tag{21}$$

Formula (21) is also used when deciding if a candidate rule should be added to the resulting rule-base and when computing the weight of the newly added rule (for details refer to [3]).

For the data from Table 3, KEX will find the decision rules shown in Table 8. Notice, that unlike founded implications created by 4ft-Miner procedure, rules created by KEX are not intended for individual interpretation by domain experts. This is because some strong rules (rules with high confidence) need not to be included if their confidence can be inferred from the rule-base and, on the contrary, some weak rules are included if their confidence cannot be inferred. So the rules should be used as a whole by the inference mechanism during classification. E.g., a new example described by the attribute values *income*(*high*), *balance*(*medium*), *sex*(*male*) and *unemployed*(*no*) will be classified using the rules[1]

$$\Rightarrow loan(yes)[0.6667]$$

$$income(high) \Rightarrow loan(yes)[0.9802]$$

$$balance(medium) \wedge unemployed(no) \Rightarrow loan(yes)[0.9512]$$

as belonging to class *loan*(*yes*) with the weight $0.6667 \oplus 0.9802 \oplus 0.9512 = 0.9995$.

The **ETree-Miner procedure** (authors M. Šimůnek and P. Berka) mines for so called exploration trees [4, 27]. This procedure is based on an extension of the well known top-down induction of decision trees (TDIDT) approach (see e.g. [19]). Instead of selecting single best attribute to make a split of the data, we select more suitable attributes. The result thus will be a set of trees (forest) where each tree represents one model applicable for both description and classification. It is also possible to classify examples using all trees, in this case the trees vote on the predicted class.

[1] A default rule with "empty" antecedent which is always added to the rule base can always be used to classify an example.

$\Rightarrow$ *loan*(*yes*)[0.6667]
income(*high*) $\Rightarrow$ *loan*(*yes*)[0.9802]
balance(*high*) $\Rightarrow$ *loan*(*yes*)[0.9753]
balance(*low*) $\wedge$ *income*(*low*) $\Rightarrow$ *loan*(*yes*)[0.0127]
balance(*medium*) $\wedge$ *unemployed*(*yes*) $\Rightarrow$ *loan*(*yes*)[0.0127]
balance(*medium*) $\wedge$ *unemployed*(*no*) $\Rightarrow$ *loan*(*yes*)[0.9512]
unemployed(*no*) $\wedge$ *sex*(*female*) $\Rightarrow$ *loan*(*yes*)[0.9512]
balance(*medium*) $\wedge$ *sex*(*female*) $\wedge$ *income*(*low*) $\Rightarrow$ *loan*(*yes*)[0.0256]

Table 8. Result of KEX example run

ETree-Miner uses the chi-square criterion to find a splitting attribute; the attributes in question are sorted according this criterion and then only k best attributes are used, k being the user-given parameter[2]. This criterion not only ranks the attributes but also evaluates the "strengths" of the relation between the evaluated attributes and the class attribute. So we can consider only significant splitting attributes. The significance testing allows also to reduce the number of generated trees; if the best splitting attribute is not significantly related with the class, we can immediately stop growing tree at this node and need not to consider other splitting possibilities.

ETree-Miner uses several stopping criteria to terminate the tree growing. Beside the standard node purity (the fraction of examples of the majority class) we can use also the node frequency (number of examples in a node), the depth of the tree and the above mentioned chi-square test of significance. The resulting exploration tree can be also filtered out according to their quality (classification accuracy on training data). When setting for our running example the number of best splitting attributes to 2, max. depth of the tree to 2, minimal node purity to 0.7, minimal node frequency to 1 and minimal quality to 0.7, we will obtain 6 exploration trees, two of them are shown in Table 9. The first tree has the quality 1 and the second tree has the quality 0.917.

The **MCluster-Miner procedure** (authors M. Šimůnek and T. Kliegr) implements a K-means clustering algorithm. A beta version of this procedure is included in the latest LISp-Miner version 22.09. This procedure is still under development, so far only the Euclidean distance (that can be properly used only for ordinal attributes) is implemented.

The procedure again follows the GUHA paradigm to "find everything interesting". So not only one partitioning of data into clusters is created. The user gives the minimal and maximal number of required clusters (k_{min} and k_{max}) and the minimal and maximal number of attributes used to describe the clustered examples ($natt_{min}$ and $natt_{max}$). The algorithm then generates the partitionings that correspond to these parameters.

[2] The number of possible trees grows exponentially, so k should be very small.

```
Root:  Tree Quality: 1.000
  |
  Balance= low
  |    |
  |    Income= low: Loan= no, Node Purity: 1.000
  |    |
  |    Income= high: Loan= yes, Node Purity: 1.000
  |
  Balance= medium
  |    |
  |    Unemployed= yes: Loan= no, Node Purity: 1.000
  |    |
  |    Unemployed= no: Loan= yes, Node Purity: 1.000
  |
  Balance= high: Loan= yes, Node Purity: 1.000


Root: Tree Quality: 0.917
  |
  Income= low
  |    |
  |    Balance= low: Loan= no, Node Purity: 1.000
  |    |
  |    Balance= medium: Loan= no, Node Purity: 0.667
  |    |
  |    Balance= high: Loan= yes, Node Purity: 1.000
  |
  Income= high: Loan= yes, Node Purity: 1.000
```

Table 9. Example ETrees

3.3 Work with LISp-Miner

The core of the LISp-Miner system are the procedures described above. Each procedure mentioned in Section 4 is realized using two modules, data processing module Task (e.g. 4ft-Task) is used to run the analysis, data interpretation module Results (e.g. 4ft-Results) is used to display, sort or select the resulting rules. Beside this, LISp-Miner also implements (in the DataSource module) a variety of data transformation and data preprocessing methods that can be used to select attributes for the specific data mining task, create derived attributes, or discretize numeric attributes [28]. There is also one administration module, LMAdmin, that assigns data and meta-data to a given task. The concept of meta-data allows (like in RapidMiner, IBM SPSS Modeler or SAS EM) to store and reuse inputs for the tasks as well the results obtained during analysis. Both data and meta-data are stored using a database (MS Access or MySQL are usually used).

The LISp-Miner data mining procedures require to set many parameters, that allow to fine tune the analysis. For instance when working with 4ft-Miner, the user must specify as input parameters:

- the definition of partial cedents for antecedent, succedent and condition respectively; each partial cedent is defied by the type of formula (conjunction/disjunction), min. and max. number of literals, and a list of possible literals (each literal is defined by the corresponding attribute, type of the coefficient, minimal and maximal length of the coefficient, specification if the literal should be used without negation, with negation or both with/without negation and specification if the literal is basic – i.e., if it must occur in the cedent or if the literal is so called remaining one),
- maximal length of antecedent, succedent and condition,
- the quantifiers $\approx$ and threshold values for their values,
- other task parameters, e.g., how to handle missing values.

To simplify the parameter setting, reasonable default values are assigned to these parameters. So the default settings for 4ft-Miner correspond to standard association rules (the quantifier is set to founded implication with parameter $p = 0.9$)[3], where the antecedent and succedent are created only from attribute-value pairs (i.e., the coefficient in the definition of a literal corresponds only to a single value of the attribute). It is also possible to "clone" an existing task, i.e. to reuse and only slightly modify its parameters.

Most of the procedures can be executed using PC grid; a large task can thus be split and solved in parallel on more computers.

The work with the LISp-Miner is not so straightforward as with other data mining systems (Weka, RapidMiner, IBM SPSS Modeler, SAS EM). This is because LISp-Miner is distributed as a number of executables that must be invoked by the user:

1. The first module to be used is the LMAdmin. The primary aim of this module is to attach the (empty) meta base to the analyzed data. This is a necessary step that must precede any analysis. The module has been recently enhanced by a control panel that allows to invoke most of the other modules.
2. The LMDataSource module must then be used to select (and preprocess) the tables and attributes to be analyzed.
3. To analyze the data, the respective xxxTask module must be used.
4. To display and evaluate the results, the corresponding xxxResult module must be run; the xxxResult module can be invoked directly from the xxxTask module.

The system is freely available on `http://lispminer.vse.cz`.

[3] When setting parameter p, it should be kept in mind that if $a/(a+b) < (a+c)/(a+b+c+d)$, the antecedent of a rule actually disconfirms the succedent.

4 LISP-MINER AND THE CRISP-DM METHODOLOGY

CRISP-DM methodology defines 6 basic steps of a data mining project: business understanding, data understanding, data preparation, modeling, evaluation and deployment [9]. LISp-Miner was intentionally created as a modular system that supports these steps. That's why each LISp-Miner procedure is implemented as two modules: xxxTask and xxxResult [28].

Business understanding. The use of domain knowledge in data mining tasks is a hot research topic within LISp-Miner. First results on using specific types of domain knowledge (meaningful range for numeric attributes, meaningful intervals, mutual influence of attributes – e.g., "if income increases then balance increases as well") for LISp-Miner runs can be found in [23].

Data understanding. The LMDataSource module can be used for initial exploration of the analyzed data.

Data preprocessing. As already mentioned, the mining procedures in LISp-Miner work only with categorial data. The LMDataSource module can perform various types of discretization (equidistant, equifrequent, user-given) to divide the range of a numeric attribute into intervals. Another supported data transformation is manual merging of values of a categorial attribute – this transformation can be used also for intervals created from a numeric attribute. It is also possible to create a so called derived attribute, here the LMDataSource uses the functionality of MS Access to create new field from existing ones.

Modeling. The data mining procedures that are used for the modeling step have been described in Section 3.

Evaluation. To evaluate the quality of classification models created using KEX and ETree-Miner, n-fold cross-validation or testing on a randomly selected testing examples can be performed. When evaluating the results from other procedures, the quality of the created models must be assessed by domain experts who should decide, if the found knowledge brings something new, interesting and useful to them. To go trough a list of found patterns (sometimes very huge) can be very tedious. LISp-Miner supports the evaluation step by various sorting, filtering and visualization possibilities, that help to speed-up the visual inspection of the resulting patterns. It is possible to sort the results according to the attributes contained in the knowledge patterns (lexicographic ordering) or according to the values of quantifiers. The filters are based on the attributes that occur in the patterns. It is also possible to visualize each found pattern using the corresponding contingency table and values of all quantifiers.

Recently a method based on the concept of meta-rules has been proposed to help with interpretation of the results obtained from 4ft-Miner [5]. The idea of this method is that we can encode the obtained 4ft-association rules as data and apply a subsequent mining step on them thus obtaining "rules about rules". To

illustrate this idea, let us assume that we found the 4ft-association rules shown in Table 5. These rules can be encoded as data as shown (for the first ten rules) in Table 10. Applying 4ft-Miner to these data we will obtain meta-rules shown in Table 11. The first meta-rule of this list can be interpreted as "IF the original rules contain the literals *income*(*low*) and *loan*(*yes*), THEN they also contain the literal *balance*(*high*)".

Rule	Income	Balance	Sex	Unemployed	Loan
r1	?	low	female	yes	?
r2	low	low	female	yes	no
r3	high	low	female	yes	yes
r4	high	low	female	yes	yes
r5	low	low	female	yes	no
r6	low	low	?	yes	no
r7	low	low	female	yes	?
r8	low	low	female	yes	no
r9	high	low	?	yes	yes
r10	high	high	female	yes	?

Table 10. Encoded association rules

$$
\begin{aligned}
\mathit{income}(\mathit{low}) \wedge \mathit{loan}(\mathit{yes}) &\Rightarrow \mathit{balance}(\mathit{high}) \\
\mathit{unemployed}(\mathit{yes}) \wedge \mathit{loan}(\mathit{yes}) &\Rightarrow \mathit{balance}(\mathit{high}) \\
\mathit{balance}(\mathit{high}) \wedge \mathit{unemployed}(\mathit{yes}) &\Rightarrow \mathit{income}(\mathit{low}) \\
\mathit{income}(\mathit{low}) \wedge \mathit{balance}(\mathit{high}) &\Rightarrow \mathit{unemployed}(\mathit{yes}) \\
\mathit{balance}(\mathit{high}) \wedge \mathit{unemployed}(\mathit{no}) &\Rightarrow \mathit{income}(\mathit{high})
\end{aligned}
$$

Table 11. Example meta-rules

Deployment. To support the deployment of the results, LISp-Miner can export the found knowledge patterns in PMML, HTML or text format. A more elaborated way of the deployment towards automatic creation of analytical reports is under investigation within the Sewebar project [15].

The classification models created by KEX and ETree-Miner can be applied on unseen data using the respective xxxResults procedure.

5 LISP-MINER AND TYPICAL DATA MINING TASKS

Knowledge discovery in databases is commonly used to perform the tasks of data description and summarization, segmentation, concept description, classification, prediction, dependency analysis, or deviation detection (Fayyad et al. 1996; Chapman et al., 2000). Let's have a look how these tasks can be solved using LISp-Miner.

Data description and summarization. The goal is the concise description of characteristics of the data, typically in elementary and aggregated form. This gives the user an overview of the structure of the data. Even such a very simple and preliminary analysis is appreciated by the data owners and users. The LMDataSource module can be used for this type of tasks as it allows to analyze and visualize the frequencies of values of the used attributes. Also the procedures CF-Miner and SDCF-Miner that analyze a single attribute in more details are well suited for this task.

Segmentation. Segmentation (or clustering) aims at the separation of data into interesting and meaningful subgroups or classes where all members of a subgroup share common characteristics. Examples of this type of task are clients profiling or clustering gene expression data. The newly added MCluster-Miner procedure can be used for this task.

Classification. Classification assumes that there is a set of objects which belong to different classes. The objective is to build classification models, which assign the correct class label to previously unseen and unlabeled objects. Examples of this type of task can be credit risk assessment and credit scoring, churn (retention) analysis, customer modeling (to evaluate the propensity to buy a product), or medical and technical diagnostics. LISp-Miner offers two procedures for this type of tasks: KEX and ETree-Miner. KEX that creates decision rules was developed especially for classification purposes, ETree-Miner creates exploration trees but these trees can be used for classification as well.

Prediction. Prediction is very similar to classification. The only difference is that in prediction the target attribute (class) is not a qualitative discrete attribute but a continuous one. Examples are exchange rate prediction, prediction of energy consumption or sales forecasting. LISp-Miner is not very suitable for this type of tasks because all mining procedures work only with categorial data. Anyway, it is possible to discretize the numeric target (e.g. into intervals "low", "medium", "high" or into intervals "decrease of the target in time", "increase of the target in time") and thus turn the prediction task into classification.

Concept description. Concept description aims at the understandable description of concepts or classes. The purpose is not to develop complete models with high prediction accuracy, but to gain insights. Examples of this type of task are description of loyal customers, bad loan applications or insurance claims frauds. LISp-Miner offers two procedures that can be used here: ETree-Miner and 4ft-Miner. The exploration trees created by the ETree-Miner procedure were actually intended for this task. It is also possible to express a concept using rules created by 4ft-Miner; here we can benefit from the fact, that 4ft-Miner generates each part of a rule separately. So we can set the succedent of the founded implications to be the concept in question (i.e. a value of the target attribute) and the antecedent of the founded implications to take values of the input attributes.

Dependency analysis. Dependency analysis consists of finding a model that describes significant dependencies (or associations) between data items or events. Market basket analysis can be an example of this type of task. LISp-Miner is especially oriented to this type of tasks. The 4ft-Miner, SD4ft-Miner and AC4ft-Miner procedures have been developed to find different types of relations between conjunctions of literals (including association rules), KL-Miner and SDKL-Miner procedures have been developed to find dependencies between categorial attributes. All these procedures go far beyond what is offered in other data mining systems.

Deviation detection. Deviation detection focuses on discovering the most significant changes in the data from previously measured or normative values. This is not a typical task to be solved using LISp-Miner, but when e.g. using in 4ft-Miner the setting $a \leq CEIL$, we will find rules that cover only small part of the data. Such rules can be interpreted as rules describing unusual situations.

6 CASE STUDY

We used a very simple illustrative example in Section 3 to describe the different types of rules that can be created using LISp-Miner. This section shows some results of analysis of a real-world medical data taken from the atherosclerosis risk domain and it follows the work presented in [6]. The aim is to show the variety of different knowledge patterns that can be found by the LISp-Miner procedures.

6.1 Problem and Data Description

The data collected within this study concern the twenty years lasting study of the risk factors of the atherosclerosis in the population of 1417 middle aged men [8]. The men were divided according to presence of risk factors (RF), overall health conditions and ECG result into following three groups: normal (a group of men showing no RF), risk (group of men with at least one RF) and pathological (group of men with a manifested cardio-vascular disease). The considered risk factors were arterial hypertension (BP $\geq$ 160/95 mm Hg), cholesterol (cholesterol level $\geq$ 200 mg %), triglycerides (triglycerides level $\geq$ 200 mg %), smoking (smoking $\geq$ 15 cig./day), overweight (Brocka index $>$ 115 %), positive family case history (death of father or mother from cardiovascular disease before reaching 65 years of age). Long-term observation of patients was based on checking and monitoring the men from normal group and risk group. The men from the pathological group were excluded from further observation. The study was realized at the 2nd Department of Medicine, 1st Faculty of Medicine of Charles University and Charles University Hospital, Prague. The data were transferred to the electronic form by the European Centre of Medical Informatics, Statistics and Epidemiology of Charles University and the Academy of Sciences, Prague.

We will report results obtained when analyzing the data about the men when entering the study. These data consider life style, personal and family history, physical examination, and special laboratory tests. The risk factors and the classification into one of the three groups (normal, risk, pathological) were included as well. The used attributes are summarized in Table 12.

Groups of Attributes	No. of Attributes
group	1
social characteristics (age, marital status, education, position in job)	4
physical activity (in job, after job)	4
smoking	3
alcohol (beer, vine, liquers)	10
sugar, coffee, tea	3
personal anamnesis (myocardial infarction, hypertension, ictus, diabetes, hyperlipidemia)	18
Questionnaire A_2 (chest pain, lower limbs pain, asthma)	3
physical examination (height, weight, blood presssure, skin fold)	8
biochemical examination (cholesterol, triglycerides, urine)	3
risk factors	5

Table 12. Summary of the data

6.2 Results of 4ft-Miner

4ft Miner was used to answer the following analytical questions:

- What are the relations between social factors and physical activity, smoking, alcohol consumption, BMI, blood pressure, cholesterol, triglycerides for men in the respective groups? Examples of found rules are

$$\begin{aligned} &\textit{Education}(\textit{university}) \wedge \textit{Job position}(\textit{manager}) \\ &\Rightarrow_{74,0.90} \textit{Physical activity in job}(\textit{mainly sits}) / \textit{Group}(\textit{risk}), \end{aligned} \tag{22}$$

  i.e., within the risk group, 90 % of patients with university education and working as managers are mainly sitting in their work (there were 74 such persons), or

$$\textit{Education}(\textit{basic}) \wedge \textit{Job position}(\textit{worker}) \Rightarrow_{101,0.82} \textit{Beer}(\textit{yes}), \tag{23}$$

  i.e., 82 % of patients with basic education and working as workers also drink beer (there were 101 such persons).

- What are the relations between physical activity and smoking, alcohol consumption, BMI, blood pressure, cholesterol, triglycerides for men in the respective groups? An example of found rules is

$$\begin{aligned} &\textit{Physical activity after work}(\textit{high}) \\ &\Rightarrow_{35,0.88} \textit{BMI}(22\text{–}27)/\textit{Group}(\textit{normal}) \end{aligned} \tag{24}$$

  i.e., in the group of normal patients, 88 % of patients who are physically active after their work have low BMI (there were 35 such persons).

- What are the relations between alcohol consumption and smoking, BMI, blood pressure, cholesterol, triglycerides for men in the respective groups? An example of found rules is

$$\begin{aligned} &\textit{Alcohol}(\textit{regularly}) \wedge \textit{Liquers}(\textit{yes}) \wedge \textit{Beer}(\textit{more than 1 liter per day}) \\ &\Rightarrow_{20,0.74} \textit{Smoking}(\textit{more than 20 years}) \end{aligned} \tag{25}$$

  i.e., 74 % of patients who regularly drink alcohol, i.e. liquers and beer, also smoke for more than 20 years (there were 20 such persons).

6.3 Results of SD4ft-Miner

Here we were interested in pairs of rules that relate together physical examinations and life-style (antecedent) with blood pressure (succedent) and that have different confidences for group of normal and group of risky men. One of the most interesting SD4ft patterns was the rule

$$\begin{aligned} &\textit{Non-smoker}(\textit{yes}) \wedge \textit{Beer}(\textit{up to 1 liter per day}) \\ &\Rightarrow \textit{Diastolic blood pressure}(60\text{–}90) \end{aligned} \tag{26}$$

for which the confidence in the normal group was 0.85 and in the risk group only 0.41.

6.4 Results of AC4ft-Miner

The attribute is stable if its value cannot be changed, e.g. *Education*, *Age* or *Height*. The attribute is flexible if its value can be subject to change, e.g. *Weight* or *Blood pressure*. Let us assume that *Systolic blood pressure* and *Weight* have categories *low*, *medium*, *high*. Then the expression

$$\begin{aligned} &\textit{Age}(38\text{–}42) \wedge \textit{Weight}(\textit{medium} \rightarrow \textit{low}) \\ &\Rightarrow_{0.51\rightarrow 0.57,21,73} \textit{Systolic blood pressure}(\textit{medium} \rightarrow \textit{low}). \end{aligned} \tag{27}$$

is an example of a G-action rule. Here *Age(38–42)* is a stable antecedent, *Weight (medium → low)* is a change of antecedent and *Systolic blood pressure(medium →*

low) is a change of succedent. This rule actually corresponds to two 4ft association rules. The rule

$$Age(38\text{–}42) \wedge Weight(medium) \Rightarrow Systolic\ blood\ pressure(medium) \tag{28}$$

has confidence 0.51 and support 21, the rule

$$Age(38\text{–}42) \wedge Weight(low) \Rightarrow Systolic\ blood\ pressure(low) \tag{29}$$

has confidence 0.57 and support 73.

6.5 Results of KL-Miner

We solved the analytical question "Are there any strong ordinal dependencies among attributes describing physical examination, biochemical examination and blood pressure in the set of all patients or in some subgroups of patients described by lifestyle?" An example of such ordinal dependency is "For the patients drinking more than 3 cups of coffee daily, there is a strong dependency: if systolic blood pressure increases, then diastolic blood pressure increases as well". There are 462 patterns of the form Systolic $\sim$ Diastolic/γ or Diastolic $\sim$ Systolic/γ , the strongest pair of patterns is for γ defined by the conjunction

$$Coffe(1\text{–}2\ cups\ per\ day) \wedge Beer(up\ to\ 1\ liter\ per\ day) \wedge Vine(not\ drink) \wedge Liquors(not\ drink). \tag{30}$$

6.6 Results of KEX

Using KEX we want to create a set of decision rules able to classify men into two groups: normal and abnormal (risk or pathological group). We run several tasks for different subsets of input attributes:

T1: classification based only on already known risk factors (this rule base should confirm the initial assignment of patients to the groups in the analyzed data),

T2: classification based on attributes concerning life style, personal and family history (but without special laboratory tests),

T3: classification based on attributes concerning life style and family history,

T4: classification based only on attributes concerning life style.

We defined these tasks to find a trade-off between the classifier accuracy and the amount of information, an unskilled user must provide to the classifier. Ideally, the input information should refer only to the person itself (this corresponds to the task T4). The classification accuracies (computed using 10 fold cross-validation) of the rule bases resulting from these analyzes are summarized in Tab 13. We can see, that when decreasing the amount of used information (from task T2 to Task T4), the

classification accuracy also decreases (which we expected). We can also see, that the classification into abnormal groups was always more accurate than the classification into normal group. So the classifiers assign more patients from abnormal groups into normal group than vice versa (e.g., in Task 2, 26 % of patients that belong to abnormal group were classified as normal while only 13 % of normal patients were classified as abnormal). Such behavior is not very suitable for a system that should recommend a visit at a physician. Here we would prefer the opposite – a classifier that rather sends a healthy person to a doctor than does not recognize a person with possible problems. This situation can be in general handled by using different misclassification costs (KEX does not allow this possibility) or by re-sampling the used data.

		Accuracy		
Task	No. of Rules	Total	Classified as Normal	Classified as Abnormal
T1	19	0.87	0.83	0.88
T2	39	0.94	0.74	0.87
T3	32	0.77	0.63	0.83
T4	27	0.73	0.48	0.83

Table 13. Classification accuracy for different tasks

6.7 Summary of the Experiments

The results of 4ft-Miner, SD4ft-Miner, AC4ft-Miner and KL-Miner can be used as a proof of concept of the usability of data mining methods in the respective domain. However, the reported patterns which were found do not enhance the knowledge of the domain experts.

The results of task T2 performed by KEX have been used when building an expert system for atherosclerosis risk assessment. This expert system was comparable with standard CVD risk calculators. More details about the created expert system can be found in [7].

7 LISP-MINER AND OTHER DATA MINING SYSTEMS

LISp-Miner was related to other data mining tools throughout the text. Let us make such comparison more explicit by looking at the state-of-the-art freely available data mining systems Weka and Rapid Miner.

Weka (Waikato Environment for Knowledge Analysis) is a data mining system that is under development at the Waikato University in New Zealand since 1997 [14]. Weka was the first widely used data mining tool. It gained its popularity because it was described (and thus recommended) in a well known textbook on data mining by Eibe Frank and Ian Witten (first edition in 1999, the latest third

edition from 2011 [29]). Weka integrates a large number of different machine learning algorithms (oriented mainly on classification and prediction tasks) and data preprocessing methods. The authors of Weka also encourage the users to include their implemented algorithms into the system. Weka offers four modes of operation: simple command line interface, Explorer (for a single analysis in an easy-to-use standard windows interface), Experimenter (to create batch of runs in a standard windows interface) and KnowledgeFlow (for a single analysis using a graphical interface that allows to compose a task as a graph where nodes correspond to particular operations and arrows indicate the data flow). The system is available at `http://www.cs.waikato.ac.nz/ml/weka/`.

Rapid Miner started in 2001 as a research project at the University of Dortmund under the name Yale (Yet Another Learning Environment) [18]. Nowadays, this system became number one among the freely available data mining tools. Rapid Miner includes all the methods and algorithms from Weka (and much more), and offers better graphical capabilities for visualization of data and models than Weka. Graphical interface is also used (as the only way) to work with the system, again a task is composed from nodes (defining different operators) into a process graph. Also this system can be extended by different plugins provided by the users, there is even a marketplace (at `http://marketplace.rapid-i.com/`) where these plugins can be shared in the user community. Rapid Miner is (after registration) available at `http://rapidminer.com`.

Both systems offer a broader range of implemented machine learning methods, as both systems have been from the beginning created as multipurpose one. LISp-Miner on the contrary is gradually evolving from the 4ft-Miner, so its main strengths is the variety of rule-like patterns that the system searches for. Some other characteristics and their comparison are shown in Table 14.

8 CONCLUSIONS

The paper reviews the data mining system LISp-Miner. When compared with the state-of-the-art freely available data mining systems like Weka or RapidMiner, LISp-Miner is focused on mining rule-like patterns. Here the system offers richer syntax and wider variety of patterns that can be mined than the standard tools. Nevertheless, by including new data mining procedures ETree-Miner and MCluster-Miner, LISp-Miner gradually becomes more universal. And new extensions of the system are under consideration. It would be very easy to enhance LISp-Miner by a naive Bayesian classifier, because the probabilities $P(E|H)$, that are necessary can be computed from 2×2 tables of 4ft-association rules $H \Rightarrow E$, where H is a category of the target attribute (i.e. class) and E are in the sequence all categories of all input attributes. A "GUHA-like" implementation could then generate more Bayesian classifiers by considering different subsets of input attributes.

The LISp-Miner system has been used in several data mining projects, e.g. in the Discovery Challenge workshops held within the ECML/PKDD Conferences,

Characteristics		LISp-Miner	Weka	Rapid Miner
usage	software requirements	MS Windows + database	(MS Windows or Linux or Mac OS) + java runtime	(MS Windows or Linux or Mac OS) + java runtime
	working mode	windows mode, command line	command line, windows mode (explorer or experimenter), graphical mode	graphical mode
	save/load tasks	yes (using MetaBase)	in experimenter mode	yes (as process)
	distributed computation	yes	in experimenter mode	no
	extendable by users	no	yes	yes
data	data tables	more tables	single	more tables
	attribute types	numeric (must be preprocessed), nominal, ordinal, string	numeric, nominal, string	numeric, nominal, string
	native data format	no (mdb files used)	arff, xrff files	repository
	import data	text files	csv, database	csv, xls, xml, sas, spss, arff, xrff, database
	export data	text files	csv, arff, xrff	csv, xls, xml, sas, spss, arff, database
	metadata	yes (within MetaBase)	no	yes (within data repository)
data preprocessing	supported transformations	unsupervised preproc. of attributes	both unsupervised and supervised preproc. of attributes and instances	both unsupervised and supervised preproc. of attributes and instances
	save and load preprocessed data	yes	yes	yes
modeling	mining methods	association rules, decision rules, dependency analysis, clustering, trees	association rules, decision rules, dependency analysis, clustering, trees, regression, neural networks SVM, Bayesian methods, instance based	association rules, decision rules, dependency analysis, clustering, trees, regression, neural networks, SVM, Bayesian methods, instance based, correlation
	combine classifiers	no	yes	yes
	save models	yes (in MetaBase)	yes	yes (as process)
evaluation	filter outputs	yes	no	yes
	combine classifiers	no	yes	yes
	vizualize resluts	yes	yes	yes
deployment	export models	pmml, html, text files	java, native format	xml, binary
	apply classifiers on new data	yes	yes	yes

Table 14. Comparison of systems

where in subsequent years the analyzed data were taken from various domains: banking, medicine or e-commerce (this event was not of a competitive nature, instead of announcing a winner there were discussions with domain experts coming to the workshop), or in the analyzes of medical data from the atherosclerosis risk domain [6].

REFERENCES

[1] Alavi, M.—Leidner, D.: Knowledge Management and Knowledge Management Systems. MIS Quarterly, 2001, Vol. 25, No. 1, pp. 107–136.

[2] Agrawal, R.—Imielinski, T.—Sawami, A.: Mining Associations Between Sets of Items in Massive Databases. Proceedings of the ACM-SIGMOD International Conference on Management of Data, Washington D.C., 1993, pp. 207–216.

[3] Berka, P.: Learning Compositional Decision Rules Using the KEX Algorithm. Intelligent Data Analysis, Vol. 16, 2012, No. 4, pp. 665–681.

[4] Berka, P.: ETree Miner: A New GUHA Procedure for Building Exploration Trees. In: Kryszkiewicz, M., Rybinski, H., Skowron, A., Ras, Z. W. (Eds.): Foundations of Intelligent Systems, 19th International Symposium on Methodologies for Intelligent Systems (ISMIS 2011). Springer, Lecture Notes in Computer Science, Vol. 6804, 2011, pp. 96–101.

[5] Berka, P.—Rauch, J.: Meta-Learning for Post-Processing of Association Rules. In: Pedersen, T. B., Mohania, M. K., Tjoa, A. M. (Eds.): 12th International Conference Data Warehousing and Knowledge Dicovery (DaWaK 2010). Springer, Lecture Notes in Computer Science, Vol. 6263, 2010, pp. 251–262.

[6] Berka, P.—Rauch, J.—Tomečková, M.: Data Mining in the Atherosclerosis Risk Factor Data. In: Berka, P., Rauch, J., Zighed, D. A. (Eds.): Data Mining and Medical Knowledge Management: Cases and Applications. IGI Global, 2009, pp. 376–397.

[7] Berka, P.—Tomečková, M: Atherosclerosis Risk Assessment Using Rule-Based Approach. In: Ras, Z. W., Dardzinska, A. (Eds.): Advances in Data Management. Springer, Studies in Computational Intelligence, Vol. 223, 2009, pp. 333–350.

[8] Boudík, F.—Reissigová, J.—Hrach, K.—Tomečková, M.—Bultas, J.—Anger, Z.—Aschermann, M.—Zvárová, J.: Primary Prevention of Ischemic Heart Disease in Middle-Aged Men Living in Prague: Results of Twenty-Year Research. Vnitřní Lékařství, Vol. 52, 2006, No. 4, pp. 339–347.

[9] Chapman, P.—Clinton, J.—Kerber, R.—Khabaza, T.—Reinartz, T.—Shearer, C.—Wirth, R.: CRISP-DM 1.0. Step-by-Step Data Mining Guide. SPSS Inc., 2000.

[10] Clark, P.—Niblett, T.: The CN2 Induction Algorithm. Machine Learning, Vol. 3, 1989, pp. 261–283.

[11] Duda, R. O.—Gasching, J. E.: Model Design in the Prospector Konsultant System for Mineral Exploration. In: Michie, D. (Ed.): Expert Systems in the Micro Electronic Age. Edinburgh University Press, UK, 1979.

[12] Hájek, P.—Havránek,T.: Mechanizing Hypothesis Formation. Mathematical Foundations for a General Theory, Springer, 1978.

[13] Hájek, P.—Holeňa, M.—Rauch, J.: The GUHA Method and Its Meaning for Data Mining. Journal of Computer and System Science, Vol. 76, 2010, No. 1, pp. 34–48.

[14] Hall, M.—Frank, E.—Holmes, G.—Pfahringer, B.—Reutemann, P.—Witten, I. H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations, Vol. 11, Issue 1, 2009.

[15] Kliegr, T.—Svátek, V.—Ralbovský, M.—Šimůnek, M.: SEWEBAR-CMS: Semantic Analytical Report Authoring for Data Mining Results. Intelligent Information Systems [online], 2010, pp. 1–25.

[16] Lín, V.—Dolejší, P.—Rauch, J.—Šimůnek, M.: The KL-Miner Procedure for Datamining. Neural Network World, 2004, No. 5, pp. 411–420.

[17] Michalski, R. S.: On the Quasi-Minimal Solution of the General Covering Problem. Proceedings 5th International Symposium on Information Processing (FCIP-69), Bled, Yugoslavia, 1969, pp. 125–128.

[18] Mierswa, I.—Wurst, M.—Klinkenberg, R.—Scholz, M.—Euler, T.: YALE: Rapid Prototyping for Complex Data Mining Tasks. Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006, pp. 935–940.

[19] Quinlan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufman, 1993.

[20] Ras, Z.—Wieczorkowska, A.: Action-Rules: How to Increase Profit of a Company. In: Zighed, D. A., Komorowski, J., Zytkow, J. (Eds.): Principles of Data Mining and Knowledge Discovery. Springer, Lecture Notes in Computer Science, Vol. 1910, 2000, pp. 587–592.

[21] Rauch, J.: Observational Calculi and Association Rules. Springer, Studies in Computational Intelligence, Vol. 469, 2013.

[22] Rauch, J.: Considerations on Logical Calculi for Dealing with Knowledge in Data Mining. In: Ras, Z. W., Dardzinska, A. (Eds.): Advances in Data Management. Springer, Studies in Computational Intelligence, Vol. 223, 2009, pp. 177–201.

[23] Rauch, J.—Šimůnek, M.: Applying Domain Knowledge in Association Rules Mining Process – First Experience. Proceedings of 19th International Symposium (ISMIS 2011), Warsaw, Poland, 2011, Foundations of Intelligent Systems. Springer, Lecture Notes in Computer Science, Vol. 6804, 2011, pp. 113–122.

[24] Rauch, J.—Šimůnek, M.: Action Rules and the GUHA Method: Preliminary Considerations and Results. In: Rauch, J., Ras, Z. W., Berka, P., Elomaa, T. (Eds.): Foundations of Intelligent Systems. Springer, Lecture Notes in Computer Science, Vol. 5722, 2009, pp. 76–87.

[25] Rauch, J.—Šimůnek, M.: An Alternative Approach to Mining Association Rules. In: Lin, T. Y. (Ed.): Proceedings Data Mining: Foundations, Methods, and Applications. Springer, 2005, pp. 219–238.

[26] Rauch, J.—Šimůnek, M.: GUHA Method and Granular Computing. In: Hu, X. et al. (Eds.): Proceedings of IEEE Conference Granular Computing 2005, IEEE, 2005, pp. 630–635.

[27] Šimůnek, M.: Nová GUHA-procedura ETree-Miner v systému LISp-Miner. Systémová Integrace, Vol. 19, 2012, No. 2, pp. 62–72 (in Czech).

[28] Šimůnek, M.: Academic KDD Project LISp-Miner. In: Abraham, A., Franke, K., Koppen, M. (Eds.): Intelligent Systems Design and Applications. Springer, Advances in Soft Computing, Vol. 23, 2003, pp. 263–272.

[29] Witten, I. H.—Frank, E.—Hall, M. A.: Data Mining: Practical Machine Learning Tools and Techniques. Third Edition. Morgan Kaufmann, 2011.

Petr Berka is Full Professor at the Department of Information and Knowledge Engineering, University of Economics, Prague. His main research interests are machine learning, data mining and knowledge-based systems. He is a member of the editorial board of the Intelligent Data Analysis journal and the International Journal on Intelligent Information Systems. He also serves as the program committee member on a number of international conferences in the area of machine learning and knowledge discovery in databases.