TAMPERE UNIVERSITY OF TECHNOLOGY

GEORGY MINAEV

Comparing GUHA and Weka Methods in Data Mining

Master Thesis

Supervisors: Associate Professor Esko Turunen
Professor Ari Visa
Inspectors and approved subject
Department of Mathematics
Department of Signal Processing
meeting 07.11.2012

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
MINAEV, GEORGY: Comparing GUHA and Weka Methods in Data Mining
Master of Science Thesis, 67 pages, 0 Appendix pages
07 November 2012
Major: Mathematics
Examiner: Associate Professor Esko Turunen and Professor Ari Visa
Keywords: data mining, GUHA, Weka, Association rules

The development of computers has enabled the collection and storage of terabytes of data and the creation of large data warehouses. The main problems with such data are their size and structure. The fundamental intellectual challenges at present are the analysis and understanding of the data in the decision-making process. This thesis introduces and compares the methods of GUHA and Weka software.

The thesis highlights the differences between GUHA and Weka software through taking 2 methods which reveal the association rules of the Weka programme and comparing them with three methods which reveal the association rules of the GUHA programme. The difficulty of the task is the amount of computation which has to be done to explain whether the methods have any differences or not.

The work has been done by taking the results from one of the Weka methods and comparing these with all the methods of GUHA. The second Weka method provides the same results as the first one, but in a different order. The results have been carefully compared and there are some comments in the discussion part of the thesis.

FOREWORD

The thesis has been written in the departments of Mathematics and Signal Processing of Tampere University of Technology.

I would like to thank the supervising associate professor Esko Turunen for his advice and inspection of my work. I would also like to say thank you to professor Ari Visa for his advice and inspection of some parts of the thesis. I would also like to mention professor Jan Rauch for providing a copy of his work.

I want to say many thanks to all my colleagues for their cooperation and for the general atmosphere during my work. I have many happy memories. Finally, I would like to say thanks to my parents, who have supported and encouraged me during the whole study process.

07 November 2012, Tampere

Georgy Minaev

CONTENTS

1. INTRODUCTION

> We are drowning in information, but starved for knowledge. - John Naisbitt

Many years ago computers were very slow and used only a small amount of memory for storing data, but as new and more powerful computers have been developed, the amount of memory available has increased by megabytes, gigabytes and lately terabytes. This has thrown up a new problem. One can store a huge amount of data, but the operator has no idea how to find anything interesting in the data, or even whether the data contains something interesting or not.

Nowadays there are many different applications based on completely different approaches, such as the Weka and GUHA programs. The Weka program uses the knowledge mining approach, while the GUHA program uses the data mining approach. Data mining is more mathematical and can be explained mathematically, because it is based on many different formulas. Knowledge mining, on the other hand, is at a level above the data mining approach, and this approach is quite difficult to explain with one formula, since it uses models.

The main objective of the thesis is to provide information about the Weka and GUHA programs and compare the two methods in action with a small database.

After the introduction and a brief history of data mining in Section 2, the thesis consists of 5 main sections. Section 3 deals with the theoretical bases of the GUHA method; (4), the theoretical bases of the Weka method; (5), the theoretical hypotheses of the GUHA and Weka softwares; (6), the results of the approaches in practice; and (7), some discussion about the theoretical hypotheses and the practical results.

The thesis explains the theoretical bases of the GUHA and Weka softwares; presents practical results on a given artificial database and discusses the actual results. The reader need not start reading from the beginning of the thesis. If he or she has the necessary GUHA and/or Weka knowledge, the reader can omit reading either or both of the theoretical GUHA or Weka parts. The reader can read the table of contents and decide which parts are of interest and only need read those parts.

The goal of the thesis is to give the reader a basic understanding of GUHA and Weka methods. The explanation begins with the basic theory of the association rules of the GUHA and Weka approaches and then, building upon these fundamental concepts, generates hypotheses and practical results and, finally, presents a discussion of the actual results.

The reader should be familiar with basic Boolean logic theory.

2. HISTORY OF DATA MINING

The definitions of the chapter are from the source [1].

Computer science has progressed rapidly over recent decades, all of which has helped to develop more powerful and ever larger databases.

Databases were introduced as files before the 1960s, but subsequently more powerful database systems were developed. The development of databases continued and by the beginning of the 1970s relation database systems had been developed. In addition, users had access to data through query language, user interface and transaction management. Users then moved onto more efficient methods like on-line transaction processing (OLTP). OLTP is a tool which allows the use of relational databases and working with a large amount of data. Increased interest in relation databases in the 1980s led to the development of different approaches. For example, the extended-relational, object relation and deductive models. The size of databases continued increasing and eventually achieved world-wide size. Heterogeneous databases and Internet-based global information systems were introduced in the form of the World Wide Web.

Nowadays, data can be stored in databases and informational repositories. A repository can be used as a data warehouse. One data warehouse can store different types of data, organized as a unified schema so that a manager can make correct decisions. Data cleaning and data integration are elements of data warehouse technology. On-line analytical processing (OLAP) is part of the technology too. OLAP is a technique which allows data to be analysed using summary, consolidation and aggregation from different angles. OLAP is a very useful tool, but it requires additional data analysis tools for in-depth analysis, such as data classification, clustering, and the characterization of data, which changes over time. Nowadays, huge amounts of data can be accumulated not only in databases or data warehouses, but also using World Wide Web technology.

The analysis of data is a very demanding and challenging task, leading to the expression *data rich but information poor*. The fast growth of data in data repositories reached such levels that the data exceeded the human capacity to analyze it without powerful tools. Therefore, users only rarely visited big databases, and decisions which should have been made using information in the databases were often made on the basis of the decision-maker's intuition, because the decision-maker did not have the proper tools needed to find the important and relevant information from the massive databases.

It was situations like this that led to the systematic development of ***data mining tools***.

There are many definitions of data mining. For example, knowledge mining from data, knowledge extraction or data archaeology. Knowledge Discovery from Data (KDD) is the term which is used nowadays by many people. The main steps of knowledge discovery are: data preprocessing, data mining, pattern evaluation and knowledge presentation. Fig. 2.1 shows these steps.

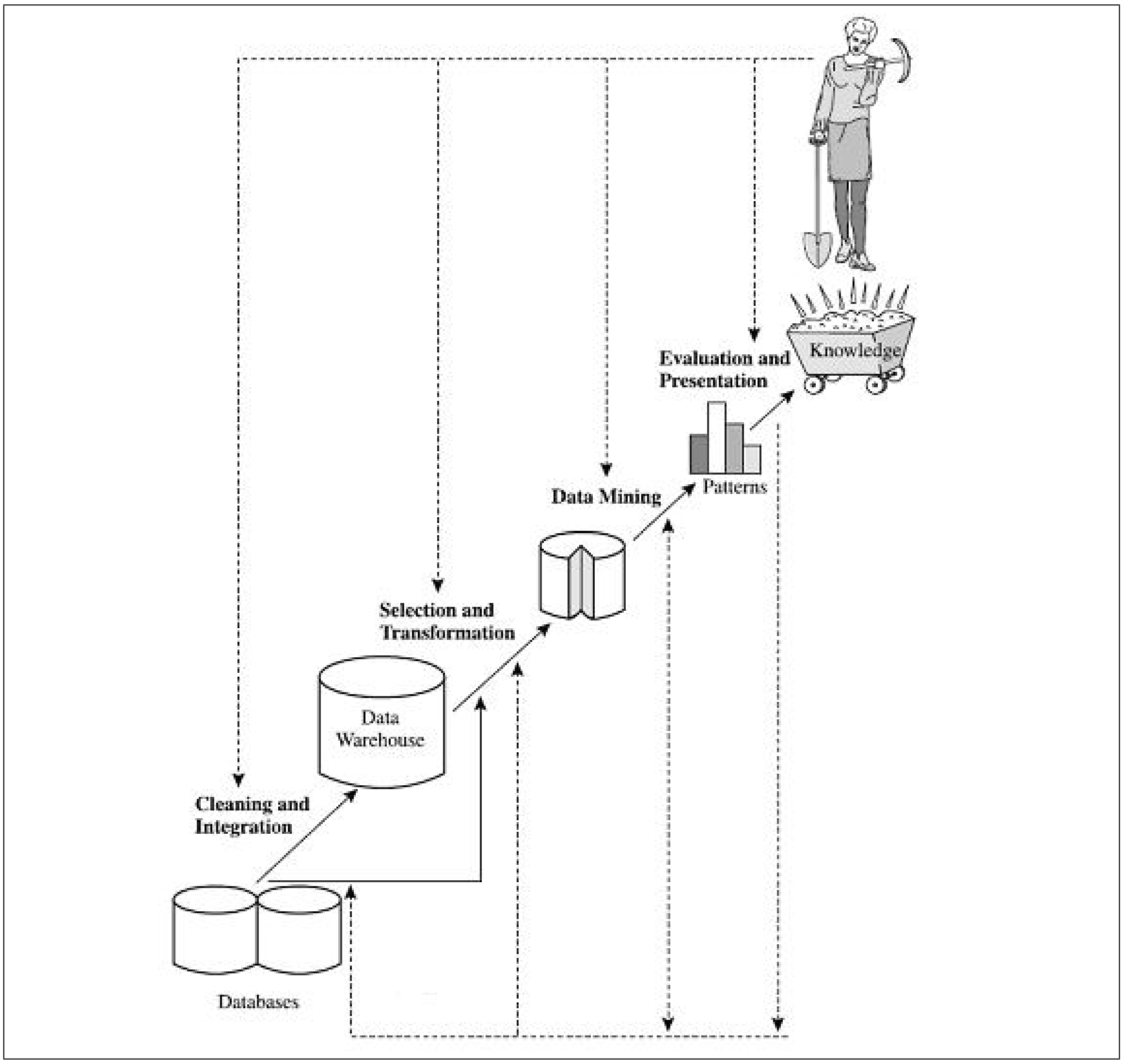

Figure 2.1: Data mining as a step of knowledge discovery [1].

Data preprocessing is the step in which the data is prepared for data mining. There are at least four possible stages in preprocessing: data cleaning, data integration, data selection and data transformation. Data cleaning is the step in which noise and inconsistent data are removed. Data integration is the step in which one can combine different data sources. Data selection is the step in which one can choose only the relevant data from the database. Data transformation is the step whereby one can transform the data for mining by performing summary or aggregation operations.

Data mining is the step in which different intelligent tools and methods are used to extract data patterns. The subsequent pattern evaluation step identifies whether a pattern is interesting for the user or not. The knowledge representation step enables the mined knowledge to be visualised for the user.

The user can interact with the data mining step and each step can interact with the knowledge database. The knowledge database can be used for storing interesting patterns, after the patterns have been checked by the user. The data mining step is only one step, but it is a very important one, because it uncovers hidden patterns from a database for subsequent evaluation.

So, **data mining** can be defined as, 'the process of discovering interesting knowledge from large amounts of data stored in databases...' [1].

3. THEORETICAL BASES OF THE GUHA METHOD

3.1 History

The definitions of the chapter are from the source [4].

The GUHA principle was introduced by Hájek-Havel-Chytil [3] in 1966. GUHA is the acronym for General Unary Hypotheses Automation (GUHA); the authors only later realised that GUHA is quite a popular name in India. The method generates interesting hypotheses from a given data base. The next book was published by Hájek and Havranek in 1978. Several other books have been published by different authors since 1978, but the response to these books has been less than overwhelming. One of the possible reasons for this might be the steadily increasing difficulty with getting one of the first books published in 1978.

3.2 The first steps in the GUHA method

The definitions of the chapter are from the source [5].

The GUHA method was developed in Czechoslovakia. The method enables the postulation of interesting hypotheses from a given database. The method is developed with GUHA-procedures. A GUHA-procedure is a computer program. The computer program uses simple definitions and a given database to raise interesting hypotheses (see the principle of GUHA method in Fig. 3.1).

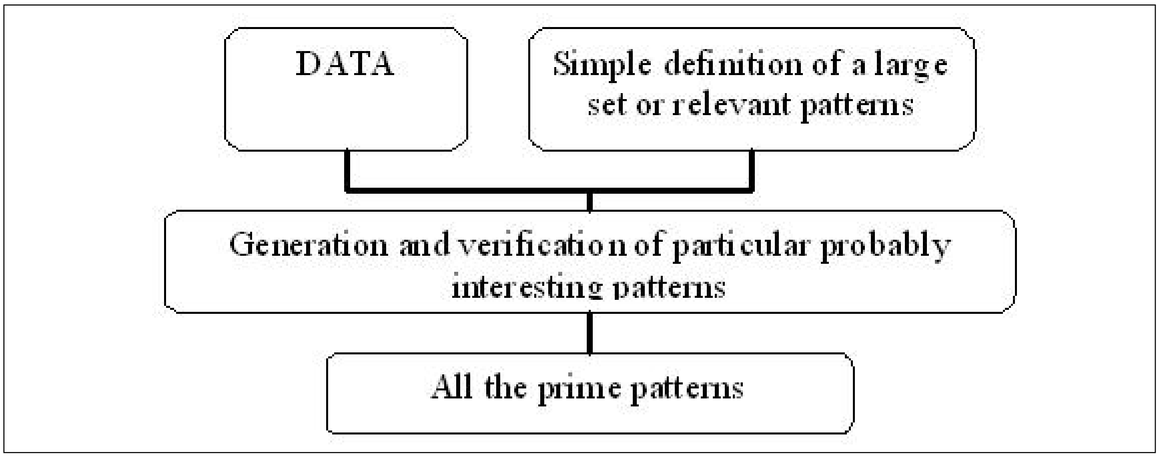
DATA
Simple definition of a large set or relevant patterns
Generation and verification of particular probably interesting patterns
All the prime patterns

Figure 3.1: GUHA method principle [4].

The pattern is prime if the simple definition is true and the pattern does not arise from a simpler pattern. GUHA methods work with data matrices. The most important GUHA procedures are those which mine with association rules (see Chapter 3.4.2).

3.3 Data matrices

The definitions of the chapter are from the source [5].

The example data matrix M in table 3.1 has 50 attributes. Each attribute is introduced in the data matrix in columns. Every attribute has a finite number of categories (values). For example, attribute A1 has categories {1,2,3,4}.

Potentially interesting patterns could be mined from the categorical attributes or Boolean attributes or both. Literals are basic Boolean attributes like $A(a)$, or $\neg A(a)$,

Table 3.1: An example of data matrix M.

Object	A_1	A_2	...	A_{50}	A_1(1,2)	$\neg A_{50}$(6)
o_1	*1*	*4*	...	*4*	*T*	*T*
o_2	*4*	*3*	...	*6*	*F*	*F*
o_3	*2*	*6*	...	*7*	*T*	*T*
...	...	...	...	...	...	...
o_n	*3*	*1*	...	*36*	*F*	*T*

where a is a set of all the categories of column A. A Literal $A(a)$ is true if the value of a row of a Data matrix is a. For example, A_1*(1)* is true in the first o_1 row in data matrix M. Two columns A_1*(1,2)* and $\neg A_{50}$*(6)* are examples of literals from the data matrix M.

Every attribute has its own *card of categories*, represented as a string of bits. For example, data matrix M in the table 3.2 shows the card A_1. A card has only one '1' bit which corresponds to the value A_1 with respect to a row.

Table 3.2: Cards of categories A_1.

Row	A_1	Cards of categories A_1			
		A_1[1]	A_1[2]	A_1[3]	A_1[4]
o_1	*1*	*1*	*0*	*0*	*0*
o_2	*4*	*0*	*0*	*0*	*1*
o_3	*2*	*0*	*1*	*0*	*0*
...	...	...	...	...	...
o_n	*3*	*0*	*0*	*1*	*0*

Boolean attributes have similar cards. A card *C(y)* is '1' if and only if y is true in the corresponding row of the card. There are three *bit-wise* operations. They are $\vee$, $\wedge$ and $\neg$. The three operations can be rapidly calculated by a computer. The number of '1's in a bit string can be quickly calculated with the command *Count.*

Table 3.3 shows an example of a random 4ft-table, where ψ and φ are Boolean attributes. Variables a,b,c and d are natural numbers. Every variable corresponds to ψ and φ. For example, variable a is a natural number of rows, which are true for ψ

and φ. Mathematically the variable a is $Count(C(\varphi)\ and\ C(\psi))$, $b = Count(C(\varphi) - a)$, $c = Count(C(\psi) - a)$, d = n-a-b-c, where n is the amount of rows in the data matrix.

Table 3.3: 4ft-table.

M	ψ	$\neg\psi$
φ	a	b
$\neg\varphi$	c	d

Definition 3.3.1. (4ft table) *4ft table is a quadruple* $\langle a,b,c,d \rangle$ *where* a,b,c,d *are non-negative integers and* $a+b+c+d>0$.

3.4 Theoretical bases of the GUHA method

3.4.1 Mathematical notions of predicate calculus

The definitions of the chapter are from the source [4].

Theorem 3.4.1. (The language of predicates) *The language of predicates* $t = \langle t_1, \dots t_n \rangle$ *consists of*

- *predicates* (attributes) $P_1 \dots P_n$ with arity (GUHA uses arity equal 1) $t_1, \dots t_n$ are an infinite sequence of $x_1, x_2, \dots, x_m, \dots$ variables;

- logic *junctors* are $\top$ or $\bot$ (nullary), negation $\neg$, conjunctions $\wedge$, disjunction $\vee$, implications $\rightarrow$ and equivalence $\longleftrightarrow$

- non-empty set of quantifiers $q_0, q_1, \dots q_m, \dots$ The set could be infinite or finite.

- The classical universal and existential quantifiers $\forall$ (for every), $\exists$ (exist)

Example 3.4.1. (Formulas) *R is a binary predicate. The variable x is free and y is bound in the following formulas (f_1, f_2, f_3):*

$$\varphi_1 = (\forall y)R(x,y)$$

$$\varphi_2 = (\exists y)R(x,y)$$

$$\varphi_3 = (Wy)R(x,y)$$

Some closed formulas:

$$\psi_1 = \psi_1 \Rightarrow^x \varphi_1$$

$$\psi_2 = \psi_1 \sim^x \varphi_2$$

Summarize main logic facts which are true 3.4.

Table 3.4: Logic facts

	logic facts	name
(1)	$\varphi\&\psi \Leftrightarrow \psi\&\varphi$	commutativity
(2)	$\varphi \vee \psi \Leftrightarrow \psi \vee \varphi$	commutativity
(3)	$\varphi\&\varphi \Leftrightarrow \varphi$	idempotence
(4)	$\varphi \vee \varphi \Leftrightarrow \varphi$	idempotence
(5)	$\varphi\&(\varphi\&\chi) \Leftrightarrow (\varphi\&\varphi)\&\chi$	associativity
(6)	$\varphi \vee (\varphi \vee \chi) \Leftrightarrow (\varphi \vee \varphi) \vee \chi$	associativity
(7)	$\varphi\&\underline{1} \Leftrightarrow \varphi \vee \underline{0} \Leftrightarrow \varphi$	
(8)	$\varphi\&\underline{0} \Leftrightarrow \varphi \vee \underline{1} \Leftrightarrow \underline{1}$	
(9)	$(\varphi \rightarrow \psi) \Leftrightarrow (\neg\varphi \vee \psi) \Leftrightarrow \neg(\varphi\&\neg\psi)$	
(10)	$\varphi\&(\psi \vee \chi) \Leftrightarrow (\varphi\&\psi) \vee ((\varphi\&\chi)$	distributivity
(11)	$\varphi \vee ((\psi \vee \chi) \Leftrightarrow (\varphi \vee \psi)\&(\varphi \vee \chi)$	distributivity
(12)	$\neg\neg\varphi \Leftrightarrow \varphi$	
(13)	$\neg(\varphi\&\psi) \Leftrightarrow \neg\varphi \vee \neg\psi$	de Morgan law
(14)	$\neg(\varphi \vee \psi) \Leftrightarrow \neg\varphi\&\neg\psi$	de Morgan law
(15)	$\varphi\&\neg\varphi \Leftrightarrow \underline{0}$	complementation
(16)	$\varphi \vee \neg\varphi \Leftrightarrow \underline{1}$	complementation

3.4.2 Associational rules

The definitions of the chapter are from the source [7].

An expression $\varphi \approx \psi$ is an *association rule.* The formulas φ and ψ are Boolean attributes. The 4ft-quantifier is shown as $\approx$. The four-fold table of φ and ψ is used to denote a condition between the variables. Propositional logic (connections $\vee$, $\wedge$ and $\neg$) is using to create the variables of a four-fold table.

The 4ft-quantifier $\approx$ gives the rule for associating Boolean attributes φ and ψ of a four-fold table.

Table 3.3 shows a 4ft-table. The full four-fold table is shown in Table 3.5.

Table 3.5: The four-fold table.

M	ψ	$\neg\psi$	
φ	a	b	r
$\neg\varphi$	c	d	s
	k	l	m

where:

- a,b,c,d are the number of objects, satisfying the corresponding φ and ψ

- ψ and φ are built on unary predicates using Boolean connectives $\vee, \wedge, \neg$(conjunction, disjunction, negation)

- $r=a+b$; $s=c+d$; $k=a+c$ and $l=b+d$.

- $m=a+b+c+d$;

The formula $\varphi \approx \psi$ obtains the value `TRUE` if the function which is defined by $\approx$ obtains value *1* on the four-fold table 3.5, otherwise it is labelled as `FALSE`.

Remark 3.4.1. `TRUE` *is a label which satisfies* $v(\varphi \approx \psi)$*=1.*

The term *association* shows that a and d are big enough and c and d are not too big.

The name of the function which assigns to each four-fold table either *1* (TRUE) or *0* (FALSE) and satisfies some natural conditions is the association quantifier.

Example 3.4.2. (Survey among students) *Assume somebody wants to carry out a survey on a tranche of students. There are a number of students and the user wants to know the relation between their ages and their Grade Point Averages (GPA).*

Table 3.6: The table of students GPA and ages.

Student number	*4*≤GPA≤*5*	*3*≤GPA<*4*	GPA<*3*	Age<*25*	Age ≥ *25*
1	*T*	*F*	*F*	*T*	*F*
2	*T*	*F*	*F*	*F*	*T*
3	*F*	*T*	*F*	*T*	*F*
4	*F*	*T*	*F*	*T*	*F*
5	F	T	F	T	F

Table 3.6 provides an example of a truth student-GPA-ages table. There could be many observations which are based on the table. An interesting observation could be, for example, $\varphi = (GPA \geq 4) \vee (3 \leq GPA < 4)$, $\psi = Age < 25$

Table 3.7 shows the four-fold table resulting from this observation:

Table 3.7: The four-fold table M.

M	ψ	$\neg\psi$	
φ	*4*	*1*	*5*
$\neg\varphi$	*0*	*0*	*0*
	4	*1*	*5*

3.4.3 Classes of Associational rules

The definitions of the chapter are from the source [7].

There are five main classes of Associational rules. They are defined by classes of 4ft quantifiers. The classes of 4ft quantifiers are called *truth-preservation conditions.*

Definition 3.4.1. (Truth-preservation condition) *The truth-preservation condition is the Boolean condition $TPC_c\,(a,b,c,d,a',b',c',d')$, where $\langle a,b,c,d\rangle$ belongs to one four-fold contingency table and $\langle a',b',c',d'\rangle$ belongs to another table.*

$$v(\approx (a,b,c,d)) = 1 \; and \; TPC_c\,(a,b,c,d,a',b',c',d')$$

$$implies \; v(\approx (a',b',c',d')) = 1$$

For all valuations v:Formulas→{0,1}.

Definition 3.4.2. (Implication quantifier) *The implication quantifier is the 4ft-quantifier $\approx$, if $v(\approx (a,b,c,d)) = 1$ and $a^{'} \geq a$ and $b^{'} \leq b$ and v($\approx (a^{'},b^{'},c^{'},d^{'})) = 1$, for tables M and $M^{'}$, where $a^{'} \geq a$ and $b^{'} \leq b$ is the truth-preservation condition.*

The implication quantifier can be defined by the truth-preservation condition.

Definition 3.4.3. (TPC for implication quantifiers) *The truth-preservation condition for implication quatntifiers can be defined as*

$$TPC_{\Rightarrow} = a' \geq a \; and \; b' \leq b$$

Therefore the quantifier is an ***implication quantifier*** *if*

$$v(\approx (a,b,c,d)) = 1 \; and \; a' \geq a \; and \; b' \leq b$$

$$implies \; v(\approx (a',b',c',d')) = 1$$

For all valuations v:Formulas→0,1.

Table 3.8 shows the classes of associational rules which are defined by truth-preservation conditions (TPC)

Table 3.8: Classes of association rules.

Class		TPC	examples
Implicational	$TPC_{\Rightarrow}$	$a' \geq a$ *and* $b' \leq b$	$\Rightarrow_{p,Base}$ $\Rightarrow^{!}_{p,\alpha,Base}$
Double implicational	$TPC_{\Leftrightarrow}$	$a' \geq a$ *and* $b' \leq b$ *and* $c' \leq c$	$\Rightarrow^{*}_{0.9,0.1}$ $\Leftrightarrow_{p,Base}$
Σ-double implication	$TPC_{\Sigma,\Leftrightarrow}$	$a' \geq a$ *and* $b' + c' \leq b + c$	$\Leftrightarrow_{p,Base}$ $\Leftrightarrow^{!}_{p,\alpha,Base}$
Equivalency	$TPC_{\equiv}$	$a' \geq a$ *and* $b' \leq b$ *and* $c' \leq c$ *and* $d' \geq d$	$\sim_{\alpha,Base}$ $\equiv_{p,Base}$
Σ-equivalency	$TPC_{\Sigma,\equiv}$	$a' + d' \geq a + d$ *and* $b' + c' \leq b + c$	$\equiv_{p,Base}$ $\equiv^{!}_{p,\alpha,Base}$

3.4.4 Non-standard quantifiers

The definitions of the chapter are from the source [6].

Classical associations, which look like $A \rightarrow B$, where A and B are sets, are often not so interesting. The GUHA method offers non-classical associations. Association rules can be expressed as $\varphi \approx \psi$, where φ and ψ are Boolean attributes.

All quantifiers are expressed by a formula $\varphi \approx \psi$. The rule $\varphi \approx \psi$ is the associational rule.

All the following definitions are taken using the model M in Table 3.5.

Definition 3.4.4. (Founded implication) *A founded implication is the 4ft-quantifier* $\Rightarrow_{p,Base}$*, where* $Base \in \mathbb{N}$*,* p *is a rational number, and so* $0 < p \leq 1$*. A model* M *is given,* $v\left(\varphi(x) \Rightarrow_{p,Base} \psi(x)\right) = 1$*, i.e.* $v\left(\varphi(x) \Rightarrow_{p,Base} \psi(x)\right) =$ `TRUE`*, if and only if*

$$\frac{a}{a+b} \geq p \text{ } and \text{ } a \geq Base.$$

A founded implication quantifier shows at least *100p* percent of objects satisfy φ and also ψ. The variable *Base* means that the number of objects which satisfy both φ and ψ is at least *Base.*

Definition 3.4.5. (Lower critical implication) *The lower critical implication is a 4ft-quantifier* $v\left(\varphi(x) \Rightarrow^{!}_{p,Base} \psi(x)\right) = 1$*, i.e.* $\varphi(x) \Rightarrow^{!}_{p,Base} \psi(x)$ *is labelled* `TRUE`*, which is defined to hold in the given model M, where* $0 < p \leq 1$, $0 < \alpha < 0.5$ *and* $Base > 0$*:*

$$\sum_{i=a}^{a+b} \left(\begin{array}{c} a+b \\ i \end{array} \right) p^{i} \left(1-p\right)^{a+b-i} \leq \alpha \; and \; a \geq Base$$

The lower critical implication quantifier is based on a statistical test. H_0 and H_1 are statistical hypotheses. $H_0 : P\left(\psi|\varphi\right) \leq p$ against the alternative one $H_1 : P\left(\psi|\varphi\right) \succ p$, where $P\left(\psi|\varphi\right)$ is a conditional probability.

Definition 3.4.6. (Founded double implication) *A founded double implication is the 4ft-quantifier* $v\left(\varphi(x) \Leftrightarrow_{p,Base} \psi(x)\right) = 1$*, i.e.* $\varphi(x) \Leftrightarrow_{p,Base} \psi(x)$ *is labelled* `TRUE`*, which is defined to hold in the given model M, where* $0 < p \leq 1$ *and* $Base > 0$

$$\frac{a}{a+b+c} \geq p \; and \; a \geq Base$$

A founded double implication quantifier shows that at least *100p* percent of objects satisfying φ or ψ satisfy both φ and ψ. Variable Base means that the number of objects of a model which satisfy φ and also ψ is at least Base [6].

Definition 3.4.7. (Lower critical double implication) *A lower critical double implication is a 4ft-quantifier* $v\left(\varphi(x) \Rightarrow^{!}_{p,\alpha,Base} \psi(x)\right) = 1$*, i.e.* $\varphi(x) \Rightarrow^{!}_{p,\alpha,Base} \psi(x)$ *is labelled* `TRUE`*, which is defined to hold in the given model M, where* $0 < p \leq 1$, $0 < \alpha < 0.5$ *and* $Base > 0$*:*

$$\sum_{i=a}^{a+b+c} \left(\begin{array}{c} a+b+c \\ i \end{array} \right) p^i \left(1-p\right)^{a+b+c-i} \leq \alpha \; and \; a \geq Base$$

Definition 3.4.8. (Founded equivalence) *This is the 4ft-quantifier* $v(\varphi(x) \Leftrightarrow_{p,Base} \psi(x)) = 1$*, i.e.* $\varphi(x) \Leftrightarrow_{p,Base} \psi(x)$ *is labelled* `TRUE`*, which is defined to hold in the given model M, where* $0 < p \leq 1$ *and* $Base > 0$

$$\frac{a+d}{a+b+c+d} = \frac{a+d}{m} \geq p \; and \; a \geq Base$$

A founded equivalence implication quantifier shows that at least *100p* percent of objects φ and ψ have the same value. Variable Base means that the number of objects which satisfy both φ and ψ is at least Base.

Definition 3.4.9. (Lower critical equivalence) *lower critical equivalence is a 4ft-quantifier* $v\left(\varphi(x) \equiv^!_{p,\alpha,Base} \psi(x)\right) = 1$*, i.e.* $\varphi(x) \equiv^!_{p,\alpha,Base} \psi(x)$ *is labelled* `TRUE`*,which is defined to hold in the given model M, where* $0 < p \leq 1$*,* $0 < \alpha < 0.5$ *and* $Base > 0$*:*

$$\sum_{i=a+d}^{m} \left(\begin{array}{c} m \\ i \end{array} \right) p^i \left(1-p\right)^m \leq \alpha \; and \; a \geq Base$$

Definition 3.4.10. (The Fisher´s quantifier) *The Fisher´s quantifier is a 4ft-quantifier* $v\left(\varphi(x) \sim_{\alpha,Base} \psi(x)\right) = 1$*, i.e.* $\varphi(x) \sim_{\alpha,Base} \psi(x)$ *is labelled* `TRUE`*, which is defined to hold in the given model M, where* $0 < \alpha < 0.5$ *and* $Base > 0$*:*

$$\sum_{i=a}^{min(r,k)} \frac{\left(\begin{array}{c} k \\ i \end{array} \right) \left(\begin{array}{c} n-k \\ r-i \end{array} \right)}{\left(\begin{array}{c} r \\ n \end{array} \right)} \geq p \; and \; a \geq Base$$

Fisher´s quantifier is based on a statistical test, where one hypothesis of independence of φ and ψ is posited against the alternative, positive dependence one.

Definition 3.4.11. (Above average dependence) *above average dependence is the 4ft-quantifier* $v\left(\varphi(x) \sim^{!}_{+,Base} \psi(x)\right) = 1$, *i.e.* $\varphi(x) \sim^{!}_{+,Base} \psi(x)$ *is labelled* `TRUE`, *which is defined to hold in the given model M, where* $0 < p$ *and* $Base > 0$

$$\frac{a}{a+b} \geq (1+p)\frac{a+c}{a+b+c+d} \textit{ and } a \geq Base$$

Definition 3.4.12. (E-equivalence) *E-equivalence is the 4ft-quantifier* $v(\varphi(x) \sim^{E}_{\delta,Base} \psi(x)) =$ `TRUE`, *which is defined to hold in the given model M.*

$$\max\left(\frac{a}{a+b}, \frac{c}{d+c}\right) < \delta$$

3.4.5 Implicational class

The definitions of the chapter are from the sources [4] and [7].

This chapter shows how to demonstrate that the first class of Table 3.8 is implicational.

To show whether a class is Associational or Implicational the definitions have to be made.

Definition 3.4.13. *Model N is associationally better than model M if* $a_2 \geq a_1$,$d_2 \geq d_1$, $b_2 \leq b_1$ *and* $c_2 \leq c_1$ *[8].*

Definition 3.4.14. (Associational quantifier) *A binary quantifier* $\approx$ *is associational if* $v_M(\varphi(x) \approx \psi(x)) =$ `TRUE`, *N associationally better than M (Definition 3.4.13), than* $v_N(\varphi(x) \approx \psi(x)) =$ `TRUE` *for all formulae* $\varphi(x)$ *and* $\psi(x)$ *and all models M and N[8].*

[8].

There are M and N four-fold tables: Table 3.9.

M	ψ	$\neg\psi$	N	ψ	$\neg\psi$
φ	a_1	b_1	φ	a_2	b_2
$\neg\varphi$	c_1	d_1	$\neg\varphi$	c_2	d_2

Table 3.9: M and N models

Definition 3.4.15. *Model N is implicational better than model M if* $a_2 \geq a_1$ *and* $b_2 \leq b_1$ *[8].*

Definition 3.4.16. (Implicational quantifier) *A binary quantifier* $\approx$ *is implicational if* $v_M(\varphi(x) \approx \psi(x)) = \texttt{TRUE}$*, N implicational better than M (Definition 3.4.15), than* $v_N(\varphi(x) \approx \psi(x)) = \texttt{TRUE}$ *for all formulae* $\varphi(x)$ *and* $\psi(x)$ *and all models M and N[8].*

Lemma 3.4.2. (All associational quantifiers are Implicational) *Let M and N be models and M is* a-better than N, then M is i-better than N [4].

The M and N four-fold tables: Table 3.9.

The first class in Table 3.8 is Implicational. Take the founded implication quantifier 3.4.4 and check that the class is implicational.

Prove 3.4.1. *To show that the basic founded implicational is implicational, the founded implicational quantifier has the label* TRUE *if the equation* $\frac{a}{a+b} \geq p$ *and* $a \geq$ *Base, where* $p \in (0, 1]$ *and Base* > 0*, holds.*

Lemma 3.4.2 tells that an associational quantifier is implicational. Show firstly that the quantifier is associational and after show that the quantifier is implicational too.

A Founded implicational quantifier is associational if four conditions holds true:

1. $a \rightarrow a + 1$

An associational quantifier should evaluate the $a \rightarrow a+1$ equation for models M and N with the label `TRUE`*, where models M and N are represented in Table 3.10.*

M	ψ	$\neg\psi$	N	ψ	$\neg\psi$
φ	*a*	*b*	φ	*a+1*	*b*
$\neg\varphi$	*c*	*d*	$\neg\varphi$	*c*	*d*

Table 3.10: M and N models $(a \rightarrow a+1)$

The equation $p \leq \frac{a}{a+b} \leq \frac{a+1}{(a+1)+b}$ *should hold true.*

$$\frac{a}{a+b} \leq \frac{a+1}{(a+1)+b}$$

$$a(a+1+b) \leq (a+1)(a+b)$$

$$a^2+a+ab \leq a^2+ab+a+b$$

$$\cancel{a^2}+\cancel{a}+\cancel{ab} \leq \cancel{a^2}+\cancel{ab}+\cancel{a}+b$$

$$0 \leq b$$

Obviously, $0 \leq b$

2. $b \rightarrow b-1$ *An associational quantifier should evaluate the* $b \rightarrow b-1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in the table 3.11.*

M	ψ	$\neg\psi$	N	ψ	$\neg\psi$
φ	*a*	*b*	φ	*a*	*b-1*
$\neg\varphi$	*c*	*d*	$\neg\varphi$	*c*	*d*

Table 3.11: M and N models $(b \rightarrow b-1)$

The equation $p \leq \frac{a}{a+b} \leq \frac{a}{a+(b-1)}$ *should hold true.*

$$\frac{a}{a+b} \leq \frac{a}{a+(b-1)}$$

Obvious, $\frac{a}{a+b} \leq \frac{a}{a+b-1}$.

3. $d \rightarrow d+1$

An associational quantifier should evaluate the $d \rightarrow d+1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.12.*

M	ψ	$\neg\psi$	N	ψ	$\neg\psi$
φ	*a*	*b*	φ	*a*	*b*
$\neg\varphi$	*c*	*d*	$\neg\varphi$	*c*	*d+1*

Table 3.12: M and N models ($d \rightarrow d+1$)

The equation $p \leq \frac{a}{a+b} \leq \frac{a}{a+b}$ *should hold true.*

It is obvious that the equation $p \leq \frac{a}{a+b} \leq \frac{a}{a+b}$ *holds true.*

4. $c \rightarrow c-1$

An associational quantifier should evaluate the $c \rightarrow c-1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.13.*

M	ψ	$\neg\psi$	N	ψ	$\neg\psi$
φ	*a*	*b*	φ	*a*	*b*
$\neg\varphi$	*c*	*d*	$\neg\varphi$	*c-1*	*d*

Table 3.13: M and N models ($c \rightarrow c-1$)

The equation $p \leq \frac{a}{a+b} \leq \frac{a}{a+b}$ *should hold true.*

It is obvious that the equation $p \leq \frac{a}{a+b} \leq \frac{a}{a+b}$ *holds true.*

Therefore, a basic founded implicational quantifier is associational.

The basic founded implicational quantifier is implicational, because items 1 and 2 of the proof 3.4.1 show that the definition 3.4.16 holds true.

Remark 3.4.2. *An implicational quantifier depends only on* a *and* b*. Therefore it is possible to show only* $\Rightarrow^* (a, b)$*, instead of* $\Rightarrow^* (a, b, c, d)$

The remark shows that an implicational quantifier is 'weaker' than an associational quantifier. Thus, every associational quantifier is implicational, but not every implicational quantifier is associational.

3.4.6 Double implicational class

The definitions of the chapter are from the sources [4] and [7].

The second class in Table 3.8 is Double Implicational. Take the founded double implication quantifier 3.4.6 and check whether the class is associational or implicational.

Table 3.8 defines double implicational class as $TPC_{\Leftrightarrow} = a^{'} \geq a \; and \; b^{'} \leq b \; and \; c^{'} \leq c$.

Prove 3.4.2. *To show that double implicational quantifiers are associational or implicational. Double implicational quantifiers obtain the value* `TRUE` *if the equation* $\frac{a}{a+b+c} \geq p$ *and* $a \geq Base$*, where* $p \in (0, 1]$ *and* $Base > 0$*, holds true.*

A double implicational quantifier is associational if the four conditions are held to *be true.*

1. $a \rightarrow a+1$ *An associational quantifier should evaluate the* $a \rightarrow a+1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.10.*

   The equation $p \leq \frac{a}{a+b+c} \leq \frac{a+1}{(a+1)+b+c}$ *should hold true.*

$$\frac{a}{a+b+c} \leq \frac{a+1}{(a+1)+b+c}$$

$$a\left[(a+1)+b+c\right] \leq (a+1)(a+b+c)$$

$$a^2+a+ab+ac \leq a^2+ab+ac+a+b+c$$

$$\cancel{a^2}+\cancel{a}+\cancel{ab}+\cancel{ac} \leq \cancel{a^2}+\cancel{ab}+\cancel{ac}+\cancel{a}+b+c$$

$$0 \leq b+c$$

   Obviously, $0 \leq b+c$

2. $b \rightarrow b-1$

   An associational quantifier should evaluate the $b \rightarrow b-1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.11.*

   The equation $p \leq \frac{a}{a+b+c} \leq \frac{a}{a+(b-1)+c}$ *should hold true.*

$$\frac{a}{a+b+c} \leq \frac{a}{a+(b-1)+c}$$

Obviously, $\frac{a}{a+b+c} \leq \frac{a}{a+(b-1)+c}$.

3. $d \to d+1$

An associational quantifier should evaluate the $d \to d+1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.12.*

The equation $p \leq \frac{a}{a+b+c} \leq \frac{a}{a+b+c}$ *should hold true.*

Obviously $\frac{a}{a+b+c} \leq \frac{a}{a+b+c}$.

4. $c \to c-1$

An associational quantifier should evaluate the $c \to c-1$ *equation for models M and N with the label* `TRUE`*, where models M and N are represented in Table 3.13.*

$$\frac{a}{a+b+c} \leq \frac{a}{a+b+c-1}$$

Obviously, $\frac{a}{a+b+c} \leq \frac{a}{a+b+c-1}$.

Therefore, founded double implication quantifiers are associational.

Remark 3.4.3. *A double implicational class depends only on* a, b *and* c, *therefore* *it is possible to show only* $\Leftrightarrow^{*}(a,b,c)$, *instead of* $\Leftrightarrow^{*}(a,b,c,d)$

The remark shows that a Double implicational quantifier is 'weaker' than an associational quantifier. Thus, every associational quantifier is double implicational, but not every double implicational quantifier is associational.

3.4.7 Σ-double implication class

The definitions of the chapter are from the source [7].

Table 3.8 defines the Σ-double implication class as $TPC_{\Sigma,\Leftrightarrow} = a^{'} \geq a \; and \; b^{'} + c^{'} \leq b + c$.

The double implicational quantifier is associational, as was shown in 3.4.2. The class is also double implicational, which was shown in 3.4.6.

The Σ-double implication class is 'weaker' than the double implicational class. The Σ-double implication class is a sub-class of the double implicational class [7]. That means that there are quantifiers which will be double implicational but not Σ-double implicational. The source [7] provides one example of a quantifier $\Leftrightarrow^{*}_{0.9,\omega}$.

Example 3.4.3. (Non Σ-double implication quantifier) *Example of a non Σ-double implication [7].*

The quantifier $\Leftrightarrow^{*}_{0.9,\omega}$*, where* $0 < \omega$ *could be defined as:*

$$\Leftrightarrow^{*}_{0.9,\omega}(a,b,c,d) = \begin{cases} 1, & iff \; \frac{a}{a+b+\omega c} \geq 0.9 \; and \; a+b+c > 0 \\ 0, & otherwise. \end{cases}$$

The quantifier $\Leftrightarrow^*_{0.9,\omega}$ *is double implicational if* $\frac{a}{a+(b-1)+\omega c} \geq 0.9$, $\frac{a}{a+b+\omega(c-1)}$ *and* $\frac{a+1}{(a+1)+b+\omega c} \geq 0.9$. *The first two equations are obvious, thus the equation* $\frac{a+1}{(a+1)+b+\omega c} \geq \frac{a}{a+b+\omega c} \geq 0.9$ *should hold true.*

$$\frac{a+1}{(a+1)+b+\omega c} \geq \frac{a}{a+b+\omega c} \geq 0.9$$

$$\frac{a+1}{(a+1)+b+\omega c} \geq \frac{a}{a+b+\omega c}$$

$$(a+1)(a+b+\omega c) \geq a(a+1+b+\omega c)$$

$$a(a+b+\omega c)+a+b+\omega c \geq a^2+a+ab+a\omega c$$

$$a^2+ab+a\omega c+a+b+\omega c \geq a^2+a+ab+a\omega c$$

$$\cancel{a^2}+\cancel{ab}+\cancel{a\omega c}+\cancel{a}+b+\omega c \geq \cancel{a^2}+\cancel{a}+\cancel{ab}+\cancel{a\omega c}$$

$$b+\omega c \geq 0$$

Obviously, $b+\omega c \geq 0$.

Therefore, the quantifier is double implicational.

Assume that there is a model with 4ft tables:

M	ψ	$\neg\psi$
φ	a=90	b=9
$\neg\varphi$	c=2	d=0

Table 3.14: M model ($\Leftrightarrow^*_{0.9,\omega}$)

There is a quantifier $\Leftrightarrow^*_{0.9,\omega}$ *where* $0 < \omega < 0.5$ *and* $b+\omega c < 10$.

First we show that the quantifier is double implicational:

$$\frac{a}{a+b+\omega c} = \frac{90}{90+b+\omega c} > \frac{90}{90+10} = 0.9$$

Therefore, the $\Leftrightarrow^{*}_{0.9,\omega}$ *quantifier is double implicational because* $\Leftrightarrow^{*}_{0.9,\omega} (90, 9, 2, 0) = 1$.

Then we show that the quantifier is not in the Σ*-double implication class.*

Assume that the quantifier is Σ*-double implicational, and* $\Leftrightarrow^{*}_{0.9,\omega} (90, 9+2, 0, 0) = 1$.

$$\frac{90}{90+9+2+\omega 0} > \frac{90}{90+11} < 0.9$$

Therefore, there is a contradiction, because $\Leftrightarrow^{*}_{0.9,\omega} (90, 9+2, 0, 0) = 0$.

The quantifier $\Leftrightarrow^{*}_{0.9,\omega}$ *is not an* Σ*-double implication quantifier.*

3.4.8 Equivalency class

The definitions of the chapter are from the source [7].

Table 3.8 defines the equivalency class as $TPC_{\equiv} = a^{'} \geq a \ and \ b^{'} \leq b \ and \ c_{\prime} \leq c \ and \ d_{\prime} \geq d$.

The next class is basic equivalence quantifiers. Let us show that the class is associational.

Prove 3.4.3. *We will prove that basic equivalence quantifiers are associational. Basic equivalence quantifiers 3.4.8 are labelled* TRUE *if the equation* $\frac{a+d}{a+b+c+d} = \frac{a+d}{m} \geq p$, *where* $p \in (0, 1]$ *and* $a \geq Base$, *holds true.*

A basic equivalence quantifier is associational if the four conditions hold true.

1. $a \to a + 1$ *An associational quantifier should evaluate the* $a \to a + 1$ *equation for models M and N with the label* TRUE, *where models M and N are represented in Table 3.10.*

   The equation $p \leq \frac{a+d}{m} \leq \frac{(a+1)+d}{m+1}$ *should hold true.*

   $$\frac{a+d}{m} \leq \frac{(a+1)+d}{m+1}$$

   $$(a+d)(m+1) \leq m(a+1+d)$$

   $$am + a + dm + d \leq am + m + dm$$

   $$\cancel{am +} a + \cancel{dm} + d \leq \cancel{am +} m + \cancel{dm}$$

   $$a + d \leq m$$

   where $m = a + b + c + d$.

   Obviously, $m \geq a + d$

2. $b \to b - 1$

An associational quantifier should evaluate the $b \rightarrow b-1$ equation for models M and N with the label `TRUE`, *where models M and N are represented in Table 3.11.*

The equation $p \leq \frac{a+d}{m} \leq \frac{a+d}{m-1}$ should hold true.

$$\frac{a+d}{m} \leq \frac{a+d}{m-1}$$

Obviously, $\frac{a+d}{m} \leq \frac{a+d}{m-1}$.

3. $d \rightarrow d+1$

An associational quantifier should evaluate the $d \rightarrow d+1$ equation for models M and N with the label `TRUE`, *where models M and N are represented in Table 3.12.*

Obviously, the same as the first item.

4. $c \rightarrow c-1$

An associational quantifier should evaluate the $c \rightarrow c-1$ equation for models M and N with the label `TRUE`, *where models M and N are represented in Table 3.13.*

$$p \leq \frac{a+d}{m} \leq \frac{a+d}{m-1}$$

Obviously, the same holds true as for item 2.

Therefore, basic equivalence quantifiers are associational.

3.4.9 Σ-equivalency class

The definitions of the chapter are from the source [7].

The table table 3.8 defines the Σ-*equivalency* class as $TPC_{\Sigma,\equiv} = a^{'} + d^{'} \geq a + d \; and \; b^{'} + c\prime \leq b + c.$

The Σ-equivalency class is 'weaker' than the equivalency class. The Σ-equivalency class is a sub-class of the equivalency class [7].

3.4.10 Association rules with support and confidence

The definitions of the chapter are from the sources [7] and [10].

The 'classical' rules, defined in Chapter 4, are not associational by the truth-preservation condition.

Definition 3.4.17. ('Classical' association rule) *The A-quantifier is the 4ft-quantifier* $v\left(\varphi(x) \sim^{A}_{\gamma,\sigma} \psi(x)\right) = \texttt{TRUE}$*, which is defined to be true in the given model M, where* $0 < \gamma \leq 1$ *and* $0 \leq \sigma \leq 1$*. The minconf is* γ *and the minsup is* σ*.*

$$\sim^{A}_{\gamma,\sigma}(a,b,c,d) = \begin{cases} 1, & \textit{iff } \frac{a}{a+b} \geq \gamma \wedge \frac{a}{a+b+c+d} \geq \sigma \\ 0, & \textit{otherwise.} \end{cases}$$

The confidence *is* $\frac{a}{a+b}$.

The support *is* $\frac{a}{a+b+c+d}$.

Remark 3.4.4. *The user should remember the difference between the confidence and support of an association rule and the minimum support (σ) and confidence (γ) thresholds. The confidence and support are variables which show the current condition between the variables on a 4ft-table by applying the formulas from 3.4.17. The minimum support (σ) and confidence (γ) thresholds are variables which a user should determine before applying the association rule.*

Prove 3.4.4. (Show that the A-quantifier is not associational) *The A-quantifier is not associational. The definition of the A-quantifier was made in 3.4.17.*

The table 3.15 shows when an A-quantifier is implicational and non-associational.

Table 3.15: An A-quantifier is implicational and non-associational.

	γ	σ	$\sim^{A}_{\gamma,\sigma}$
1	$0 < \gamma < 1$	$\sigma = 0$	implicational
2	$0 < \gamma < 1$	$0 < \sigma \leq 1$	non associational
3	$\gamma = 1$	$\sigma = 0$	implicational
4	$\gamma = 1$	$0 < \sigma \leq 1$	non associational

1. *Assume that $0 < \gamma < 1$ and $\sigma = 0$, where γ and σ are the minumum confidence and support thresholds. The assumption $\sigma = 0$ shows that $\frac{a}{a+b+c+d} \geq \sigma$ is true with any of a, b, c or d. The $\frac{a}{a+b} \geq \gamma$ condition is obvious and has already been proved in Chapter 3.4.5. The assumption is true.*

2. *Assume that $0 < \gamma < 1$ and $0 < \sigma \leq 1$, where γ and σ are the minumum confidence and support thresholds. The source [10] provides an explanation of the assumption. Assume there are two 4-ft tables: $M = \langle a, b, c, d\rangle$ and $N = \langle a^{'}, b^{'}, c^{'}, d^{'}\rangle$. If the quantifier is associational, the equations should hold true: $\sim^{A}_{\gamma,\sigma} (a, b, c, d) = 1$ and $\sim^{A}_{\gamma,\sigma} (a^{'}, b^{'}, c^{'}, d^{'}) = 0$.*

   Let M = (a,0,0,0), then $\frac{a}{a+b} = 1 \geq \gamma$ and $\frac{a}{a+b+c+d} = 1 \geq \sigma$, therefore $\sim^{A}_{\gamma,\sigma}$ $(a, b, c, d) = 1$.

Take $d^{'} > a\frac{1-\sigma}{\sigma}$ *and* $b^{'} = c^{'} = 0$*, therefore:*

$$\frac{a^{'}}{a^{'} + b^{'} + c^{'} + d^{'}} = \frac{a}{a + d^{'}} < \frac{a}{a + a\frac{1-\sigma}{\sigma}} = \frac{1}{1 + \frac{1-\sigma}{\sigma}} = \frac{1}{\frac{1}{\sigma}} = \sigma$$

Therefore, there is a contradiction, because in this case $\sim^{A}_{\gamma,\sigma} (a^{'}, b^{'}, c^{'}, d^{'}) = 0$*.*

Definition 3.4.14 tells that it is necessary to look through all cases $a_2 \geq a_1, d_2 \geq d_1, b_2 \leq b_1$ *and* $c_2 \leq c_1$*, but for simplicity let's show only the case* $d_2 \geq d_1$*. Thus, look through the case* $d \to d'$*, where* $d' > d$*.*

The counter example M = (a,0,0,d), where $d = 0$*, and N = (a',0,0,d'), where* $a' = a$ *and* $d' > a\frac{1-\sigma}{\sigma}$*. Remember* $0 < \sigma \leq 1$*.*

The equation $\frac{a}{a+b+c+d} \leq \frac{a^{'}}{a^{'}+b^{'}+c^{'}+d^{'}}$ *should hold true.*

$$\frac{a}{a + b + c + d} = \frac{a}{a + 0 + 0 + 0} = 1 \geq \sigma$$

$$\frac{a^{'}}{a^{'} + b^{'} + c^{'} + d^{'}} = \frac{a}{a + d^{'}} < \sigma$$

Obviously, $\frac{a}{a+b+c+d} > \frac{a^{'}}{a^{'}+b^{'}+c^{'}+d^{'}}$*. Therefore, there is a contradiction.*

The counter example shows the A-quantifier, with the parameters minsup ($0 < \sigma \leq 1$*) and minconf (*$0 < \gamma < 1$*), is not associational.*

3. *Assume that $\gamma = 1$ and $\sigma = 0$, where γ and σ are the minumum confidence and support thresholds. Obvious, as in item 1. The assumption $\sigma = 0$ shows that $\frac{a}{a+b+c+d} \geq \sigma$ holds true with any of a, b, c or d. The $\frac{a}{a+b} \geq \gamma = 1$ condition holds true only when $b = 0$. The $\frac{a^{'}}{a^{'}+b^{'}} \geq \gamma = 1$ condition is labelled* `TRUE` *only when $b^{'} = 0$, and the equation $b^{'} \leq b$ holds true when $b = b^{'} = 0$. The assumption is true.*

4. *Assume that $\gamma = 1$ and $0 < \sigma \leq 1$, where γ and σ are the minumum confidence and support thresholds. This assumption can be proved in the same way as for items 2 and 3.*

Therefore the A-quantifier is not associational.

Remark 3.4.5. *Assume a rule with the variable $a = 0$. In this case the* support *and* confidence *for the rule are 0, as is shown by the formulas in 3.4.17. Remember the threshold requirements* Confidence threshold $\in (0, 1]$ *and* Support threshold $\in [0, 1]$. *Thus, although the rule may hold true for the support requirement (*Support threshold $=$ 0*), the rule does not hold for the* Confidence threshold, *because the* Confidence threshold *should be more than 0.*

4. THEORETICAL BASIS OF THE WEKA METHOD

4.1 History

The definitions of the chapter are from the source [9].

The New Zealand government has funded the WEKA project since late 1992. The goal of the project is *'... developing techniques of machine learning and investigating their application in key areas of the New Zealand economy... '*. The first internal version of Weka was published in 1994. The first public version was released in October, 1996. Finally, the 2.2 version was released in July, 1997, in which the first eight learning algorithms were introduced. The first algorithms were implemented by the authors of the algorithms.

The Pentano Corporation became the main sponsor in 2006 and the Weka software was adapted to a data mining and predictive analytic component form. The latest version of Weka software is 3.6, which was released in December, 2008.

The version 3.6 will be used in the thesis.

4.2 Association rules

The definitions of the chapter are from the source [1].

The pattern which appears in the given dataset most frequently is called *the frequent pattern.* A sequence of goods which appears frequently in any given database is the (frequent) *sequential pattern.* For example, if the same two books frequently appear in shopping database transactions, they are sequential patterns. **Market basket analysis** is a typical example of frequent itemset mining. Below is an example of how market basket analysis is used.

Example 4.2.1. (Market basket analysis.) *Suppose there is a bookstore. A manager wants to know 'Which groups or sets of items are customers likely to purchase on a given trip to the store?'. The manager can use market basket analysis to answer the question. The market basket analysis uses transactions from the store's database. The manager can use the results to plan marketing or advertising strategies.*

He can prepare recomendations for a seller about what to recommend a buyer to buy. For example, if a buyer wants to buy book 1, the seller can recommend book 2, which other buyers have bought together with book 1. Alternatively, the manager can put book 1 and book 2 in different places in the store, compelling the buyer to go through the store and look for and buy more books. Another strategy could be that the manager gives a discount on book 2 if the customer buys book 2 with book 1, or vice versa.

Frequent patterns can be represented by Boolean vectors. The items which are frequently *associated* or purchased together can be analysed by the Boolean vectors. The items can be represented by Association Rule:

$$\textit{book}\ 1 \Rightarrow \textit{book}\ 2\,[\text{support} = 5\%, \text{confidence}70\%]$$

The measures of interest in the pattern are **support** and **confidence**.

$$X\&Y \Rightarrow Z\,(\text{support} = s\%, \text{confidence}\,c\%) \tag{4.1}$$

Definition 4.2.1. (support) *The **support** of the (4.1) association rule shows the probablity of transactions which contain* $\{X \wedge Y \wedge Z\}$.

$$\text{support}\,(X\&Y \Rightarrow Z) = P\,(X \wedge Y \wedge Z) \tag{4.2}$$

Definition 4.2.2. (confidence) *The **confidence** of the (4.1) association rule shows conditional probability* $P\,(Z|X \wedge Y)$. *The confidence shows the probability of transactions which contains* $\{X \wedge Y\}$ *and also contain* Z.

$$\text{confidence}\,(X\&Y \Rightarrow Z) = P\,(Z|X \wedge Y) \tag{4.3}$$

Assume, that an association rule is interesting if the minimum support and confidence requirements in the rule hold (minimum support or confidence threshold). A **strong** rule is a rule which satisfies the minimum support and confidence thresholds. Conventionally, the support and confidence levels are shown between 0% and 100%.
Example 4.2.2. (Support and Confidence.) *Suppose there is a book shop. The book shop database has several transactions, some of which are presented in Table 4.1.*

Count support and confidence for some examples:

Table 4.1: The book shop database.

Transaction ID	Sold Books
2000	book 1, book 2, book 3
2500	book 1
3000	book 1, book 3
3300	book 2, book 3
4000	book 4, book 1, book 2

$$book1 \Rightarrow book2 \, (support = 40\%, confidence \, 50\%)$$

$$book2 \Rightarrow book1 \, (support = 40\%, confidence \, 66.6\%)$$

$$book1\&book2 \Rightarrow book3 \, (support = 20\%, confidence \, 50\%)$$

The itemset is the set of items, for example $\{book_1, book_2\}$. The support count (frequency or count) is the number of trunsactions which contain the itemset. The confidence can be expressed as (4.5).

$$\text{confidence} \left(X\&Y \Rightarrow Z\right) = P\left(Z|X \wedge Y\right) = \frac{\text{support}\left(\left(X \wedge Y\right) \vee Z\right)}{\text{support}\left(X \wedge Y\right)} \tag{4.4}$$

$$\text{confidence} \left(X\&Y \Rightarrow Z\right) = \frac{\text{support\_count}\left(\left(X \wedge Y\right) \vee Z\right)}{\text{support\_count}\left(X \wedge Y\right)} \tag{4.5}$$

Equation (4.5) shows the possibility of applying the confidence rule $(X\&Y \Rightarrow Z)$ for support counts $((X \wedge Y) \vee Z)$ and $(X \wedge Y)$.

There are two steps in mining an association rule:

1. Find all frequent patterns

2. Generate strong association rules from the item set

4.3 Categories of frequent patterns mining

The definitions of the chapter are from the source [1].

There are plenty of different types of frequent patterns or association rules. They can be classified with some basic criteria.

- **Completeness** of patterns. There are different types of frequent itemsets, such as closed itemset, complete itemset or maximal-frequent itemset. There are different applications with different requirements for the completeness of the mined patterns. The requirements can affect the evaluation and optimization methods.

- **Level of abstraction**. For example, there is a book shop where there are different levels of abstraction. The level *books* is 'higher' than the *Books 1* and *Books 2* levels. Assume that there is the 'book 1' book in the *'Books 1'* level and the 'book 2' in the *'Books 2'*. Take the measure of interest as (4.1).

$$buys(\text{Customer},'\textit{Books }1') \Rightarrow buys(\text{Customer},'\ \textit{book }2') \tag{4.6}$$

$$buys(\text{Customer},'\textit{book }1') \Rightarrow buys(\text{Customer},'\ \textit{book }2') \tag{4.7}$$

The level of abstraction of the formula (4.6) is different than for formula (4.7), because the level 'Books 1' in (4.6) is 'higher' than 'book 1' in (4.7) (remember: *books* > *Books 1* > 'book 1').

- **Number of data dimensions**. The formula (4.8) has only one data dimension while (4.9) has two dimensions.

$$buys(\text{Customer}, X) \Rightarrow buys(\text{Customer}, Z) \tag{4.8}$$

$$buys(\text{Customer}, X) \wedge isStudent(X) \Rightarrow buys(\text{Customer}, Z) \tag{4.9}$$

- **Types of values**. If the presence or absence of items are included in an association rule the rule is a *Boolean association rule.* For example, equation (4.1) is the *Boolean association rule* from a market basket analysis.

- **Kinds of rules**. There are many different types or kinds of rules, such as association or correlation. Association rules are most popular when finding frequent patterns.

- **Kinds of patterns**. There are different patterns, which depend on the type of database. The frequent itemset is mined from relational or transactional datasets. Sequential patterns can be mined from a sequence dataset, where records are kept of the order of events.

4.4 Apriori algorithm

The definitions of the chapter are from the source [1].

The Apriori algorithm was developed by R.Agrawal and R. Srikant in 1994. The algorithm is used for finding frequent patterns for Boolean association rules. The algorithm is based on a *level-wise* search. A level-wise search is a search where each preceding set is used to discover the following one. The first step of the algorithm is to count all the items in the database and collect the results into a set, L1, which satisfies the minimum support. The L1 set is used to calculate the L2 set. The L2 set is a frequent 2-items set. The operation should continue until there are no more frequent itemsets. The last frequent itemset is obtained by scanning the whole database.

Definition 4.4.1. (Apriori property) *The **Apriori property** is 'all non-empty subsets of a frequent itemset must also be frequent' [1]. The property is used to improve the efficiency of the algorithm.*

The Apriori property says that if a pattern at a level i of an itemset, where $0<i<k$, and k is the last frequent itemset, is not frequent, then the pattern for the next $i+1$ level will not be frequent either.

Example 4.4.1. (Apriori algorithm) *Assume there is a bookstore. The bookstore database has several transactions. Take some of them in 4.2 (the table is just an extended version of table 4.1).*

Table 4.2: The bookstore database.

Transaction ID	Sold Books
2000	book 1, book 2, book 3
2500	book 1
3000	book 1, book 3
3300	book 2, book 3
4000	book 1, book 2
4000	book 2
4000	book 4, book 2
4000	book 3, book 1, book 2
4000	book 3, book 2

There are different steps, as was mentioned earlier, to generate frequent itemsets. Minimum support should be declared before the explanation. The minimum support *equals 2, which is the absolute support value.*

$$\textit{Minimum support} = 2 \tag{4.10}$$

1. *The first step is to create a 1-item set, to calculate the support, and to compare this with the minimum support (4.10).*

   Firstly, scan the table 4.2 to get all the 1-item itemsets.

   Table 4.3: The L1 itemset.

itemset	support
$\{book\ 1\}$	5
$\{book\ 2\}$	7
$\{book\ 3\}$	5
$\{book\ 4\}$	1

   Check the support and compare with the minimum support (4.10). Then put the result in the table, 4.4.

   Table 4.4: The resulting L1 itemset.

itemset	support
$\{book\ 1\}$	5
$\{book\ 2\}$	7
$\{book\ 3\}$	5

2. *The second step is generating an L2 set, counting the support and comparing the results with the minimum support (4.10).*

   Firstly, generate the 2-item set, based on Table 4.4, and calculate the support.

   Check the support and compare with the minimum support (4.10). It is obvious, that the resulting table will be the same as in Table 4.5

Table 4.5: The L2 itemset.

itemset	support
$\{book\ 1, book\ 2\}$	3
$\{book\ 1, book\ 3\}$	3
$\{book\ 2, book\ 3\}$	4

3. *The third step is to create a 1-item set, calculate the support, and compare with the minimum support (4.10).*

   Firstly, generate the 3-item set, based on Table 4.5 and count the support.

   Table 4.6: The L3 itemset.

itemset	support
$\{book\ 1, book\ 2, book\ 3\}$	2

   Check the support and compare with the minimum support 4.10. The result is the same as in Table 4.6.

The next step is to generate strong association rules from the frequent itemsets L2 and L3. A strong association rule is a rule which satisfies both the minimum support and the minimum confidence. The confidence rule was shown in equation (4.3). Below is the extended equation:

$$\text{confidence}\,(X\&Y \Rightarrow Z) = P\,(Z|X \wedge Y) = \frac{\text{support\_count}\,((X \wedge Y) \vee Z)}{\text{support\_count}(X \wedge Y)} \tag{4.11}$$

Association rules can be generated using these steps:

- Generate all subsets S, which are non-empty, of every frequent itemset L 4.4.1

- Calculate minimum confidence threshold for every subset s of S, the rule $s \Rightarrow (L - s)$. The minimum confidence threshold holds true if $\frac{\text{support\_count}(L)}{\text{support\_count}(s)} \geq minconf$.

All rules sutisfy the minimum support condition because they were generated from frequent itemsets.

Example 4.4.2. (Generating association rules.) *Assume there is a book shop. The book shop database has several transactions. Take some of them from Table 4.2 (the example follows the example 4.4.1).*

The example 4.4.1 provides only one 3-item pattern $\{book\ 1, book\ 2, book\ 3\}$.

The first step is to generate all subsets S, which are non-empty, from the 3-item pattern. They are $\{book1\}$, $\{book\ 2\}$, $\{book\ 3\}$, $\{book\ 1, book\ 2\}$, $\{book\ 1, book\ 3\}$ *and* $\{book\ 2, book\ 3\}$. *Generate from the subsets' association rules with confidence. The confidence is calculated using the formula (4.11).*

Table 4.7: Association rules.

association rule	confidence
$book\ 2 \wedge book\ 3 \Rightarrow book\ 1$	$2/4 = 0.50$
$book\ 1 \wedge book\ 3 \Rightarrow book\ 2$	$2/3 = 0.67$
$book\ 1 \wedge book\ 2 \Rightarrow book\ 3$	$2/3 = 0.67$
$book\ 1 \Rightarrow book\ 2 \wedge book\ 3$	$2/5 = 0.40$
$book\ 2 \Rightarrow book\ 1 \wedge book\ 3$	$2/7 = 0.29$
$book\ 3 \Rightarrow book\ 1 \wedge book\ 2$	$2/5 = 0.40$

The minimum confidence threshold is, for example, 0.60. In this case, only the second and the third association rules are strong.

4.5 Predictive apriori

The definitions of the chapter are from the source [12].

Chapter 4.4 shows the association rule with support and confidence. Higher confidence implies greater support. The problem is how to improve the efficiency and accuracy of the method. Accuracy depends on the values of support and confidence. A Bayesian framework can help in finding out the values for the expected accuracy. Predictive apriori is a fast method, which helps to find the n best rules with accuracy.

Definition 4.5.1. (Predictive accuracy) *Assume that there is a database D, where a static process P has generated a record r. An association rule is* $[x \Rightarrow y]$*. The conditional probability* $P(y \subseteq r|x \subseteq r)$ *is the predictive accuracy* $c([x \Rightarrow y]) = Pr[r \textit{ satisfies } y|r \textit{ satisfies } x]$*.*

The main problem with the method is finding n rules to optimise the expected predictive accuracy. The algorithm should return only a fixed number of the best association rules, but should not return all the rules which satisfy the given threshhold.

Bayesian frequency correction is an approach where the formula 4.12 takes confidence and returns a lower predictive accuracy.

$$E\left(c\left([x \Rightarrow y]\right)|\hat{c}\left([x \Rightarrow y]\right), s(x)\right) = \frac{\int cB\left[c, s(x)\right]\left(\hat{c}\left([x \Rightarrow y]\right)\right)\pi(c)dc}{\int B\left[c, s(x)\right]\left(\hat{c}\left([x \Rightarrow y]\right)\right)\pi(c)dc} \tag{4.12}$$

The equation (4.12) calculates the expected confidence of a rule, where $\hat{c}$ is the confidence. The algorithm selects the rules with greatest support and confidence instead of drawing them randomly.

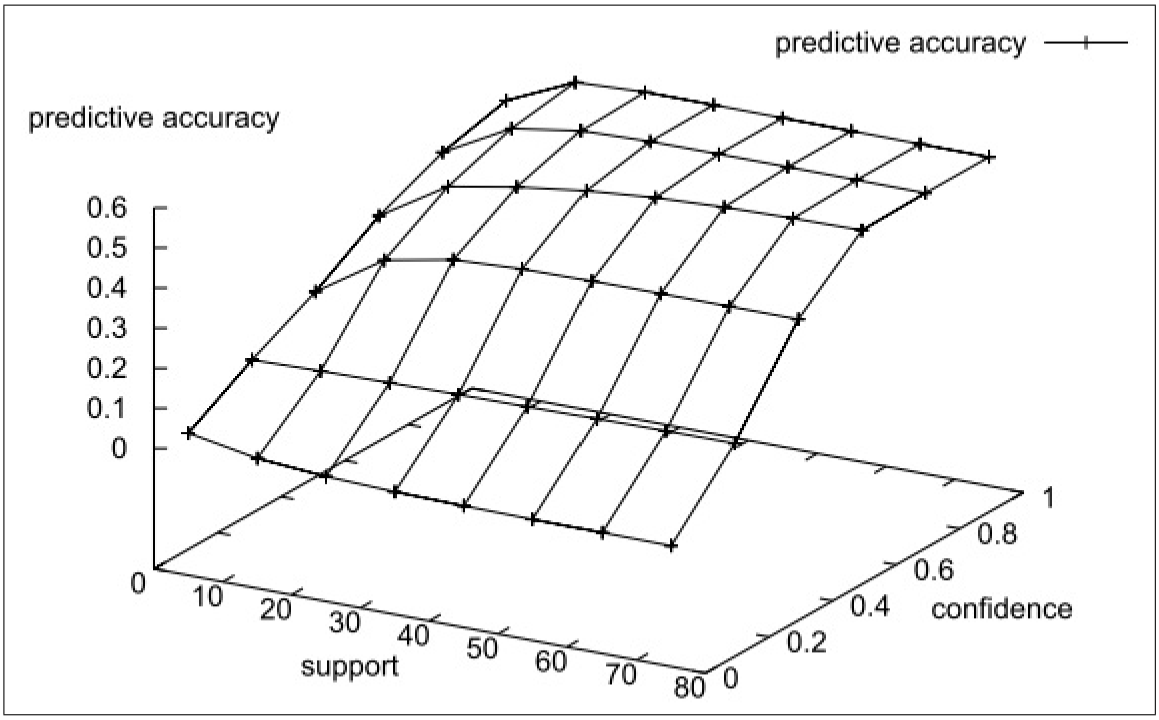

Figure 4.1: Example of distribution of predictive accuracy $c\left([x \Rightarrow y]\right)$ of the rule $[x \Rightarrow y]$ [12].

Figure 4.1 shows the predictive accuracy $c\left([x \Rightarrow y]\right)$, where support is $s(x)$ and confidence $\hat{c}\left([x \Rightarrow y]\right)$, for the rule $[x \Rightarrow y]$.

There are two steps in finding association rules with an Apriori algorithm, wich were introduced in Chapter 4.4. The support for all items is calculated in the first step. The confidence is calculated during the second step after combining all the items. The predictive algorithm should have some variations with these two steps, because the algorithm does not have fixed confidence and support thresholds.

The algorithm starts with estimating the prior $\pi(c)$. The list shows the following steps of the algorithm.

- *Frequent item sets* should be generated.

- *The hypothesis space* must be reduced by applying the minimum support threshold.

- *Association rules* should be generated.

- *Redundant association rules* should be removed.

The goal of the algorithm is to find n best rules.

One method of estimating $\pi(c)$ is to draw several hypotheses at random using uniform distribution, check the confidence and construct a histogram. Unfortunately, there are more long rules than short ones and it would be difficult to discover the short ones using this method. The solution is to put a loop through the rules with a given length and draw out only a fixed number of rules.

The uniform distrubution favours long rules, but the method should draw equal amounts of rules for each size of rule. The number of itemsets is $\begin{pmatrix} k \\ i \end{pmatrix}$, where k is database of items of size i. The number of association rules is $2^i - 1$, where the right part of a set should be non-empty.

$$P[i \text{ items}] = \frac{\begin{pmatrix} k \\ i \end{pmatrix} (2^i - 1)}{\sum_{j=1}^{k} \begin{pmatrix} k \\ i \end{pmatrix} (2^i - 1)} \tag{4.13}$$

The formula (4.13) shows the probability of i items appearing in a rule, which has been drawn by uniform distribution.

Cheking the priority of all association rules is the last step in the method. Each priority is weighted for a rule with length i by counting the probability of rule (4.13).

The predictive apriori method uses the enumeration of items as in the apriori algorithm, but with *dynamically increasing minimum support threshold* τ.

The article [12] provides an algorithm 4.5.1, which generates all rules from a body x, for every $x \in X_i$.

Algorithm 4.5.1 (RuleGen Algorithm)**.**

```
Algorithm RuleGen(x) (find the best rules with the body x, efficiently)
10. Let $\gamma$ be the smallest number such that $E\left(c|\gamma/s(x), s(x)\right) >$
    $E\left(c(best[n])|\hat{c}(best[n]), s(best[n])\right)$.
11. For $j = 1..k - |z|$ (number of items not in x)
    (a) If j=1 Then Let $Y_1 = (a_1 \dots a_k)\backslash x$.
    (b) Else Let $Y_j = \left\{y \cup y^{'}|y, y^{'} \in Y_{j-1}|y \cup y^{'}| = j\right\}$
    (c) For all $y \in Y_j$ Do
        i. Measure the support $s(x \cup y)$. If $s(x \cup y) \leq \gamma$,
           Then, eliminate $y$ from $Y_j$ and Continue the For loop with the next $y$.
        ii. Calculate predictive accuracy $E(c([x \Rightarrow y])|s(x \cup y)/x(x), s(x)))$
        iii. If the predictive accuracy is among the n best found so far (recorded in
             best). Then update best, remove rules in best that are subsumed by other,
             at least equally accurate rules, and Increase $\gamma$ to be the smallest number
             that allows $E(c|\gamma/s(x), s(x)) \geq E(c(best[n])|\hat{c}(best[n]), s(best[n]))$
11. If any subsumed rule has been erased in 11(c)iii. Then recur from step 10.
```

Redundant rules can appear when evaluating an algorithm. Assume there is a rule

$[a \Rightarrow c, d]$. If the rule satisfies a database, the other rules should be satisfied too. For example, $[a, c \Rightarrow c, d]$, $[a \Rightarrow c]$, $[a \Rightarrow d]$ among others. Redundant rules can appear when evaluating the algorithm, and they should be removed.

Theorem 4.5.1. (Correctness of the Predictive Apriori method) *The most accurate rules are returned with the predictive apriori algorithm. An array of best* $[1 \ldots n]$ *association rules* $[x_i \Rightarrow y_i]$ *is returned by the algorithm, where* $x_i \cap y_i = \emptyset$. *All the rules which are not best are* $E\left(c\left([x \Rightarrow y]\right) | \hat{c}\left([x \Rightarrow y]\right), s(x)\right) \leq E\left(c\left(best[i]\right) | \hat{c}\left(best[i]\right)\right)$, *where* $1 \leq i \leq n$. *The best rules are expressed as* $E\left(c\left([x' \Rightarrow y']\right) | \hat{c}\left([x \Rightarrow y]\right), x(s)\right)$, *and* $[x' \Rightarrow y'] \models [x \Rightarrow y]$ *[12].*

5. THEORETICAL HYPOTHESES OF GUHA AND WEKA SOFTWARES

Chapters 3 and 4 introduced some association rules. The rules are different, so the goal of this chapter is to find some theoretical hypotheses which can later be proved or disproved.

Chapter 4 showed the theory behind Apriori 4.4 and Predictive apriori 4.5 association rules. Five different classes were introduced, and the methods.

Chapter 3 provided an explanation of different classes of association rules in GUHA 3.4.3. The Implicational class was introduced in 3.4.5, where the *founded implication* quantifier 3.4.4 was used. The double implicational class was introduced in 3.4.6, where the *founded double implication* quantifier 3.4.6 was used. The *equivalency* class was introduced in subchapter 3.4.8, where the *founded equivalence* 3.4.8 quantifier was used.

There are some methods based on comparing associational rules and associational quantifiers.

- Find the similarity between the Weka apriori association rule and the GUHA founded implication, founded double implication, and founded equivalence associational quantifiers.

- Find the similarity between the Weka predictive apriori association rule and the GUHA founded implication, founded double implication, and founded equivalence associational quantifiers.

- Find the similarity between the Weka generalized sequential patterns association rule and the GUHA founded implication, founded double implication, and founded equivalence associational quantifiers.

The idea is to launch methods with different parameters and compare the results. To avoid repetition the three Weka methods should first be used with different specific parameters. Secondly, the GUHA methods should be used with similar parameters (where it is possible) and finally the results should be compared.

The methods show that the hypotheses can be constructed easily.

1. The apriori association rule can provide the same results as the founded implication association rule.

2. The apriori association rule can provide the same results as the founded double implication association rule.

3. The apriori association rule can provide the same results as the founded equivalence association rule.

4. The predictive apriori association rule can provide the same results as the founded implication association rule.

5. The predictive apriori association rule can provide the same results as the founded double implication association rule.

6. The predictive apriori association rule can provide the same results as the founded equivalence association rule.

6. PRACTICAL RESULTS OF THE DIFFERENT APPROACHES

6.1 Test data base

The goal of the thesis is to compare the two approaches of GUHA and Weka.

The database 'Weather' has been chosen because of the size requirements. The database has been taken from a list of standard databases from the Weka program. The database has only *14* records and all quantifiers or association rules can be counted and explained easily.

Table 6.1 shows the data of the 'Weather' database.

6.2 Weka software results

The database name is 'Weather', shown in Table 6.1. Apriori and predictive apriori parameters will be used with the database.

Data preprocessing should be carried out first. The database has been preprocessed with the 'Unsupervised/Attribute/Discretize' Weka method. The number of rules

Table 6.1: Weather database.

No	Outlook nominal	Temperature numeric	Humidity numeric	Windy nominal	Play nominal
1	sunny	85.0	85.0	false	no
2	sunny	80.0	90.0	true	no
3	overcast	83.0	86.0	false	yes
4	rainy	70.0	96.0	false	yes
5	rainy	68.0	80.0	false	yes
6	rainy	65.0	70.0	true	no
7	overcast	64.0	65.0	true	yes
8	sunny	72.0	95.0	false	no
9	sunny	69.0	70.0	false	yes
10	rainy	75.0	80.0	false	yes
11	sunny	75.0	70.0	true	yes
12	overcast	72.0	90.0	true	yes
13	overcast	81.0	75.0	false	yes
14	rainy	71.0	91.0	true	no

has been increased to *30* rules. There are no other changes to the database or the Apriori parameters.

The *Apriori* method requires using the *minimum support* and *confidence*. Assume that minimum support is *2/14=0.14* and confidence is *0.7*.

The *Apriori* method returned *52* different association rules. The first *10* rules are shown in Table 6.2.

Table 6.2: Weather database. Apriori results

No	Association rule	support	confidence
1.	outlook(overcast) 4 ⇒ play(yes) 4	*0.29*	*1*
2.	humidity='(89.8-92.9]' 3 ⇒ windy(true) 3	*0.21*	*1*
3.	outlook(rainy) play(yes) 3 ⇒ windy(false) 3	*0.21*	*1*
4.	outlook(rainy) windy(false) 3 ⇒ play(yes) 3	*0.21*	*1*
5.	humidity='(77.4-80.5]' 2 ⇒ outlook(rainy) 2	*0.14*	*1*
6.	temperature='(-inf-66.1]' 2 ⇒ windy(true) 2	*0.14*	*1*
7.	temperature='(68.2-70.3]' 2 ⇒ windy(false) 2	*0.14*	*1*
8.	temperature='(68.2-70.3]' 2 ⇒ play(yes) 2	*0.14*	*1*
9.	temperature='(74.5-76.6]' 2 ⇒ play(yes) 2	*0.14*	*1*
10.	humidity='(83.6-86.7]' 2 ⇒ temperature='(82.9-inf)' 2	*0.14*	*1*

The predictive apriori method returned *64* different association rules with minimum accuracy *0.274*. The first *10* rules are shown in Table 6.2. The support column was calculated manually and added to the table.

Table 6.3: Weather database. Predictive apriori results

No	Association rule	support	accuracy
1.	outlook(overcast) 4 ⇒ play(yes) 4	*0.29*	*0.98*
2.	humidity='(89.8-92.9]' 3 ⇒ windy(true) 3	*0.21*	*0.96745*
3.	outlook(rainy) windy(false) 3 ⇒ play(yes) 3	*0.21*	*0.96745*
4.	outlook(rainy) play(yes) 3 ⇒ windy(false) 3	*0.21*	*0.96745*
5.	temperature='(-inf-66.1]' 2 ⇒ windy(true) 2	*0.14*	*0.94102*
6.	temperature='(68.2-70.3]' 2 ⇒ windy(false) play(yes) 2	*0.14*	*0.94102*
7.	temperature='(74.5-76.6]' 2 ⇒ play(yes) 2	*0.14*	*0.94102*
8.	temperature='(82.9-inf)' 2 ⇒ humidity='(83.6-86.7]' windy(false) 2	*0.14*	*0.94102*
9.	humidity='(77.4-80.5]' 2 ⇒ outlook(rainy) windy(false) 2	*0.14*	*0.94102*
10.	humidity='(77.4-80.5]' 2 ⇒ outlook(rainy) play(yes) 2	*0.14*	*0.94102*

The *Predictive apriori* method returned *43* results which are different from the Apriori results. The Apriori method returned *31* results which are different from the Predictive apriori method. The number of indentical association rules is *21.*

Association rules are found using the Predictive Apriori method and revising them via the differences with the Apriori results. The accuracy of predictive apriori rules is from *0.5438* to *0.27427*, but they do not support the minimum confidence threshold. Table 6.4 shows the first 10 association rules with accuracy of equal to or less than *0.5438.*

The lowest confidence is *0.222*, which is not shown in Table 6.4. Therefore those association rules found using the Predictive Apriori method and which are different

Table 6.4: Weather database. Predictive apriori results (different from Apriori results)

No	Association rule	conf	accuracy
1.	play(yes) 9 ⇒ windy(false) 6	*0.667*	*0.5438*
2.	temperature(70.3-72.4] (71,72) 3 ⇒ humidity(89.8-92.9] (90,91) windy(true) 2	*0.667*	*0.51932*
3.	humidity(68.1-71.2] (70) 3 ⇒ windy(true) 2	*0.667*	*0.51932*
4.	humidity(68.1-71.2] (70) 3 ⇒ outlook(sunny) play(yes) 2	*0.667*	*0.51932*
5.	humidity(89.8-92.9] (90,91) 3 ⇒ windy(true) play(no) 2	*0.667*	*0.51932*
6.	humidity(89.8-92.9] (90,91) 3 ⇒ temperature(70.3-72.4] (71,72) windy(true) 2	*0.667*	*0.51932*
7.	outlook(sunny) windy(false) 3 ⇒ play(no) 2	*0.667*	*0.51932*
8.	outlook(sunny) play(no) 3 ⇒ windy(false) 2	*0.667*	*0.51932*
9.	outlook(rainy) windy(false) 3 ⇒ humidity(77.4-80.5] (80) play(yes) 2	*0.667*	*0.51932*
10.	outlook(rainy) play(yes) 3 ⇒ humidity(77.4-80.5] (80) windy(false) 2	*0.667*	*0.51932*

from the Apriori results do not sutisfy the minimum confidence threshold.

Remark 6.2.1. *The Predictive apriori method returned the same results as the Apriori method and there are no reasons to check them separately from the Apriori method.*

6.2.1 Apriori algorithm results

An explanation of the results would be much easier with an implementation of the Apriori algorithm. The Apriori method was explained in Chapter 4.4. The database 'Weather' is small enough to calculate manually with the Apriori method.

Table 6.1 shows the rows of the 'weather' database. Assume that *minimum support* is *0.14* (*2/14=0.143*). The same minimum support has been used to produce Table 6.2. Assume that *minimum confidence* is *0.7*.

The apriori algorithm requires the data preprocessing step when using numerical values. The algorithm can only work with categorical data and so the 'Weather' database should be discretized.

The data preprocessing step changes the data of the database. The Weka program offers filters to do discretization. They are 'Supervised/Attribute/Discretize' and 'Unsupervised/Attribute/Discretize'. The discretization returns different results, depending on whether supervised or unsupervised discretisation is used. All the results were obtained using unsupervised discretization.

Remark 6.2.2. *A similar data preprocessing step should be done with the GUHA database, too.*

The tables show the results after applying method and support for every label. Table 6.5 shows 1-item sets. Calculating support for 1-item sets is the first step of the apriori algorithm, as explained in Chapter 4.4.

The apriori property 4.4.1 says that there are no reasons to retain non-frequent patterns where the minimum support rule does not hold true. The support should be more than the minumum support.

The 1-item sets should be frequent up to the apriori property 4.4.1. The *windy*, *play* and *outlook* tables satisfy the minimum support requirement (minimum support = *0.14*), but *temperature* and *humidity* do not. Table 6.6 shows temperature and humidity labels after applying the apriori property.

The second step is to generate 2-item sets. There are too many 2-item sets, 34 sets exactly, to show them all here, so Table 6.7 shows only *10* rules.

The third step is to generate 3-item sets and calculate the support. The number of 3-item sets, which have the minimum support more than *0.14*, is *13*. Table 6.8

Table 6.5: 1-item sets

windy	count	Support
true	6	0.43
false	8	0.57

play	count	Support
yes	9	0.64
no	5	0.36

Outlook	count	Support
sunny	5	0.36
overcast	4	0.29
rainy	5	0.36

label	temperature	count	Support
’(-inf-66.1]’	$\langle 64, 65\rangle$	2	0.14
’(66.1-68.2]’	$\langle 68\rangle$	1	0.07
’(68.2-70.3]’	$\langle 69, 70\rangle$	2	0.14
’(70.3-72.4]’	$\langle 71, 72\rangle$	3	0.21
’(72.4-74.5]’	$\langle\emptyset\rangle$	0	0
’(74.5-76.6]’	$\langle 75\rangle$	2	0.14
’(76.6-78.7]’	$\langle\emptyset\rangle$	0	0
’(78.7-80.8]’	$\langle 80\rangle$	1	0.07
’(80.8-82.9]’	$\langle 81\rangle$	1	0.07
’(82.9-inf)’	$\langle 83, 85\rangle$	2	0.14

label	humidity	count	Support
’(-inf-68.1]’	$\langle 65\rangle$	1	0.07
’(68.1-71.2]’	$\langle 70\rangle$	3	0.21
’(71.2-74.3]’	$\langle\emptyset\rangle$	0	0
’(74.3-77.4]’	$\langle 75\rangle$	1	0.07
’(77.4-80.5]’	$\langle 80\rangle$	2	0.14
’(80.5-83.6]’	$\langle\emptyset\rangle$	0	0
’(83.6-86.7]’	$\langle 85, 86\rangle$	2	0.14
’(86.7-89.8]’	$\langle\emptyset\rangle$	0	0
’(89.8-92.9]’	$\langle 90, 91\rangle$	3	0.21
’(92.9-inf)’	$\langle 95, 96\rangle$	2	0.14

shows them all.

The fourth step is to generate 4-item sets and calculate the support. The step revealed only one 4-item set $\langle windy(false), play(yes), outlook(rainy), humidity(77.4-80.5](80)$. Therefore, this is the last step because there is no possibility to create a 5-item set.

The final step is to generate association rules and calculate confidence using the formula 4.11. There are plenty of association rules which can be generated from these frequent sets, explained by the rules which are in Table 6.2.

Table 6.6: 1-item frequent sets

label	temperature	count	Support
'(-inf-66.1]'	$\langle 64, 65\rangle$	*2*	*0.14*
'(68.2-70.3]'	$\langle 69, 70\rangle$	*2*	*0.14*
'(70.3-72.4]'	$\langle 71, 72\rangle$	*3*	*0.21*
'(74.5-76.6]'	$\langle 75\rangle$	*2*	*0.14*
'(82.9-inf)'	$\langle 83, 85\rangle$	*2*	*0.14*

label	humidity	count	Support
'(68.1-71.2]'	$\langle 70\rangle$	*3*	*0.21*
'(77.4-80.5]'	$\langle 80\rangle$	*2*	*0.14*
'(83.6-86.7]'	$\langle 85, 86\rangle$	*2*	*0.14*
'(89.8-92.9]'	$\langle 90, 91\rangle$	*3*	*0.21*
'(92.9-inf)'	$\langle 95, 96\rangle$	*2*	*0.14*

Table 6.7: 2-items frequent sets

N0	2-item set	count	Support
1	$\langle windy(false), play(yes)\rangle$	*6*	*0.43*
2	$\langle play(yes), outlook(overcast)\rangle$	*4*	*0.29*
3	$\langle windy(true), play(yes)\rangle$	*3*	*0.21*
4	$\langle windy(true), play(no)\rangle$	*3*	*0.21*
5	$\langle windy(true), humidity(89.8-92.2](90, 91)\rangle$	*3*	*0.21*
6	$\langle windy(false), outlook(rainy)\rangle$	*3*	*0.21*
7	$\langle play(no), outlook(sunny)\rangle$	*3*	*0.21*
8	$\langle outlook(rainy), humidity(77.4-80.5](80)\rangle$	*2*	*0.14*
9	$\langle Windy(true), temperature(-inf-66.1](64, 65)\rangle$	*2*	*0.14*
10	$\langle Windy(false), temperature(68.2-70.3](69, 70)\rangle$	*2*	*0.14*

The first association rule is *outlook(overcast)* ⇒ *play(yes)*. The rule should be constructed from the second row of Table 6.7.

The next association rule is *humidity='(89.8-92.9]'* ⇒ *windy(true)*. The rule should be constructed from the fifth row of Table 6.7.

The next association rule is *outlook(rainy) play(yes)* ⇒ *windy(false)* and *outlook(rainy) windy(false)* ⇒ *play(yes)*. The rule should be constructed from the first row of Table 6.8.

The next association rule is *humidity='(77.4-80.5]'* ⇒ *outlook(rainy)*. The rule

Table 6.8: 3-item frequent sets

3-item sets	count	Support
$\langle play(yes), outlook(sunny), humidity(68.1-71.2](70)\rangle$	*2*	*0.14*
$\langle play(no), outlook(sunny), windy(no)\rangle$	*2*	*0.14*
$\langle windy(true), play(yes), outlook(overcast)\rangle$	*2*	*0.14*
$\langle windy(false), play(yes), outlook(overcast)\rangle$	*2*	*0.14*
$\langle windy(false), outlook(rainy), humidity(77.4-80.5](80)\rangle$	*2*	*0.14*
$\langle play(yes), outlook(rainy), humidity(77.4-80.5](70)\rangle$	*2*	*0.14*
$\langle windy(true), play(no), outlook(rainy)\rangle$	*2*	*0.14*
$\langle windy(false), play(yes), outlook(rainy)\rangle$	*3*	*0.21*
$\langle windy(false), play(yes), temperature(68.2-70.3](69,70)\rangle$	*2*	*0.14*
$\langle windy(true), temperature(70.3-72.4](71,72), humidity(89.8-92.9](90,91)\rangle$	*2*	*0.14*
$\langle windy(false), temperature(82.9-inf](83,85), humidity(83.6-86.7](85,86)\rangle$	*2*	*0.14*
$\langle windy(false), play(yes), humidity(77.4-80.5](80)\rangle$	*2*	*0.14*
$\langle windy(true), play(no), humidity(89.8-92.9](90,91)\rangle$	*2*	*0.14*

should be constructed from the eighth row of Table 6.7

The next association rule is *temperature='(-inf-66.1]']'* ⇒ *windy(true)*. The rule should be constructed from the ninth row of Table 6.7.

The next association rule is *temperature='(68.2-70.3]'* ⇒ *windy(false)*. The rule should be constructed from the tenth row of Table 6.7.

The next association rule is *temperature='(68.2-70.3]'* ⇒ *play(yes)*. The rule should be constructed from the eleventh row of Table 6.7.

The next association rule is *temperature='(74.5-76.6]'* ⇒ *play(yes)*. The rule should be constructed from the twelfth row of Table 6.7.

The last association rule is *humidity='(83.6-86.7]'* ⇒ *temperature='(82.9-inf)'*. The rule should be constructed from the thirteenth row of Table 6.7.

The last step is to show why the 4-item set did not appear in the resulting table 6.2.

Example 6.2.1. (4-item set association rules) *The 4-item set is ⟨windy(false), play(yes), outlook(rainy), humidity(77.4-80.5] (80)⟩. Chapter 4.4 and the example 4.4.2 show how to create association rules from a given item set. The subsets of the 4-item set are shown in Table 6.9.*

Table 6.9: Subsets of 4-item frequent set

subset
$\langle windy(false)\rangle$
$\langle play(yes)\rangle$
$\langle outlook(rainy)\rangle$
$\langle humidity(77.4-80.5](80)\rangle$
$\langle windy(false), play(yes)\rangle$
$\langle windy(false), outlook(rainy)\rangle$
$\langle windy(false), humidity(77.4-80.5](80)\rangle$
$\langle play(yes), outlook(rainy)\rangle$
$\langle play(yes), humidity(77.4-80.5](80)\rangle$
$\langle outlook(rainy), humidity(77.4-80.5](80)\rangle$
$\langle windy(false), play(yes), outlook(rainy)\rangle$
$\langle windy(false), play(yes), humidity(77.4-80.5](80)\rangle$
$\langle play(yes), outlook(rainy), humidity(77.4-80.5](80)\rangle$

The Association rules which are interesting and are used below are shown in Table 6.10. The support of all association rules in Table 6.10 is $2/14 = 0.14$

Table 6.10: Association rules of the 4-item set.

No	row in results	Association rule	confidence
1	50	$outlook(rainy), humidity(77.4-80.5](80) \Rightarrow play(yes), windy(false)$	*2/2=1*
2	48	$play(yes), humidity(77.4-80.5](80) \Rightarrow windy(false), outlook(rainy)$	*2/2=1*
3	49	$windy(false), humidity(77.4-80.5](80) \Rightarrow play(yes), outlook(rainy)$	*2/2=1*

Therefore, the 4-item rules appeared in the resulting table in rows with numbers more than *30*, but did not appear in the resulting table 6.2.

6.3 GUHA software results

6.3.1 Founded implication results

The GUHA *founded implication* is used with parameters which allow as many association rules as possible to be obtained. The *Base Absolute value* is *2*, $a \geq BASE$, and the *Founded Implication value* is *p=0.7*, *a/(a+b)≥p.*

The GUHA *founded implication* method has discovered *52* hypotheses. They are the same as the *52* hypotheses outlined in Chapter 6.2.

Remark 6.3.1. *Chapter 3.4.10 shows the definition 3.4.17 of Association rules with the 4ft table 3.5.*

The equation for minimum support σ is

$$\sigma \leq \frac{a}{a+b+c+d}$$

The equation for minimum confidence γ is

$$\gamma \leq \frac{a}{a+b}$$

It is obvious that all rules discovered by Apriori will be the same as the rules discovered by the Founded Implication method, but the Founded Implication method can find some methods, where a/(a+b+c+d)≤ σ.

The formula 4.11 or 6.1 [1] shows the confidence rule .

$$confidence(A \Rightarrow B) = P(B|A) = support\_count(A \cup B)/support\_count(A) \quad (6.1)$$

It is plain to see that the equations (6.2) and (6.3) hold true.

$$support\_count(A) = (A \wedge B) \cup (A \wedge \neg B) = a + b \tag{6.2}$$

$$support\_count(A \cup B) = (A \wedge B) = a \tag{6.3}$$

Therefore, the formula (6.4) shows the confidence where a *and* b *are variables of the 4ft-table 3.3.*

$$confidence(A \Rightarrow B) = a/(a + b) \tag{6.4}$$

6.3.2 Double Founded implication results

The GUHA *double founded implication* is used with parameters which allow as many association rules as possible to be obtained. The *Base Absolute value* is *2*, *a≥BASE*, and the *Double Founded Implication value* is *p=0.7*, *a/(a+b+c)≥p*.

The GUHA *double founded implication* method has discovered *6* hypotheses. All *6* hypotheses have been shown by the Apriori method. Table 6.11 shows the first *9* discovered Apriori rules.

Table 6.11: Weather database. Apriori results (not from double implication)

No	Association rule	*a*	*b*	*c*	Confidence *a/(a+b+c)*
1.	outlook(overcast) 4 ⇒ play(yes) 4	*4*	*0*	*5*	*0.444*
2.	humidity='(89.8-92.9]' 3 ⇒ windy(true) 3	*3*	*0*	*3*	*0.500*
3.	outlook(rainy) play(yes) 3 ⇒ windy(false) 3	*3*	*0*	*5*	*0.375*
4.	outlook(rainy) windy(false) 3 ⇒ play(yes) 3	*3*	*0*	*6*	*0.333*
5.	humidity='(77.4-80.5]' 2 ⇒ outlook(rainy) 2	*2*	*0*	*3*	*0.400*
6.	temperature='(-inf-66.1]' 2 ⇒ windy(true) 2	*2*	*0*	*4*	*0.333*
7.	temperature='(68.2-70.3]' 2 ⇒ windy(false) 2	*2*	*0*	*6*	*0.250*
8.	temperature='(68.2-70.3]' 2 ⇒ play(yes) 2	*2*	*0*	*7*	*0.222*
9.	temperature='(74.5-76.6]' 2 ⇒ play(yes) 2	*2*	*0*	*7*	*0.222*

Remark 6.3.2. *The Double founded implication method found only 6 association rules which are the same as the Apriori or Founded Implication results. The other*

association rules found by Apriori do not satisfy the Double Founded Implication minimum confidence requirement. The Apriori association rules can be discovered by reducing the confidence value of the Double Founded implication.

6.3.3 Founded Equivalence results

The GUHA *founded equivalence* is used with parameters which allow as many association rules as possible to be obtained. The *Base Absolute value* is *2*, *a≥BASE*, and the *Founded Equivalence value* is *p=0.7*, *(a+d)/(a+b+c+d)≥p.*

The GUHA *founded equivalence* method discovered *54* hypotheses, *28* of which are the same as the Apriori method.

Table 6.12 shows the first *10* hypotheses which have been obtained using the founded equivalence quantifier (apart from for the *28* found with the same rules as the Apriori method). The confidence column of Table 6.12 has been calculated using the formula 6.4.

The Apriori method found *24* association rules which are different from the founded equivalence method. Table 6.13 shows the first *10* rules sorted by support. Confidence equals *1* for every shown rule.

Remark 6.3.3. *The Founded Equivalence method discovered 28 association rules which are the same as the Apriori method. The other rules found by Apriori do not satisfy the Double Founded implication minimum confidence requirement. Otherwise, the minimum confidence can be reduced and the rules would appear in the Founded Equivalence results, and vice versa.*

So, all the association rules have been discovered, and the next step is discussion

Table 6.12: Weather database. Founded equivalence results

No	Antecedent ⇒ Succedent	Conf $a/(a+b)$	a	b	c	d
1.	Humidity 68.1-71.2 (70) ⇒ Play(yes) & Outlook(sunny)	*0.667*	*2*	*1*	*0*	*11*
2.	Temperature 70.3-72.4 (71,72) ⇒ Humidity 89.8-92.9 (90,91)	*0.667*	*2*	*1*	*1*	*10*
3.	Temperature 70.3-72.4 (71,72) ⇒ Windy(true) & Humidity 89.8-92.9 (90,91)	*0.667*	*2*	*1*	*1*	*10*
4.	Humidity 89.8-92.9 (90,91) ⇒ Play(no) & Windy(true)	*0.667*	*2*	*1*	*1*	*10*
5.	Play(no) & Windy(true) ⇒ Humidity 89.8-92.9 (90,91)	*0.667*	*2*	*1*	*1*	*10*
6.	Windy(true) ⇒ Humidity 89.8-92.9 (90,91)	*0.5*	*3*	*3*	*0*	*8*
7.	Play(yes) & Windy(true) ⇒ Outlook(overcast)	*0.667*	*2*	*1*	*2*	*9*
8.	Play(no) ⇒ Windy(true) & Outlook(rainy)	*0.4*	*2*	*3*	*0*	*9*
9.	Outlook(sunny) ⇒ Play(yes) & Humidity 68.1-71.2 (70)	*0.4*	*2*	*3*	*0*	*9*
10.	Outlook(sunny) ⇒ Play(no) & Windy(false)	*0.4*	*2*	*3*	*0*	*9*

and analysis of the obtained results.

Table 6.13: Weather database. Apriori results (not from founded equivalence results)

No	Association rule	*a*	*b*	*c*	*d*	*confidence* *(a+d)/(a+b+c+d)*
1.	outlook(overcast) 4 ⇒ play(yes) 4	*4*	*0*	*5*	*5*	*0.642*
2.	outlook(rainy) play(yes) 3 ⇒ windy(false) 3	*3*	*0*	*5*	*6*	*0.642*
3.	outlook(rainy) windy(false) 3 ⇒ play(yes) 3	*3*	*0*	*6*	*5*	*0.571*
4.	temperature(68.2-70.3] 2 ⇒ windy(false) 2	*2*	*0*	*6*	*6*	*0.572*
5.	temperature(68.2-70.3] 2 ⇒ play(yes) 2	*2*	*0*	*7*	*5*	*0.500*
6.	temperature(74.5-76.6] 2 ⇒ play(yes) 2	*2*	*0*	*7*	*5*	*0.500*
7.	temperature(82.9-inf) 2 ⇒ windy(false) 2	*2*	*0*	*6*	*6*	*0.571*
8.	humidity(77.4-80.5] 2 ⇒ windy(false) 2	*2*	*0*	*6*	*6*	*0.571*
9.	humidity(77.4-80.5] 2 ⇒ play(yes) 2	*2*	*0*	*7*	*5*	*0.500*
10.	humidity(83.6-86.7] 2 ⇒ windy(false) 2	*2*	*0*	*6*	*6*	*0.571*

7. DISCUSSION OF THE THEORETICAL HYPOTHESES AND THE PRACTICAL RESULTS

7.1 Founded Implication discussion

The *Founded implication* method returned the same association rules as the Apriori method. The remark 6.3.1 shows that the Founded Implication and Apriori methods are the same. Therefore, all the results of both methods are identical.

7.2 Double Founded Implication discussion

The *Double Founded Implication* DFUI method returned only *6* methods which are the same as the Apriori method. On the other hand, there are plenty of rules which were not discovered by DFUI. The base 4ft-tables are the same for all Association rules in both methods, but the methods for calculating the confidence are different. The *Apriori* confidence 6.4 should be more than $a/(a+b)$, as opposed to DFUI, where the confidence should be more than $a/(a+b+c)$. It is obvious that the rules discovered by DFUI should have the variable c small. The variable c shows the number of rules where the *succedent* is labeled `TRUE`, but the *antecedent* is labeled `FALSE`, or $A \Rightarrow B$, where A is labeled `TRUE` and B is labeled `FALSE`.

Remark 7.2.1. *The Double Founded Implication method can be used when a user wants to discover only those strong rules where there are not many rules with succedent labeled* `TRUE` *and antecedent labeled* `FALSE`*.*

7.3 Founded Equivalence discussion

The *Founded Equivalence* discovered *54* hypotheses, *28* of which are the same as the Apriori method. Strong rules of Founded Equivalence are calculated as a proportion of *(a+d)/(a+b+c+d)*, where a,b,c and d are variables of Table 3.5. Thus, the rule is $A \Rightarrow B$ and the number $a+d$ shows the number of all hypotheses which satisfy the conditions $\#(\neg A \wedge \neg B) = d$ and, in addition, rules with the condition $\#(A \wedge B) = a$. The number $a+b+c+d$ is the number of all the rows in the database.

The Founded Equivalence method shows hypotheses where there are proportionally small numbers b and c instead of a and d numbers.

Remark 7.3.1. *The Founded Equivalence method can be used when a user wants to discover only those strong rules which have proportionally small numbers b and c in Table 3.5. The method does not require a small number of 'a' variable, the lower bound of the number of hypotheses which satisfy both the succedent and antecedent is controlled by the Base variable.*

7.4 General discussion

There are two different methods, GUHA and Weka. If a user decides to use both methods, he or she can reduce the time needed for discovering and improving the efficiency by first using the Apriori method. The Apriori method automatically discovers all patterns with strong support, but the method can return too many potentially interesting hypotheses and depends on the confidence parameter. A user can use other GUHA methods with the interesting hypotheses later. The Double Implication method returns fewer potentially interesting rules. The GUHA method allows a user to find only those patterns which are potentially interesting for the user. Of course, the Weka method allows hypotheses to be found using the classIndex property, but GUHA methods can construct different patterns manually.

The Weka method generates all possible interesting hypotheses which satisfy support and confidence. The number of those hypotheses could be high and reducing the amount of hypotheses is possible only by changing the values for support and confidence, or by using the value 'classIndex', which allows the class attribute to be selected.

The GUHA method, otherwise, allows the creation of only those hypotheses which are interesting from the point of view of a given general problem [8]. That fact allows a hypothesis to be constructed manually, discovering only those which are interesting for the user, instead of creating all of them.

8. CONCLUSION

The development of computers allows huge amounts of data to be stored in databases. The problem with those huge databases is how to analyse all the data in them. A manager may not be able to analyse the data and so he may write a report based on how he thinks things 'could be'. The report is not based on the data in this case. There are many different methods which can be used to mine the data to obtain factual information.

The methods which can be used are developing continuously. There are programmes which allow the data in those databases to be analysed with these methods. Two programmes have been introduced in this thesis. They are GUHA and Weka. Both of them can discover similar, partial or almost identical results.

Weka methods do not require any knowledge of the database. The methods construct all the possible interesting patterns and a user only has to check whether or not the patterns have something new. The GUHA approach requires some knowledge of the database. A user should at least have an idea about the columns of the database. The GUHA methods allows the creation of only those patterns which are interesting for a user.

GUHA software uses the Founded Implication method, which provides exactly the same results as the Apriori method. The Double Founded Implication method can be used to discover only those rules in which there not many rules where the succedent is true and the antecedent is false. The Founded Equivalence method can be used to discover only those rules where the variables b and c are proportionally small.

The reader should remember that Weka data preparation includes discretization. The thesis did not check the possibility of generating more interesting rules without discretisation, but this could be easily checked with GUHA software by constructing rules manually. The Weka software uses models instead of the mathematical formulas which are used in the GUHA software.

REFERENCES

[1] Jiawei Han, Micheline Kamber, *Data Mining, Concepts and Techniques.* Morgan Kaufmann Publishers is an imprint of Elsevier, 2nd Edition, 2006.

[2] Jan Rauch, David Coufal and others, *The GUHA Method, Data Preprocessing and Mining.*

[3] Hajek P., Havel I., Chytil M., *The method of automatic hypotheses determination, Computing 1.* 1966.

[4] Hajek, P., Havranek, T., *Mechanizing Hypothesis Formation - Mathematical Foundations for a General Theory.* 1978.

[5] Rauch, J. Simunek, M., *GUHA Method and Granular Computing.* 2005.

[6] Rauch, J. *Logic of Association Rules. Applied Intelligence*, 22(2005), pp. 9-28.

[7] Rauch, J. *Classes of Association Rules - an Overview.*, In: Data Mining: Foundations and Practice. Studies in Computational Intelligence. Vol.118. Berlin 2008, Springer-Verlag. pp.314-337

[8] Esko Turunen *MAT-42106 Applied Logics*, http://matwww.ee.tut.fi/ eturunen/AppliedLogics2008.html

[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reute-

mann, Ian H. Witten *The WEKA Data Mining Software: An Update; SIGKDD Explorations*, 2009, Vol. 11, Issue 1

[10] J.Rauch (1998) *Contribution to Logical Foundations of KDD*, Assoc. Prof. Thesis, Faculty of Informatics and Statistics, University of Economics, Prague (in Czech)

[11] R.Srikant and R. Arrawal (1996) *Mining seqential patterns:Gentralizations and perfomance improvements.*, In *In Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT)*, Avignon, France, March 1996.

[12] Scheffer, T. , (2005) *Finding association rules that trade support optimally against confidence*, *Intelligent Data Analysis* 9(4), 381-395