A Comparison of the Quality of Rule Induction from Inconsistent Data Sets and Incomplete Data Sets

By

Xiaomeng Su

Submitted to the Department of Electrical Engineering and Computer Science and the
Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Master of Science

Jerzy W. Grzymala-Busse, Chairperson

Committee members

Prasad A. Kulkarni

Zongbo Wang

Date defended: June 9, 2015

The Project Report Committee for Xiaomeng Su certifies
that this is the approved version of the following project report :

A Comparison of the Quality of Rule Induction from Inconsistent Data Sets and Incomplete Data Sets

Jerzy W. Grzymala-Busse, Chairperson

Date approved: June 9, 2015

Abstract

In data mining, decision rules induced from known examples are used to classify unseen cases. There are various rule induction algorithms, such as LEM1 (Learning from Examples Module version 1), LEM2 (Learning from Examples Module version 2) and MLEM2 (Modified Learning from Examples Module version 2). In the real world, many data sets are imperfect, either inconsistent or incomplete. The idea of lower and upper approximations, or more generally, the probabilistic approximation, provides an effective way to induce rules from inconsistent data sets and incomplete data sets. But the accuracies of rule sets induced from imperfect data sets are expected to be lower. The objective of this project is to investigate which kind of imperfect data sets (inconsistent or incomplete) is worse in terms of the quality of rule induction. In this project, experiments were conducted on eight inconsistent data sets and eight incomplete data sets with lost values. We implemented the MLEM2 algorithm to induce certain and possible rules from inconsistent data sets, and implemented the local probabilistic version of MLEM2 algorithm to induce certain and possible rules from incomplete data sets. A program called Rule Checker was also developed to classify unseen cases with induced rules and measure the classification error rate. Ten-fold cross validation was carried out and the average error rate was used as the criterion for comparison. The Mann-Whitney nonparametric tests were performed to compare, separately for certain and possible rules, incompleteness with inconsistency. The results show that there is no significant difference between inconsistent and incomplete data sets in terms of the quality of rule induction.

Acknowledgements

Looking back to the years I spent at KU, there are so many people who had helped me. This thesis would not have been finished without their support.

I would like to express my deepest thanks to my advisor Dr. Jerzy Grzymala-Busse for his persistent guidance on my study and thesis. His enthusiasm of teaching and research impressed me a lot, and will have long lasting effects on me. I also thank Dr. Prasad Kulkarni, Dr. Zongbo Wang for serving in my committee, and for their fantastic suggestions and inspiring questions. Thank the professors in my department for the wonderful teaching, I really learned a lot from their lectures. I've expanded scope of knowledge, and acquired skills for research and work in the future.

My special thanks go to Dr. Heather Desaire and Dr. Man Kong. They took care of me and were always there for me. I would not be able to sutdy here without their generous help. I also thank Pam Shadoin and Anna Paradis, for taking care of all the "little" things, but together they make a whole difference.

I am also grateful for my family and friends. They always cheer me up and stand by me through ups and downs. Thank all the people who made this report possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As we encounter exponential growth of data acquisition in recent years, it is of great interest to discover the hidden information lying beneath the data, so that explanation of patterns and prediction of new examples are possible. Data mining is a research area aiming to extract such previously unknown information. Classification is a type of problems in which the decision values are category labels, and the decisions of training examples are known. One way to represent the extracted knowledge of classification problems is to use decision rules.

There are many rule induction algorithms. In this project, we focused on the LERS (Learning from Examples based on Rough Set) learning system, which is a data mining system that induces a rule set from examples represented as a decision table, and classifies unseen examples using the previously induced rule set (Grzymala-Busse, 1992). LEM1 (Learning from Examples Module version 1), LEM2 (Learning from Examples Module version 2) and MLEM2 (Modified Learning from Examples Module version 2) are different rule induction algorithms frequently used in LERS system.

Ideally we want training sets to be consistent and complete, from such sets rule sets with high accuracy could be induced. However, in the real world, we usually have imperfect data sets, either inconsistent or incomplete, which make rule induction more difficult. Lower and upper approximations, or more generally, probabilistic approximations, provide an effective and practical way to induce certain and possible rules from inconsistent data sets and incomplete data sets (Pawlak, 1982) (Pawlak, 1992) (Clark & Grzymala-Busse, 2011) (Grzymala-Busse & Siddhaye, 2004) (Grzymala-Busse et al., 2014). The approximation is based on "indiscernibility relation", which is an equivalence relation on inconsistent data sets. For incomplete data set, the indiscernibility relation is reflexive, but is not necessarily symmetric or transitive. Therefore the approximation on incomplete data set is not unique. One has several choices on which approximation to use, for example, subset approximation, concept approximation, or local approximation. However, no matter what approximation is used, the qualities of rule sets induced from inconsistent data sets and incomplete data sets are expected to be lower than rules induced from perfect data set. Since inconsistent data sets and incomplete data sets are very common, it is worth the effort to investigate which one is worse.

The objective of this project is to find out which type of imperfect data sets, inconsistent or incomplete, has more negative effects on rule induction. In this project, we implemented the conventional MLEM2 algorithm and the local probabilistic version of MLEM2 algorithm. We also developed a program to classify cases from testing set and measure the classification error rate. Experiments were carried out on eight inconsistent data sets and eight incomplete data sets. For inconsistent data sets, certain rules and possible rules were induced with MLEM2 algorithm. For incomplete data sets, we only considered one type of missing attribute values, which are *lost* values (denoted by ?), and applied the local probabilistic version of MLEM2 algorithm to induce certain rules and possible rules. Ten-fold cross validation was conducted on each data set, and the average error rate was used as the criterion for comparison. The Mann-Whitney nonparametric test was performed to compare, separately for certain and possible rules, incompleteness with inconsistency. The results show that inconsistent data sets and incomplete data sets are comparable with each other in terms of the quality of rule induction. There is no significant difference between the two types of imperfect data sets.

Chapter 2

Data Representation

2.1 Decision Table

In this thesis, we follow the terminologies of the LERS system (Grzymala-Busse, 1992). In the LERS system, an input data set is presented as a *decision table*. Each row of a decision table is called a *case*, and each column corresponds to a *variable*. One special variable is selected as *decision*, and the other variables are *attributes*. Conventionally, we use d to denote decision, U to denote the entire set of cases, and A to denote the set of all attributes. Table 2.1 is an example of a decision table which keeps a record on the symptoms of a patient and whether he/she has flu or not. In this table, $U = \{1,2,3,4,5,6,7\}$, $d = Flu$, and $A = \{Temperature, Headache, Cough\}$.

Table 2.1: An example of a decision table

		Attributes		Decision
Case id	Temperature	Headache	Cough	Flu
1	100.6	yes	yes	yes
2	102.4	yes	no	yes
3	98.0	no	yes	yes
4	102.4	yes	no	no
5	98.0	yes	no	no
6	98.0	no	yes	no
7	102.4	yes	no	no

Table 2.2: All attribute-value blocks of Table 2.1

(a,v)	$[(a,v)]$
$(Temperature, 98.0)$	$\{3,5,6\}$
$(Temperature, 100.6)$	$\{1\}$
$(Temperature, 102.4)$	$\{2,4,7\}$
$(Headache, yes)$	$\{1,2,4,5,7\}$
$(Headache, no)$	$\{3,6\}$
$(Cough, yes)$	$\{1,3,6\}$
$(Cough, no)$	$\{2,4,5,7\}$

"Headache" and "Cough" are *symbolic attributes* because the values of these attributes are discrete labels. On the contrary, "Temperature" is a *numerical attribute* because the values of this attribute are continuous numerical numbers. Numerical attributes are usually discretized to intervals. The MLEM2 algorithm handles discretization at the same time with rule induction, which will be discussed in Chapter 3.

Let a denote an attribute, x denote a case, v denote a value of a particular attribute. The pair (a,v) is called an *attribute-value pair*. For example, $(Headache, yes)$ is an attribute-value pair. The *attribute-value pair block*, denoted by $[(a,v)]$ for complete data sets is defined as

$$[(a,v)] = \{x | x \in U, a(x) = v\} \tag{2.1}$$

A full list of all attribute-value blocks of Table 2.1 is summarized in Table 2.2.

Similarly, we could define *concept*, which is the block of a decision-value pair.

$$[(d,w)] = \{x | x \in U, d(x) = w\} \tag{2.2}$$

In Table 2.1, there are two concepts: $[(Flu, yes)] = \{1,2,3\}$ and $[(Flu, no)] = \{4,5,6,7\}$. With the knowledge of attribute-value pair blocks and concepts, we are able to discuss consistency, completeness, and approximations.

2.2 Inconsistent Data Sets

2.2.1 Indiscernibility Relation and Elementary Sets

A decision table is consistent if there does not exist more than one case that share the same values for all attributes, while belong to different concepts. A formal definition is based on *indiscernibility* relation of cases, which is denoted by $\underset{B}{\sim}$. Let x and y be two cases and B be a subset of all variables $(B \subseteq A \cup \{d\})$. Case x is indiscernible from case y by B if and only if for every element a in B, $a(x) = a(y)$ (Pawlak, 1982), as is shown in Equation 2.3.

$$x \underset{B}{\sim} y \Leftrightarrow \forall a \in B, a(x) = a(y) \tag{2.3}$$

In Table 2.1, we have

$$\begin{array}{llllll}
1 \underset{A}{\sim} 1 & 2 \underset{A}{\sim} 2 & 2 \underset{A}{\sim} 4 & 2 \underset{A}{\sim} 7 & & \\
3 \underset{A}{\sim} 3 & 3 \underset{A}{\sim} 6 & 4 \underset{A}{\sim} 2 & 4 \underset{A}{\sim} 4 & 4 \underset{A}{\sim} 7 & \\
5 \underset{A}{\sim} 5 & 6 \underset{A}{\sim} 3 & 6 \underset{A}{\sim} 6 & 7 \underset{A}{\sim} 2 & 7 \underset{A}{\sim} 4 & 7 \underset{A}{\sim} 7
\end{array}$$

Observe that the indiscernibility relation on a complete data set is reflexive, symmetric and transitive. In other words, the indiscernibility relation on complete data sets is an equivalence relation on U. Therefore, the idea of partition could be used to describe pairwise indiscernibility relations in a more concise way. The set of all cases that are indiscernible from x by B, denoted by $[x]_B$, is called an *equivalent set.*

$$[x]_B = \cap\{[(a, a(x))] | a \in B\} \tag{2.4}$$

For example, in Table 2.1, $[1]_A = \{1\}$, and $[2]_A = \{2, 4, 7\}$.

Partition B^* of U is the set of all equivalent sets of indiscernibility relation generated by B.

$$B^* = \{[x]_B | x \in U\} \tag{2.5}$$

If $B = A$, the equivalent sets of A^* are called *elementary sets*.

For Table 2.1,

$$\{Temperature\}^* = \{\{1\},\{2,4,7\},\{3,5,6\}\}$$
$$\{Headache\}^* = \{\{1,2,4,5,7\},(3,6\}\}$$
$$\{Cough\}^* = \{\{1,3,6\},\{2,4,5,7\}\}$$
$$\{Temperature, Cough\}^* = \{\{1\},\{2,4,7\},\{3,6\},\{5\}\}$$

Consider two partitions on U, partition P^* is said to be smaller than or equal to partition Q^* if and only if for each block $E \in P^*$, there exists a block $E' \in Q^*$, such that E is a subset of E'. For example, in Table 2.1, $\{Temperature, Cough\}^* \leqslant \{Temperature\}^*$.

The consistency of a data set is based on the inequality of partitions. A decision table is said to be *consistent* if and only if $A^* \leqslant \{d\}^*$.

Table 2.1 is inconsistent because

$$A^* = \{Temperature, Headache, Cough\}^* = \{\{1\},\{2,4,7\},\{3,6\},\{5\}\}$$
$$\{d\}^* = \{Flu\}^* = \{\{1,2,3\},\{4,5,6,7\}\}$$
$$A^* \nleqslant \{d\}^*$$

Note that when we discuss consistency, we assume the decision table to be complete, i.e., there are no missing attribute values. If a decision contains missing attribute values, it is categorized as incomplete data set.

2.2.2 Lower and Upper Approximations

For consistent data sets, rules for a concept could be directly induced with algorithms like LEM1 and LEM2 without preliminary processing. However, for inconsistent data sets, we need to use approximations of concepts to induce rules (Pawlak, 1992) (Grzymala-Busse, 1992). Consider an inconsistent data set, given a set $X \subseteq U$ and a set $B \subseteq A$, the *B-singleton lower approximation* of X, denoted by $\underline{appr}_B^{singleton}(X)$ is defined as follows

$$\underline{appr}_B^{singleton}(X) = \{x | x \in U, [x]_B \subseteq X\} \tag{2.6}$$

The *B-subset lower approximations* of X, denoted by $\underline{appr}_B^{subset}(X)$ is defined as follows

$$\underline{appr}_B^{subset}(X) = \cup\{[x]_B | x \in U, [x]_B \subseteq X\} \tag{2.7}$$

The *B-concept lower approximations* of X, denoted by $\underline{appr}_B^{concept}(X)$ is defined as follows

$$\underline{appr}_B^{concept}(X) = \cup\{[x]_B | x \in X, [x]_B \subseteq X\} \tag{2.8}$$

The *B-singleton upper approximation* of X, denoted by $\overline{appr_B}^{singleton}(X)$ is defined as follows

$$\overline{appr}_B^{singleton}(X) = \{x | x \in U, [x]_B \cap X \neq \emptyset\} \tag{2.9}$$

The *B-subset upper approximation* of X, denoted by $\overline{appr}_B^{subset}(X)$ is defined as follows

$$\overline{appr}_B^{subset}(X) = \cup\{[x]_B | x \in U, [x]_B \cap X \neq \emptyset\} \tag{2.10}$$

The *B-concept upper approximation* of X, denoted by $\overline{appr}_B^{concept}(X)$ is defined as follows

$$\overline{appr}_B^{concept}(X) = \cup\{[x]_B | x \in X, [x]_B \cap X \neq \emptyset\} \tag{2.11}$$

For example, the singleton, subset, and concept lower and upper approximations of concept $[(Flu, yes)] = \{1, 2, 3\}$ of Table 2.1 are

$$\underline{appr}_A^{singleton}(X) = \underline{appr}_A^{subset}(X) = \underline{appr}_A^{concept}(X) = \{1\}$$
$$\overline{appr}_A^{singleton}(X) = \overline{appr}_A^{subset}(X) = \overline{appr}_A^{concept}(X) = \{1, 2, 3, 4, 6, 7\}$$

Table 2.3: Probabilistic approximations for $[(Flu, yes)]$ of Table 2.1

α	$appr_{\alpha,A}(\{1,2,3\})$
1/3	$\{1,2,3,4,6,7\}$
1/2	$\{1,3,6\}$
1	$\{1\}$

For inconsistent data sets, the singleton, subset and concept definitions of lower and upper approximations give the same results. Thus sometimes we could omit the word singleton, subset or concept, and simply say lower and upper approximations.

2.2.3 Probabilistic Approximations

Lower and upper approximations are special cases of a more generalized approximation called probabilistic approximation. For inconsistent data sets, the singleton, subset and concept definitions of probabilistic approximations also give the same approximation results. Therefore we do not distinguish the three definitions. Given a set $X \subseteq U$, and a set $B \subseteq A$, the probabilistic approximations of X with a parameter α, $0 < \alpha \leqslant 1$, are defined as follows (Clark & Grzymala-Busse, 2011)

$$appr_{\alpha,B}^{singleton}(X) = appr_{\alpha,B}^{subset}(X) = appr_{\alpha,B}^{concept}(X) = \cup\{[x]_B | x \in U, P(X|[x]_B) \geqslant \alpha\} \tag{2.12}$$

where

$$P(X|[x]_B) = \frac{|X \cap [x]_B|}{|[x]_B|}.$$

The probabilistic approximations of concept $[(Flu, yes)] = \{1,2,3\}$ of Table 2.1 are listed in Table 2.3. Observe that with very small α, the probabilistic approximation of X equals to the B-lower approximation of X, and with $\alpha = 1$ the probabilistic approximation of X becomes the B-upper approximation of X.

2.3 Incomplete Data Sets

2.3.1 Missing Attribute Values and Characteristic Sets

Although Table 2.1 is inconsistent, it is complete since every attribute value is specified. In the real world, there are many incomplete data sets with missing attribute values due to a lot of reasons. We consider three interpretations of missing attribute values: *lost* values, *"do not care"* values and *attribute-concept* values.

- Lost values, denoted by ?

  If the original value of an attribute is erased or is never obtained, we interpret it as a lost value. For lost values, we have no idea to guess what the original values are. Therefore, if $a(x) = ?$, x is not included in any $[(a,v)]$ blocks for all specified values v of attribute a.

- "Do not care" values, denoted by *

  "Do not care" values could be replaced by any specified values from the attribute domain. This usually happens when people do not feel comfortable to reply to some unpleasing questions, like age, income, and so on. If the corresponding value of case x is "do not care" value for attribute a, x is included in all $[(a,v)]$ blocks for all specified values v of attribute a.

- Attribute-concept values, denoted by -

  The property of attribute-concept values lies somewhere between lost values and "do not care" values. The original attribute value is lost, but it is reasonable for us to replace it by "typical" attribute values of other cases from the same concept to which the case belongs. When computing attribute-value pair blocks, if $a(x) = $ -, x is included in blocks $[(a,v)]$ for all specified values $v \in V(x,a)$, where

$$V(x,a) = \{a(y) | a(y) \text{ is specified}, y \in U, d(y) = d(x)\} \tag{2.13}$$

Table 2.4: An incomplete data set

	Attributes			Decision
Case id	Wind	Humidity	Temperature	Trip
1	low	low	medium	yes
2	-	low	*	yes
3	*	medium	medium	yes
4	low	?	high	maybe
5	medium	-	medium	maybe
6	medium	high	low	no
7	?	high	*	no
8	low	low	high	no

Table 2.5: All attribute-value blocks of Table 2.4

(a,v)	$[(a,v)]$
$(Wind, low)$	$\{1,2,3,4,8\}$
$(Wind, medium)$	$\{3,5,6\}$
$(Humidity, low)$	$\{1,2,8\}$
$(Humidity, medium)$	$\{3\}$
$(Humidity, high)$	$\{6,7\}$
$(Temperature, low)$	$\{2,6,7\}$
$(Temperature, medium)$	$\{1,2,3,5,7\}$
$(Temperature, high)$	$\{2,4,7,8\}$

Note: $V(2, Wind) = \{low\}$, and $V(5, Humidity) = \emptyset$

Table 2.4 is an example of an incomplete data set, and the attribute-value pair blocks of this decision table is shown in Table 2.5.

In Section 2.2.1 we have defined elementary sets for complete data sets. For incomplete data sets, the idea of elementary sets is extended to the idea of *characteristic sets* (Grzymala-Busse,

2004) denoted by K_A.

$$
K_A(x) = \bigcap_{a \in A} K(x,a)
$$

$$
\text{where} \quad K(x,a) = \begin{cases} [(a,a(x)] & \text{if } a(x) \text{ is specified} \\ U & \text{if } a(x) = ?\text{, or } a(x) = *\text{, or } a(x) = \text{- and } V(x,a) = \emptyset \\ \bigcup_{v \in V(x,a)} [(a,v)] & \text{if } a(x) = \text{- and } V(x,a) \neq \emptyset \end{cases} \tag{2.14}
$$

For the data set of Table 2.4, the characteristic sets are computed as

$$
\begin{aligned}
K_A(1) &= \{1,2,3,4,8\} \cap \{1,2,8\} \cap \{1,2,3,5,7\} = \{1,2\} \\
K_A(2) &= \{1,2,3,4,8\} \cap \{1,2,8\} \cap U = \{1,2,8\} \\
K_A(3) &= U \cap \{3\} \cap \{1,2,3,5,7\} = \{3\} \\
K_A(4) &= \{1,2,3,4,8\} \cap U \cap \{2,4,7,8\} = \{2,4,8\} \\
K_A(5) &= \{3,5,6\} \cap U \cap \{1,2,3,5,7\} = \{3,5\} \\
K_A(6) &= \{3,5,6\} \cap \{6,7\} \cap \{2,6,7\} = \{6\} \\
K_A(7) &= U \cap \{6,7\} \cap U = \{6,7\} \\
K_A(8) &= \{1,2,3,4,8\} \cap \{1,2,8\} \cap \{2,4,7,8\} = \{2,8\}
\end{aligned}
$$

Note that for incomplete data set, the indiscernibility relation is reflexive, but is not necessarily symmetric or transitive. Thus the indiscernibility relation on incomplete data sets is not an equivalence relation, which will make approximations not unique for incomplete data sets.

2.3.2 Lower and Upper Approximations

In Section 2.2.2, we have discussed the lower and upper approximations of concepts from an inconsistent (but complete) data set. For incomplete data sets, the lower and upper approximations could be defined similarly by replacing the elementary sets by characteristic sets in all equations (Grzymala-Busse & Siddhaye, 2004). Consider an incomplete data set, given a set $X \subseteq U$ and a set $B \subseteq A$, the *B-singleton lower approximation*of X, denoted by $\underline{appr}_B^{singleton}(X)$ is defined as follows

$$\underline{appr}_B^{singleton}(X) = \{x | x \in U, K_B(x) \subseteq X\}. \tag{2.15}$$

The *B-subset lower approximations* of X, denoted by $\underline{appr}_B^{subset}(X)$ is defined as follows

$$\underline{appr}_B^{subset}(X) = \cup\{K_B(x) | x \in U, K_B(x) \subseteq X\}. \tag{2.16}$$

The *B-concept lower approximations* of X, denoted by $\underline{appr}_B^{concept}(X)$ is defined as follows

$$\underline{appr}_B^{concept}(X) = \cup\{K_B(x) | x \in X, K_B(x) \subseteq X\}. \tag{2.17}$$

The *B-singleton upper approximation* of X, denoted by $\overline{appr_B}^{singleton}(X)$ is defined as follows

$$\overline{appr}_B^{singleton}(X) = \{x | x \in U, K_B(x) \cap X \neq \emptyset\}. \tag{2.18}$$

The *B-subset upper approximation* of X, denoted by $\overline{appr}_B^{subset}(X)$ is defined as follows

$$\overline{appr}_B^{subset}(X) = \cup\{K_B(x) | x \in U, K_B(x) \cap X \neq \emptyset\}. \tag{2.19}$$

The *B-concept upper approximation* of X, denoted by $\overline{appr}_B^{concept}(X)$ is defined as follows

$$\overline{appr}_B^{concept}(X) = \cup\{K_B(x) | x \in X, K_B(x) \cap X \neq \emptyset\}. \tag{2.20}$$

For example, for the concept $[(Trip, yes)] = \{1,2,3\}$ of Table 2.4, the approximations are

$$\underline{appr}_A^{singleton}(X) = \{1,3\}$$
$$\underline{appr}_A^{subset}(X) = \{1,2,3\}$$
$$\underline{appr}_A^{concept}(X) = \{1,2,3\}$$
$$\overline{appr}_A^{singleton}(X) = \{1,2,3,4,5,8\}$$
$$\overline{appr}_A^{subset}(X) = \{1,2,3,4,5,8\}$$
$$\overline{appr}_A^{concept}(X) = \{1,2,3,8\}$$

Observe that for incomplete data sets, the three definitions (singleton, subset and concept) of lower and upper approximations could give different results, which is different from approximations on inconsistent data sets.

2.3.3 Probabilistic Approximations

Lower and upper approximations are special cases of probabilistic approximations with very small α and with $\alpha = 1$ respectively. Similar to the discussion in 2.2.3, we can also define the general probabilistic approximations for incomplete data sets (Grzymala-Busse et al., 2014). Given a set $X \subseteq U$, and a set $B \subseteq A$, the probabilistic approximations of X with a threshold α, $0 < \alpha \leqslant 1$, are defined as follows

$$appr_{\alpha,B}^{singleton}(X) = \{x | x \in U, P(X|K_B(x]) \geqslant \alpha\} \tag{2.21}$$

$$appr_{\alpha,B}^{subset}(X) = \cup\{K_B(x) | x \in U, P(X|K_B(x) \geqslant \alpha\} \tag{2.22}$$

$$appr_{\alpha,B}^{concept}(X) = \cup\{[K_B(x) | x \in X, P(X|K_B(x)) \geqslant \alpha\} \tag{2.23}$$

where

$$P(X|K_B(x)) = \frac{|X \cap K_B(x)|}{|K_B(x)|}.$$

Table 2.6: Probabilistic approximations for $[(Trip, yes)]$ of Table 2.4

α	$appr_{\alpha,A}(\{1,2,3\}^{singleton})$	$appr_{\alpha,A}(\{1,2,3\}^{subset})$	$appr_{\alpha,A}(\{1,2,3\}^{concept})$
1/3	$\{1,2,3,4,5,8\}$	$\{1,2,3,4,5,8\}$	$\{1,2,3,8\}$
1/2	$\{1,2,3,5,8\}$	$\{1,2,3,5,8\}$	$\{1,2,3,8\}$
2/3	$\{1,2,3\}$	$\{1,2,3,8\}$	$\{1,2,3,8\}$
1	$\{1,3\}$	$\{1,2,3\}$	$\{1,2,3\}$

The probabilistic approximations of concept $[(Trip, yes)] = \{1,2,3\}$ of Table 2.4 are summarized in Table 2.6.

The singleton, subset and concept definitions of probabilistic approximations are also not unique on incomplete data sets. There is no absolute answer to the question "which definition would give the best results for rule induction". However, it is found that in many cases, if the approximation is closer to the original concept, the quality of induced rules is better. Also, it has been proved in (Clark et al., 2014) that for any concept X and a subset $B \subseteq A$,

$$appr_{\alpha,B}^{singleton}(X) \subseteq appr_{\alpha,B}^{subset}(X)$$

$$appr_{\alpha,B}^{concept}(X) \subseteq appr_{\alpha,B}^{subset}(X)$$

Consider that $\underline{appr}_A(X) \subseteq X \subseteq \overline{appr}_A(X)$, if one is using lower approximation for rule induction, it is recommended to start with subset definition, and if one is using upper approximation, concept definition might be a good choice. Singleton definition is not recommended because it may not be locally definable, which will be discussed in the next section.

2.3.4 Local approximations

In Section 2.3.3, we have mentioned that singleton approximations are not always locally definable. Let B be a subset of A, T be a set of attribute-value pairs, i.e. $T = \{t_1, t_2, ...t_n\}, t_i = \{a_i, v_i\}$. If all a_i are distinct and belong to B, T is called a *B-complex*. If $B = A$, we simply say T is a complex. T is nontrivial if $[T] = \bigcap_{i=1}^{n} [(a_i, v_i)] \neq \emptyset$. A set X is *globally definable* if and only if X is a union of some characteristic sets $K_A(x), x \in U$. X is *locally definable* if and only if it is a union of intersections of attribute-value pair blocks from some complexes.

Definability of a set is very important because decision rules and the rule induction algorithms we are going to discuss in Chapter 3 only work on locally definable sets (Grzymala-Busse & Rzasa, 2006).

We also mentioned in 2.3.3 that usually we would like to choose an approximation that is close to the original set. Sometimes subset and concept probabilistic approximations are not satisfying. For example, consider the incomplete data set in Table 2.7, the (a,v) blocks and characteristic sets are

$$\begin{aligned}
[(Age, under-21)] &= \{1,8\} & [(Age, 21-40)] &= \{3,5,7\}\\
[(Age, 41-and-over)] &= \{4\} & [(Education, elementary)] &= \{1,3,6,8\}\\
[(Education, secondary)] &= \{2,3,6,7\} & [(Education, higher)] &= \{3,4,6\}\\
[(Gender, male)] &= \{1,4,8\} & [(Gender, female)] &= \{2,5,6,7,8\}
\end{aligned}$$

$$\begin{aligned}
K_A(1) &= \{1,8\} & K_A(2) &= \{2,6,7\}\\
K_A(3) &= \{3,5,7\} & K_A(4) &= \{4\}\\
K_A(5) &= \{5,7\} & K_A(6) &= \{2,5,6,7,8\}\\
K_A(7) &= \{7\} & K_A(8) &= \{1,8\}
\end{aligned}$$

The subset and concept probabilistic approximations of concept $[(Hobby, fishing)] = \{1,2,3\}$ are listed in Table 2.8. If we restrict to subset and concept probabilistic approximations, we only have two choices on which approximation to use to induce rules for concept $[(Hobby, fishing)]$, either $\{1,2,3,5,6,7,8\}$ or $\{1,8\}$. However, since any locally definable set could be expressed by a rule set, a more promising choice of approximation of concept $[(Hobby, fishing)]$ could be

$$\begin{aligned}
\{1,3,7,8\} &= [(Age, under-21)] \cup ([(Education, secondary)] \cap [(Age, 21-41)])\\
&= \{1,8\} \cup (\{2,3,6,7\} \cap \{3,5,7\}).
\end{aligned}$$

Table 2.7: An incomplete data set

	Attributes			Decision
Case id	Age	Education	Gender	Hobby
1	under-21	elementary	male	fishing
2	?	secondary	female	fishing
3	21-40	*	?	fishing
4	41-and-over	higher	male	hunting
5	21-40	?	female	hunting
6	?	*	female	jogging
7	21-40	secondary	female	jogging
8	under-21	elementary	*	jogging

Table 2.8: Probabilistic approximations for $[(Hobby, fishing)]$ of Table 2.7

α	$appr_{\alpha,A}(\{1,2,3\}^{subset})$	$appr_{\alpha,A}(\{1,2,3\}^{concept})$
1/5	$\{1,2,3,5,6,7,8\}$	$\{1,2,3,5,6,7,8\}$
1/3	$\{1,2,3,5,6,7,8\}$	$\{1,2,3,5,6,7,8\}$
1/2	$\{1,8\}$	$\{1,8\}$
1	$\emptyset$	$\emptyset$

Therefore, we introduce local lower approximation, local upper approximation, and local probabilistic approximation (Grzymala-Busse & Rzasa, 2006) (Clark et al., 2012).

- *B-local lower approximation*, denoted by $L\underline{B}X$, is defined as

$$L\underline{B}X = \cup\{[T] | T \text{ is a nontrivial B-complex of } X, [T] \subseteq X\} \tag{2.24}$$

For example,

$$\begin{aligned} L\underline{(A)}\{6,7,8\} &= ([(Education, elementary)] \cap [(Gender, female)]) \\ &\cup ([(Age, 21-40)] \cap [(Education, secondary)] \cap [(Gender, male)]) \\ &= \{6,7,8\} \end{aligned}$$

- *B-local upper approximation*, denoted by $L\overline{B}X$, is defined as

$$L\overline{B}X = \{[T] | \exists \text{ a family } \mathbb{T} \text{ of nontrivial B-complexes of } X, T \in \mathbb{T}, [T] \cap X \neq \emptyset\} \tag{2.25}$$

  Local upper approximations are not unique, for example, $\{6,7,8\}$ and $\{1,2,6,7,8\}$ are both local upper approximations of concept $[(Hobby, jogging)]$.
  Note:$\{1,2,6,7,8\} = [(Age, under-21)] \cup ([(Education, secondary)] \cap [(Gender, female)])$.

- *B-local probabilistic approximation*, denoted by $appr_{\alpha}^{local}(X)$, is defined as

$$appr_{\alpha,B}^{local}(X) = \cup\{[T] | \exists \text{ a family } \mathbb{T} \text{ of nontrivial B-complexes of } X, T \in \mathbb{T}, P(X|[T] \geqslant \alpha)\} \tag{2.26}$$

  Local probabilistic approximations are also not unique.

The local lower and upper approximations are special cases of the local probabilistic approximation. Local approximations are very promising choices of approximations for rule induction on incomplete data sets. However, since local approximations are not unique, finding the optimal approximation (i.e. the approximation that is closest to the original set) is quite time consuming. One needs to examine all possible combinations of (a, v) pairs, which could not be done in polynomial time. A heuristic approach called *local probabilistic version of MLEM2 algorithm* to calculate a single local probabilistic approximation (sub-optimal) and induce rules from incomplete data sets will be discussed in Chapter 3.

Chapter 3

Rule Induction

3.1 Decision Rules

A *rule* is represented in the LERS format as shown in Equation 3.1

$$\begin{aligned}&number1, number2, number3\\&(a_1, v_1)\&(a_2, v_2)\&(a_3, v_3)\&\ldots\&(a_m, v_m) \rightarrow (d, w)\end{aligned} \tag{3.1}$$

where $a_i \in A$, $v_i \in V(a_i)$, $\forall i = 1, 2, 3, \ldots, m$.

The three numbers are *specificity*, *strength*, and the *rule domain size* respectively, which are important features of a rule (Gryzmala-Busse & Wang, 1996).

- Specificity

  The attribute-value pairs on the left-hand side of a rule are called conditions. The total number of conditions indicates the complexity of a rule, which is called specificity. Specificity is usually a concern in the health and medication field, where conditions are often associated with results of lab tests. The more conditions a rule contains, the more tests a patient need to take to obtain the diagnosis.

- Rule Domain Size

  If the attribute values of a training case x are in accord with all the conditions of a rule r, we say that case x is covered by rule r. The total number of training cases that completely match the left-hand side of a rule is called the rule domain size. (The idea of training cases and testing cases are discussed in Chapter 4.) In general, we want a rule to cover as many cases as possible.

- Strength

  Apart from the rule domain size, we are also interested in the correctness of a rule. The total number of training cases that are correctly classified by a rule is called strength. If all the cases that are covered by a rule r are also correctly classified by r, we say that rule r is consistent with the data set.

In most cases, a single rule only covers a few cases, thus we need a set of rules to cover as many cases in a decision table as possible. A rule set R is complete if and only if for any case $i \in U$, there exists a rule r, such that case i is covered by r. A rule set R is consistent with the decision table if and only if every rule $r \in R$ is consistent with the decision table. The most common rule induction task is to induce a rule set R that is both complete and consistent with the decision table.

3.2 The LEM2 Algorithm

The LEM2 (Learn from examples, module version 2) algorithm is a module of the LERS learning system, which computes a local covering for every concept of the training data set. Then the local coverings are converted to a rule set (Grzymala-Busse, 1992).

In Section 2.3.4, we have defined complex $T = \{t_1, t_2, \ldots, t_n\}$, where t_1, t_2, ..., t_n are attribute-value pairs with distinct attribute names. Observe that T corresponds to the left-hand side of a rule. Let B represent a nonempty concept or an nonempty approximation of a concert. T is a *minimal complex* of B if and only if (1) $[T] = \bigcap_{i=1}^{n} t_i \subseteq B$; (2) There does not exist a subset $T' \subset T$ such that $[T'] \subseteq B$. Let $\mathbb{T} = \{T_1, T_2, \ldots, T_m\}$ be a nonempty set of nontrivial complexes. $\mathbb{T}$ is a *local covering* of B if and only if it satisfies the following conditions:

1. Each member of $T_i \in \mathbb{T}$ is a minimal complex of B, $\forall i = 1, 2, \ldots, m$.

Table 3.1: A decision table for demonstrating LEM2 algorithm

	Attributes			Decision
Case id	Wind	Humidity	Temperature	Trip
1	low	low	medium	yes
2	low	low	low	yes
3	low	medium	medium	yes
4	low	medium	high	maybe
5	medium	low	medium	maybe
6	medium	high	low	no
7	high	high	high	no
8	medium	high	high	no

2. $\bigcup_{T \in \mathbb{T}} T = B$.

3. $\mathbb{T}$ is minimal, i.e., if any T is removed from $\mathbb{T}$, the second condition is violated.

As is mentioned earlier, a local covering could be converted to a set of rules. Condition 1 ensures that every rule in the rule set only covers cases belong to B, and every rule is as simple as possible. Condition 2 guarantees that every case in B is covered by the rule set, and condition 3 indicates that there are no redundant rules. The following example illustrates how to convert a local covering to a set of rules.

Consider a decision table as shown in Table 3.1, a local covering of concept $[(Trip, yes)] = \{1,2,3\}$ is $\mathbb{T} = \{\{Wind, low), (Humidity, low)\}, \{(Humidity, medium), (Temperature, medium)\}\}$, which is corresponding to two rules for concept $[(Trip, yes)]$:

$$(Wind, low) \rightarrow (Trip, yes)$$

$$(Wind, medium) \rightarrow (Trip, maybe)$$

$$(Temperature, high) \rightarrow (Trip, no)$$

The first rule covers case 1 and case 2, and the second rule covers case 3. Both rules are consistent with the data set, and there are no more cases of concept $[(Trip, yes)]$ that are not covered by the two rules. Also, both rules are minimal because no conditions could be removed to maintain the consistency of the rules. Furthermore, there is no redundant rule, both rules are necessary to cover all cases in $[(Trip, yes)]$. Similarly, we can find rules for concepts $[(Trip, maybe)]$ and $[(Trip, no)]$.

Therefore, an effective way to induce a rule set from a decision table is to compute a local covering for every concept. The LEM2 algorithm is designed to find a single local covering for a given concept. The essential part of the LEM2 algorithm is to select the best attribute-value pair t from all available attribute-value pairs. Given a goal G, which is a set of cases that need to be covered, the criteria of selecting t are

1. Select a t such that t is most relevant to G. $\Leftrightarrow |G \cap [t]|$ is maximum.

2. If a tie occurs from criterion 1, select a t such that the conditional probability $P(G|[t])$ is maximized.

Note that the order of the criteria brings a bonus of simplifying the computation of conditional probability. Consider $P(G|[t]) = |G \cap [t]|/|[t]|$, the first criterion makes sure that if multiple t $(t_1, t_2, \ldots, t_k)$ give the same maximum relevance, we must have $|G \cap [t_1]| = |G \cap [t_2]| = \ldots = |G \cap [t_k]|$. Then smaller cardinality size of $[t]$ results in larger value of $P(G|[t])$. Thus we only need to pick up a t with the smallest $|[t]|$, and it will automatically give the largest $P(G|[t])$.

The procedure of LEM2 is described by Algorithm 1. In this algorithm, $|X|$ denotes the cardinality of set X.

Here is an example showing how the LEM2 algorithm would work on Table 3.1. First, all attribute-value pairs are computed.

$$
\begin{aligned}
[(Wind, low)] &= \{1,2,3,4\} & [(Wind, medium)] &= \{5,6,8\} \\
[(Wind, high)] &= \{7\} & [(Humidity, low)] &= \{1,2,5\} \\
[(Humidity, medium)] &= \{3,4\} & [(Humidity, high)] &= \{6,7,8\} \\
[(Temperature, low)] &= \{2,6\} & [(Temperature, medium)] &= \{1,3,5\} \\
[(Temperature, high)] &= \{7,8\} & &
\end{aligned}
$$

Algorithm 1 LEM2 Procedure

Input: a nonempty set B, which is a concept, or some approximation of a concept

Output: a single local covering $\mathbb{T}$ of set B

```
 1: begin procedure
 2: $G := B$;
 3: $\mathbb{T} := \emptyset$;
 4: while $G \neq 0$ do
 5:    $T := \emptyset$;
 6:    $T(G) := \{t|[t] \cap G \neq \emptyset\}$;
 7:    while $T = \emptyset$ or $[T] \not\subseteq B$ do
 8:       select a pair $t \in T(G)$ such that $|t \cap G|$ is maximum; if a tie occurs, select a pair $t \in T(G)$
          with the smallest cardinality of $[t]$; if another tie occurs, select first pair;
 9:       $T := T \cup \{t\}$;
10:       $G := [t] \cap G$;
11:       $T(G) := \{t|[t] \cup G \neq \emptyset\}$;
12:       $T(G) := T(G) - T$;
13:    end while
14:    for each $t \in T$ do
15:       if $[T - \{t\}] \subseteq B$ then
16:          $T := T - \{t\}$;
17:       end if
18:    end for
19:    $\mathbb{T} := \mathbb{T} \cup T$;
20:    $G := B - \bigcup_{T \in \mathbb{T}} T$;
21: end while
22: for each $T \in \mathbb{T}$ do
23:    if $\bigcup_{S \in \mathbb{T} - \{T\}} [S] = B$ then
24:       $\mathbb{T} := \mathbb{T} - \{T\}$;
25:    end if
26: end for
27: end procedure
```

Let us induce rules for concept $[(Trip, yes)] = \{1,2,3\}$. Initially, $B = G = \{1,2,3\}$. The set of all relevant attribute-value pairs is $T(G) = \{(Wind, low), (Humidity, low), (Humidity, medium), (Temperature, low), (Temperature, medium)\}$. Among all $t \in T(G)$, $(Wind, low)$ yields to the maximum $|t \cap G|$. Therefore $(Wind, low)$ is selected and $T = \{(Wind, low)\}$. Since $T \nsubseteq B$, we have to go through another iteration of the inner while loop (lines 7 to 13 of Algorithm 1) to select the next attribute-value pair with $G = \{1,2,3\}$ and $T(G) = \{(Humidity, low), (Humidity, medium), (Temperature, low), (Temperature, medium)\}$.

In the second iteration, both $(Humidity, low)$ and $(Temperature, medium)$ give the maximum $|t \cap G| = 2$, and both attribute-pairs have the same cardinality of 2. We select $(Humidity, low)$ by the heuristic of choosing first pair, then T is updated to $\{(Wind, low), (Humidity, low)\}$. The inner while loop is terminated because $[T] = \{1,2\} \subseteq B$.

Next, we need to check if T is minimal by running the first for loop (lines 14 to 18 of Algorithm 1), and find that no pairs could be dropped from T. As a result, $\{(Wind, low), (Humidity, low)\}$ is the first minimal complex of B. Then G is updated to $\{3\}$ and we identify the second minimal complex, which is $\{(Humidity, medium), (Temperature, medium)\}$. The new G is now an empty set. As we come out from the outer while loop, we need to examine if $\mathbb{T}$ is minimal by running the second for loop (lines 22 to 27 of Algorithm 1). It is found that both complexes should be kept. Eventually, the local covering of concept $[(Trip, yes)] = \{1,2,3\}$ is

$$\mathbb{T} = \{\{(Wind, low), (Humidity, low)\}, \{(Humidity, medium), (Temperature, medium)\}\}.$$

3.3 Discretization and MLEM2 Algorithm

Consider some attributes like height, age, interest rate, speed and so on, usually the values of these attributes are continuous (numerical). Numerical attributes need to be discretized for rule induction. In general, there are two ways to handle data sets with numerical attributes. The first way is to perform a preliminary discretization step, and then apply rule induction algorithms. Frequently used discretization methods include dominant attribute approach, multiple scanning approach, and many other approaches. The second way is to include the discretization procedure into the rule induction procedure. In this project, we focus on the second approach, and adapt a modified version of the LEM2 algorithm named MLEM2 for rule induction with data sets containing numerical attributes (Grzymala-Busse, 2008).

Consider an attribute a with numerical values from an interval $[b,c]$, the discretization procedure is to divide interval $[b,c]$ into subintervals $[b_1,b_2)$, $[b_2,b_3)$, ..., $[b_{m-1},b_m]$ where $b_1 = b$, $b_m = c$, and $b_{i-i} < b_i, \forall i = 1,2,3,\dots,m$. The numbers b_2, b_3, ..., b_{m-1} are called *cut points*. We use "$b_{i-1}..b_i$" to represent the interval $[b_{i-1},b_i)$ for $i < m$, and use "$b_{m-1}..b_m$" to represent the interval $[b_{m-1},b_m]$.

The MLEM2 algorithm is analogous to LEM2 in searching for best attribute-value pairs and identifying a local covering for each concept. The main difference between MLEM2 and LEM2 algorithms is the calculation of candidates t_1, t_2, ..., t_n (also called elementary conditions) in the searching space. In LEM2 the elementary conditions are (a,v) pairs. In MLEM2, the elementary conditions for symbolic attributes are also (a,v) pairs, while the elementary conditions for numerical attributes are presented as $(a,b_{i-1}..b_i)$, which means the value of attribute a is in between of b_{i-1} and b_i. Another difference between MLEM2 and LEM2 is the updating of $T(G)$ at the end of each iteration of the inner while loop. In LEM2, $T(G) := T(G) - T$, which means only the selected elementary conditions are deleted from $T(G)$. While in MLEM2, $\forall t = (a_t,x..y) \subseteq T$, if a_t is an numerical attribute, the set $\{t'|t' = (a_t,u..v), t' \subseteq T(G), x..y$ and $u..v$ are disjoint , or $x..y \subseteq u..v\}$ is also subtracted from $T(G)$. This is because selecting t' won't bring any benefit in making $T(G)$ a subset of B.

Let us show how MLEM2 works using the following example. The decision table is shown in Table 3.2. There is one numerical attribute (Temperature) and two symbolic attributes (Headache and Cough). First, we calculate all the cut points for attribute "Temperature" as follows:

1. Sort the distinct numerical values of "Temperature" in ascending order.

Table 3.2: A decision table containing numerical for demonstrating MLEM2 attributes

	Attributes			Decision
Case id	Temperature	Headache	Cough	Flu
1	101.6	yes	yes	yes
2	102.4	yes	no	yes
3	100.8	no	yes	yes
4	101.6	yes	no	no
5	98.8	yes	no	no
6	98.0	no	no	no
7	102.4	no	yes	no

98.0 98.8 100.8 101.6 102.4

2. Calculate the average of two adjacent numbers, which are the cut points.

$$(98.0+98.8)/2 = 98.4 \qquad (98.8+100.8)/2 = 99.8$$

$$(100.8+101.6)/2 = 101.2 \qquad (101.6+102.4)/2 = 102.0$$

Therefore, cut points are 98.4, 99.8, 101.2, and 102.0.

Second, all elementary conditions are determined and the blocks of elementary conditions are computed. For symbolic attributes "Headache" and "Cough", the elementary conditions are attribute-value pairs. For numerical attribute "Temperature", we create two elementary conditions for every cut point b_i as $(Temperature, start\text{-}number..b_i)$ and $(Temperature, b_i..end\text{-}number)$, where *start-number* is the smallest numerical value and *end-number* is the largest numerical value of the attribute. The blocks of such elementary conditions are calculated as

$$[(a, start\text{-}number..b_i)] = \{x | start\text{-}number \leqslant a(x) < b_i\}$$

$$[(a, b_i..end\text{-}number)] = \{x | b_i < a(x) \leqslant end\text{-}number\}$$

Let us induce rules for concept $[(Flu, yes)] = \{1,2,3\}$. We need to identify a local covering for {1,2,3}, and the criteria of selecting the best elementary condition t are the same as in the

LEM2 algorithm. Initially, $B = G = \{1,2,3\}$. All the available elementary conditions are listed as follows.

$$\begin{array}{rl}
[(Temperature, 98.0..98.4)] = \{6\} & [(Temperature, 98.4..102.4)] = \{1,2,3,4,5,7\} \\
[(Temperature, 98.0..99.8)] = \{5,6\} & [(Temperature, 99.8..102.4)] = \{1,2,3,4,7\} \\
(Temperature, 98.0..101.2) = \{3,5,6\} & [(Temperature, 101.2..102.4)] = \{1,2,4,7\} \\
[(Temperature, 98..102.0)] = \{1,3,4,5,6\} & [(Temperature, 102.0..102.4)] = \{2,7\} \\
[(Headache, yes)] = \{1,2,4,5\} & [(Headache, no)] = \{3,6,7\} \\
[(Cough, yes)] = \{1,3,7\} & [(Cough, no)] = \{2,4,5,6\}
\end{array}$$

The set of all relevant attribute-value pairs is

$$\begin{aligned}
T(G) = \{&(Temperature, 98.4..102.4), (Temperature, 99.8..102.4), \\
&(Temperature, 98..101.2), (Temperature, 101.2..102.4), \\
&(Temperature, 98..102.0), (Temperature, 102.0..102.4), \\
&(Headache, yes), (Headache, no), (Cough, yes), (Cough, no)\}
\end{aligned}$$

.

Among all $t \in T(G)$, $(Temperature, 98.4..102.4)$ and $(Temperature, 99.8..102.4)$ both give the maximum $|t \cap T(G)|$. Since $|[(Temperature, 99.8..102.4)]|$ is smaller than $|[(Temperature, 98.4..102.4)]|$, the first elementary condition t selected by MLEM2 algorithm is $(Temperature, 99.8..102.4)$. Because $[(Temperature, 99.8..102.4)] = \{1,2,3,4,7\} \nsubseteq B$, we need to select the next elementary condition t. To update $T(G)$, $(Temperature, 99.8..102.4)$ should be deleted because it is already selected. $(Temperature, 98..98.4)$ should also be deleted since the two intervals $[99.8, 102.4]$ and $[98, 98.4)$ have no intersection.

The next selected elementary condition is $(Cough, yes)$. Since $[(Temperature, 99.8..102.4)] \cap [(Cough, yes)] = \{1,3,7\} \nsubseteq B$, we need to select the third elementary condition, which is

$(Temperature, 98..102.0)$. The first minimal complex of B is

$$T = \{(Temperature, 99.8..102.4), (Cough, yes), (Temperature, 98..102.0)\}.$$

T is minimal, all the three elementary conditions are necessary. However, $(Temperature, 99.8..102.4)$ and $(Temperature, 98..102.0)$ could be combined to $(Temperature, 99.8..102.0)$. The procedure of combining elementary conditions for numerical attributes is called *merging intervals*.

The second minimal complex of B could be identified in a similar way, and finally we get a local covering of B as

$$\begin{aligned}\mathbb{T} = \{&\{(Temperature, 99.8..102.0), (Cough, yes)\},\\ &\{(Temperature, 102.0..102.4), (Headache, yes)\}\}.\end{aligned}$$

The rules for $[(Flu, yes)]$ are

$2, 2, 2$

$(Temperature, 99.8..102) \& (Cough, yes) \rightarrow (Flu, yes)$

$2, 1, 1$

$(Temperature, 102.0..102.4) \& (Headache, yes) \rightarrow (Flu, yes)$

So far we only discussed how to induce rules with LEM2 and MLEM2 from consistent data sets. In Chapter 2, we discussed approximations of concepts. For inconsistent data sets and incomplete data sets, we could apply LEM2 and MLEM2 algorithms by replacing the original concept by some approximation (e.g. lower approximation, probabilistic approximation, etc.) of the concept. Rules that are induced with lower approximations are called *certain rules*, and those induced with upper approximations are called *possible rules*.

3.4 Local Probabilistic version of MLEM2 Algorithm

We introduced local approximations for incomplete data sets. It has been mentioned that local approximations are quite promising. However, one needs to check all possible combinations of elementary conditions t to find the optimal approximation, the computation complexity is exponential to the total number of t. The local probabilistic version of MLEM2 algorithm is a heuristic approach which identifies one single local approximation and induce rules at the same time (Grzymala-Busse & Rzasa, 2010). Note that the identified local approximation may not be the optimal approximation, but we are able to finish calculation in polynomial time.

The procedure of the local probabilistic version of MLEM2 algorithm (LocalPrMLEM2) is analogous to MLEM2 except for the following modifications.

1. In MELM2, we check if $[T] \subseteq B$ each time an elementary condition is selected, and B remains the same during the procedure of rule induction. While in LocalPrMLEM2, we introduce a set D, and check if $[T] \subseteq D$. Originally, $D = X$, and D is updated during the rule induction procedure. X is some concept.

2. For incomplete data sets, one may not get $[T] \subseteq D$ when one runs out of available elementary conditions. In this case, the conditional probability $P(X|[T]) = |X \cap [T]|/|[T]|$ is calculated. If $P(X|[T] \geqslant \alpha$, then $D = D \cup [T]$, indicating that all the cases belong to $[T]$ are included into the approximation. Otherwise, we introduce a set of "junk complexes" $\mathbb{J}$, and update $\mathbb{J} := \mathbb{J} \cup [T]$, which means all the cases belong to $[T]$ are excluded from the approximation, and T is excluded from the local covering $\mathbb{T}$.

3. When the algorithm terminates, $\bigcup\limits_{S \in \mathbb{T}} [S]$ represents a single local approximation of X, and $\mathbb{T}$ could be converted into rules.

The full description of LocalPrMLEM2 is shown in Algorithm 2. If the input parameter α equals to 1, the algorithm induces certain rules. If α is very small, possible rules are induced.

To demonstrate how LocalPrMLEM2 works, let us induce rules for $[(Hobby, fishing)] = \{1,2,3\}$ of Table 3.3 with $\alpha = 0.5$. Initially $G = D = X = \{1,2,3\}$ and $\mathbb{J} = \emptyset$. After selecting $(Eduction,$

Algorithm 2 Local Probabilistic version of MLEM2 Procedure

Input: a set X (a subset of U) and a parameter α

Output: a single local probabilistic covering $\mathbb{T}$ of set X

```
1:  begin procedure
2:  $G := X$;   $D := X$;
3:  $\mathbb{T} := \emptyset$;   $\mathbb{J} := \emptyset$;
4:  while $G \neq 0$ do
5:     $T := \emptyset$;   $T_s := \emptyset$;   $T_n := \emptyset$;
6:     $T(G) := \{t | [t] \cap G \neq \emptyset\}$;
7:     while ($T = \emptyset$ or $[T] \not\subseteq D$) and $T(G) \neq \emptyset$ do
8:        select a pair $t = (a_t, v_t) \in T(G)$ such that $|t \cap G|$ is maximum; if a tie occurs, select a pair
          $t \in T(G)$ with the smallest cardinality of $[t]$; if another tie occurs, select first pair;
9:        $T := T \cup \{t\}$;
10:       $G := [t] \cap G$;
11:       $T(G) := \{t | [t] \cup G \neq \emptyset\}$;
12:       if $a_t$ is symbolic{ let $V_{a_t}$ be the domain of $a_t$} then
13:          $T_s := T_s \cup \{(a_t, v) | v \in V_{a_t}\}$;
14:       else
15:          {$a_t$ is numerical, let $t = (a_t, u..v)$} $T_n := T_n \cup \{(a_t, x..y) |$disjoint $x..y$ and $u..v\} \cup$
             $\{(a_t, x..y) | x..y \supseteq u..v\}$;
16:       end if
17:       $T(G) := T(G) - (T_s \cup T_n\}$;
18:    end while
19:    if $P_T(X|[T]) \geqslant \alpha$ then
20:       $D := D \cup [T]$;   $\mathbb{T} := \mathbb{T} \cup \{T\}$;
21:    else
22:       $\mathbb{J} := \mathbb{J} \cup \{T\}$;
23:    end if
24:    $G := D - \bigcup_{S \in \mathbb{T} \cup \mathbb{J}} [S]$;
25: end while
26: for each $T \in \mathbb{T}$ do
27:    for each numerical attribute $a_t$ with $(a_t, u..v) \in T$ do
28:       while ($T$ contains at least two different pairs $(a_t, u..v)$ and $(a_t, x..y)$ with the same numer-
          ical attribute $a_t$) do
29:          replace these two pairs with a new pair $(a_t,$common part of $(u..v)$ and $(x..y))$;
30:       end while
31:    end for
32:    for each $t \in T$ do
33:       if $[T - \{t\}] \subseteq D$ then
34:          $T := T - \{t\}$;
35:       end if
36:    end for
37: end for
38: for each $T \in \mathbb{T}$ do
39:    if $\bigcup_{s \in (\mathbb{T} - \{T\})} [S] = \bigcup_{S \in \mathbb{T}} [S]$ then
40:       $\mathbb{T} := \mathbb{T} - \{T\}$;
41:    end if
42: end for
43: end procedure
```

Table 3.3: An incomplete data set for demonstrating the local probabilistic version of MLEM2 algorithm

	Attributes			Decision
Case id	Age	Education	Gender	Hobby
1	under-21	elementary	male	fishing
2	?	secondary	female	fishing
3	21-40	*	?	fishing
4	41-and-over	higher	male	hunting
5	21-40	?	female	hunting
6	?	*	female	jogging
7	21-40	secondary	female	jogging
8	under-21	elementary	*	jogging

$elementary)$, $(Age, under-21)$ and $(Gender, male)$ successively, we run out of available elementary conditions and $[T] = \{1,8\} \nsubseteq D$. Therefore, we need to calculate the conditional probability, which is $P(X|[T]) = |\{1,2,3\} \cap \{1,8\}|/|\{1,8\}| = 0.5$. Since $P(X|[T]) \geqslant \alpha$, we update $D := D \cup [T] = \{1,2,3,8\}$, and

$$\mathbb{T} = \{\{(Education, elementary), (Age, under-21), (Gender, male)\}\}.$$

The new goal G becomes {2,3}, and in a similar way we obtain the second T as $\{(Education, secondary), (Age, 21-40)\}$ and update D to $\{1,2,3,7,8\}$.

Next, we continue to identify another T for the goal $G = \{2\}$. This time we end up with $T = \{(Education, secondary), (Gender, female)\}$. Since $[T] = \{2,6,7\} \nsubseteq D$, we calculate the conditional probability $P(X|[T]) = |\{2\}|/|\{2,6,7\}| = 0.333$. Because $P(X|[T]) < \alpha$, T is put to $\mathbb{J}$ and $[T] = \{2,6,7\}$ is subtracted from G, which makes the new G to be an empty set. Then we run the for loops (lines 26 to 42 of Algorithm 2) to merge intervals and remove redundancy.

Eventually the approximation of concept $[(Hobby, fishing)]$ is $\bigcup_{S \in \mathbb{T}} [S] = \{1,3,7,8\}$, the local covering is

$$\mathbb{T} = \{\{(Age, under-21)\}, \{(Education, secondary), (Age, 21-41)\}\},$$

and the rules for concept $[(Hobby, fishing)]$ with conditional probability larger than or equal to 0.5 are

$1, 1, 2$

$(Age, under-21) \rightarrow (Hobby, fishing)$

$2, 1, 2$

$(Education, secondary) \& (Age, 21-41) \rightarrow (Hobby, fishing)$

Chapter 4

Experiments

The objective of this project is to investigate which type of data sets, inconsistent or incomplete, would have more negative effects on rule induction. In order to answer this question, we conducted experiments to induce rules from inconsistent data sets and incomplete data sets, classify testing cases with induced rules, and measure the error rate of classification. The data set from which rules are induced is called *training* set, and the data set on which the accuracy of classification with the induced rules is measured is called *testing* set. The method of classifying unseen cases is introduced in Section 4.1, the idea of ten-fold cross validation is discussed in Section 4.2. Section 4.3 described the set up and procedures of the experiments, and the results of the experiments are presented in Section 4.4.

4.1 Classification System

In this project, we followed the classification method of the LERS learning system (Gryzmala-Busse & Wang, 1996). The determination of which concept should an unseen case be classified to depends on three factors: *strength*, *support*, and *partial matching factor*. Strength is defined in Chapter 3 as the total number of training cases that are correctly classified by the rule. For concept *C*, the support of *C* is defined as the sum of products of strength and specificity for all the complete matching rules that refer to concept C.

$$Support(C) = \sum_{\text{completely matching rule } r \text{ describing } C} Strength(r) \times Specificity(r) \tag{4.1}$$

If several rules $r_1, r_2, r_3, \ldots, r_n$ completely match a case x, and all refer to the same concept C, x will be classified as a member of C. However, if the complete matching rules refer to different concepts, supports of every concept are calculated. The concept C with the largest support will be the winner, and x will be classified to C.

If complete matching is impossible for a case x, we will search for all partially matching rules. Partial matching means the attribute values of a case only match some of the conditions of a rule. Partial matching factor (*pmf*) is defined as the number of matched conditions divided by the total number of conditions. When complete matching is impossible, the support of concept C is calculated as

$$Support(C) = \sum_{\text{partially matching rule } r \text{ describing } C} pmf(r) \times Strength(r) \times Sepcificity(r) \tag{4.2}$$

Again, case x is classified to the concept C with the largest support.

For example, consider the following two cases and four rules,

Caseid	*Temperature*	*Headache*	*Cough*	*Flu*
1	*high*	*yes*	*yes*	*yes*
2	*high*	*no*	*no*	*no*

$2, 12, 15$

$(Temperature, high)\&(Headache, yes) \rightarrow (Flu, yes) \qquad (r_1)$

$2, 14, 20$

$(Temperature, high)\&(Cough, yes) \rightarrow (Flu, yes) \qquad (r_2)$

$2, 16, 18$

$(Temperature, high)\&(Cough, yes) \rightarrow (Flu, no) \qquad (r_3)$

$3, 20, 25$

$(Temperature, normal)\&(Headache, no)\&(Cough, no) \rightarrow (Flu, no) \qquad (r_4)$

For case 1, there are three completely matching rules r_1, r_2, and r_3. Rules r_1 and r_2 would classify case 1 to concept $[(Flu, yes)]$, and the support of $[(Flu, yes)]$ is $12 \times 2 + 14 \times 2 = 52$. Rule r_3 would classify case 1 to $[(Flu, no)]$, and the support of this concept is $16 \times 2 = 32$. Therefore case 1 is classified as a member of $[(Flu, yes)]$.

For case 2, there is no complete matching. However, we are able to find partial matching rules r_1, r_3 and r_4, with partial matching factors 0.5, 0.5 and 0.667 respectively. Rule r_1 refers to concept $[(Flu, yes)]$ and the support is $12 \times 2 \times 0.5 = 12$. Rules r_3 and r_4 refer to concept $[(Flu, no)]$ and the support is $16 \times 2 \times 0.5 + 20 \times 3 \times 0.667 = 56$. Therefore case 2 would be classified as a member of $[(Flu, no)]$.

4.2 Ten-fold Cross Validation

Ten-fold cross validation is commonly accepted as a standard method to measure the accuracy of classification with a rule set. In ten-fold cross validation, the data set is randomly shuffled. The re-ordered cases are divided into ten mutually disjoint subsets, and each subset contains roughly 10% cases. For each subset S_i, the other nine subsets $S_1, S_2, \ldots, S_{i-1}, S_{i+1}, \ldots, S_{10}$ are combined as the training set for rule induction, and S_i is used as the testing set. The rule set induced from the training set is used to classify every case from the testing set, and the error rate is measured as

$$error\_rate = \frac{\text{number of incorrectly classified cases}}{\text{number of total cases}} \times 100\% \tag{4.3}$$

In total, ten runs of rule induction and classification are conducted and the average error rate is used to measure the accuracy of classification.

4.3 Experiment Procedure

We implemented the conventional MLEM2 algorithm and the local probabilistic version of MLEM2 algorithm for rule induction. We also developed a program to classify cases with induced rule sets and measure the classification error rate. Experiments were conducted on eight inconsistent data sets and eight incomplete data sets. The data sets were obtained by making modifications on a data set called *Iris*. The original *Iris* data sets contains 150 cases, 5 numerical attributes, and 3 concepts. There are no missing attribute values or inconsistent cases in the original *Iris* data set. Incomplete data sets were created by replacing a certain percentage of existing attribute values to lost values (denoted by ?). We obtained eight incomplete data sets, with 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35% missing attribute values respectively. To set up inconsistent data sets, we discretize the data set to a point such that there were some inconsistent cases. The inconsistent data sets were discretized using a method based on agglomerative cluster analysis. Eight inconsistent data sets were created in this way, with 6, 16, 26, 39, 49, 59, 80, and 92 inconsistent cases respectively.

The procedure of the experiments is shown in Figure 4.1. For every inconsistent experiment data set, we applied MLEM2 algorithm to induce two sets of rules: certain rule set and possible rule set (using lower approximation and upper approximation). For every incomplete experiment data set, the local probabilistic version of MLEM2 algorithm was applied with $\alpha = 1$ to induce certain rules, and with $\alpha = 0.0001$ to induce possible rules. Ten-cross validation was conducted

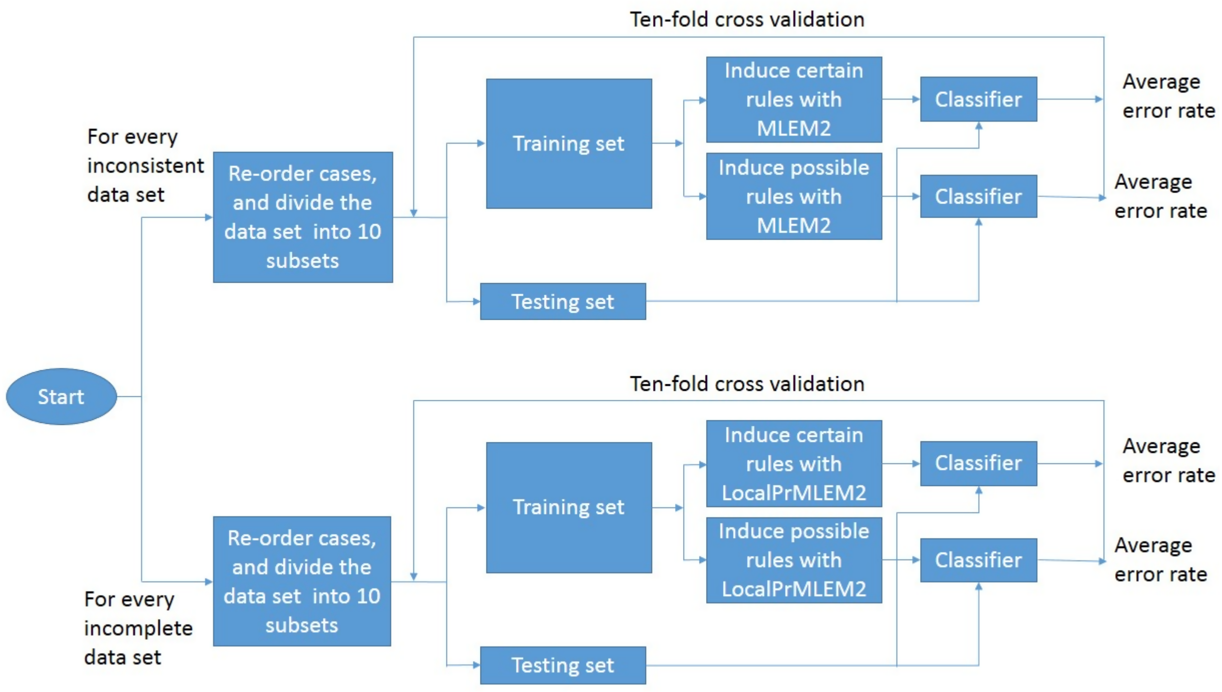
Ten-fold cross validation
For every inconsistent data set
Re-order cases, and divide the data set into 10 subsets
Training set
Induce certain rules with MLEM2
Induce possible rules with MLEM2
Classifier
Classifier
Average error rate
Average error rate
Testing set
Start
Ten-fold cross validation
For every incomplete data set
Re-order cases, and divide the data set into 10 subsets
Training set
Induce certain rules with LocalPrMLEM2
Induce possible rules with LocalPrMLEM2
Classifier
Classifier
Average error rate
Average error rate
Testing set

Figure 4.1: Experiment procedure

to measure the average error rates of classifications with certain and possible rule sets separately. In this project, we exclude specificity (set specificity to 1) in the classification procedure, because this set up usually gives better results from previous experience (Grzymala-Busse & Zou, 1998).

4.4 Experiment Results

The average error rates of ten-fold cross validation for inconsistent data sets are shown in Table 4.1, the percentage of inconsistency of an inconsistent data set was calculated as the ratio of the number of inconsistent cases and the number of total cases. From the table, it is observed that although there are some exceptions, the average error rates tends to be higher as the percentage of inconsistency gets larger. This conclusion holds for both certain rules and possible rules. This is because the if two inconsistent cases are covered by a rule, no matter what concept the rule refers to, at least one case is incorrectly classified. Therefore when there are many inconsistent cases, the error rate of classification is expected to be higher.

Table 4.1: Average error rates of inconsistent data sets

	Avg. error rate (%)	
Percentage of inconsistency (%)	Certain rules	Possible rules
4.0	4.67	5.33
10.7	6.67	6.67
17.3	12.67	6.00
26.0	8.67	6.67
32.7	25.33	25.33
39.3	26.00	16.67
53.3	26.00	26.00
61.3	34.00	36.67

Table 4.2: Average error rates of incomplete data sets

	Avg. error rate (%)	
Percentage of missing attribute values (%)	Certain rules	possible rules
0.0	4.67	4.67
5.0	8.67	8.00
10.0	7.80	6.67
15.0	7.33	6.00
20.0	11.33	10.00
25.0	14.67	11.20
30.0	15.33	15.33
35.0	14.67	13.33

The average error rates for incomplete data sets are shown in Table 4.2. The similar tendency is observed that as the percentage of missing attribute values gets larger, the error rates for both certain rules and possible rules would be higher. It makes sense because with a lot of missing attribute values, useful information is lost, and the lack of information results in lower quality of rule induction.

In Figure 4.2, the blue curve shows the average error rates of certain rules induced from inconsistent data sets versus the percentage of inconsistency, and the yellow curve represents the average error rate of certain rules induced from incomplete data sets versus the percentage of missing attribute values. To compare the two curves, we concentrate on 4% to 40% inconsistency or missing

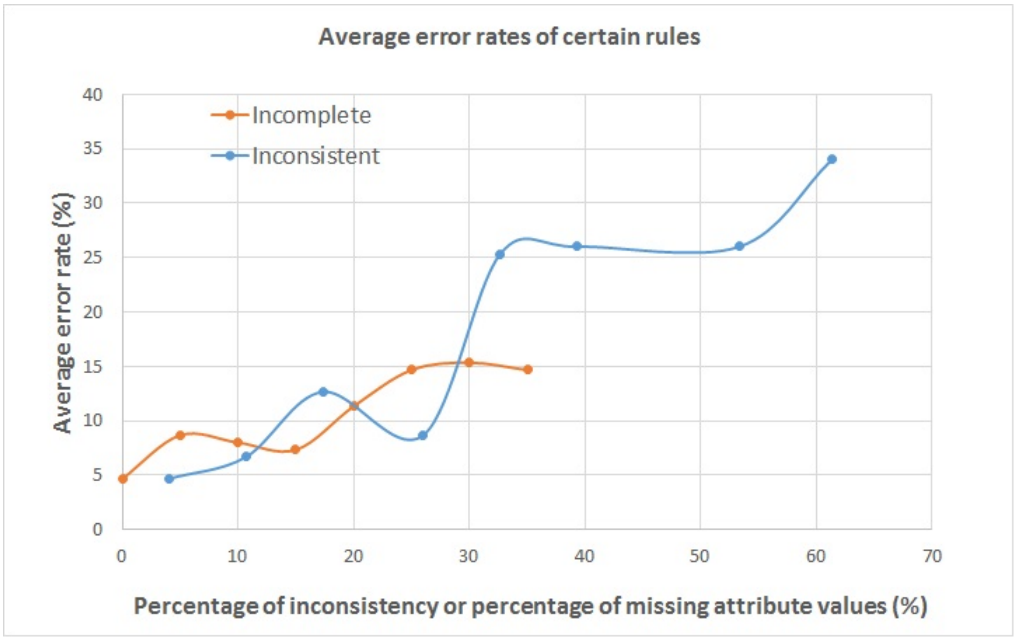

Figure 4.2: Comparison of error rates of certain rules for inconsistent and incomplete data sets

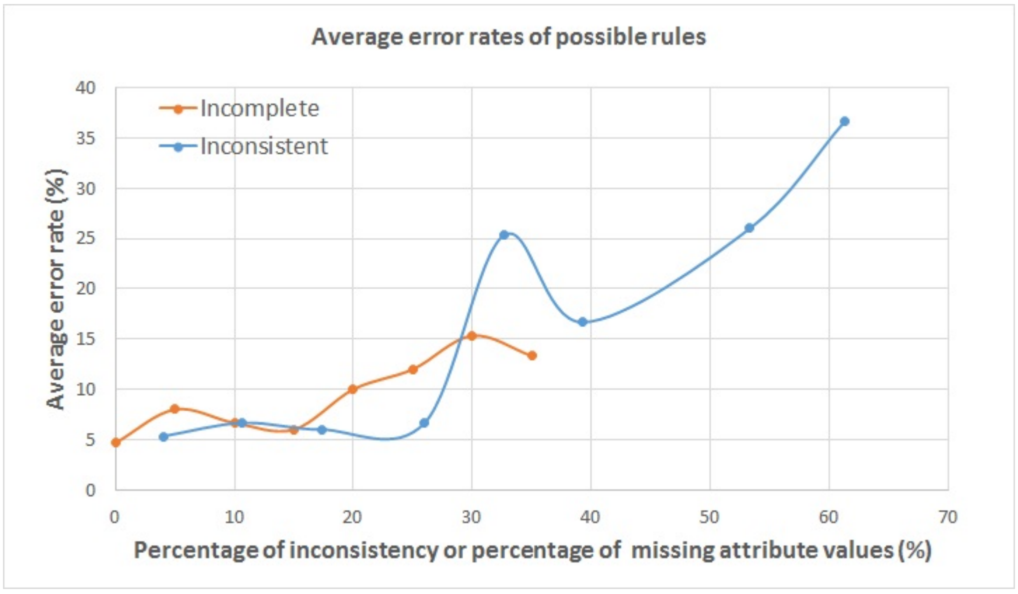

Figure 4.3: Comparison of error rates of possible rules for inconsistent and incomplete data sets

attribute value. In this interval, the two curves intersect three times, and it is hard to say in general that certain rules induced from which data sets would have higher error rates.

A comparison of average error rates of possible rules for inconsistent and incomplete data sets is shown in Figure 4.3. The blue curve shows the results related to inconsistent data sets and the results of incomplete data sets are plotted as the yellow curve. We also concentrate on 4% to 40% inconsistency or missing attribute values. It seems that possible rules induced from incomplete data sets have higher error rates than those induced from inconsistent data sets, however, the Mann-Whitney nonparametric test shows there is no significant difference.

To check if there is significant difference in error rates between inconsistent and incomplete data sets, the Mann-Whitney nonparametric test was conducted (Mann & Whitney, 1947). Since the error rates get higher when the percentage of inconsistency or the percentage of missing attribute values is larger, the result would be more precise if we only compare error rates with similar percentage of inconsistency and percentage of missing attribute values. Therefore, we focused on the data of error rates from the interval of 4% to 40% inconsistency or missing attribute values.

Table 4.3 shows the ranking details of Mann-Whitney test to compare inconsistent data sets with incomplete data sets for certain rules. In this test,

$$n_1 = 6 \quad n_2 = 7$$

$$R_1 = \sum \text{ranks in group } 1 = 43.5 \quad R_2 = \sum \text{ranks in group } 2 = 47.5$$

$$U_1 = n_1 n_2 + n_1(n_1+1)/2 - R_1 = 19.5 \quad U_2 = n_1 n_2 + n_2(n_2+1)/2 - R_2 = 22.5$$

$$U = min\{U_1, U_2\} = 19.5$$

The critical value of two-sided level with $n_1 = 6$, $n_2 = 7$, and 0.05 significance level is 6. Since $19.5 > 6$, there is no significant difference between inconsistent data sets and incomplete data sets in terms of average error rates of certain rules.

Similarly, the ranks of Mann-Whitney test for the two types of data sets for possible rules are

Table 4.3: Ranking details of Mann-Whitney test to compare the error rates of certain rules induced from inconsist and incomplete data sets

Error rates of inconsistent data sets	Rank	Error rates of incomplete data sets	Rank
4.67	1	8.67	5.5
6.67	2	8.00	4
12.67	10	7.33	3
8.67	5.5	11.33	7
25.33	12	14.67	8.5
26.00	13	15.33	11
		14.67	8.5

Table 4.4: Ranking details of Mann-Whitney test to compare the error rates of possible rules induced from inconsist and incomplete data sets

Error rates of inconsistent data sets	Rank	Error rates of incomplete data sets	Rank
5.33	1	8	7
6.67	5	6.67	5
6	2.5	6	2.5
6.67	5	10	8
25.33	13	12.0	9
16.67	12	15.33	11
		13.33	10

shown in Table 4.4. In this test,

$$n_1 = 6 \quad n_2 = 7$$

$$R_1 = \sum \text{ranks in group } 1 = 38.5 \quad R_2 = \sum \text{ranks in group } 2 = 52.5$$

$$U_1 = n_1 n_2 + n_1(n_1+1)/2 - R_1 = 24.5 \quad U_2 = n_1 n_2 + n_2(n_2+1)/2 - R_2 = 17.5$$

$$U = min\{U_1, U_2\} = 17.5$$

The critical value of two-sided level with $n_1 = 6$, $n_2 = 7$, and 0.05 significance level is 6. Since $17.5 > 6$, we do not have sufficient evidence to conclude which kind of data sets has higher error rates for possible rules.

Chapter 5

Summary

5.1 Conclusions

In this project, we investigated which kind of imperfect data sets, inconsistent or incomplete, would have worse performance on the quality of induced certain and possible rules. To find answers to this problem, we implemented the MLEM2 algorithm and the local probabilistic version of MLEM2 algorithm. Eight inconsistent data sets with different levels of inconsistency, and eight incomplete data sets with different percentages of missing attribute values were created from *Iris* data set. For inconsistent data sets, we induced certain and possible rules with MLEM2 algorithm, and for incomplete data sets, certain and possible rules were induced with the local probabilistic version of MLEM2 algorithm. We also implemented a rule checker program to classify cases from testing data set and measure the classification error rate.

With the rule induction program and rule checker program, we conducted ten-fold cross validation for every data set, and the average error rate was used to measure the quality of induced rule sets. The experiment results were further analyzed with Mann-Whitney nonparametric tests to compare, separately for certain and possible rules, incompleteness with inconsistency.

Three main conclusions could be drawn from the experiment results.

1. For inconsistent data sets, although there are some exceptions, the average error rates for

both certain and possible rules tend to be higher as the percentage of inconsistency gets larger. The similar tendency was also seen on incomplete data sets. As the percentage of missing attribute values gets larger, the error rates for both certain rules and possible rules would be higher.

2. The Mann-Whitney nonparametric test shows that there is no significant difference between inconsistent data sets and incomplete data sets in terms of average error rates for certain rules. For possible rule sets induced from inconsistent data sets and incomplete data sets, we do not have sufficient evidence to conclude which kind of data sets has higher error rates either.

3. Therefore, we may conclude that the experiment results of this project indicate that inconsistent data sets and incomplete data sets have similar effect on the quality of rule induction.

5.2 Future Work

In this project, we only considered lost values for incomplete data sets. Future work could be conducted to include all three types of missing attribute values (lost values, "do not care" values, and attribute-concept values). In addition, all the experiment data sets were obtained by modifying *Iris* data set in this project. In the future, we may conduct more experiments on some other data sets and see if we could draw more conclusions.

Moreover, we only tested the error rates of certain and possible rules in this project. However, rules may be induced with other conditional probability threshold α. It might be interesting to compare the error rates for those rules.

References

Clark, P. & Grzymala-Busse, J. (2011). Experiments on probabilistic approximations. In *Granular Computing (GrC), 2011 IEEE International Conference on* (pp. 144–149).

Clark, P. G., Grzymala-Busse, J. W., & Kuehnhausen, M. (2012). Local probabilistic approximations for incomplete data. In *Foundations of Intelligent Systems* (pp. 93–98). Springer.

Clark, P. G., Grzymala-Busse, J. W., & Rzasa, W. (2014). Mining incomplete data with singleton, subset and concept probabilistic approximations. *Information Sciences*, 280(0), 368 – 384.

Gryzmala-Busse, J. & Wang, C. (1996). Classification and rule induction based on rough sets. In *Fuzzy Systems, 1996., Proceedings of the Fifth IEEE International Conference on*, volume 2 (pp. 744–747 vol.2).

Grzymala-Busse, J. (1992). Lers-a system for learning from examples based on rough sets. In S. Roman (Ed.), *Intelligent Decision Support*, volume 11 of *Theory and Decision Library* (pp. 3–18). Springer Netherlands.

Grzymala-Busse, J. (2004). Characteristic relations for incomplete data: A generalization of the indiscernibility relation. In S. Tsumoto, R. Slowinski, J. Komorowski, & J. W. Grzymala-Busse (Eds.), *Rough Sets and Current Trends in Computing*, volume 3066 of *Lecture Notes in Computer Science* (pp. 244–253). Springer Berlin Heidelberg.

Grzymala-Busse, J. (2008). Mlem2 rule induction algorithms: With and without merging intervals.

In T. Lin, Y. Xie, A. Wasilewska, & C.-J. Liau (Eds.), *Data Mining: Foundations and Practice*, volume 118 of *Studies in Computational Intelligence* (pp. 153–164). Springer Berlin Heidelberg.

Grzymala-Busse, J. & Rzasa, W. (2006). Local and global approximations for incomplete data. In S. Greco, Y. Hata, S. Hirano, M. Inuiguchi, S. Miyamoto, H. Nguyen, & R. Słowiński (Eds.), *Rough Sets and Current Trends in Computing*, volume 4259 of *Lecture Notes in Computer Science* (pp. 244–253). Springer Berlin Heidelberg.

Grzymala-Busse, J. W., Clark, P. G., & Kuehnhausen, M. (2014). Generalized probabilistic approximations of incomplete data. *International Journal of Approximate Reasoning*, 55(1, Part 2), 180 – 196. Special issue on Decision-Theoretic Rough Sets.

Grzymala-Busse, J. W. & Rzasa, W. (2010). A local version of the mlem2 algorithm for rule induction. *Fundam. Inf.*, 100(1-4), 99–116.

Grzymala-Busse, J. W. & Siddhaye, S. (2004). Rough set approaches to rule induction from incomplete data. In *Proceedings of the IPMU*, volume 2 (pp. 923–930).

Grzymala-Busse, J. W. & Zou, X. (1998). Classification strategies using certain and possible rules. In *Proceedings of the First International Conference on Rough Sets and Current Trends in Computing*, RSCTC '98 (pp. 37–44). London, UK, UK: Springer-Verlag.

Mann, H. B. & Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, (pp. 50–60).

Pawlak, Z. (1982). Rough sets. *International Journal of Computer & Information Sciences*, 11(5), 341–356.

Pawlak, Z. (1992). *Rough Sets: Theoretical Aspects of Reasoning About Data*. Norwell, MA, USA: Kluwer Academic Publishers.