MINING STUDENT DATABASE USING ROUGH SET THEORY

PROJECT REPORT

Submitted By

P. KATHIRAVAN

Register No.: 0720300017

in partial fulfilment for the award of the degree

Of

MASTER OF COMPUTER APPLICATIONS

in

COMPUTER APPLICATIONS

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

COIMBATORE – 641 006.

Department of Computer Applications

PROJECT WORK

MAY 2010

This is to certify that the project entitled

MINING STUDENT DATABASE USING ROUGH SET THEORY

is the bonafide record of project work done by

P.KATHIRAVAN

Register No: 0720300017

of MCA (Computer Applications) during the year 2009-2010.

6/5/10

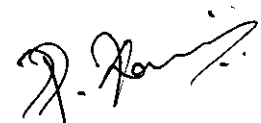
Project Guide

Head of the Department

Submitted for the Project Viva-Voce examination held on 17.05.2010

DECLARATION

I affirm that the project work titled **MINING STUDENT DATABASE USING ROUGH SET THEORY** being submitted in partial fulfilment for the award of **MASTER OF COMPUTER APPLICATIONS** is the original work carried out by me. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.



(Signature of the Candidate)

P. KATHIRAVAN

0720300017

I certify that the declaration made above by the candidate is true.

Signature of the Guide,

Mrs. V. Geetha

Assistant Professor, MCA

KUMARAGURU COLLEGE OF TECHNOLOGY

[An Autonomous Institution]
(Approved by AICTE / Affiliated to Anna University, Coimbatore / Accredited by NBA & NAAC)
POST BOX No. 2034 – COIMBATORE - 641 006

Estd. 1984

ISO 9001-2008 Certified

TO WHOMSOEVER IT MAY CONCERN

05/05/2010

Certified that **Mr. P. KATHIRAVAN** student of final year M.C.A (Master of Computer Applications) from **Kumaraguru College of Technology** has done the said project titled as **"Mining Student Database Using Rough Set Theory"** in our college from December 1,2009 to April 30, 2010.

During this period of project his attendance was found to be regular and satisfactory.

Ms. V. Geetha
Assistant Professor/MCA

ACKNOWLEDGEMENT

I wish to express sincerest thanks to **Dr.J.Shanmugam,** Director, Kumaraguru College of Technology, **Dr.S.Ramachandran,** Principal, Kumaraguru College of Technology, and **Dr.S.Thangasamy**, Dean, Department of Computer Science and Engineering for providing necessary facilities in carrying out my project work.

I am very glad to express a special word of thanks to **Dr.A.Muthukumar,** Professor and course coordinator, Department of Computer Applications, Kumaraguru College of Technology, Coimbatore for encouraging me to do this work.

I wish to thank my Project guide and Project Coordinator **Mrs.V.Geetha,** Assistant Professor, for her sincere advice, thought provoking discussions and immense help throughout the project and encouragement given by her.

I wish to thank **Mr.P. Ramasubramanian,** Professor, Department of Computer Science and Engineering, Francis Xavier Engineering College, Tirunelveli., for his sincere advice, thought provoking discussions and immense help throughout the project and encouragement given by him.

I wish to thank all my staff members for their timely help and guidance to complete the project successfully.

TABLE OF CONTENTS

ABSTRACT

The project titled "**Mining Student Database Using Rough Set Theory**" is used to mine the academic performance of the students and analyze the students' data using rough set theory.

The student details are stored in a database as a two dimensional table. The table consists of set of objects and attributes. The attribute set consists of condition attributes and a decision attribute. The decision attribute is an observed decision about a student's academic performance. The decision attribute consists of three classes namely normal, average and below average. The characteristic sets are computed for each record in the table. After computing the characteristic sets for each record, the input class is obtained from the user interface as an input.

After obtaining the input class, the approximation space is computed by which the decision rules are generated. Approximation space is computed from the lower and upper approximation of the given class.

Then decision rules are generated to predict the different categories of studies using ROSE tool. Finally, only the rules that satisfy the minimum support and confidence are considered, which will in turn be used to predict the performance of students'.

This project is done by using Visual Basic 6.0 as a front end and MS Access as a back end. The computation of approximation space is calculated by the Visual Basic application. The decision rules are generated using the ROSE tool.

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Data mining (or data discovery) is the process of autonomously extracting useful information or knowledge from large data stores or sets. It involves the use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data sets. Data mining consists of more than collecting and managing data; it also includes analysis and prediction. These tools can include statistical models, mathematical algorithms, and machine learning methods such as neural networks or decision trees etc by using rough set algorithms. This project in data mining field shows that one of the application domains that can take advantage of data mining benefits in education.

One of the biggest challenges that higher education faces today is predicting the paths of students. Institutions are interested to know, about the performance of the students like which type of students will complete the course successfully and which type of students need more assistance. Normally, the students' performance is classified into three levels namely normal, average and below average. This project attempts to analyze the Student Information System (SIS) database using rough set theory to predict the future of students. In fact two main cases of missing attribute values are considered here "lost" (the original value was erased) and "do not care" (the original value was irrelevant).

The system, mining student database using rough set analysis, is a windows application which is used for decision making like predicting the performance of the students. Classification, a data mining technique, is used in this system. Rough set theory is one of the mining techniques comes under classification.

In this project, a two dimensional table is used to store the student details. For

sets, the approximation space is computed for the given input class. This input class is given through a user interface. Then decision rules are generated to predict the different categories of studies using ROSE tool. Finally, only the rules that satisfy the minimum support and confidence are considered, which will in turn be used to predict the performance of students'.

CHAPTER 2

SYSTEM ANALYSIS

System analysis involves the process of diagnosing, interpreting and helps us to propose a new system. This chapter describes existing and proposed system.

2.1 EXISTING SYSTEM

There is no existing system available with this concept of mining technique. There are many data mining techniques used for mining yet, no specific system in existence. Generally, statistical methods are used for data mining.

The existing system has the following drawbacks:

- The results from the other data mining techniques are unclear.
- Large set of decision rules are produced using the other data mining techniques.
- We have to preprocess for the incomplete data prior to mining.

2.2 PROPOSED SYSTEM

The proposed system uses the concept called rough set theory, a mathematical concept.

2.2.1. Rough Set Theory

The Rough Set Theory is a recent mathematical theory employed as a data mining tool with many favorable advantages. It offers the mathematic tools for discovering hidden patterns in data through the use of identification of partial and total dependencies in data. Since this theory has been applied to various domains, the majority of these applications are used to solve the classification problems, which exclude the temporal factor in data sets. The rough set analysis is presented as a technique to direct the knowledge discovery process from data. In 1982, Pawlak introduced the Rough Set Theory. This rough set theory was initially developed for a finite universe of discourse in

which the knowledge base is a partition, which is obtained by any equivalence relation defined on the universe of discourse. It also enables work with null or missing values. Rough sets can be used separately but usually they are used together with other methods such as fuzzy sets, statistic methods, genetic algorithms etc. The rough sets theory uses different approach to uncertainty. As well as fuzzy sets this theory is only part of the classic theory, not an alternative.

2.2.2. Rough Set Terminology and Notions

The following are the important terms in rough set theory. They are as follows.

- Information system
- Indiscernibility relation/ Characteristic relation
- Set approximation(Lower and Upper)
- Definability
- Reducts
- Core
- Rough membership

Information system

A data set is represented as a table, where each row represents a case, an event, a student or simply an object. Every column represents an attribute (a variable, an observation or a property, etc.) that can be measured for each object; the attribute may also be supplied by a human expert or user. This table is called an *information system*. More formally, it is a pair A= (U, A), where U is non-empty finite set of objects called the *universe* and A is the non-empty finite set of attributes such that a: U$\rightarrow V_a$ for every a $\in$ A. The set V_a is called the value set of a. We can also represent the above function in another way, i.e., any decision table defines a function ρ that maps the direct product of U and A into the set of all values. For example, in Table 2.1, $\rho(7, Academic) = 0.8$. The independent variables are called attributes (or condition attributes), the dependent variables are called decisions (or decision attributes). The following table is a sample information system of students with attributes academic, non-academic and human

rough sets, the decision table of any information system is given by T= (U, A, C, D), where U is the universe of discourse, A is a set of primitive features, C and D are the subsets of A, called condition and decision features respectively.

The information system is sorted into two kinds by the nature of the data it contains. They are

- Complete
- Incomplete

A decision table with completely specified function *a* will be called *completely specified*, or, for the sake of simplicity, *complete*. On the other hand, a dataset is said to be incomplete if and only if the function *a* for the given table is incompletely specified. The following tables represent the two categories of datasets.

SNo.	Academic	Non-academic	Human Behavior relationship	Decision
1	0.8	0.6	0.8	Normal
2	0.7	0.4	0.9	Normal
3	0.5	0.4	0.4	Average
4	0.4	0.3	0.7	Below average
5	0.3	0.4	0.9	Below average
6	0.4	0.5	0.5	Below average
7	0.8	0.6	0.8	Normal
8	0.8	0.6	0.8	Normal
9	0.8	0.3	0.8	Normal
10	0.3	0.2	0.4	Below average
11	0.7	0.7	0.7	Normal

Table 2.1 A sample complete information system

SNo.	Academic	Non-academic	Human Behavior relationship	Decision
1	0.8	*	0.8	Normal
2	0.7	0.4	0.9	Average
3	?	*	0.4	Average
4	*	0.3	0.7	Average
5	*	0.4	0.9	Average
6	0.4	0.5	?	Below average
7	0.8	0.6	0.8	Normal
8	0.8	0.6	0.8	Normal
9	0.8	*	0.8	Normal
10	0.3	*	?	Below average
11	0.7	0.7	*	Average

Table 2.2 A sample incomplete information system

Indiscernibility Relation

Rough set theory is based on the idea of an indiscernibility relation, defined for complete decision tables. Let B be a nonempty subset of the set A of all attributes. The indiscernibility relation IND (B) is a relation on U defined for $x, y \in U$ as follows

$$(x, y) \in IND\ (B)\ \textit{if and only if}\ \rho(x, a) = \rho(y, a)\ \textit{for all}\ a \in B.$$

. The indiscernibility relation IND (B) is an equivalence relation. Equivalence classes of IND (B) are called *elementary sets* of B and are denoted by $[x]_B$.

The equivalence relation partitions U. Since, by theorem (the theorem is shown in appendices) any two equivalence classes are either identical or disjoint. For example, for Table 2.1, elementary sets of IND(A) are {1,7,8},{2},{3},{4},{5},{6},{9},{10},{11}. The indiscernibility relation IND (B) may be computed using the idea of blocks of attribute-value pairs. Let a be an attribute, i.e., $a \in A$ and let v be a value of a for some case. For complete decision tables if $t = (a, v)$ is an attribute-value pair then a block of t, denoted $[t]$, is a set of all cases from U that for attribute a have value v. The

(B). Such elementary sets of B are intersections of the corresponding attribute-value pair blocks. They are denoted by $[x]_B$, where B is the subset of the attribute set A.

The idea of how to compute elementary sets of B for Table 2.1 and $B = A$ is given below.

$[1]_A = [7]_A = [8]_A$ = [(Academic, 0.8)] ∩ [(Non-academic, 0.6)] ∩ [(Human Behavior Relationship, 0.8)] = {1,7,8}

$[2]_A$= [(Academic, 0.7)] ∩ [(Non-academic, 0.4)] ∩ [(Human Behavior Relationship, 0.9)] = {2}

$[3]_A$ = [(Academic, 0.5)] ∩ [(Non-academic, 0.4)] ∩ [(Human Behavior Relationship, 0.4)] = {3}

$[4]_A$ = [(Academic, 0.4)] ∩ [(Non-academic, 0.3)] ∩ [(Human Behavior Relationship, 0.7)] = {4}

$[5]_A$ = [(Academic, 0.3)] ∩ [(Non-academic, 0.4)] ∩ [(Human Behavior Relationship, 0.9)] = {5}

$[6]_A$ = [(Academic, 0.4)] ∩ [(Non-academic, 0.5)] ∩ [(Human Behavior Relationship, 0.5)] = {6}

$[9]_A$ = [(Academic, 0.8)] ∩ [(Non-academic, 0.3)] ∩ [(Human Behavior Relationship, 0.8)] = {9}

$[10]_A$ = [(Academic, 0.3)] ∩ [(Non-academic, 0.2)] ∩ [(Human Behavior Relationship, 0.4)] = {10}

$[11]_A$ = [(Academic, 0.7)] ∩ [(Non-academic, 0.7)] ∩ [(Human Behavior Relationship, 0.7)] = {11}

Characteristic Relation

For data sets with missing attribute values, the corresponding function ρ is incompletely specified (partial). A decision table with incompletely specified function will be called *incompletely specified*, or *incomplete*.

In the sequel it is assumed that all decision values are specified, i.e., they are not missing. Also, it is assumed that all missing attribute values are denoted by "?", by "*" or by "?", lost values will be denoted by "?", "do not care" conditions will be denoted by "*", and attribute-concept values by "?". Additionally, it is assumed that for each case at

least one attribute value is specified. Incomplete decision tables are described by characteristic relations instead of indiscernibility relations. Also, elementary sets are replaced by characteristic sets. An example of incomplete data is presented in Table 2.2.

For incomplete decision tables the definition of a block of an attribute value pair must be modified.

• If an attribute a there exists a case x such that $\rho(x, a) = ?$, i.e., the corresponding value is lost, then the case x should not be included in any block $[(a, v)]$ for all values v of attribute a.

• If for an attribute a there exists a case x such that the corresponding value is a "do not care" condition, i.e., $\rho(x, a) = *$, then the corresponding case x should be included in blocks $[(a, v)]$ for all specified values v of attribute a.

• If for an attribute a there exists a case x such that the corresponding value is a attribute-concept value, i.e., $\rho(x, a) = -$, then the corresponding case x should be included in blocks $[(a, v)]$ for all specified values v of attribute a that are members of the set $V(x, a)$, where

$$V(x, a) = \{\rho(y, a) \mid \rho(y, a) \text{ is specified}, y \in U, \rho(y, d) = \rho(x, d)\},$$

and d is the decision.

These modifications of the definition of the block of attribute-value pair are consistent with the interpretation of missing attribute values, lost," do not care conditions and attribute-concept values. Also, note that the attribute-concept value is the most universal, since if $V(x, a) = \emptyset$, the definition of the attribute-concept value is reduced to the lost value, and if $V(x, a)$ is the set of all values of an attribute a, the attribute-concept value becomes a "do not care" condition.

For a case $x \in U$, the *characteristic set* $KB(x)$ is defined as the intersection of the sets $K(x, a)$, for all $a \in B$. If $\rho(x, a)$ is specified, then $K(x, a)$ is the block $[(a, \rho(x, a)]$ of attribute a and its value $\rho(x, a)$. If $\rho(x, a) = *$ or $\rho(x, a) = ?$ then the set $K(x, a) = U$. If $\rho(x, a) = -$ and $V(x, a)$ is nonempty, then the corresponding set $K(x, a)$ is equal to the union of all blocks of attribute-value pairs (a, v), where $v \in V(x, a)$. If $V(x, a)$ is empty, then $K(x, a) = \{x\}$. The way of computing characteristic sets needs a comment. For both"do not care" conditions and lost values the corresponding set $K(x, a)$ is equal to U

value of the attribute a is irrelevant; if $\rho(x, a) = ?$, only existing values need to be checked. However, the case when $\rho(x, a) = -$ is different, since the attribute a restricts the set $K_B(x)$. Furthermore, the description of $K_B(x)$ should be consistent with other (but similar) possible approaches to missing attribute values, e.g., an approach in which each missing attribute value is replaced by the most common attribute value restricted to a concept. Here the set $V(x, a)$ contains a single element and the characteristic relation is an equivalence relation. For Table 2.2 and $B = A$,

$K_A(1) = \{1,4,5,7,8,9\} \cap \{1,3,9,10\} \cap \{1,7,8,9,11\} = \{1,9\}$

$K_A(2) = \{2, 5\}$,

$K_A(3) = \{3\}$,

$K_A(4) = \{4\}$,

$K_A(5) = \{5\}$,

$K_A(6) = \{6\}$,

$K_A(7) = \{1, 7, 8, 9\}$

$K_A(8) = \{1, 7, 8, 9\}$

$K_A(9) = \{1, 7, 8, 9\}$

$K_A(10) = \{10\}$ and,

$K_A(11) = \{11\}$.

The characteristic set $K_B(x)$ may be interpreted as the smallest set of cases that are indistinguishable from x using all attributes from B, and using given interpretation of missing attribute values. Thus, $K_A(x)$ is the set of all cases that cannot be distinguished from x using all attributes. Also, note that the previous definition is an extension of a definition of $K_B(x)$ for decision tables with only lost values and "do not care" conditions, both definitions are identical.

The *characteristic relation* $R(B)$ is a relation on U defined for $x, y \in U$ as follows

$$(x, y) \in R(B) \text{ if and only if } y \in K_B(x).$$

The characteristic relation $R(B)$ is reflexive but—in general—it does not need to be symmetric or transitive. Also, the characteristic relation $R(B)$ is known if the characteristic sets $K_B(x)$ for all $x \in U$ is known. In our example,

$R(A) = \{(1, 1), (1, 9), (2, 2), (2, 5), (3, 3), (4, 4), (5, 5), (6, 6), (7, 1), (7, 7), (7,8), (7, 9), (8, 1), (8, 7), (8, 8), (8, 9), (9, 1), (9, 9), (10, 10), (11, 11)\}$

For decision tables, in which all missing attribute values are lost, a special characteristic relation $LV(B)$ was defined by J. Stefanowski and A. Tsoukias. Characteristic relation $LV(B)$ is reflexive, but—in general—it does not need to be symmetric or transitive.

For decision tables where all missing attribute values are"do not care" conditions a special characteristic relation $DCC(B)$ was defined byM. Kryszkiewicz. Relation $DCC(B)$ is reflexive and symmetric but— in general—is not transitive.

Obviously, characteristic relations $LV(B)$ and $DCC(B)$ are special cases of the characteristic relation $R(B)$. For a completely specified decision table, the characteristic relation $R(B)$ is reduced to IND (B).

Set Approximations

Let A be the set of all attributes, B is a subset of A. For completely specified tables, the lower and upper approximations are defined using the indiscernibility relation. Any finite union of elementary sets of B is called a B- definable set. Let X be any subset of the set U of all cases. The set X is called concept and is usually defined as the set of all cases defined by a specific value of the decision. In general, X is not a B-definable set. However, set X may be approximated by two B-definable sets; the first one is called a *B-lower approximation* of X, denoted by BX and defined as follows

$$\underline{B}X = \{x \mid [x]_B \subseteq X\},$$

The second set is called an *B-upper approximation* of X, denoted by BX and defined as follows

$$\overline{B}X = \{x \mid [x]_B \cap X \neq \phi\}.$$

For incompletely specified tables, the lower and upper approximations are defined by the characteristic relation. Let X be a concept, let B be a subset of the set A of all attributes, and let $R(B)$ be the characteristic relation of the incomplete decision table with characteristic sets $K_B(x)$, where $x \in U$. The lower approximation of the given class X

$$\underline{B}X = \{x \in U \mid K_B(x) \subseteq X\},$$

The upper approximation of the given class X using B is defined as follows

$$BX = \{x \in U \mid K_B(x) \cap X \neq \Phi\}$$

More precisely, the lower approximation is defined as the set of items which can be certainly classified as items of X while the upper approximation is defined as the set of items which can be possibly classified as items of X. The boundary region, denoted as BND(X) or $BN_B(X)$, is defined as the set of items which can be classified either as item of X or not. It is calculated as follows

$$\text{BND}(X) = BN_B(X) = \overline{B}X - \underline{B}X,$$

The outside the boundary region is called the negative region of X. Fig 2.1 illustrates the typical lower and upper approximation of a given set and its boundary region.

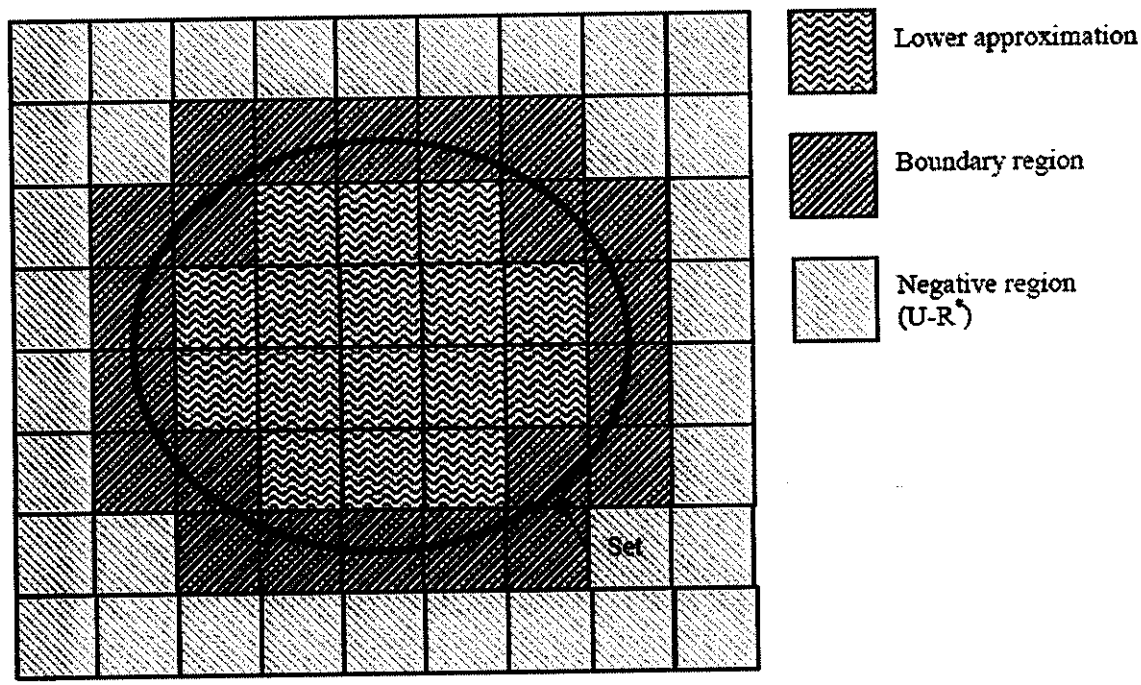


Fig. 2.1 An illustration of the boundary region of rough set.

Definability

There are four basic classes of rough set X. They are

a) X is roughly B- definable, iff $\underline{B}(X) \neq \Phi$ and $B(X) \neq U$

b) X is internally B- undefinable iff $B(X) = \Phi$ and $B(X) \neq U$

c) X is externally B- undefinable iff $B(X) \neq \Phi$ and $B(X) = U$

d) X is totally B- undefinable iff $B(X) = \Phi$ and $B(X) = U$

The intuitive meaning of this classification is as follow:

a) X is roughly B- definable means that with the help of B it is possible to decide for some elements of U that they belong to X and for some elements of U that they belong to U-X

b) X is internally B- undefinable means that using B it is possible to decide for some elements U that they belong to U-X but it cannot be decided for any element of U whether it belongs to X

c) X is externally B- undefinable means that using B it is possible to decide for some elements U that they belong to X but it cannot be decided for any element of U whether it belongs toU- X

d) X is totally B- undefinable means that using B it cannot be decided for any elements of U whether it belongs to X or U-X.

Reducts

A *reduct* is the minimal subset of attributes that enables the same classification of objects of the universe as the complete set of attributes.

Core

One of the important properties of reducts is the *core of attributes*. The core of attributes is the intersection of the attributes in the reducts. In a sense, the core is the most important set of attributes, since none of its elements can be removed without affecting the classification power of the attributes.

Attributes for mining

The following attributes are taken for the mining the students' performance.

- Academic mark
- Entrance mark
- Attendance percentage
- Zone
- Mode of admission
- Family member working in IT industry
- Computer background

The explanation of these attributes to be explained in later chapter.

2.2.3 Algorithms

The following algorithms are used to calculate the characteristic sets and boundary.

The characteristic relation R (B) is known if the characteristic sets K_B (a) for all a ϵ U is known. For completely specified decision tables, if t = (a,v) is an attribute value pair, then a block of t, denoted by [t], is a set of all cases from U that for attribute a have value v. For incompletely specified decision tables the definition of a block of an attribute value pair must be modified. If an attribute 'a' there exists a case such that p(x,a)= ?, that is the corresponding value is lost, that the case x is not included in the block[(a,v)] for any value v of attribute a. If for an attribute a there exists a case x such that the corresponding value is a "don't care" condition, that is p(x,a) = *, then the corresponding case x should be included in blocks[(a,v)] for all values v of attribute a. The characteristic set KB(x) is the intersection of blocks of attribute value pairs (a,v) for all attributes a from B for which p(x,a) = v.

Algorithm 1: characteristic sets

Input: Dataset, U and the subset, B, of the attribute set A. Say, B= {a, b, c}

Output: Characteristic sets K_B (x)

For each $x \in U$,

$K_a(x) = \{y \in U: p(x, a) = p(y, a)\} \cup \{y \in U: p(y, a) = \text{'* '}\}$.

$K_B(x) = K_a(x) \cap K_b(x) \cap K_c(x)$.

Explanation:

For example, the characteristic sets are computed for table 2.2 as follows. For $K_B(1)$,

$$K_B(1) = \{1,4,5,7,8,9\} \cap \{1,3,9,10\} \cap \{1,7,8,9,11\} = \{1,9\}$$

In the same manner, the characteristic sets for the remaining objects are computed as follows.

$K_B(2) = \{2,5\}$;

$K_B(3) = \{3\}$;

$K_B(4) = \{4\}$;

$K_B(5) = \{5\}$;

$K_B(6) = \{6\}$;

$K_B(7) = \{1,7,8,9\}$;

$K_B(8) = \{1,7,8,9\}$;

$K_B(9) = \{1,7,8,9\}$;

$K_B(10) = \{10\}$ and

$K_B(11) = \{11\}$.

Computing the Boundary of X

Let A = (U, A) be an information system and Let B be a subset of A and X be a subset of U. The approximation X is computed by constructing the B-lower and B-upper approximations of X respectively using only the information contained in B, where

$$\underline{B}X = \{x \in U \mid K_B(x) \subseteq X\},$$

$$\overline{B}X = \{x \in U \mid K_B(x) \cap X \neq \Phi\}$$

The set BND(X) = $\overline{B}X - \underline{B}X$ is called the B-boundary region of X, and thus consists of those objects that one cannot decisively classify into X on the basis of knowledge in B. A set is said to be rough, if the boundary region is non-empty.

Algorithm 2: Compute Boundary of X

Input: B, the subset of attributes

Input: X, the subset of U, the decision space

Output: Boundary of X

1. [Initialize]

G: =U

a $\in$ U

Lower(X) = ϕ

Upper(X) = ϕ

2. Repeat step 3 to 5 while G $\neq \phi$

3. If $K_B(a) \subseteq X$ then Lower(X) = Lower(X) $U K_B(a)$

4. If $K_B(a) \cap X \neq \varphi$ then Upper(X) = Upper(X) $U K_B(a)$

5. G = G – {a}

[End of the loop]

6. BND(X) = Upper(X) – Lower(X)

This algorithm is known as **Modified Learning from Examples, version 2 (MLEM2).**

Explanation:

Consider a set (namely, a decision) X= {2, 3, 4, 5, 11}. From table 2.2, the B-lower and B- upper approximations are

Lower (X) = {2, 3, 4, 5, 11} and Upper (X) = {2, 3, 4, 5, 11}

Decision Rules:

Decision rules are of the form *if→ then.* The *if* part of the rule contains the combination of attribute- value pairs with AND or OR operator. The same idea of blocks of attribute value pairs is used in the rule induction algorithm LEM2. LEM2 explores the

search space of attribute-value pairs. Its input data file is a lower or upper approximation of a concept, so its input data file is always consistent. Rules induced from the lower approximation of the concept *certainly* describe the concept, so they are called *certain.* On the other hand, rules induced from the upper approximation of the concept describe the concept only *possibly* (or *plausibly*), so they are called *possible.*

The decision rules generated for the student data with 50% minimum support and 50% confidence are as follows.

Rule1. (ADMISSION = COUNSELING) & (FAMILY_MEMBER_IN_IT = YES) & (COMPUTER_BACKGROUND = YES) => (PERFORMANCE = NORMAL);

Rule2. (ACADEMIC = 9) & (ATTENDANCE = 85-90) => (PERFORMANCE = NORMAL);

Rule3. (ENTRANCE = 5) & (ATTENDANCE = ABOVE_95) => (PERFORMANCE = NORMAL);

Rule4. (ATTENDANCE = BELOW_85) & (ADMISSION = COUNSELING) => (PERFORMANCE = NORMAL);

Rule5. (ZONE = RURAL) & (ADMISSION = COUNSELING) & (COMPUTER_BACKGROUND = YES) => (PERFORMANCE = NORMAL);

Rule 6. (ENTRANCE = 4) & (ATTENDANCE = 90-95) => (PERFORMANCE = NORMAL);

Rule 7. (ACADEMIC = 8) & (ATTENDANCE = 85-90) & (FAMILY_MEMBER_IN_IT = NO) => (PERFORMANCE = AVERAGE);

Rule 8. (ACADEMIC = 8) & (ZONE = RURAL) & (COMPUTER_BACKGROUND = NO) => (PERFORMANCE = AVERAGE);

Rule9. (ACADEMIC = 8) & (ADMISSION = MANAGEMENT) & (COMPUTER_BACKGROUND = YES) => (PERFORMANCE = AVERAGE);

Rule10. (ATTENDANCE = ABOVE_95) & (ADMISSION = MANAGEMENT) => (PERFORMANCE = AVERAGE);

Rule11. (ACADEMIC = 7) & (ENTRANCE = 4) => (PERFORMANCE = AVERAGE);

Rule12. (ACADEMIC = 8) & (FAMILY_MEMBER_IN_IT = YES) & (COMPUTER_BACKGROUND = NO) => (PERFORMANCE = AVERAGE);

Rule13. (ACADEMIC = 7) & (FAMILY_MEMBER_IN_IT = YES) => (PERFORMANCE = BELOW_AVERAGE);

Rule14. (ACADEMIC = 6) => (PERFORMANCE = BELOW_AVERAGE);

Advantages of Rough sets

Mining students' information system using rough sets has the following advantages over the other data mining techniques. These are

- Data Analysis of quantitative and qualitative is easy.
- Since it is a mathematical tool, the decisions from these methods are very accurate and reliable.
- Minimal set of decision rules are generated.
- Since the method preprocess the missing values by itself, so there is no need to preprocess the data prior to mining.

2.3 MODULE FUNCTIONALITIES

The modules of the project are as follows:

- Characteristic sets
- Lower approximation
- Upper approximation
- Boundary
- Decision rules

Characteristic sets

This module calculates the characteristic sets for each object in the dataset. A characteristic set for a particular object 'x' is nothing but the set of objects having the same attribute-value for all attributes as 'x' has. Further, the characteristic sets are used to find the lower and upper approximation of the given class X.

Lower approximation

This module calculates the lower approximation of the given class X. The objects in the lower approximation of the set are the set of all objects which certainly belongs to the given class X. More precisely, it is the maximum number of objects which belongs to the given class X with 100% certainty.

Upper approximation

This module calculates the upper approximation of the given class X. The upper approximation of the given class X is itself a set which contains both the elements that belongs to the given class X and does not belong to X.

Boundary

This module describes the set of elements which lies in the boundary region. The boundary of X is calculated by the subtracting the lower approximation of X from the upper approximation of X. The boundary of X contains the elements which may or may

Decision Rules

This module describes the generation of the decision rules from the approximation space. The decision rules are generated using the ROSE tool. The minimal set of decision rules are generated with minimum support and confidence.

CHAPTER 3

DEVELOPMENT ENVIRONMENT

This chapter describes the hardware and software requirements for the application.

3.1 HARDWARE REQUIREMENTS

Processor	-	Intel Dual-core
Hard Disk Capacity	-	80GB
RAM	-	512MB
Speed	-	2.66 GHz
Monitor	-	Samsung
Printer	-	Epson EX-1000
Floppy Disk Drive	-	1.44MB
Compact Disk Drive	-	Sony

3.2 SOFTWARE REQUIREMENTS

When an application project is considered the three basic software requirements are the platform in which the project is developed, the front-end tool that provides the interaction with the user and back-end tool that stores the data.

Operating system	: Windows XP (SP 2)
Front end	: Visual Basic 6.0, ROSE 2
Back end	: MS- Access

3.3 PROGRAMMING ENVIRONMENT

VISUAL BASIC 6.0

The front end VISUAL BASIC 6.0 is a powerful programming language developed by Microsoft Corporation. It was developed from the BASIC programming language. It makes use of graphical user interface for creating powerful applications.

The GUI, as the name suggests user's illustrations to text enables users to interact return application. This feature provides the quickest and easiest way to create application.

Like the BASIC programming language, Visual Basic was designed to be easily learned and used by beginner programmers. The language not only allows programmers to create simple GUI applications, but can also develop complex applications. Programming in VB is a combination of visually arranging components or controls on a form, specifying attributes and actions of those components, and writing additional lines of code for more functionality. Since default attributes and actions are defined for the components, a simple program can be created without the programmer having to write many lines of code. Performance problems were experienced by earlier versions, but with faster computers and native code compilation this has become less of an issue.

Although programs can be compiled into native code executables from version 5 onwards, they still require the presence of runtime libraries of approximately 1 MB in size. This runtime is included by default in Windows 2000 and later, but for earlier versions of Windows like 95/98/NT it must be distributed together with the executable.

Forms are created using drag-and-drop techniques. A tool is used to place controls (e.g., text boxes, buttons, etc.) on the form (window). Controls have attributes and event handlers associated with them. Default values are provided when the control is created, but may be changed by the programmer. Many attribute values can be modified during run time based on user actions or changes in the environment, providing a dynamic application. For example, code can be inserted into the form resize event handler to reposition a control so that it remains centered on the form, expands to fill up the form,

etc. By inserting code into the event handler for a key press in a text box, the program can automatically translate the case of the text being entered, or even prevent certain characters from being inserted.

Visual Basic can create executables (EXE files), ActiveX controls, or DLL files, but is primarily used to develop Windows applications and to interface database systems. Dialog boxes with less functionality can be used to provide pop-up capabilities. Controls provide the basic functionality of the application, while programmers can insert additional logic within the appropriate event handlers. For example, a drop-down combination box will automatically display its list and allow the user to select any element. An event handler is called when an item is selected, which can then execute additional code created by the programmer to perform some action based on which element was selected, such as populating a related list.

The Visual Basic compiler is shared with other Visual Studio languages (C, C++), but restrictions in the IDE do not allow the creation of some targets (Windows model DLLs) and threading models.

MS Datagrid control:

The MS Datagrid control displays and operates on tabular data. It allows complete flexibility to sort, merge and format tables containing strings and pictures when bound to data control, MS Datagrid displays read-only data.

Picture can be included in any cell of MS Datagrid. The row and column property specify the current cell in MS Datagrid the current can be specified in the code, or the user can change it at runtime using the mouse or the arrow keys. The text property references the contents of the current cell.

Use the cols and rows property to determine the number of columns and rows in a MS Datagrid control.

Microsoft MS Access:

Performance level

MS Access Enterprise Edition supports features such as federated servers, indexed views, and large memory support that allow it to scale to the performance level.

Relational database engine

The MS Access relational database engine supports the features required to support demanding data processing environments. The database engine protects data integrity while minimizing the overhead of managing thousands of users concurrently modifying the database.

Distributed Queries

MS Access distributed queries allow user to reference data from multiple sources as if it were a part of a MS Access database, while at the same time, the distributed transaction support protects the integrity of any updates of the distributed data.

Replication

Replication allows user to also maintain multiple copies of data, while ensuring that the separate copies remain synchronized. User can replicate a set of data to multiple, disconnected users, have them work autonomously, and then merge their modifications back to the publisher.

ROSE 2

ROSE2 (Rough Sets Data Explorer) is a software implementing basic elements of the rough set theory and rule discovery techniques. It has been created at the Laboratory of Intelligent Decision Support Systems of the Institute of Computing Science in Poznan, basing on fourteen-year experience in rough set based knowledge discovery and decision analysis.

All computations are based on rough set fundamentals introduced by Z. Pawlak. One of implemented extensions applies the variable precision rough set model defined by W. Ziarko. It is particularly useful in analysis of data sets with large boundary regions.

proposed by R. Slowinski. The similarity relation is assessed from data via inductive learning.

The ROSE2 system is a successor of RoughDAS and RoughClass systems. RoughDAS is historically one of the first successful implementations of the rough set theory, which has been used in many real life applications.

The system contains several tools for rough set based knowledge discovery, e.g.:

- data preprocessing, including discretization of numerical attributes,
- performing a standard and an extended rough set based analysis of data,
- search of a core and reducts of attributes permitting data reduction,
- inducing sets of decision rules from rough approximations of decision classes,
- evaluating sets of rules in classification experiments,
- using sets of decision rules as classifiers.

ROSE started as several independent modules that were later put together in one system. First we were motivated to create computational engine working on more powerful computers (e.g. UNIX workstations), allowing faster analysis of large data sets. Then we came to the point of creating user friendly interface, where Microsoft Windows was chosen as our basic platform. So the modules can be separately redesigned and recompiled without much interference from user's point of view. The only component that is strictly platform dependent is graphical user interface (GUI). All this guarantees that the system can be easily adapted for future operating systems and platforms.

CHAPTER 4

SYSTEM DESIGN

The most important and challenging phase of the system life cycle is system design. The design focuses on the detailed implementation of the system. The first step in system design phase is to determine how the outputs are produced and in which format. Secondly, input data and the tables have to be designed to meet the requirements of proposed system. The system is designed as discussed in the analysis phase, i.e. the lower and upper approximations are computed. Further, the decision rules are generated from these two sets.

4.1 SYSTEM FLOW DIAGRAM

A system flow chart explains how a system works using a diagram. The diagram shows the flow of the data through a system in Figure 4.1

4.2 DATABASE DESIGN

Database design is designed to manage large bodies of information. These large bodies of information do not exist in isolation. Database design mainly involves the design of the data base schema. The design of the complete database application environment meets the needs of the enterprise being modeled requires attention to a broader set of issues.

The process of moving from an abstract data model to the implementation of the database proceeds in two design phases. In the logical schema, the designer maps the high level conceptual schema onto the implementation data model of the database system that will be used. The resulting specific database schema will be used in the subsequent physical design phase.

The table used for mining is shown in table 4.1

SYSTEM FLOW DIAGRAM

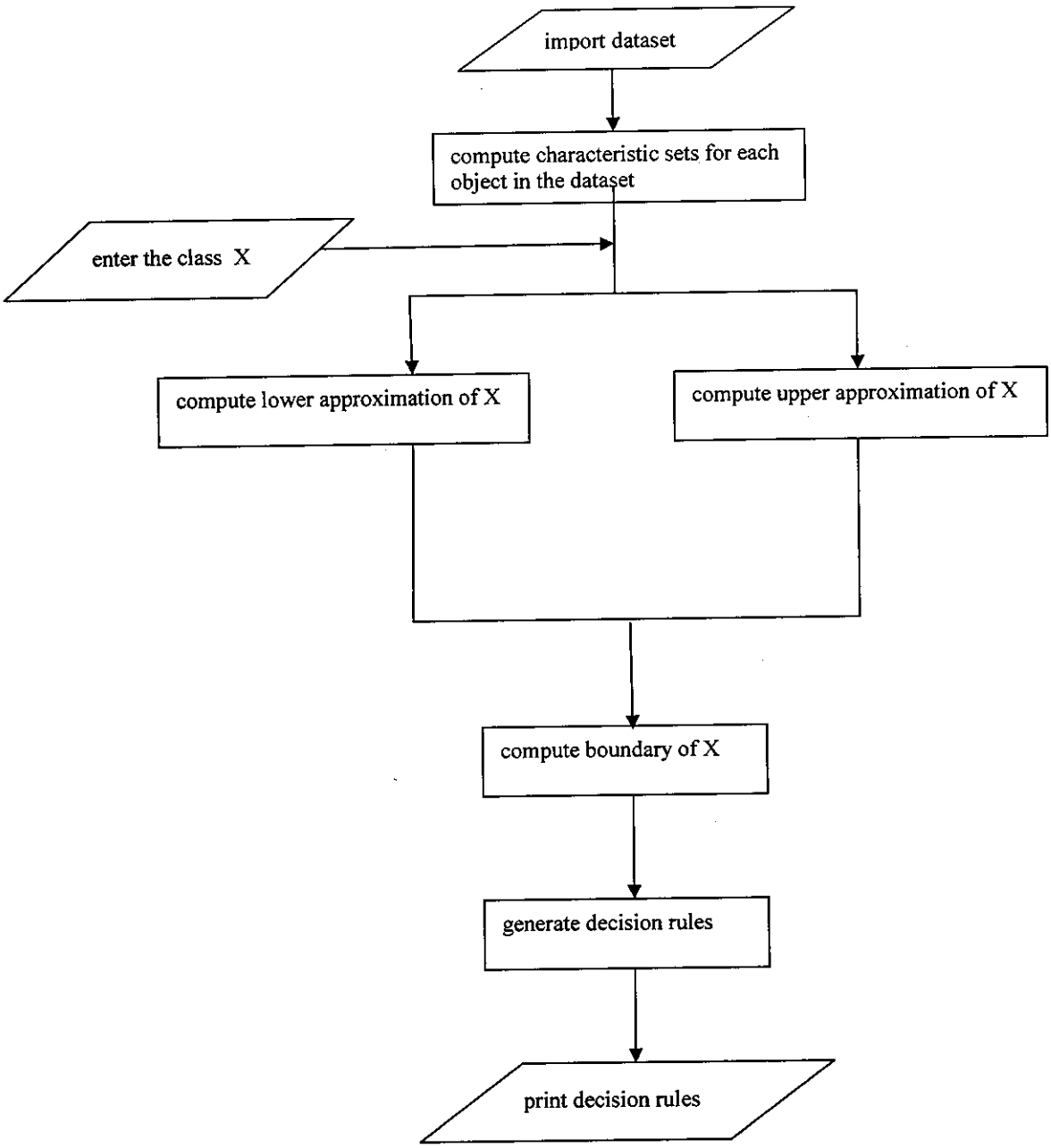


Fig. 4.1 System flow diagram for mining

Table name: MCA 3

This table stores the student details for mining with the following attributes. The following attributes are taken for mining.

Primary key: SNO

Field	Data Type	Description
SNO	Integer	Object or record id
ACAEMIC	Integer	Academic mark of the student up to 4 semesters
ENTRANCE	Integer	Entrance mark of the student, prior to join the course
ATTENDANCE	Text	Attendance mark of the student up to 4 semesters
ZONE	Text	Zone where the student is living
ADMISSION	Text	Mode of admission to this course
FAMILY_MEMBERS_IN_IT	Text	Whether the student's family members or friends working in IT industry or not?
COMPUTER_BACKGROUND	Text	Students' computer background
PERFORMANCE	Text	Performance of the student (decision attribute)

Table 4.2.1 MCA3 table

4.3 INPUT AND OUTPUT DESIGN

INPUT DESIGN

Input design is the process of correcting a user-oriented description of the inputs to a computer. Inaccurate data is one of the most common causes of data processing errors. The input design is very obvious as far as this project is concerned. The application uses MS Flexgrid and combo box as input. The dataset is imported to the flexgrid by clicking a command button. The input class X is given through a combo box as a text. The input screens are shown in Appendix A.1 and A.2

OUTPUT DESIGN

An application is successful only when it can produce efficient and effective reports. The output of this project is the approximation sets. They are displayed in a label. Moreover, decision rules are displayed in a text editor. The system allows us to print these decision rules. The output formats are shown in Appendix A.3 and A.4

CHAPTER 5

SYSTEM IMPLEMENTATION

An implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system. Many implementations may exist for a given specification or standard.

5.1 IMPLEMENTATION PROCESS

System Implementation is the part of the software engineering life cycle, where, the design artifacts are converted to a working application. The implementation of this system is done using Visual Basic 6.0 as front end. The coding is done using Visual Basic 6.0 and MS Access is used to store the dataset. Once the design is coded into a working application, it has to be verified, validated and tested in detail. The tested product if successful is deployed in the user environment.

5.2 SYSTEM VERIFICATION

System verification is the process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. The system is verified according to the requirements and design specifications. System verification is done at each stage of the implementation. Unit testing is done for this project.

5.3 SYSTEM VALIDATION

The system validates each module of the system. The system meets all the requirements given at the analysis phase. Finally, the system gives the required approximation space and the decision rules which designed at the system analysis phase.

CHAPTER 6

TESTING

Testing is a critical element of software quality and assurance and represents the ultimate review of specification design and coding. It is a vital activity that has to be enforced in the development of any system. This could be done in parallel during all the phases of system development. The feedback received from these tests can be used for further enhancement of the system under consideration.

The main type of testing carried out in this system is:

- Unit Testing

6.1 Unit Testing

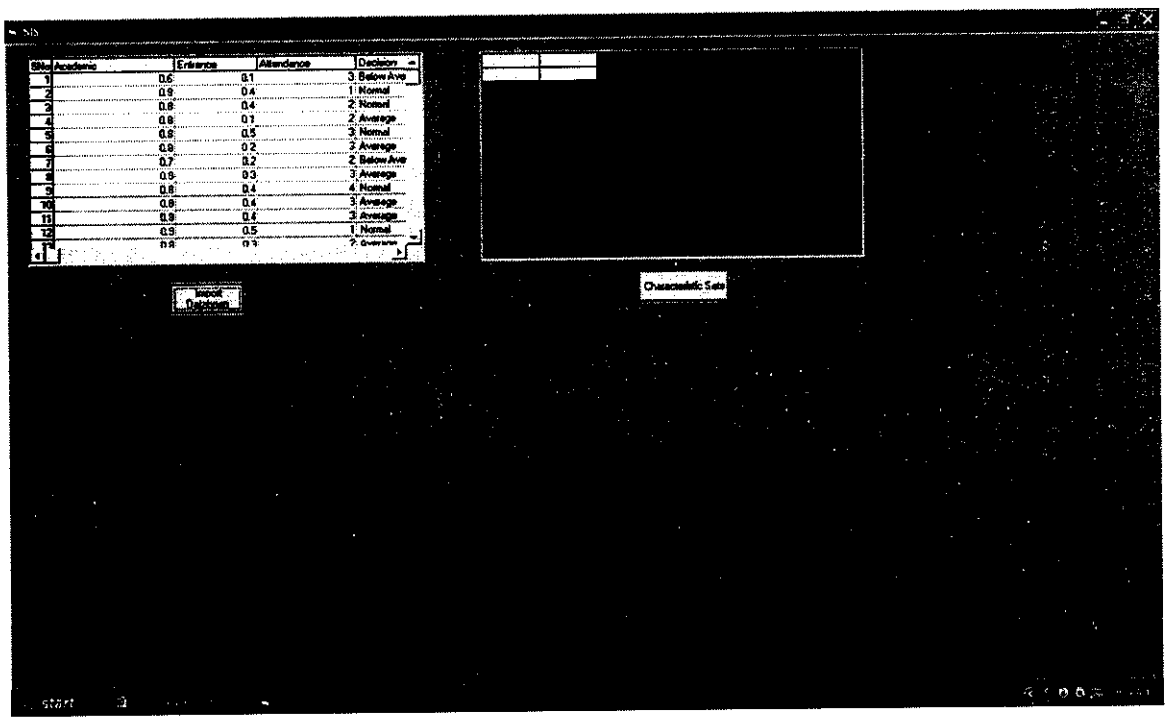
Unit testing focuses verification effort on the smallest unit of the software design the module. The local data structure is examined to ensure that data stored temporarily maintains its integrity.

Unit testing is successfully done for the system. Each unit is tested to give the correct output for the given requirement. The computation of characteristic sets and the approximation space are checked using unit testing.

APPENDIX

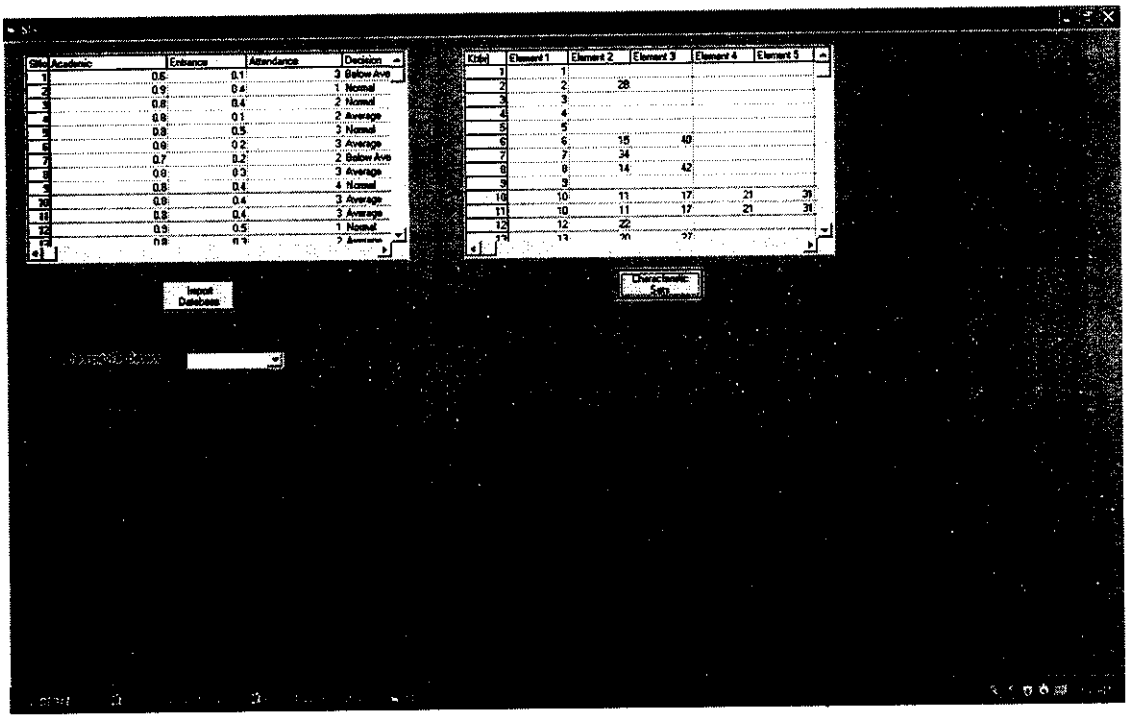
SAMPLE SCREEN

A.1 Import Dataset



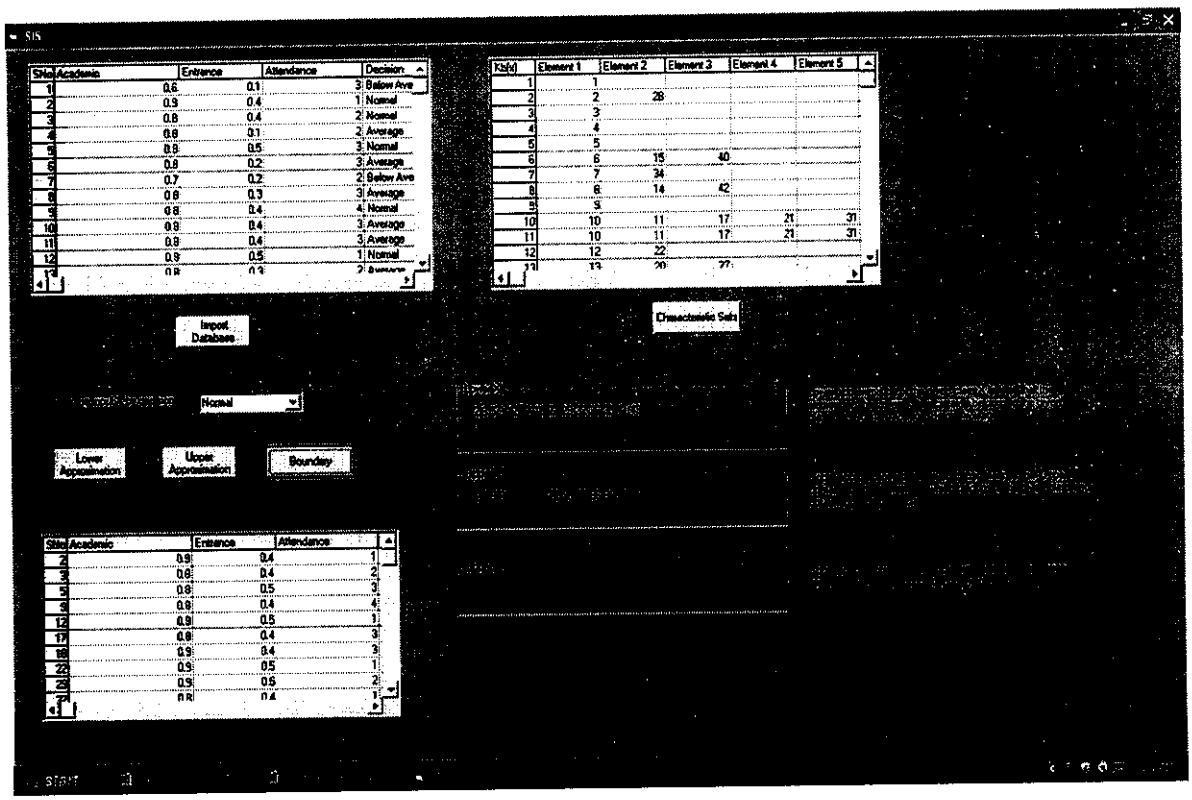
This screen shot is taken after importing the dataset to the application.

A.2 Characteristic sets



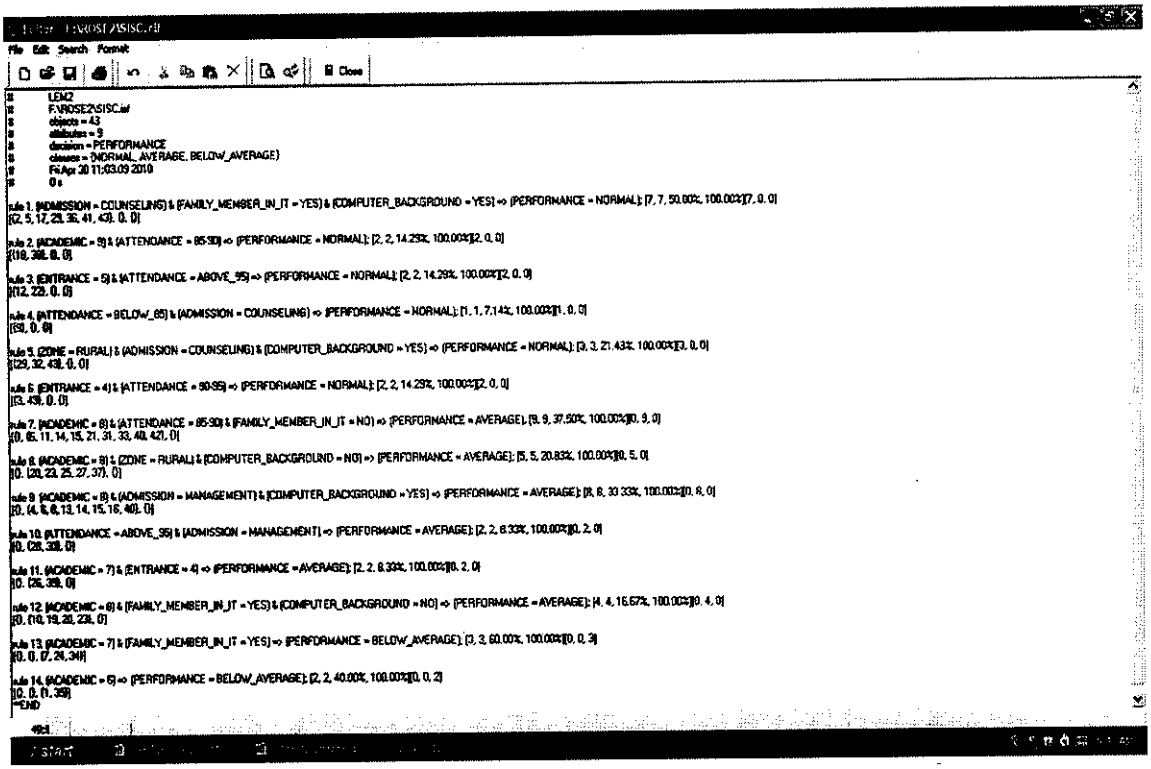
This screen shot is taken after calculating the characteristic sets.

A.3 Lower and Upper approximation



This screen shot is taken after calculating the lower and upper approximation and boundary of the given class.

A.4 Decision Rules



This screen shot is taken after the decision rules are generated using ROSE2 tool.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The project titled **"Mining Student Database Using Rough Set Theory"** is implemented to predict the students' academic performance. Though the algorithm for calculating the approximation space involves complex mathematical operations, it is implemented in Visual Basic 6.0 successfully. The decision rules are generated for the student dataset with the given attributes. From the decision rules, it is clear that

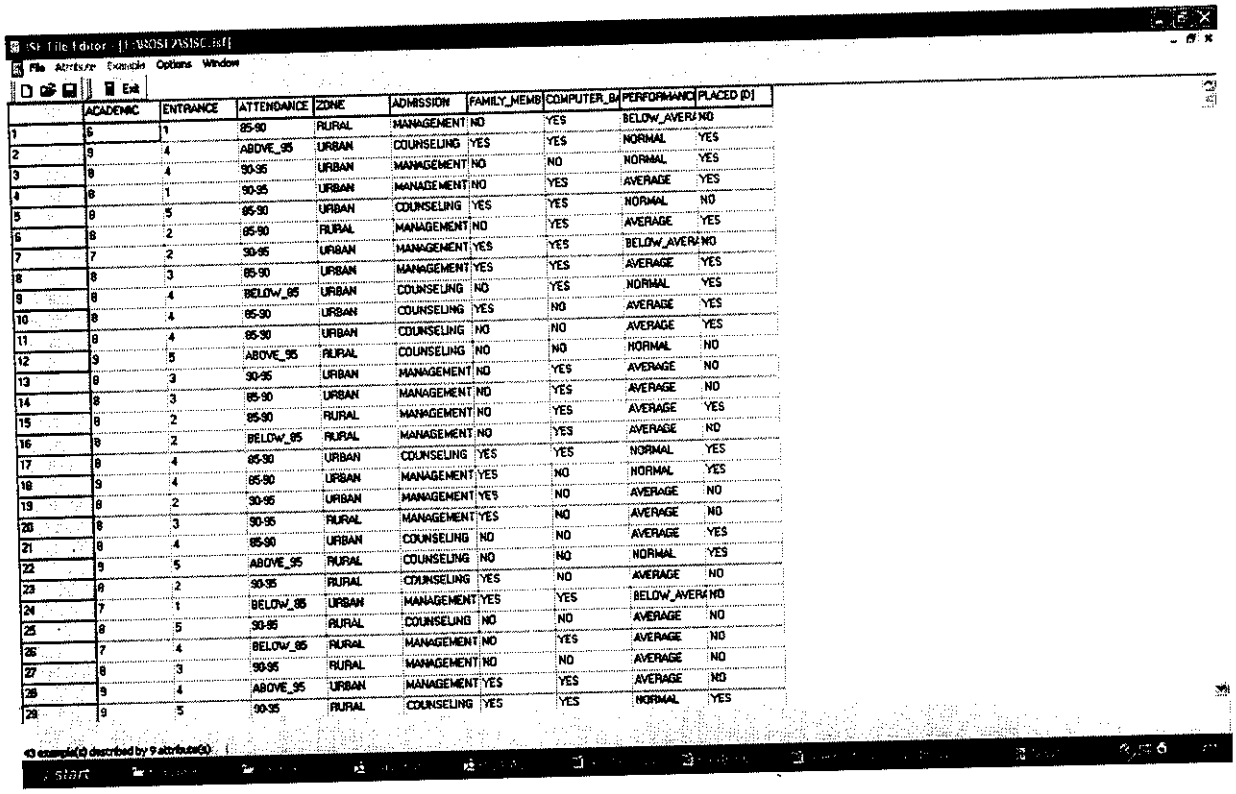
- The students having the academic marks in the range 85-95% and attendance percentage in the range 85-90% are classified into "normal" category.
- The students having academic marks in the range 75-85% and their mode of admission is management and if they have computer background in their previous course are classified into "average" category.
- The students having the academic marks in the range 55-65% are classified into "below average" category.

From the decision rules, it is easy to predict the performance of the future students.

7.2 FUTURE ENHANCEMENT

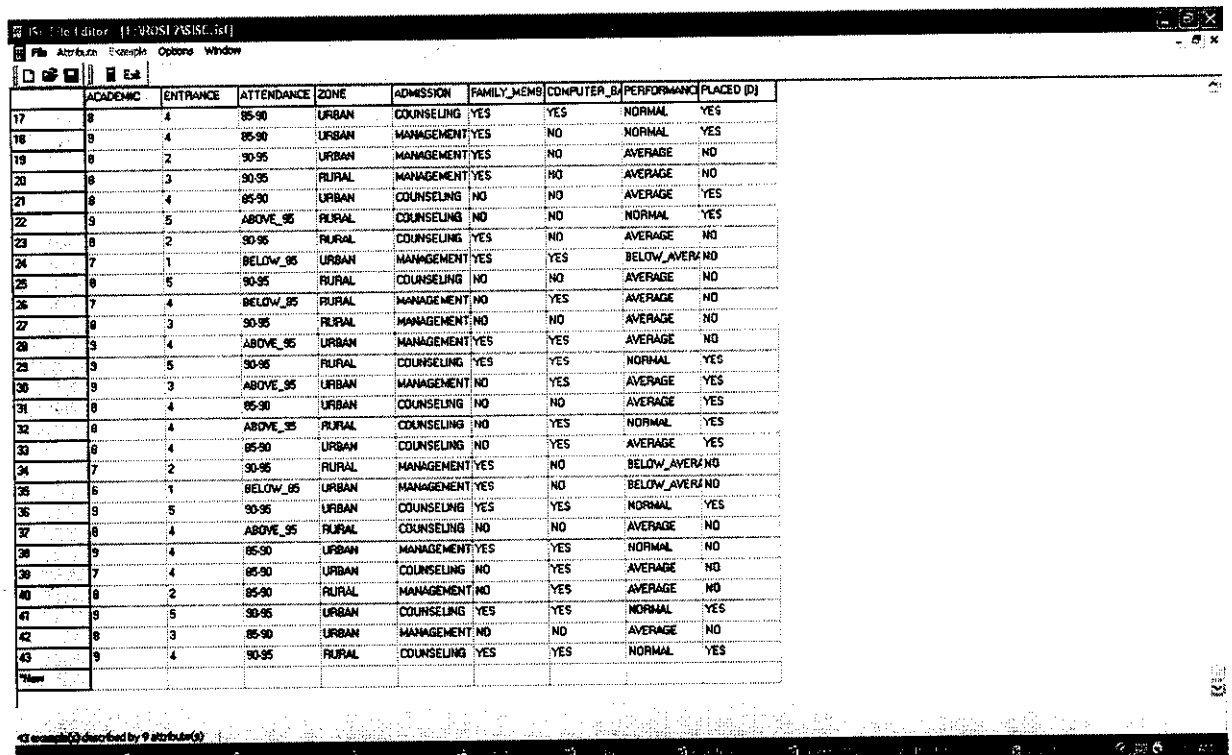
- The system uses only one database for mining. The system will also be able to incorporate the specification for importing any other database in runtime.
- The system can be made to compute the core and reducts of the attribute set of the given dataset.
- The system can be made to generate the decision rules automatically instead of ROSE2.

A.5 Dataset



	ACADEMIC	ENTRANCE	ATTENDANCE	ZONE	ADMISSION	FAMILY_MEMB	COMPUTER_BA	PERFORMANC	PLACED [D]
1	6	1	85-90	RURAL	MANAGEMENT	NO	YES	BELOW_AVERA	NO
2	9	4	ABOVE_95	URBAN	COUNSELING	YES	YES	NORMAL	YES
3	8	4	90-95	URBAN	MANAGEMENT	NO	NO	NORMAL	YES
4	9	1	90-95	URBAN	MANAGEMENT	NO	YES	AVERAGE	YES
5	8	5	85-90	URBAN	COUNSELING	YES	YES	NORMAL	NO
6	8	2	85-90	RURAL	MANAGEMENT	NO	YES	AVERAGE	YES
7	7	2	90-95	URBAN	MANAGEMENT	YES	YES	BELOW_AVERA	NO
8	8	3	85-90	URBAN	MANAGEMENT	YES	YES	AVERAGE	YES
9	8	4	BELOW_85	URBAN	COUNSELING	NO	YES	NORMAL	YES
10	8	4	85-90	URBAN	COUNSELING	YES	NO	AVERAGE	YES
11	8	4	85-90	URBAN	COUNSELING	NO	NO	AVERAGE	YES
12	9	5	ABOVE_95	RURAL	COUNSELING	NO	NO	NORMAL	NO
13	8	3	90-95	URBAN	MANAGEMENT	NO	YES	AVERAGE	NO
14	8	3	85-90	URBAN	MANAGEMENT	NO	YES	AVERAGE	NO
15	8	2	85-90	RURAL	MANAGEMENT	NO	YES	AVERAGE	YES
16	8	2	BELOW_85	RURAL	MANAGEMENT	NO	YES	AVERAGE	NO
17	8	4	85-90	URBAN	COUNSELING	YES	YES	NORMAL	YES
18	9	4	85-90	URBAN	MANAGEMENT	YES	NO	NORMAL	YES
19	8	2	90-95	URBAN	MANAGEMENT	YES	NO	AVERAGE	NO
20	8	3	90-95	RURAL	MANAGEMENT	YES	NO	AVERAGE	NO
21	8	4	85-90	URBAN	COUNSELING	NO	NO	AVERAGE	YES
22	9	5	ABOVE_95	RURAL	COUNSELING	NO	NO	NORMAL	YES
23	8	2	90-95	RURAL	COUNSELING	YES	NO	AVERAGE	NO
24	7	1	BELOW_85	URBAN	MANAGEMENT	YES	YES	BELOW_AVERA	NO
25	8	5	90-95	RURAL	COUNSELING	NO	NO	AVERAGE	NO
26	7	4	BELOW_85	RURAL	MANAGEMENT	NO	YES	AVERAGE	NO
27	8	3	90-95	RURAL	MANAGEMENT	NO	NO	AVERAGE	NO
28	9	4	ABOVE_95	URBAN	MANAGEMENT	YES	YES	AVERAGE	NO
29	9	5	90-95	RURAL	COUNSELING	YES	YES	NORMAL	YES

A.6 Dataset



	ACADEMIC	ENTRANCE	ATTENDANCE	ZONE	ADMISSION	FAMILY_MEMB	COMPUTER_BA	PERFORMANC	PLACED (D)
17	8	4	85-90	URBAN	COUNSELING	YES	YES	NORMAL	YES
18	9	4	85-90	URBAN	MANAGEMENT	YES	NO	NORMAL	YES
19	8	2	90-95	URBAN	MANAGEMENT	YES	NO	AVERAGE	NO
20	8	3	90-95	RURAL	MANAGEMENT	YES	NO	AVERAGE	NO
21	8	4	85-90	URBAN	COUNSELING	NO	NO	AVERAGE	YES
22	9	5	ABOVE_95	RURAL	COUNSELING	NO	NO	NORMAL	YES
23	8	2	90-95	RURAL	COUNSELING	YES	NO	AVERAGE	NO
24	7	1	BELOW_85	URBAN	MANAGEMENT	YES	YES	BELOW_AVERA	NO
25	8	5	90-95	RURAL	COUNSELING	NO	NO	AVERAGE	NO
26	7	4	BELOW_85	RURAL	MANAGEMENT	NO	YES	AVERAGE	NO
27	8	3	90-95	RURAL	MANAGEMENT	NO	NO	AVERAGE	NO
28	9	4	ABOVE_95	URBAN	MANAGEMENT	YES	YES	AVERAGE	NO
29	9	5	90-95	RURAL	COUNSELING	YES	YES	NORMAL	YES
30	9	3	ABOVE_95	URBAN	MANAGEMENT	NO	YES	AVERAGE	YES
31	8	4	85-90	URBAN	COUNSELING	NO	NO	AVERAGE	YES
32	8	4	ABOVE_95	RURAL	COUNSELING	NO	YES	NORMAL	YES
33	8	4	85-90	URBAN	COUNSELING	NO	YES	AVERAGE	YES
34	7	2	90-95	RURAL	MANAGEMENT	YES	NO	BELOW_AVERA	NO
35	6	1	BELOW_85	URBAN	MANAGEMENT	YES	NO	BELOW_AVERA	NO
36	9	5	90-95	URBAN	COUNSELING	YES	YES	NORMAL	YES
37	8	4	ABOVE_95	RURAL	COUNSELING	NO	NO	AVERAGE	NO
38	9	4	85-90	URBAN	MANAGEMENT	YES	YES	NORMAL	NO
39	7	4	85-90	URBAN	COUNSELING	NO	YES	AVERAGE	NO
40	8	2	85-90	RURAL	MANAGEMENT	NO	YES	AVERAGE	NO
41	9	5	90-95	URBAN	COUNSELING	YES	YES	NORMAL	YES
42	8	3	85-90	URBAN	MANAGEMENT	NO	NO	AVERAGE	NO
43	9	4	90-95	RURAL	COUNSELING	YES	YES	NORMAL	YES

This screen shot shows the two dimensional student table used for mining.

Glossary

The following terms are used throughout in this document and their explanation is given below.

Information system: An information system is four tuple T= (U, A, C, D) where U is the set of all objects taken into the account for mining, A is the set of all attributes. C and D is the subsets of A, called condition and decision attributes, where C and D are disjoint.

Dataset: A dataset is a 2-D table having rows and columns.

Object: An object in a dataset or information system is a record in the dataset.

Complete dataset: Simply, if every object in an information system has some value for all attributes, then the information system or the data set is said to complete.

Incomplete dataset: An information system or a dataset is said to be incomplete, if any one of the objects in the dataset has no value for at least one attribute.

Indiscernibility relation: For any subset P of A, a binary relation IND(P), called the indiscernibility relation is defined as IND(P) = {(x,y) $\in U \times U : a(x) = a(y)$ for all a in P, where A is the set of all attributes.

Characteristic relation: The *characteristic relation* $R(B)$ is a relation on U defined for x, $y \in U$ as follows

$(x, y) \in R(B)$ *if and only if* $y \in K_B(x)$.

Lower approximation of X: The lower approximation of X is the of all elements which certainly belong to the class X.

Upper approximation of X: The upper approximation of X is the set of all elements which contains both elements that belongs to X and does not belong to X.

Boundary of X: The boundary of X is the set of all elements that may or may not belong to X.

Exact: If the boundary region of X is empty, then the given class X is said to be exact.

Rough set: A class X is said to be rough if the boundary region of X is non-empty.

REFERENCES

REFERENCE BOOKS

- Roger S. Pressman, "Software Engineering, A practitioner's approach (2003)", by Tata McGraw-Hill Edition.
- Elias M Awad, "System Analysis and Design (1999)", by Galgotia Publications.
- Jittery R. Shapiro, "The complete reference Visual Basic 6.0 (2002)", by Tata McGraw-Hill publications.

PAPERS

- P. Ramasubramaniam, "Mining Analysis of SIS database using Rough Set Theory (2007)" – Journal DOI 10.1109/ICCIMA.2007. Page no 81-87
- "Three Approaches to Missing Attribute Values—A Rough Set Perspective" by Jerzy W. Grzymala-Busse

WEb SITES

- www.codeguru.com
- www.profsr.com
- www.expertexchange.com
- http://idss.cs.put.poznan.pl/site/rose.html