

**AN INDUCTIVE LEARNING ALGORITHM  
FOR  
CENSORED PRODUCTION RULE (CPR) DISCOVERY**

*Dissertation submitted in partial fulfillment  
of the requirements for the award of the  
degree of*

**MASTER OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & TECHNOLOGY**

**BY**

**BASHEER MOHAMAD AHMAD AL-MAQALEH**

**UNDER THE GUIDANCE OF**

**PROF. K.K. BHARADWAJ**



**SCHOOL OF COMPUTER & SYSTEM SCIENCES  
JAWAHARLAL NEHRU UNIVERSITY  
NEW DELHI- 110 067**

**JANUARY 2003**

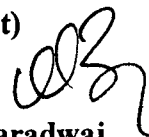
## CERTIFICATE

This is to certify that the dissertation entitled “An Inductive Learning Algorithm for Censored Production Rule (CPR) Discovery” being submitted by **Basheer Mohamad Ahmad Al-Maqaleh** to the School of Computer and System Sciences, Jawaharlal Nehru University, New Delhi, in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in Computer Science is a bonafied work carried by him under the guidance and supervision of **Prof. K.K. Bharadwaj**.


The matter embodied in the dissertation has not been submitted for the award of any other degree or diploma.



**Basheer Mohamad Ahmad Al-Maqaleh**  
(Student)



**Prof. K.K. Bharadwaj**  
(Supervisor)

  
6-01-03

**Prof. K.K. Bharadwaj**  
Dean, SC&SS  
Jawaharlal Nehru University  
New Delhi 110 067.

**Professor-K. K. Bharadwaj**  
Dean  
School of Computer & Systems Sciences,  
Jawaharlal Nehru University  
New Delhi - 110067

*...dedicated to my dear parents*

## ACKNOWLEDGEMENT

It is my privilege to have worked under the supervision of **Prof. K.K. Bharadwaj**, Dean of School of Computer & System Sciences, Jawaharlal Nehru University, New Delhi.

I am deeply indebted to him for his great help and support. This dissertation could have never been accomplished without his guidance and support.

I am also grateful to the faculty members of SC & SS for their insightful ideas and comments.

My thanks are also due to my dear **parents** who have spiritually supported me throughout my studies.

I highly appreciate the help and support shown by my friends. Special thanks are due to my friend researcher **Fadl Ba-Alawi** who has always helped me and supported me in many ways.



**Basheer Mohamad Ahmad Al-Maqaleh**

## ABSTRACT

Data mining or Knowledge Discovery in Databases (KDD) is the nontrivial extraction of implicit, previously unknown and potentially useful information from data. As a result useful relationships in the large databases are discovered as association rules.

In the recent years Data Mining has attracted a large number of researchers and several successful attempts have been made towards developing techniques for extracting interesting and previously unknown knowledge from databases.

Most of the approaches have underlying knowledge representations as standard production rules expressed as **IF** <condition> **THEN** <action> .

The standard production rules have a rigid structure and cannot handle uncertain, incomplete and imprecise knowledge with resource constraint.

Michalski and Winston extended these ordinary logic-based systems to exhibit variable precision in which certainty varies while specificity remains constant. As a vehicle for the concept of Variable Precision Logic (VPL) the Censored Production Rules (CPRs) with the underlying representation were developed.

**IF** <condition> **THEN** <action> **UNLESS** <sensor>

Where <sensor> is an exception to the rule. Such rules are employed in situations in which the conditional statement **IF** <condition> **THEN** <action>, holds frequently and the assertion <sensor> holds rarely. By using a rule of this type we are free to ignore the exception condition, when the resources needed to establish are tight or there is simply no information available as to whether it holds or not. Thus the **IF** <condition> **THEN** <action> part of the CPR expresses important

information while the UNLEES <sensor> part acts only as a switch which changes the polarity of <action> to  $\sim$ <action>.

The present work includes a study of inductive learning algorithms for data mining. Most of the approaches have standard Production Rules (PR) as underlying knowledge representation. In this work an Inductive Learning Algorithm is developed for the Censored Production Rules (CRP) Discovery. Adaptation of the algorithm to incremental learning of new sensors is also shown.

Experimental results are presented to demonstrate the performance of the proposed algorithm.

# CONTENTS

	<b>Page No.</b>
<b>1. INTRODUCTION</b>	<b>1-9</b>
1.1 Data Mining	1
1.1.2 Data Mining and Knowledge Discovery	2
1.1.3 Classification of Data Mining Systems	4
1.2 Data Mining Technologies	5
1.2.1 Query tools	6
1.2.2 Visualization techniques	6
1.2.3 Statistics	7
1.2.4 Online Analytical Processing Tools (OLAP)	7
1.2.5 k-nearest neighbor	7
1.2.6 Decision trees	8
1.2.7 Association Rules	8
1.2.8 Neural networks	9
1.2.9 Genetic algorithms	9
1.3 Censored Production Rule	9
<b>2. INDUCTIVE LEARNING</b>	<b>11-18</b>
2.1 Rule Induction Approach	11
2.1.1 ID <sub>3</sub> algorithm	12
2.1.2 AQ algorithm	13
2.1.3 CN2 algorithm	14
2.1.4 Other algorithm	14
2.2 The Inductive Learning Algorithm (ILA)	14
2.2.1 The Inductive Learning Algorithm (ILA)	15
2.2.2 A Description of the Inductive Learning Algorithm	16
2.2.3 Comparison of ILA and ID <sub>3</sub>	18

<b>3. EXTENDED ILA FOR CPR DISCOVERY</b>	<b>19-33</b>
3.1 Production Rule System	19
3.2 Variable Precision Logic	19
3.3 Censored Production Rules System	20
3.3.1 Example of Censored Production Rule System	21
3.3.2 The Augmented Unless Operator Makes Expectation Quantitative	23
3.4 The Extended Inductive Learning Algorithm (EILA)	26
3.4.1 General Requirements	27
3.4.2 The Extended Inductive Learning Algorithm (EILA)	27
3.5 Incremental Learning	33
<b>4. IMPLEMENTATION AND RESULTS</b>	<b>34-50</b>
4.1 Introduction to VC++ & ODBC	34
4.1.1 Desktop Data bases (FoxPro and Access)	34
4.1.2 User DSN (Data Source Name) Tab	35
4.1.3 Connection between VC++ and Microsoft Access	37
4.2 First Training (Phase)	39
4.3 Second Training (Phase)	44
4.4 Third Training (Phase)	47
4.5 Post Processing	50
<b>5. CONCLUSION</b>	<b>52</b>
Bibliography	53



# CHAPTER-1

## INTRODUCTION

### 1.1 Data Mining

Simply stated, data mining refers to extracting or ‘mining’ knowledge from large amounts of data. The term is actually a misnomer. Remember that the mining of gold from rocks or sand is referred to as gold mining rather than rock or sand mining.

Thus, data mining should have been more appropriately named “knowledge mining”, a shorter term, may not reflect the emphasis on mining from large amounts of data. Nevertheless, mining is a vivid term characterizing the process that finds a small set of precious nuggets from a great deal of raw material new line. Thus such a misnomer that carries both “data” and “mining” became a popular choice. There are many other terms carrying a similar or slightly different meaning to data mining, such as knowledge mining from databases, knowledge extraction data / pattern analysis, data archaeology, and data dredging (Figure 1.1). Although some define data mining to be the process of discovering interesting knowledge from large amounts of data stored either in databases, data warehouses, or other information repositories.

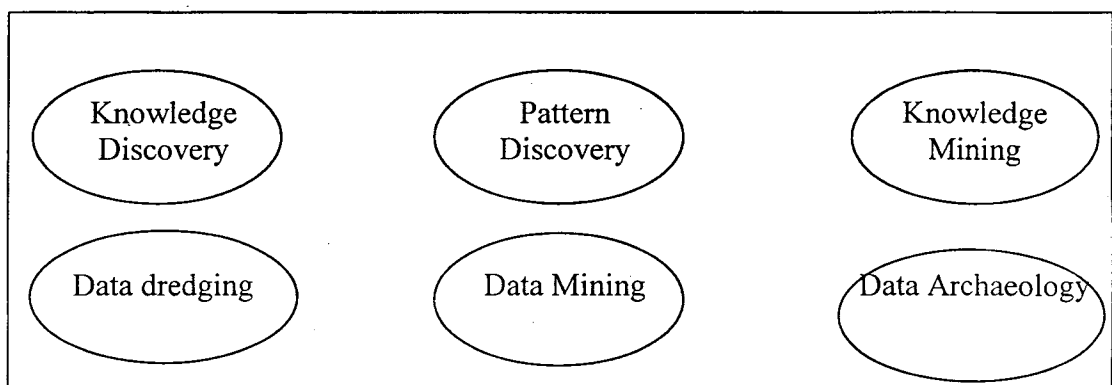


Figure 1.1 : Different definitions of data mining

Data mining can be viewed as a result of the natural evolution of information technology. An evolutionary path has been witnessed in the database industry in development of data collection, database creation, data management, data analysis and understanding.

Many industrial companies are approaching this important area and realize that data mining will provide an opportunity of major revenue.

### **1.1.2 Data Mining and Knowledge Discovery**

Many people treat data mining as a synonym for another popularly used term, Knowledge Discovery in Databases, or KDD [3].

Knowledge discovery as a process is depicted in (Figure 1.2). And consists of an iterative sequence of the following steps:

1. **Data cleaning** (to remove noise and inconsistent data)
2. **Data integration** (where multiple data sources may be combined)
3. **Data selection** (where data relevant to the analysis task are retrieved from the database).
4. **Data transformation** (where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations, for instance).
5. **Data mining** (an essential process where intelligent methods are applied in order to extract data patterns)
6. **Pattern evaluation** (to identify the truly interesting patterns representing knowledge based on some interestingness measures).

7. **Knowledge presentation** (where visualization and knowledge representation techniques are used to present the mined knowledge to the user).

The data mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user, and may be stored as new knowledge in the knowledge base. Note that according to this view, data mining is only one step in the entire process, albeit an essential one since it uncovers hidden patterns for evaluation.

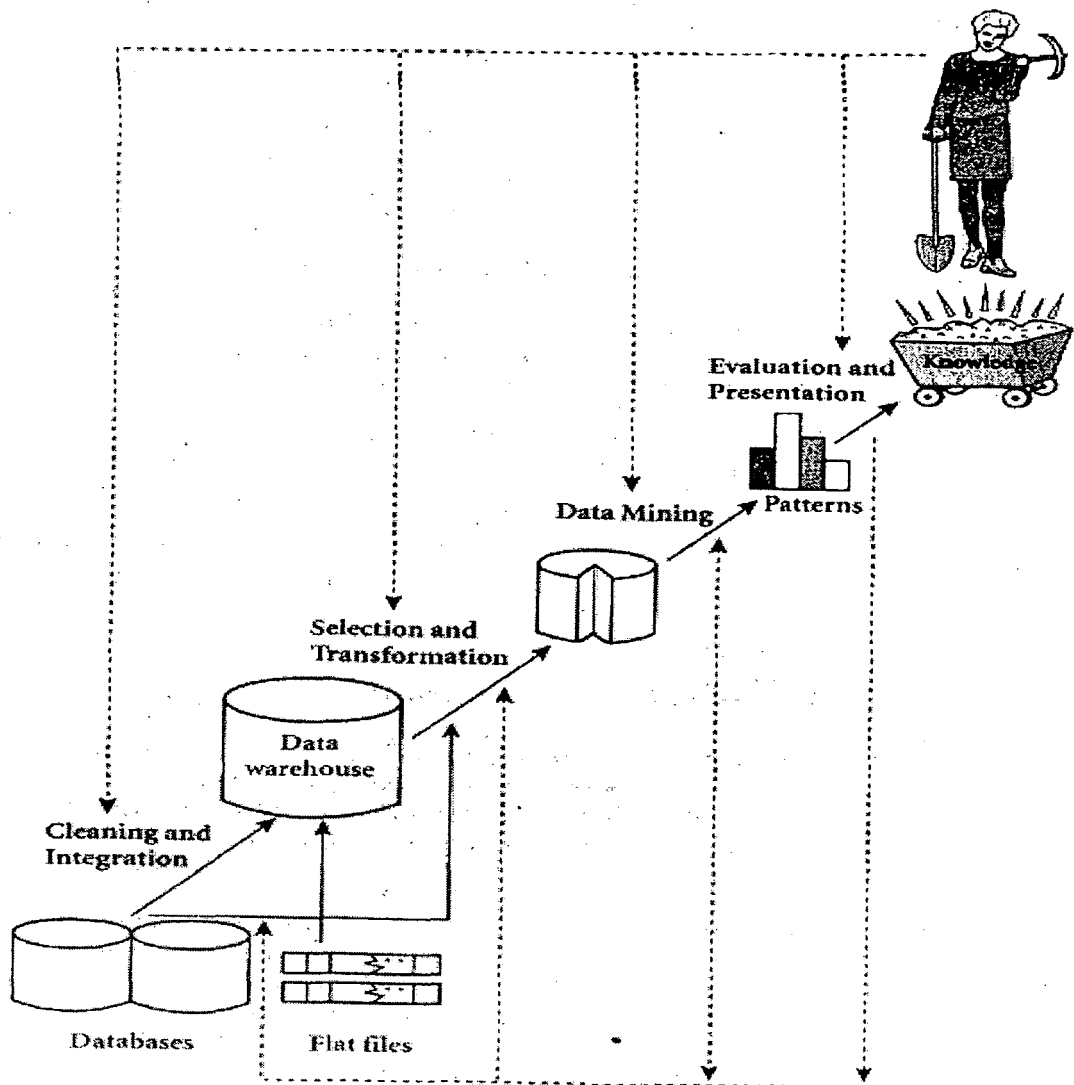
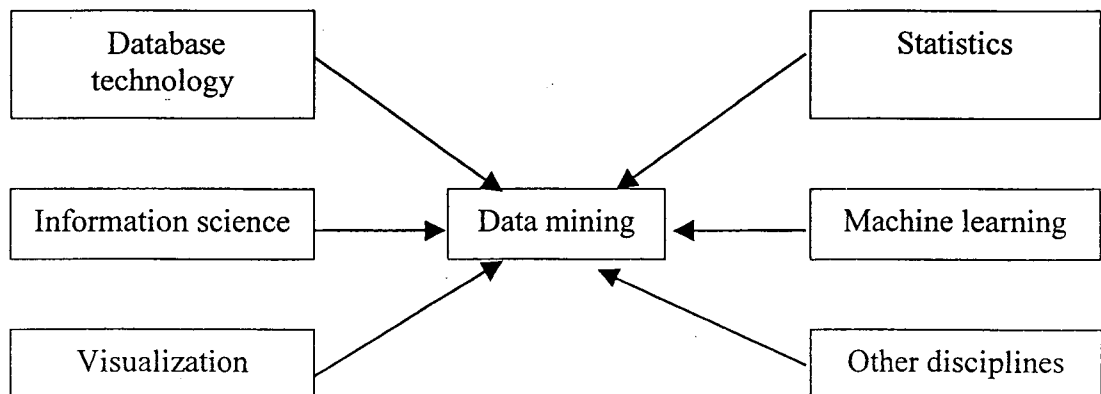


Figure 1.2 : Data Mining as a step in the process of knowledge discovery.

### 1.1.3 Classification of Data Mining Systems

Data mining is an interdisciplinary field, the confluence of a set of disciplines (Figure 1.3), including database systems, statistics, machine learning, visualization and information science [3]. Moreover, depending on the data mining approach, used techniques from other disciplines may be applied, such as neural networks, fuzzy and/or rough set theory, knowledge representation, inductive logic programming, or high performance computing. Depending on the kinds of data to be mined or on the given data mining application, the data mining systems may also integrate techniques from spatial data analysis, information retrieval, pattern recognition, image analysis, signal processing, computer graphic, Web technology, business, economics. Data mining systems can be categorized to various criteria, as follows [3].

- Classification according to the kinds of databases mined.
- Classification according to the kinds of knowledge mined.
- Classification according to the kinds of techniques mined.
- Classification according to the application adapted.



**Figure 1.3 : Data mining as a confluence of multiple disciplines**

## 1.2 Data Mining Technologies

During the last several years, data mining techniques have been used by companies to understand the demographics of their customers and to provide them with personalized interactions.

There are various data mining techniques that have been deployed in order to identify hidden trends and new opportunities within the data. These various data mining techniques have been embedded into software applications that process complex algorithms in order to provide meaningful information [7].

Data mining is not so much a single technique as the idea that there is more knowledge hidden in the data than shows itself on the surface. Any technique that helps extract more out of your data is useful, so data mining techniques form quite a heterogeneous group. Although various different techniques are used for different purposes, those that are of interest in the present context are [6,7]:

- Query tools
- Visualization
- Statistical techniques
- Online analytical processing (OLAP)
- Case based learning (K-nearest neighbor)
- Decision trees
- Association rules
- Neural networks

- Genetic algorithms

### **1.2.1 Query tools**

With an ad-hoc query application, users have the ability to access information on demand.

Just by applying simple structured query language (SQL) to a data set, you can obtain a wealth of information. However, before we can apply more advanced pattern analysis algorithms, we need to know some basic aspects and structures of the data set. With SQL we can uncover only shallow data, which is information that is easily accessible from the data set; yet although we cannot find hidden data, for the most 80% of the interesting information can be abstracted from a database using SQL. The remaining 20% of hidden information requires more advanced techniques, and for large marketing-driven organization, this 20% can prove of vital importance. For example, user creates and executes an ad-hoc query that answers the question, “How much revenue was generated by each customer during this year”? the results from the query would contain customer name and revenue for the year selected.

### **1.2.2 Visualization techniques**

Visualization techniques are a very useful method of discovering patterns in data sets, and may be used at the beginning of a data mining process to get a rough feeling of a quality of the data set and where patterns are to be found [6].

Interesting possibilities are offered by object-oriented three-dimensional tool kits, such as Inventor, which enables the user to explore three – dimensional structures interactively.

While an image that is produced provides another perspective of data relationships, visualization is often incorporated in data mining applications.

### **1.2.3 Statistics**

This data mining technique is the basis of all data mining techniques and requires individuals highly skilled in mathematics to build and interpret the results.

### **1.2.4 Online Analytical Processing Tools (OLAP)**

Information of this nature is called multidimensional and such relationships can not be easily analyzed when the table has standard two-dimensional representation. We need to explore the relationship between several dimensions and standard relational databases are not very good at this.

OLAP tools store wanted data in a special multi-dimensional format, often in memory, and a manager can ask any question at all, although the data cannot be updated [6].

OLAP can be an important stage in a data mining process, but OLAP tools do not learn, they create no new knowledge, and they cannot search for new solutions. Data mining is more powerful than OLAP. Another advantage is that data mining algorithms do not need a special form of storage, since they can work directly on data stored in a relational database.

### **1.2.5 k-nearest neighbor**

A class of learning algorithm that uses a simple form of table look-up to classify examples. For each new case, a constant number ( $k$ ) of examples that are closest to this case are selected. The classification of the new case is simply the average of the  $k$ -selected cases.

### 1.2.6 Decision trees

A decision tree is a flow-chart like tree structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distributions. The top-most node in a tree is the root node.

One of the approaches to inductive machine learning that is often used is to form a decision tree from a set of training examples.

### 1.2.7 Association Rules

Association analysis is the discovery of association rules showing attribute value conditions that occur frequently together in a given set of data. Association analysis widely used for market basket transaction data analysis [3].

More formally, association rules are of the form

$$X \rightarrow Y.$$

The association rule  $X \rightarrow Y$  is interpreted, as “database tuples that satisfy the conditions in  $X$  are also likely to satisfy the conditions in  $Y$ ”.

i.e.  $\text{age}(X, \text{“20...29”}) \wedge \text{income}(X, \text{“20K...29K”}) \rightarrow \text{buys}(X, \text{“CD player”})$

[Support = 2% confidence = 60%]

Rule support and confidence are two measures of rule interestingness:

$\text{Support}(A \rightarrow B) = [\text{tuples containing both A and B}] / [\text{total no. of tuples}]$ .

Support defines potential usefulness of a pattern or utility function.

$\text{Confidence}(A \rightarrow B) = [\text{tuples containing both A and B}] / [\text{tuples containing A}]$ .



It is the measure of certainty that assesses the validity or trustworthiness of the pattern.

### **1.2.8 Neural networks**

Typically, a neural network consists of a set of nodes: input nodes receive the input signals, output nodes give the output signals and a potentially unlimited number of intermediate layers contain in the intermediate nodes [6].

The network is first trained with training sets and examples. Once the learning is over, new patterns are given to the network. The network then uses its training experience and analyzes the new data. Data may be used for clustering identifying entities, deviation analysis and various other data mining tasks.

### **1.2.9 Genetic algorithms**

An optimization technique that use processes such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.

## **1.3 Censored Production Rule**

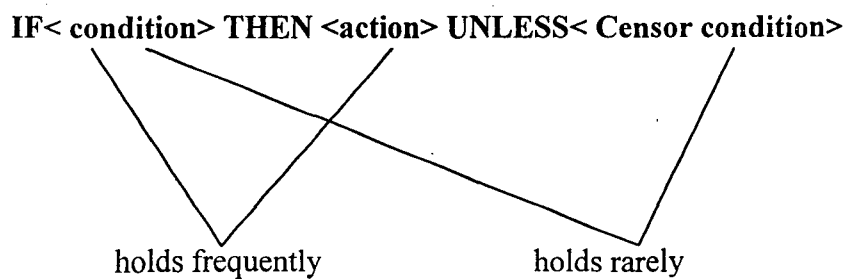
When a problem arises from the use of target knowledge, the chosen algorithm needs be revised so as to capture the source of errors and produce more sophisticated and appropriate rule.

Occurrence of most unusual cases is attributed to presence of noisy or exceptional data.

Researchers have comp up with various new designs and developed hybrid techniques or modified the exiting one in order to minimize the drawbacks and deliver a better result.

Censored Production Rule (CPR) suggested by Michalski and Winston (1986), is one such underlying knowledge representation technique. The rationale behind the introduction of the Censored Production Rule was to capture the uncertain and imprecise knowledge about the real world.

The general form of CPR is



The present work includes a study of inductive learning algorithms for data mining. Most of the approaches have standard Production Rules (PR) as underlying knowledge representation. In this work an Inductive Learning Algorithm is developed for the Censored Production Rules (CRP) Discovery. Adaptation of the algorithm to incremental learning of new censors is also shown.

Experimental results are presented to demonstrate the performance of the proposed algorithm.

## CHAPTER-2

# INDUCTIVE LEARNING

### 2.1 Rule Induction Approach

It is a data mining technique that induces values inherent within the data. The rules are used to understand the relationships that exist.

It is a technique for discovering a set of “IF/Then” rules from data to classify the different cases. Because it looks for all possible interesting patterns in a data set, the technique is powerful. But it can be overwhelming with the large number of rules that can be generated. Because the rules are independent many may contradict each other and they may not cover all possible situations.

Decision rules may be induced from training data in a bottom-up specific-to-general style, or in a top-down general-to-specific style.

Most decision rule modeling systems employ a search process to evolve this set of highly specific and individual instance to more general rules.

This search process is iterative, and usually terminates when rules can no longer be generalized, or some other alternate stopping criteria satisfied. Rule induction methods attempt to find a compact or covering rule set that completely partitions the examples into their correct classes. The covering set is found by heuristically searching for a single or best rule that covers cases for only one class. Having found a (best) conjunctive rule for a class C, the rule is added to the rule set, and the cases satisfying it are removed from further consideration. The process is repeated until no cases remain to be covered.

Various inductive learning systems are developed and discussed in the machine learning literature. These algorithms generate different types of rule sets like unordered rule sets, ordered rule set or decision lists and decision trees. Concepts of some of the well known inductive learning systems are briefly described below.

### **2.1.1 ID3 algorithm**

Very simply Quinlan's, ID3 algorithm [Quinlan, 1983] builds a decision tree from a fixed set of example. The resulting tree is used to classify future samples. The example has several attributes, and belongs to a class (like yes or no). The leaf nodes of the decision tree contain the class name whereas a non-leaf node is a decision node. The decision node is an attribute test with each branch (to another decision tree) being a possible value of the attribute. ID<sub>3</sub> uses information gain to help it decide which attribute goes into a decision node [4].

The advantage of learning a decision tree is that a program, rather than acknowledge engineer, elicits knowledge from an expert. ID<sub>3</sub> improves on Concept Learning System (CLS) algorithm by adding a feature selection heuristic. ID<sub>3</sub> searches through the attribute of the training instance and extracts the attribute that best separates the given example. If the attribute perfectly classifies the training sets then ID<sub>3</sub> stops; Otherwise it recursively operates on the  $n$  (where  $n$  = number of possible value of an attribute) partitioned subsets to get their 'best' attribute. The algorithm uses a greedy search, that is, it picks the best attribute and never looks back to reconsider earlier choices.

ID<sub>3</sub> is a nonincremental algorithm, meaning it derives its classes from a fixed set of training instance.

ID<sub>3</sub> has two new features that improved the algorithm and is extended as ID<sub>4</sub> and ID<sub>5</sub> algorithms successively.

ID3 algorithm is a top down, nonbacktracking decision tree algorithm. One of the problems with ID3 algorithm is that the decision tree produced over fits the training examples because it performs a stepwise splitting that attempts to optimize at each individual split, rather than on an overall basis [McKee, 1995]. This leads to decision trees that are too specific because they used unnecessary or irrelevant conditions. Hence this affects the ability to classify unknown examples or examples with incomplete attributes [8].

Another problem with ID3 algorithm relates to the fact that for application involving a large number of training examples which cannot be kept in computers main memory at once. The algorithm can work with a “representative” sample from the training set, called windowing, which however cannot guarantee to yield the same decision tree as would be obtained from the complete set of training examples. In this case the decision tree would be unable to classify all examples correctly [Carter and Catlett, 1987].

### **2.1.2 AQ algorithm**

AQ, another well-known inductive learning algorithm [8]. The original AQ does not handle uncertainty well. Existing AQ such as AQ15 [Michalski et al, 1986], is a modified version of the original and handle noise with pre and post processing techniques. The basic AQ algorithm performs heuristic search through a space of logical expression to determine those that account for all positive and no negative examples. The AQ algorithm when generating a conjunction of attribute value conditions (called a complex) also performs a general to specific search for the best complex. As a result, AQ searches only the space of complexes that are completely consistent with the data.

### **2.1.3 CN2 algorithm**

CN2 algorithm [Clark and Nibett, 1989], which is an adaptation of the AQ algorithm, retains the same heuristic search method of the AQ algorithm but on the other hand, removes its dependence on specific examples during search and also extends AQ's search space to inductive rules that do not perform perfectly on the training data [8]. Both AQ and CN2 are induction systems that are regarded as non-decision tree approaches.

### **2.1.4 Other algorithms**

Other algorithms include OCI [Murthy, Kasif and Salzberg 1994], which is a system for induction of oblique decision trees suitable for domains where attributes have numeric values, and RULES [Pham and Aksoy, 1995], which is a rule induction algorithm with an ability to classify unseen examples. The disadvantage of RULES lies in the increased number of rules generated to handle such data.

## **2.2 The Inductive Learning Algorithm (ILA)**

A production rule induction system ILA (Inductive Learning Algorithm) [Tolun, M.R. and S.M. Abu-Soud 1998] which produces IF-THEN rules directly from a set of training examples in a general-to-specific way [8]. ILA eliminates all unnecessary and irrelevant conditions from the extracted rules and therefore its rules are more simple and general than those obtained from ID3 and AQ. ILA also produces rules fewer in number than ID3 and AQ most of the time. The generality of rules increases the classification capability of ILA. A rule becomes more general as the number of conditions on its IF part becomes fewer. A general rule also helps in classifying incomplete examples in which one or more attributes may be unknown. They also embody the general patterns within the database. The rule can

be used to interpret and understand the active mechanisms underlying the database [8].

### **2.2.1 The Inductive Learning ALgorithm (ILA)**

**Step 1:** Partition the table which contains  $m$  examples into  $n$  sub-table. One table for each possible value of the class attribute.

(Step 2 through 8 are repeated for each sub table)

**Step 2:** Initialize attribute combination count  $J$  as  $J = 1$

**Step 3:** For the sub-table under consideration, divide the attribute list into distinct combinations, each combination with  $J$  distinct attribute.

**Step 4:** For each combination of attribute, count the number of occurrences of attribute values that appear under the same combination of attribute in unmarked rows of the sub-table. Call the first combination with the maximum number of occurrences as max-combination.

**Step 5:** If max-combination =  $\phi$  increase  $J$  by 1 and go to step-3.

**Step 6:** Mark all rows of the sub table, in which the values of max combination appear, as classified.

**Step 7:** Add a rule to  $R$ , whose left hand side comprise attribute names of max-combination with their values separated by AND operator(s) and its right hand side contains the decision attribute value associated with the sub-table.

**Step 8:** If all rows are marked as classified, then move on to process another sub table and go to step 2, otherwise go to step 4. If no sub tables are available, exit with the set of rules obtained so far.

### 2.2.2 A Description of the Inductive Learning Algorithm

Let us consider the training set for object classification given in Table 2.1, consisting of 7 examples (i.e.  $m = 7$ ) with three attributes ( $k=3$ ) and one decision (class) attribute with two possible values, {yes, no} ( $n=2$ ) in this example, “size”, “color”, and “shape” are attribute with sets of possible values {Small, medium, large}, {red, blue, green} and {brick, wedge, sphere, pillar} respectively [8].

**TABLE 2.1:** Object Classification Training Set [Thorton, 1992]

Examples	Size	Color	Shape	Decision
1	medium	blue	bridge	yes
2	small	red	wedge	no
3	small	red	sphere	yes
4	large	red	wedge	no
5	large	green	pillar	yes
6	large	red	pillar	no
7	large	green	sphere	yes

Since  $n$  is two, the first step of the algorithm generates two sub-tables which are shown in Table 2.2.



**TABLE 2.2:** Sub-Tables of the Training Set Partitioned According to Decision Classes

<b>Sub-Table 1</b>					
<b>Example No.</b>		<b>Size</b>	<b>Color</b>	<b>Shape</b>	<b>Decision</b>
<b>Old</b>	<b>New</b>				
1	1	medium	blue	brick	yes
3	2	small	red	sphere	yes
5	3	large	green	pillar	yes
7	4	large	green	sphere	yes
<b>Sub-Table 2</b>					
<b>Example No.</b>		<b>Size</b>	<b>Color</b>	<b>Shape</b>	<b>Decision</b>
<b>Old</b>	<b>New</b>				
2	1	small	Red	wedge	no
4	2	large	red	wedge	no
6	3	large	red	pillar	no

After applying the whole algorithm, the algorithm generates these rules

- 1 - IF color is green THEN the decision is yes
- 2 - IF size is medium THEN the decision is yes
- 3 - IF shape is sphere THEN the decision is yes
- 4 - IF shape is wedge THEN the decision is no
- 5 - IF size is large AND color is red THEN the decision is no

### 2.2.3 Comparison of ILA and ID3

Several distinctions between ILA and ID3 are pointed out earlier in section 2.2. For comparison purpose, the rules resulting from applying ID3 on the same training set and the ones produced by ILA are presented in Table 2.3.

**TABLE 2.3:** A Comparison Between Rules Generated by ID3 and ILA

Algorithm	Rule No	Rule
ID3 ILA	1	IF color = green AND shape = Pillar THEN yes IF color = green THEN yes
ID3 ILA	2	IF shape = brick THEN yes IF size = medium THEN yes
ID3 ILA	3	IF share = sphere THEN yes IF shape = sphere THEN yes
ID3 ILA	4	IF shape = wedge THEN no IF shape = wedge THEN no
ID3 ILA	5	IF color=red and shape = pillar THEN no IF size = large AND color = red THEN no

It is evident from Table 3 that the two algorithms generate the same number of rules but Rule 1 extracted by ILA is simpler than the same rule generated by ID3 because the latter has an unnecessary conditions (i.e. shape = pillar) [8].

Clearly rules 2, and 5 are also different in both sets of rules but with the same level of complexity. However, ILA could generate these same two rules, as for example, attribute value 'brick' was one of the choice. But we gain nothing if we change this choice since in both algorithms the two rules have the same level of specificity and classify the respective examples correctly.

## CHAPTER-3

### EXTENDED ILA FOR CPR DISCOVERY

#### 3.1 Production Rule System

A Production Rules System can capture much of the simple human problem solving capability effectively [1,2]. However not all human problem solving methods are easily Representable in the Production rules system because

- (i) for each rule information has to exist in the system somewhere as to its context use.
- (ii) sets of rules in Production Rules system have no intrinsic structure, which makes management of large knowledge bases difficult.
- (iii) the matching involved in the match-select-fire is an inherently inefficient computational process.
- (iv) the relatively simple syntax of a Production Rule is unable to capture the inherent uncertainty present in the real world knowledge.

#### 3.2 Variable Precision Logic

Variable Precision Logic (VPL) [Michalski & Winston, 1986] is concerned with problems of reasoning with incomplete information, subject to resource constraints and the problem of reasoning efficiently with exceptions [5]. VPL offers mechanisms for handling trade-off-between the precision of inferences and computational efficiency of deriving them. Specificity and certainty are the two aspects of precision. Certainty refers to the degree of belief in a statement, whereas specificity refers to the degree of detail of a description.

Following Michalski and Winston (1986), a system that gives more specific answers given more time (or resource in general) is called "variable specificity

system”. A system, that gives more certain answers, given more times, called a “variable certainty system”. There can be various combinations of the two systems, reflecting that specificity and certainty are inversely related. We can gain specificity at the expense of certainty, or vice versa.

Consider a query and “What is John doing?” given that it is Sunday. The quickest possible answer to this query be that “Probably John is working in the yard”. A more certain answer taking into account the nice weather is that “Certainly John is working in the Yard rather than reading’. Similarly, considering the time of the year, a more specific answer may be that and “John is raking leaves’.

### 3.3 Censored Production Rule System

To capture the uncertain and imprecise knowledge about the real world Michalski and Winston have introduced the concept of Censored Production Rule (CPR) as an underlying representational and computational mechanism to enable logic based system to exhibit variable precision in which certainty varies while specificity stays constant [1,2].

These Censored Production Rules are created by augmenting ordinary Production Rules with an exception condition and are written in the form “**IF A THEN B UNLESS C**”, where C is exception condition. Such rules are employed institutions in which the conditional statement ‘**IF A THEN B** holds frequently, and the assertion C holds rarely.

By using a rule of this type, we are free to ignore the Censor (exception) condition, when the resources needed to establish its presence are right, or simply no information is available as to whether it holds or does not hold [1,2]. As time permits, the Censor condition C is evaluated establishing the conclusion B with higher certainty if codes not hold, or simply changing the polarity of B to  $\sim B$  if C holds). From a logical viewpoint, the unless operator between B and C acts as the exclusive-or operator,  $A \rightarrow B \oplus C$ .

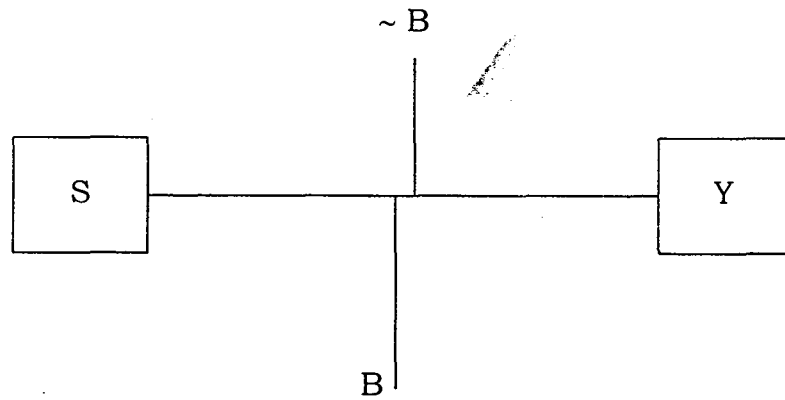
### 3.3.1 Examples of Censored Production Rule System

As an example of a Censored Production Rule. **IF** Sunday **THEN** John works in the yard **UNLESS** weather is bad,

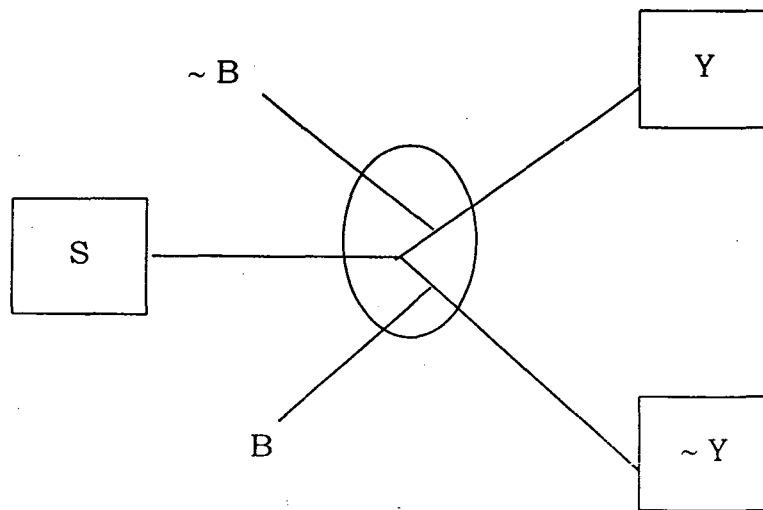
$$S \Rightarrow Y \lfloor B \quad (\lfloor \text{denotes UNLESS operator})$$

has the interpretation that if it is Sunday and weather is good, John will work in the yard, and if it is Sunday and the weather is bad, John will not work in the yard (Figure 3.1) [5].

TH-10214



a - Passive (ignore censor)



b - Active (evaluate censor)

Figure 3.1: The passive (a) and active (b) definitions of a censored rule.



A CPR may have more than one exception-denoting censor [2,5]. Consider for example the assertion that birds fly [Michalski and Winston, 1986].

$\forall x \text{ is-bird } (x) \Rightarrow \text{flies } (x)$

But not all birds fly. For example, penguins, ostriches, emus, kiwis, and domestic turkeys do not fly.

To include the information we write ( $\perp$  denotes UNLESS operator):

$\forall x \text{ is-bird } (x) \Rightarrow$

$\text{flies } (x) \perp \text{ is-penguin } (x)$   
 $\vee \text{ is-ostrich } (x)$   
 $\vee \text{ is-emus } (x)$   
 $\vee \text{ is-kiwi } (x)$   
 $\vee \text{ is-domestic-turkey } (x)$

Suppose we generalize these exceptions into one statement for special birds. Then we can write.

$\forall x \text{ is-bird } (x) \Rightarrow \text{flies } (x) \perp \text{ is-special-bird } (x).$

But then the rule is still not entirely correct. Even a flying bird cannot fly when it is dead or sick or has broken wings.

$\forall x \text{ is-bird } (x) \Rightarrow \text{flies } (x) \perp (\text{is-special-bird } (x) \vee \text{ is-in-unusual-condition } (x))$

where

$\forall x \text{ is-special-bird } (x) \Leftarrow$

$\vee \text{ is-penguin } (x)$

$v \text{ is- ostrich } (x)$

$v \text{ is- emu } (x)$

$v \text{ is- kiwi } (x)$

$v \text{ is-domestic-turkey } (x)$

and

$\forall x \text{ is - in - unusual- condition } (x) \Leftarrow$

$v \text{ is-dead } (x)$

$v \text{ is- sick } (x)$

$v \text{ has- broken- wings}(x)$

Now a bird also cannot fly when its legs are stuck in concrete. This case may also be classified as bird in an unusual condition. Thus, to update our knowledge, we need not change our rule; we need only extend our definition of unusual condition.

$\forall x \text{ is - in - unusual- condition } (x) \Leftarrow$

$v \text{ is-dead } (x)$

$v \text{ is- sick } (x)$

$v \text{ has- broken- wings}(x)$

$v \text{ has-legs-stuck-in-concrete}(x).$

Thus CPRs facilitate small rule repairs.

### 3.3.2 The Augmented Unless Operator Makes Expectations Quantitative

Consider the following rule:

$$S \Rightarrow Y \perp B$$

where  $S$  is a premise,  $Y$  is a decision, and  $C$  is a censor. Although the unless operator,  $\perp$ , is logically equivalent to the commutative exclusive-or operator, the unless operator has an expositive aspect which is not commutative. In order to

capture the asymmetry precisely, let us associate two parameters,  $\gamma_1$  and  $\gamma_2$  with the above rule

$$S \Rightarrow Y \perp B : \gamma_1, \gamma_2$$

Both  $\gamma_1$  and  $\gamma_2$  are point probabilities, one indicating the strength of the relationship between S and Y and the other between S and B.

Now consider the following sets:  $\Omega$  is a finite sample of events,  $\Omega_s$  is the set of events for which S holds,  $\Omega_{sy}$  is the subset of events for which both S and Y hold,  $\Omega_{sB}$  is a subset of events for which both S and B hold. Given these sets, the parameters  $\gamma_1$  and  $\gamma_2$  are defined as follows:

$$\gamma_1 = \frac{P_r[S, Y]}{P_r[S]} = P_r[Y/S] \approx \frac{|\Omega_{sy}|}{|\Omega_s|}$$

$$\gamma_2 = \frac{P_r[S, B]}{P_r[S]} = P_r[B/S] \approx \frac{|\Omega_{sB}|}{|\Omega_s|}$$

where  $|\Omega_i|$  denotes the cardinality of  $\Omega_i$ . Also we assume that  $\Omega_y \cap \Omega_B = \emptyset$  and  $\Omega_y \cup \Omega_B = \Omega_s$ ; thus  $P_r[Y/S] + P_r[B/S] = 1$

Relating these definitions to our example about John working in the yard unless the weather is bad-we can say that

$\Omega$ : is a set of days over a sufficiently large period of time.

$\Omega_s$ : is the set of Sundays during this period of time.

$\Omega_{sy}$ : is the set of Sundays when John works in the yard.

$\Omega_{sB}$ : is the set of Sundays with bad weathers.



Assuming that there are significantly more Sundays when John works in the yard than there are Sundays when the weather is bad, then  $\Omega_{SY}$  is considerably larger than the set  $\Omega_{SB}$ :

$$|\Omega_{SY}| \gg |\Omega_{SB}|$$

Thus, we have  $\gamma_1 \gg \gamma_2$

In our example,  $\gamma_1$  stands for the ratio of Sundays when John worked in the yard to all Sundays and,  $\gamma_2$  stands for the ratio of Sunday with bad weather to all Sundays.

Consequently the sum of  $\gamma_1 + \gamma_2$  must always equal 1

Under the Dempster – shafer interpretation of VPL, four belief values are associated with each Censored Production Rule (CPR) [Bharadwaj et al., 1994, Haddaury, 1987].

$$P \rightarrow D \lfloor C : \alpha, \beta, r, \delta$$

Such that

$$(I) \quad P \& \sim C \rightarrow D : \alpha \text{ i.e. Prob } (D|P \& \sim C) \in [\alpha \dots 1]$$

$$(II) \quad P \& C \rightarrow \sim D : \beta \text{ i.e. Prob } (\sim D|P \& C) \in [\beta \dots 1]$$

$$(III) \quad P \rightarrow D : r \quad \text{i.e. Prob } (D|P) \in [\gamma \dots 1]$$

$$(IV) \quad P \rightarrow \sim D : \delta \quad \text{i.e. Prob } (\sim D|P) \in [\delta \dots 1], \text{ where } \gamma + \delta \leq 1.$$

For example, the following rule might be used to express the fact that I read the paper before going to work unless I oversleep, which occurs once or twice a week.

Weekday – morning  $\rightarrow$  Read-paper  $\lfloor$  oversleep: 0.9, 1,0.6,0.2 where the belief factors, are interpreted as follows [2]:

- the “0.9” states that on weekday morning when I do not oversleep

I read the paper at least 0.9 of the time because there are other factors which would keep me from reading paper; such as the paper boy throwing it on the roof, which are not begin considered;

- the “1” states that on weekday mornings when I oversleep I certainly do not read the paper; and
- the “0.6” states that I read the paper at least three out of five weekday mornings (because I oversleep at most twice a week).
- the “0.2” states that I do not read the paper at least one out of five weekday mornings (because I oversleep at least once a week).

### **3.4 The Extended Inductive Learning Algorithm (EILA)**

The Extended Inductive Learning Algorithm (EILA) is an extension of Inductive Learning Algorithm (ILA) to discover Censored Production Rules (CPRs).

EILA an inductive algorithm for generating a set of classification rules for a collection of training examples. The algorithm works in an iterative fashion, each iteration searching for a rule that covers a large number of training examples as a single class. Having found a rule, EILA removes that example it covers from the training set by marking them and appends a rule at the end of its rule set.

EILA to be described in section 3.4.2 starts processing the training data by dividing the example set into sub-tables for each different class attribute value. After that EILA choose two sub-tables such that class attribute values are complimentary. EILA check if their exit two sub-table above then computes the number of rows in each sub-table.

Suppose the following rules are discovered;

Rule 1 - **IF  $\sim$ C THEN D**

Rule 2 - **IF C THEN  $\sim$ D**

The above rules confirm that attribute C is a Censor, EILA ignores the column corresponding to the Censor attribute C, and we get new rule

Rule 3 - **IF P THEN D**

and then EILA construct the following Censored Production Rule (CPR) from Rule 1, Rule 2 and Rule 3 as

**IF P THEN D UNLESS C**

### **3.4.1 General requirements**

1. The examples are to be listed in a table where each row corresponds to an example and each column contains attribute values.
2. A set of  $m$  training examples, each example composed, of  $k$  attributes, and a class attribute with  $n$  possible decisions.
3. A rule set , R, with an initial value of  $\emptyset$ .
4. All rows in the table are initially unmarked.

### **3.4.2 The Extended Inductive Learning Algorithm (EILA)**

**Steps 1:** Partition the table which contains  $m$  examples into  $n$  sub-tables. One table for each possible value of the class attribute.

**Steps 2:** Choose two sub-tables ,Sub-Table1, and Sub-Table2 such that class attribute values are complementary.

i.e. D for Sub-Table1 , and  $\sim$ D for Sub-Table2, if there doesn't exist such pair of tables then Flag=1 and go to step 4.

**Step 3:** Compute the number of rows  $m_1$  and  $m_2$  in Sub-Table1, Sub-Table2 respectively, and apply this formula

IF [ $m_2 \leq (m_1+m_2) * 0.25$ ) OR  $m_1 \leq (m_1+m_2)* 0.25$ ] then Flag =2.

**Step 4:** Initialize attribute unmarked  $j$  as  $j=1$ .

**Step 5:** For the sub-table under consideration, divide the attribute list into distinct combination, each combination with  $j$  distinct attribute.

**Step 6 :** If Flage=2 then for each attribute in Sub-Table1 with the same value ,say Atr-val having decision D and value,  $\sim$  Atr-val in Sub-Table 2 having decision  $\sim$ D ignore values of the attribute in Sub-Table 2 with value Atr-val.

**Step 7:** For each combination of attribute, count the number of occurrences of attribute values that appear under the same combination of attributes in unmarked rows of the sub-table under consideration but at the same time that shouldn't appear under the same combination of attributes of other sub-tables call the combination, with the maximum number of occurrences as max-combination. In case there are more than one attributes with the equal number of occurrences then choose all the attributes together as max-combination.

**Step 8:** If max-combination =  $\emptyset$  increase  $j$  by 1 and go to step 5.

**Step 9:** Mark all rows of the sub-table under consideration , in which the values of max-combination appear, as classified.

**Step 10:** Add a rule to R whose left hand side comprises attribute of max-combination with their values separated by AND operator(s) and its right hand side contains the decision attribute value associated with the sub-table.

**Step 11:** If all rows are marked as classified the move on to process another sub-table and go to step 4. Otherwise (i.e., if there are still unmarked rows) go to step 7. If no sub-tables are available (Flag=1), exit with set of rule obtained so far.

**Step 12 :** For each pair of discovered rules of the forms :

**IF ~ Atr-val THEN D**

**IF Atr-val THEN ~D**

confirm that Atr-val is a Censor. Ignore the column(s) corresponding to the Censor attribute(s) Atr-val.

**Step 13:** Apply the steps 4 through 11 for sub-table has large number of rows (ignore another sub-table which has fewer rows).

**Step 14:** IF (Flag=2) then

(a):- choose two Rule1 and Rule 2 discovered from Sub-Table1 and

Sub-Table2 respectively, having the structure given below:

Rule 1- **IF ~C THEN D**

Rule 2 - **IF C THEN ~D**

The above rules confirm that attribute C is a Censor. Ignore the column corresponding to the Censor attribute C and we get

Rule 3 - **IF P THEN D**

(b): Construct Censored Production Rule from Rule 1, Rule 2 and Rule 3 as

**IF P THEN D UNLESS C.**

**For example**

As an illustration of the operation of EILA, let us consider the training set for object classification given in Table 3.1, consisting of 12 example (i.e.  $m = (m_1 + m_2) = 12$ ) with two attributes ( $k = 2$ ) and are decision (class) attribute with two possible values (Fly, Dont Fly) ( $n = 2$ ).

**TABLE 3.1 : Object Classification Training Set**

<b>Example No.</b>	<b>Bird</b>	<b>Penguin</b>	<b>Decision</b>
1	Yes	No	Fly
2	Yes	Yes	Dont Fly
3	Yes	Yes	Dont Fly
4	Yes	No	Fly
5	Yes	No	Fly
6	Yes	No	Fly
7	Yes	No	Fly
8	Yes	No	Fly
9	Yes	No	Fly
10	Yes	No	Fly
11	Yes	No	Fly
12	Yes	No	Fly

Since  $n$  is two, the first step of the algorithm generates two sub-tables which are shown in Table 3.2.

**TABLE 3.2:** Sub -Tables of the Training Set Partitioned According To Decision Classes

Sub -Table 1				
Examples No old            new		Bird	Penguin	Decision
1	1	Yes	No	Fly
4	2	Yes	No	Fly
5	3	Yes	No	Fly
6	4	Yes	No	Fly
7	5	Yes	No	Fly
8	6	Yes	No	Fly
9	7	Yes	No	Fly
10	8	Yes	No	Fly
11	9	Yes	No	Fly
12	10	Yes	No	Fly
Sub- Table 2				
2	1	Yes	Yes	Dont Fly
3	2	Yes	Yes	Dont Fly

here  $m=(m_1+m_2) = 12, m_1 = 10, m_2 = 2$

Referring to Table 3.2

The Condition  $[m_2 \leq (m_1+m_2) * 0.25 \text{ OR } m_1 \leq (m_1+m_2) * 0.25]$  holds because

$[2 \leq (12 * 0.25)], 2 \leq 3$  then EILA is applied to discover CPR.

$$\text{Also } \frac{|\text{Bird} \wedge \text{Fly}|}{|\text{Bird}|} \gg \frac{|\text{Bird} \wedge \text{Penguin}|}{|\text{Bird}|}$$

$$\frac{|10|}{|12|} \gg \frac{|2|}{|12|}$$

$$0.83 \gg 0.16, \text{ True}$$

Thus formula (condition) is satisfied and after applying (EILA) we would get the following rules.

**IF** Penguin = No **THEN** Decision = Fly

**IF** Penguin = Yes **THEN** Decision = Dont Fly

**IF** Bird = Yes **THEN** Decision = Fly

The algorithm then finally produce single rule Censored Production Rule (CPR) from the above three rules:

**IF** Bird =Yes **THEN** Decision = Fly **UNLESS** Penguin

Suppose in Table 3.2 Sub-Table 1 has 8 rows and Sub-Table 2 has 4 i.e.

$$m=(m_1+ m_2)= 12, m_1 = 8, m_2 = 4.$$

Here the condition

$[m_2 \leq (m_1+m_2)* 0.25$  OR  $m_1 \leq (m_1+m_2)x 0.25]$  is not satisfied because

$$4 \leq (12 \times 0.25)$$

$$4 \leq 3$$

$$\text{Also } \frac{|\text{Bird} \wedge \text{Fly}|}{|\text{Bird}|} \gg \frac{|\text{Bird} \wedge \text{Penguin}|}{|\text{Bird}|}$$

$$\frac{|8|}{|12|} \gg \frac{|4|}{|2|}$$

$$0.66 \gg 0.33 \quad ? \text{ (false)}$$

Therefore in this case, no Censored Production Rule (CPR) is discovered and we get Production Rule (PR):

**IF** Bird = Yes **THEN** Decision = Fly.



### 3.5 Incremental Learning

Suppose through initial data mining we discovered CPR:

**IF P THEN D UNLESS C1 (1)**

and at later stage we discover following CPR through further data mining applied to new data:

**IF P THEN D UNLESS C2 (2)**

Then on a Post Processing we produce the single CPR from (1), (2) as

**IF P THEN D UNLESS C1 V C2 (3)**

**For example:**

Suppose from the first training EILA generates

**IF Bird = Yes THEN Decision = Fly UNLESS Penguin**

and from the second training EILA generates

**IF Bird = Yes THEN Decision = Fly UNLESS Dead**

and from the third training EILA generates

**IF Bird = Yes THEN Decision = Fly UNLESS Broken\_Wings v Legs-stuck-in-concrete**

Then EILA makes Post Processing from the above trainings and generates one Censored Production Rule (CPR) as

**IF Bird = Yes THEN Decision = Fly UNLESS Penguin v Dead v Broken\_Wings v Legs-stuck-in-concrete.**

All these procedures will be explained in chapter 4 through the implementation and results screens.

## **CHAPTER-4**

# **IMPLEMENTATION AND RESULTS**

### **4.1 Introduction to VC++ & ODBC**

C++ Windows developers already possess valuable knowledge of object-oriented programming in the Windows environment. However, many C++ Programmers lack knowledge of database technology. Knowledge of database technologies is crucial for building software for business applications as well as for many scientific applications.

The storing of data is an essential part of most software applications. Virtually all C++ application have the need to persist, or store, data of some kind.

A C++ Programmer is usually confident of his ability to write Software. After all, if you can master a language as complex and powerful as C++, you can no doubt write any Software, tool you need, including your own database system.

#### **4.1.1 Desktop Databases (FoxPro and Access)**

Desktop databases is a class of database software, sometimes called Indexed Sequential Access Method (ISAM) data bases because they use ISAM files. These include Microsoft Access, Microsoft FoxPro.

C++ programs can use the ODBC (Open Database Connectivity) API (Application Programming Interface) to talk to desktop databases. For instance, a C++ Program can call ODBC API functions to store and retrieve data in a Microsoft Access database file. You can even use ODBC to send language statements to the Access

interpreter and then retrieve any data that Access might return as a result of that operation.

An ODBC User data source stores information about how to connect to the indicated data provider. A user data source is only visible to you, and can only be used on the current machine.

#### 4.1.2 User DSN (Data Source Name) Tab

Adds, deletes or sets up data source with User DSNs screen (4-1). These data source are local to a computer, and can be used only by the current user.

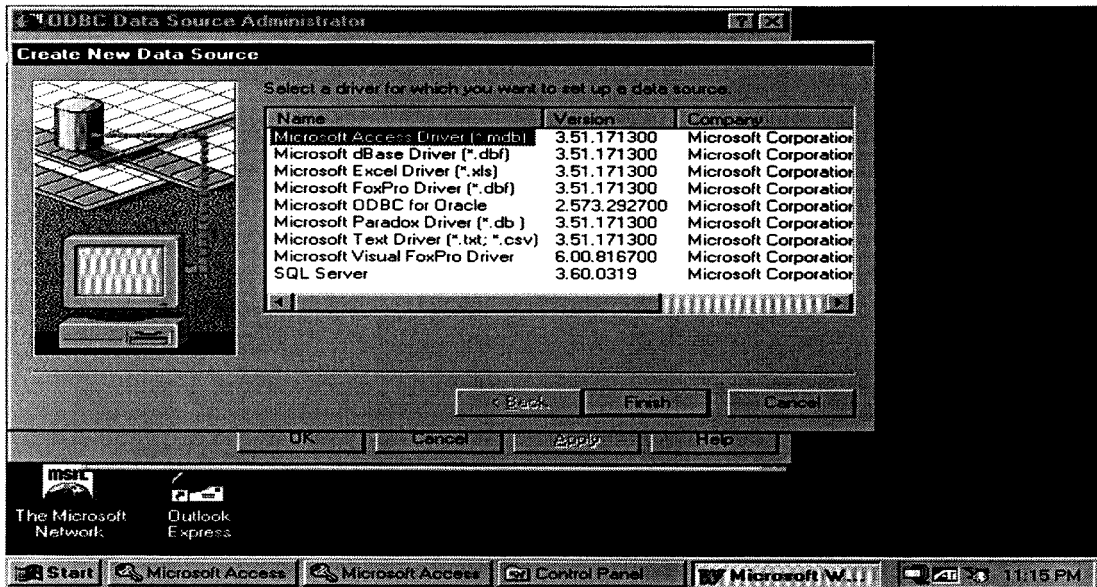
<b>Option</b>	<b>Description</b>
Configure	<p>Displays the driver specific data source setup dialog box that enables you to change the configuration of an existing user data source.</p> <p>You must select the name of a user data source from the list before clicking <i>configure</i>.</p>
Add	<p>Adds a new user data source. When you click <i>Add</i>, the <i>create New Data Source</i> dialog base appears with a list of drivers. Choose the driver for which you are adding a user data source.</p> <p>After you click <i>Finish</i>, a driver, specific setup dialog box appears.</p> <p>For my project the name of use data sources db2.mdb</p>
Remove	<p>Removes an existing user data source.</p>



Screen (4-1)

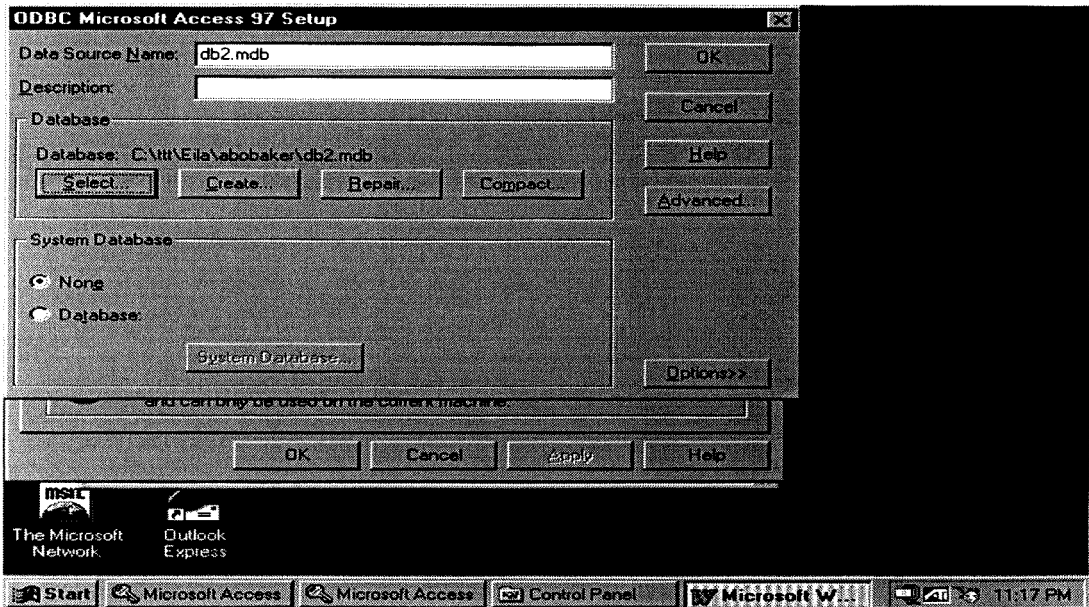
### 4.1.3 Connection between VC++ and Microsoft Access

After selecting db2. mdb from screen (4-1) (all Training Sets store in db2.mdb) click *Add* a new screen comes up screen (4-2).



Screen (4-2)

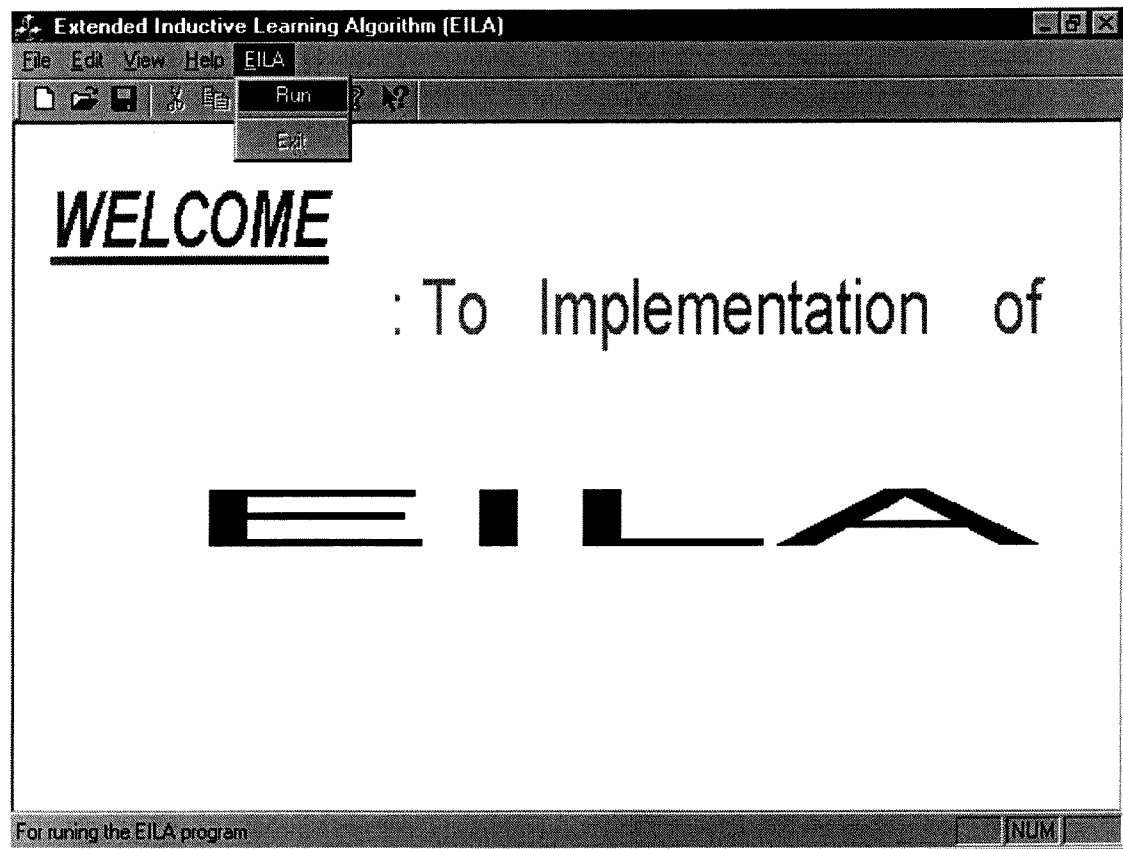
Select Microsoft Access Driver ( \*.mdb ) from screen (4-2) and click *Finish*, a new screen comes up screen (4-3).



Screen (4-3)

Write the name of Data Source Name (db2.mdb) and if *select* button is clicked from screen (4-3) drivers and directories will be accessed after that click *ok*.

After executing the Program, the main screen (4-4) appears.



Screen (4-4)

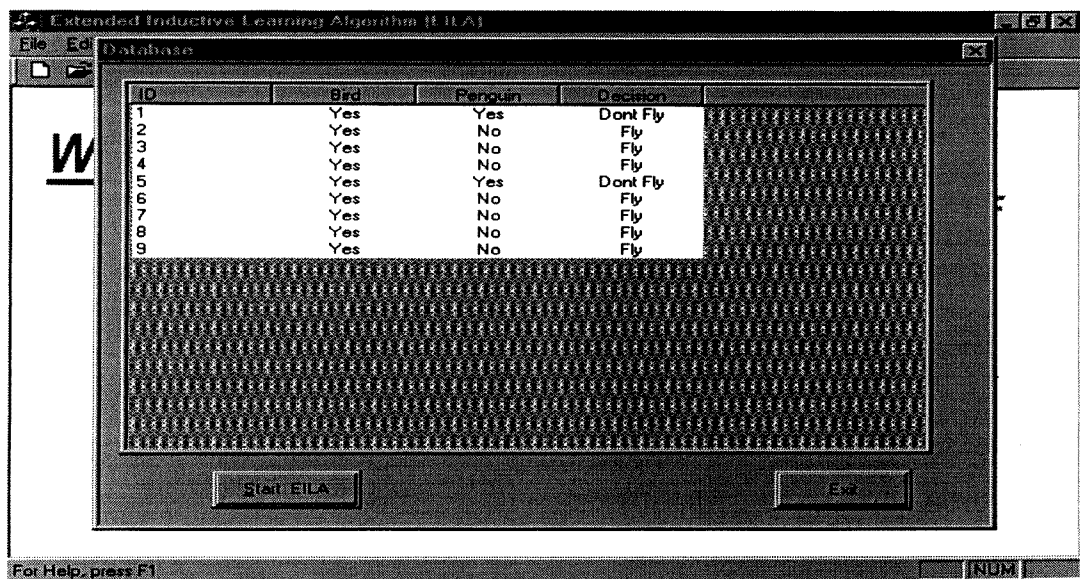
## 4.2 First Training (Phase)

Suppose we have the following Training Set Table 4.1 as input for the EILA:

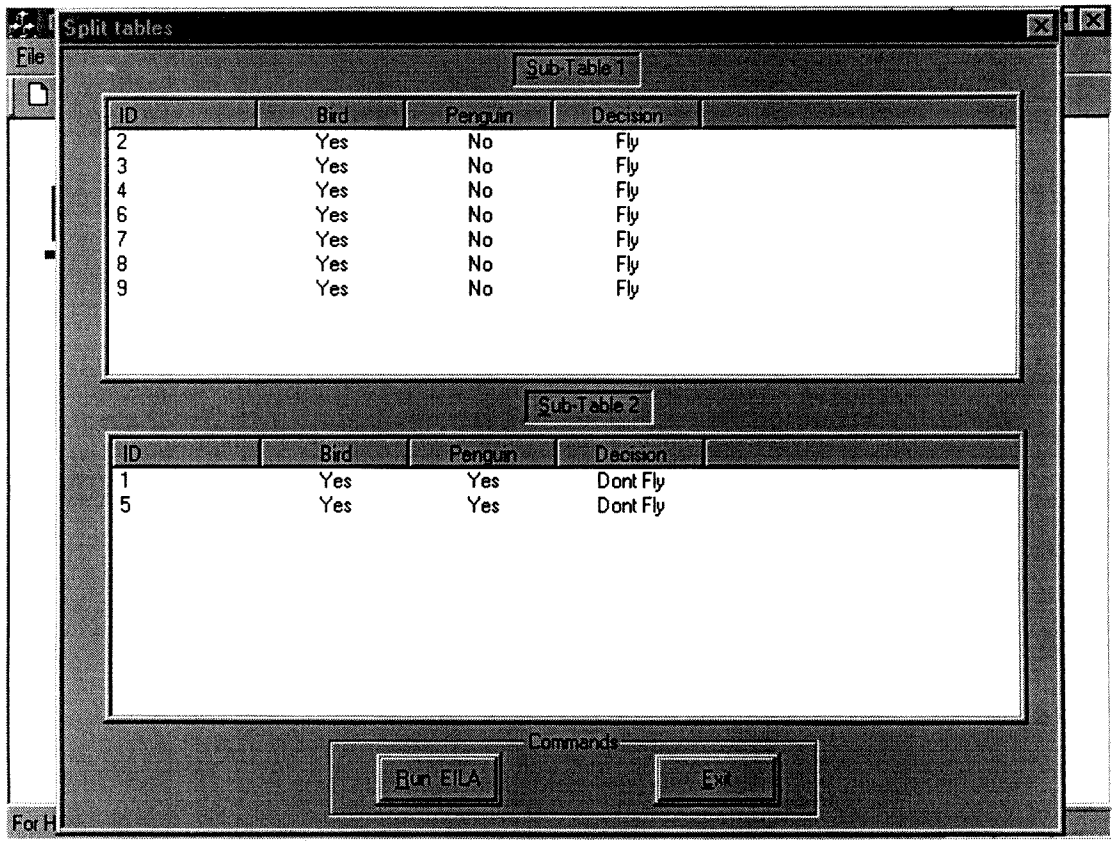
TABLE 4.1- Object Classification Training Set

ID	Bird	Penguin	Decision
1	Yes	Yes	Don't Fly
2	Yes	No	Fly
3	Yes	No	Fly
4	Yes	No	Fly
5	Yes	Yes	Dont Fly
6	Yes	No	Fly
7	Yes	No	Fly
8	Yes	No	Fly
9	Yes	No	Fly

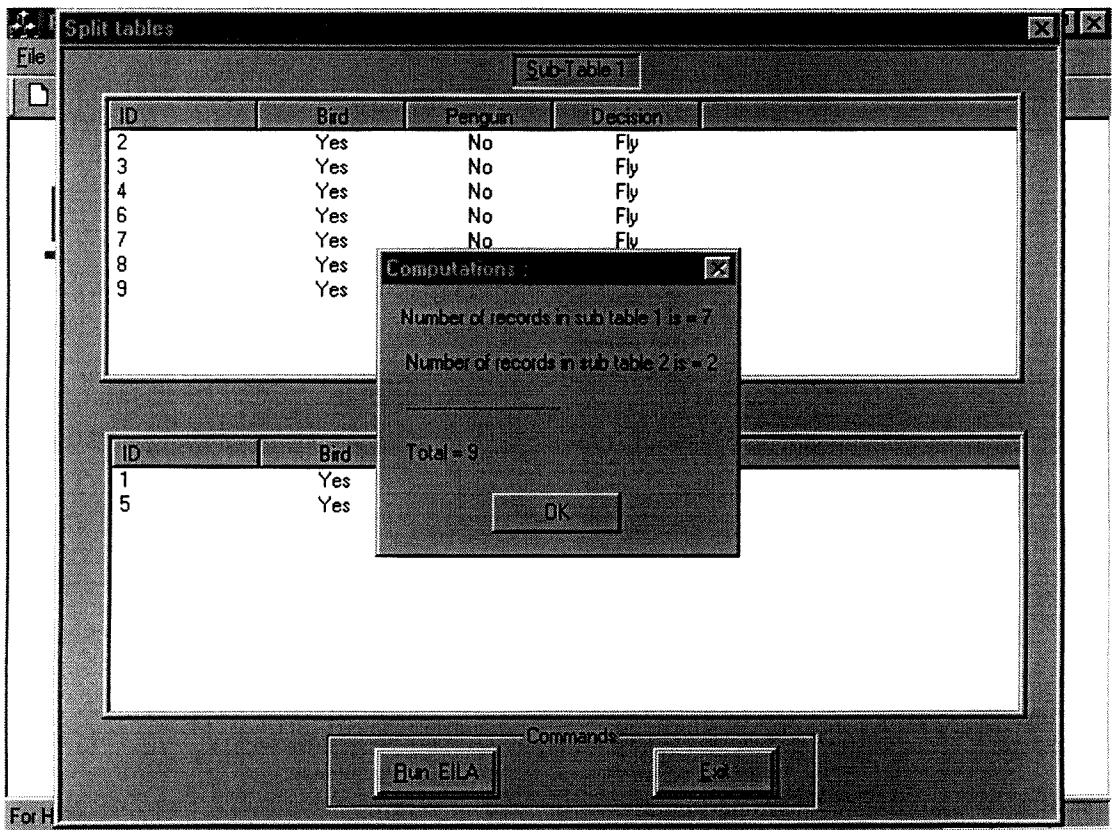
When we run the EILA program the display of input data, split into Sub-Table1 and Sub-Table2, number of rows in each sub-table and the final display for construction of Censored Production Rule (CPR) are shown in the following screens (4-5), (4-6), (4-7) and (4-8) respectively.



Screen (4-5)

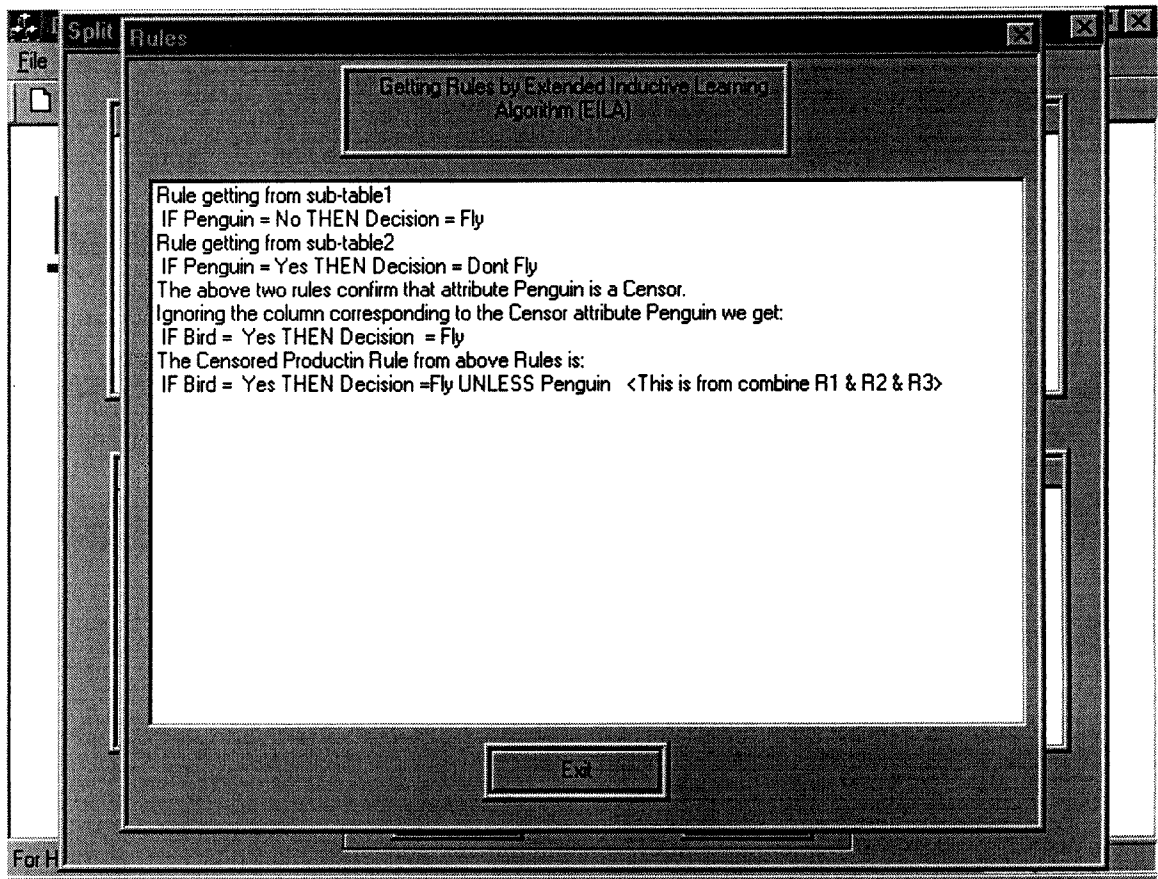


Screen (4-6)



Screen (4-7)





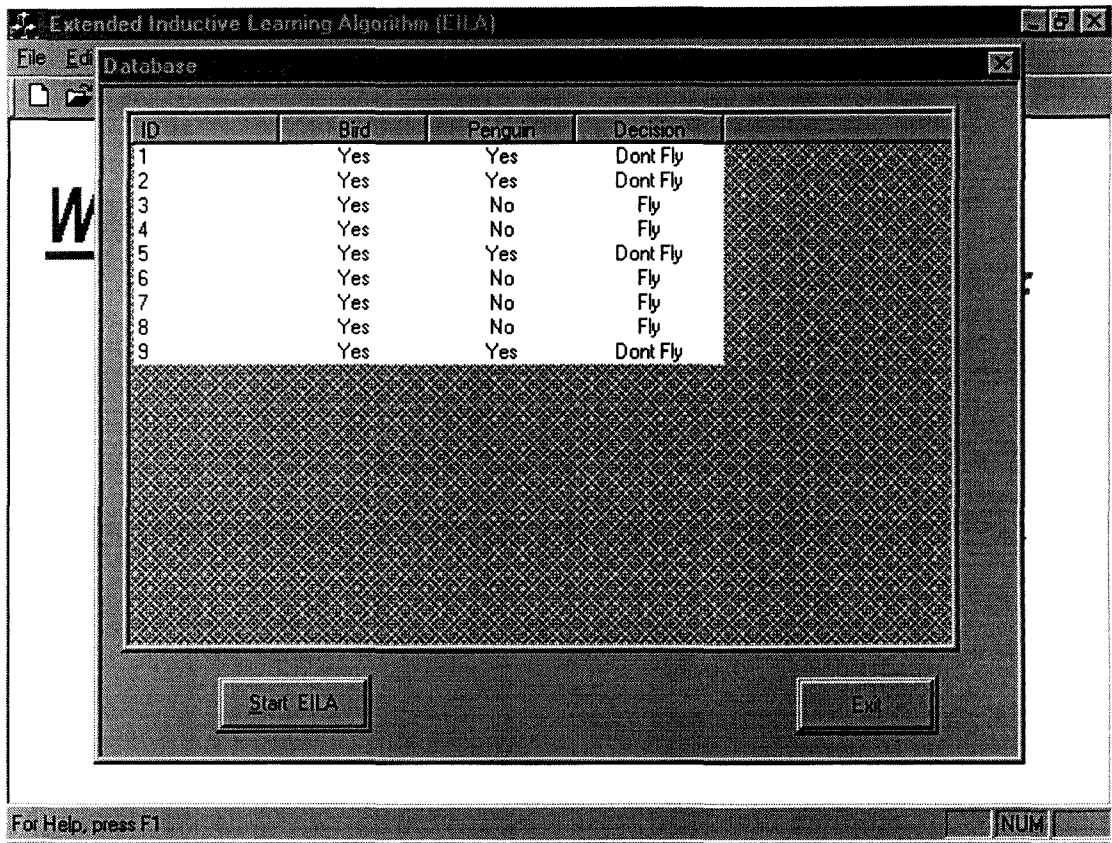
Screen(4-8)

Suppose we have new Training Set Table 4.2

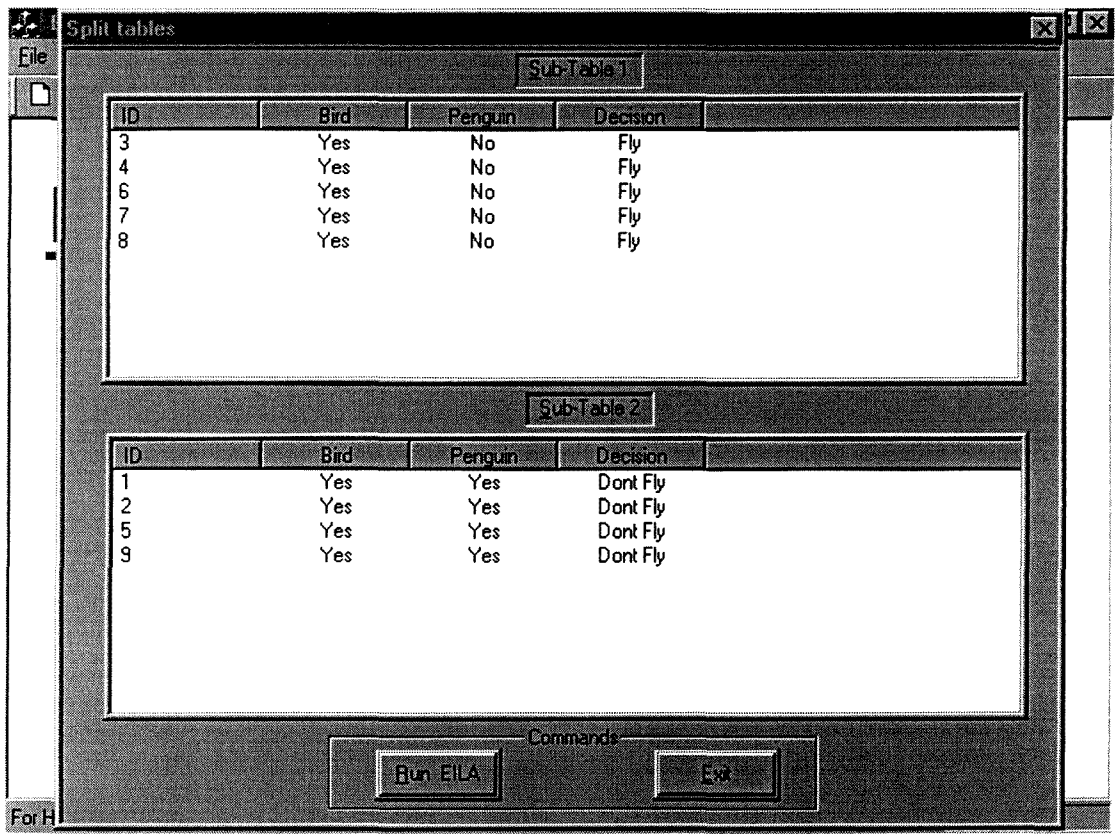
**TABLE 4.2 – Object Classification Training Set**

ID	Bird	Penguin	Decision
1	Yes	Yes	Dont Fly
2	Yes	Yes	Dont Fly
3	Yes	No	Fly
4	Yes	No	Fly
5	Yes	Yes	Dont Fly
6	Yes	No	Fly
7	Yes	No	Fly
8	Yes	No	Fly
9	Yes	Yes	Dont Fly

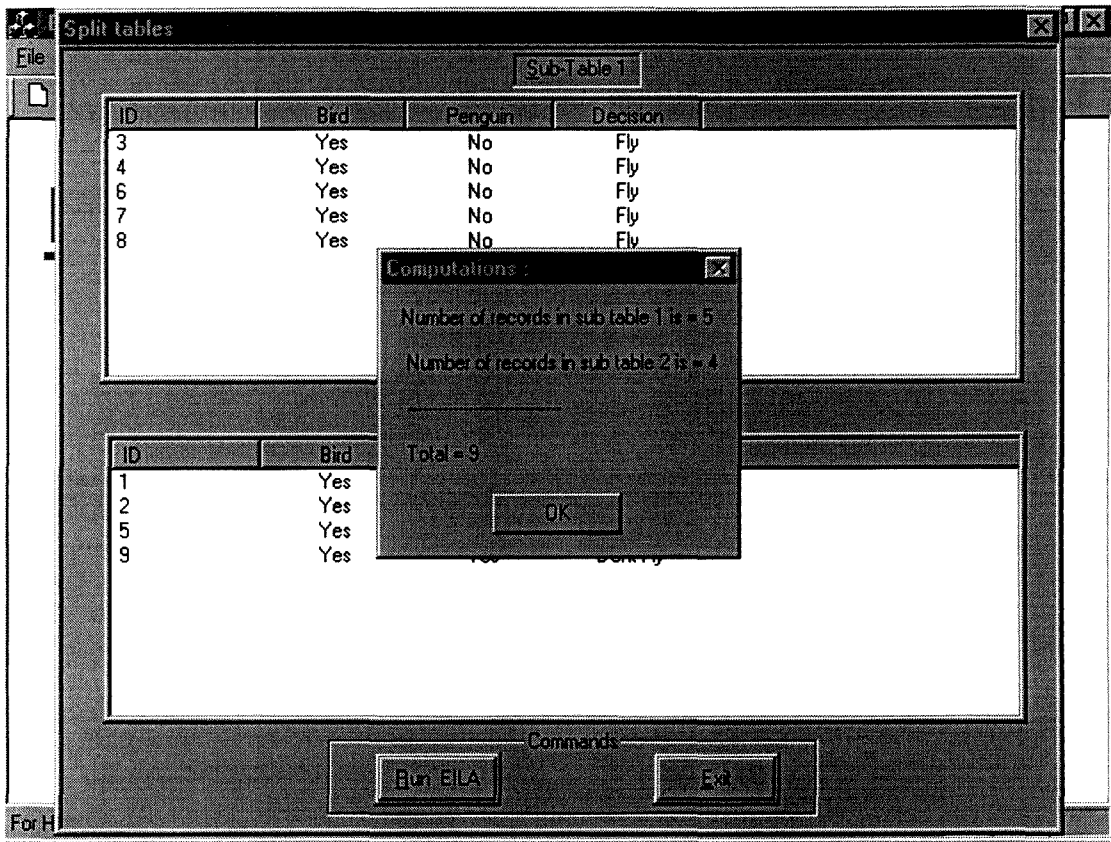
The following screens (4-9), (4-10), (4-11) and (4-12) would be displayed showing finally that screen (4-12) discovers only Production Rule (PR) and not Censored Production Rule (CPR).



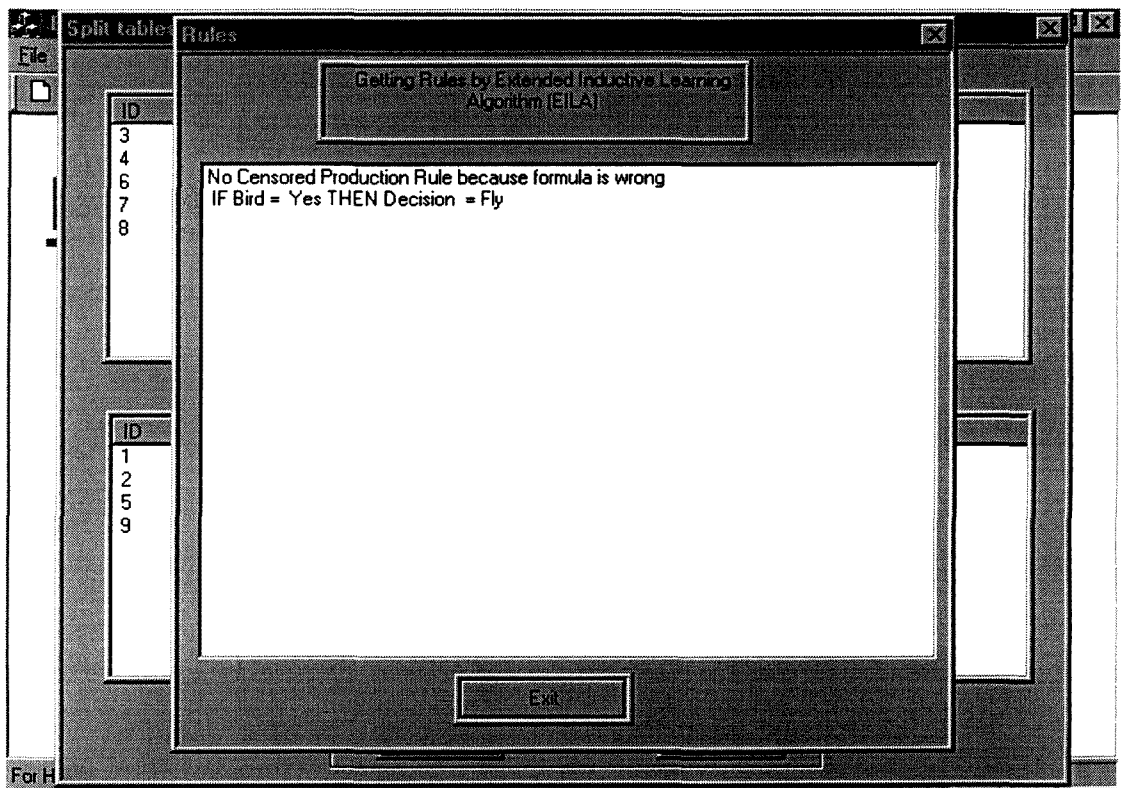
Screen (4-9)



Screen (4-10)



Screen (4-11)



Screen (4-12)

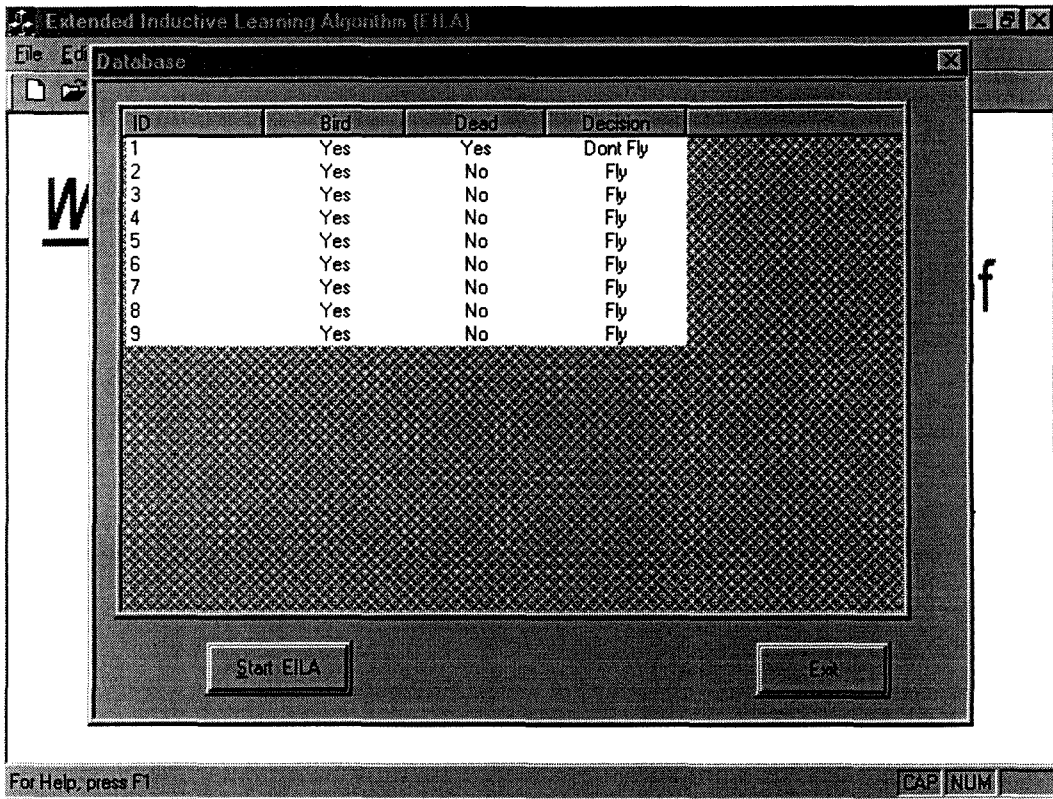
### 4.3 Second Training (Phase)

Suppose we have the following Training Set Table 4.3 as input for the EILA:

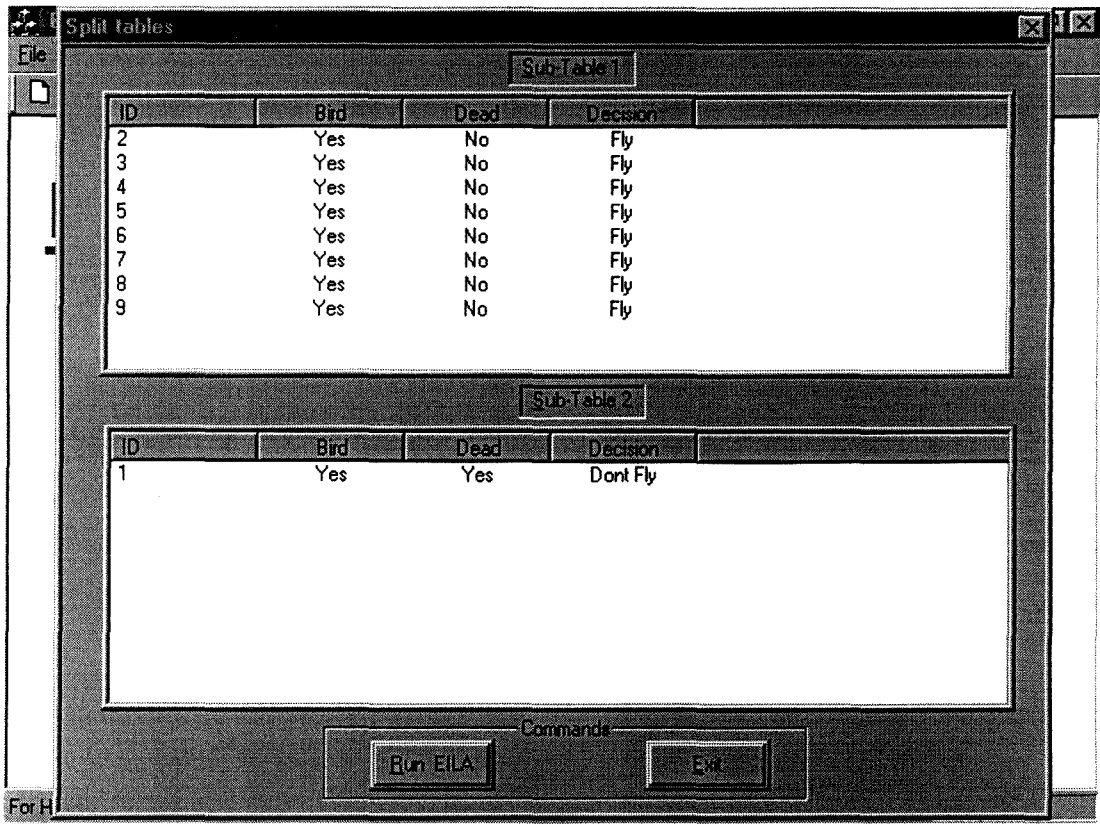
**TABLE 4.3-** Object Classification Training Set

<b>ID</b>	<b>Bird</b>	<b>Dead</b>	<b>Decision</b>
1	Yes	Yes	Dont Fly
2	Yes	No	Fly
3	Yes	No	Fly
4	Yes	No	Fly
5	Yes	No	Fly
6	Yes	No	Fly
7	Yes	No	Fly
8	Yes	No	Fly
9	Yes	No	Fly

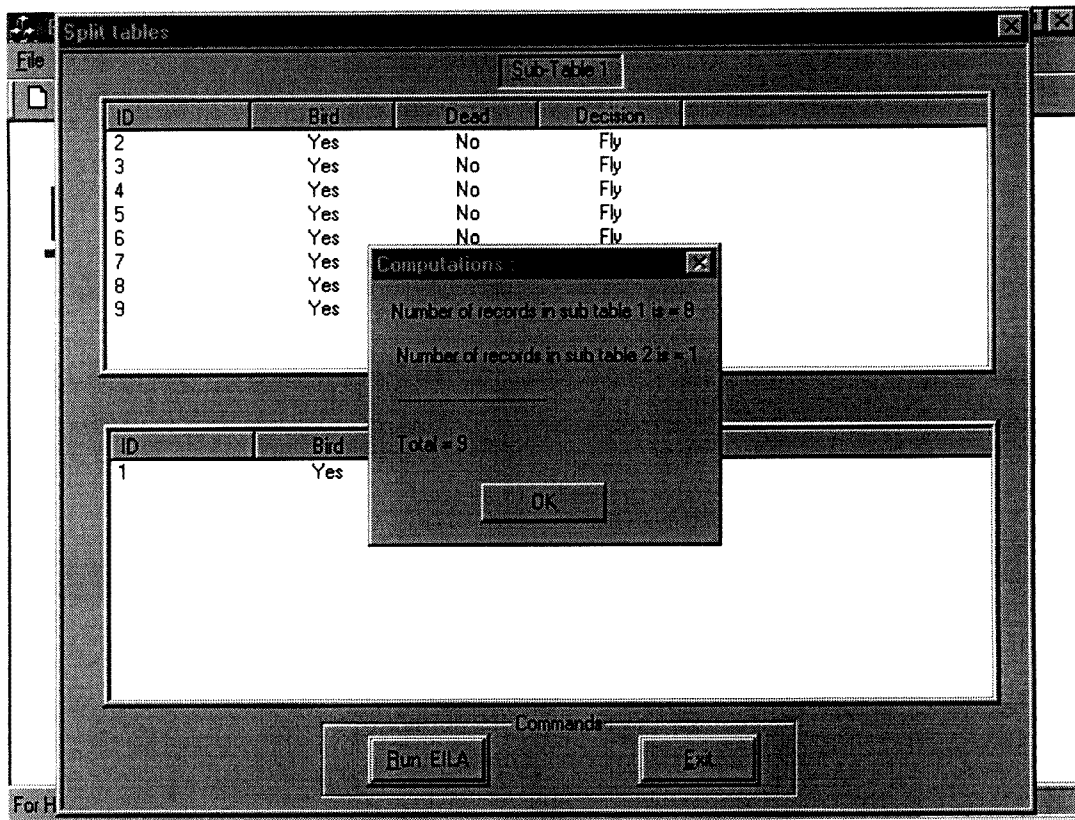
When we run the EILA program the display of input data, split into Sub-Table1 and Sub-Table2, number of rows in each sub-table and the final display for construction of Censored Production Rule (CPR) are shown in the following screens (4-13), (4-14), (4-15) and (4-16) respectively.



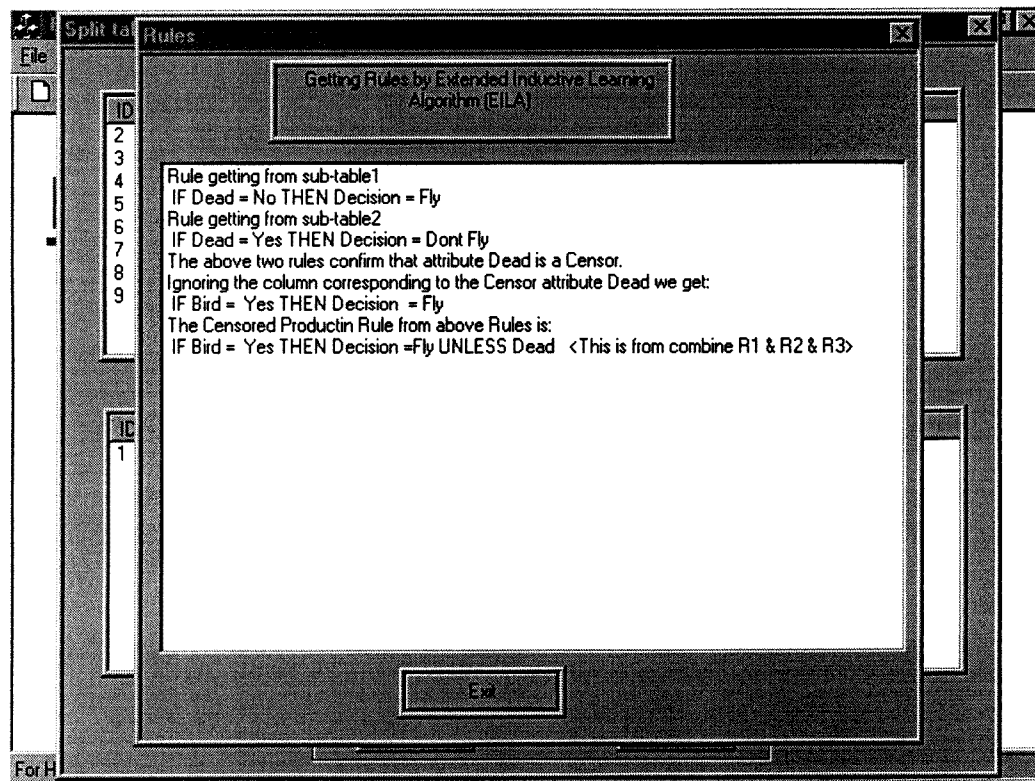
Screen (4-13)



Screen (4-14)



Screen (4-15)



Screen (4-16)

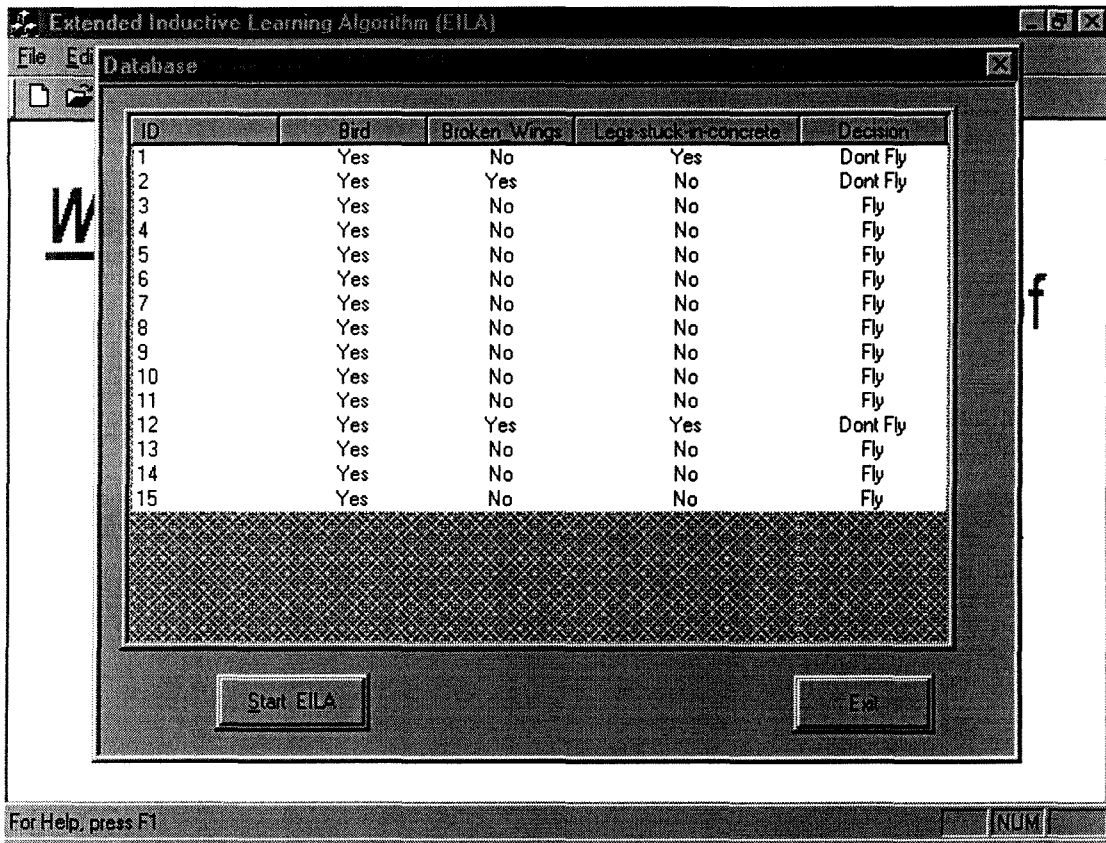
#### 4.4 Third Training (Phase)

Suppose we have the following Training Set Table 4.4 as input for the EILA:

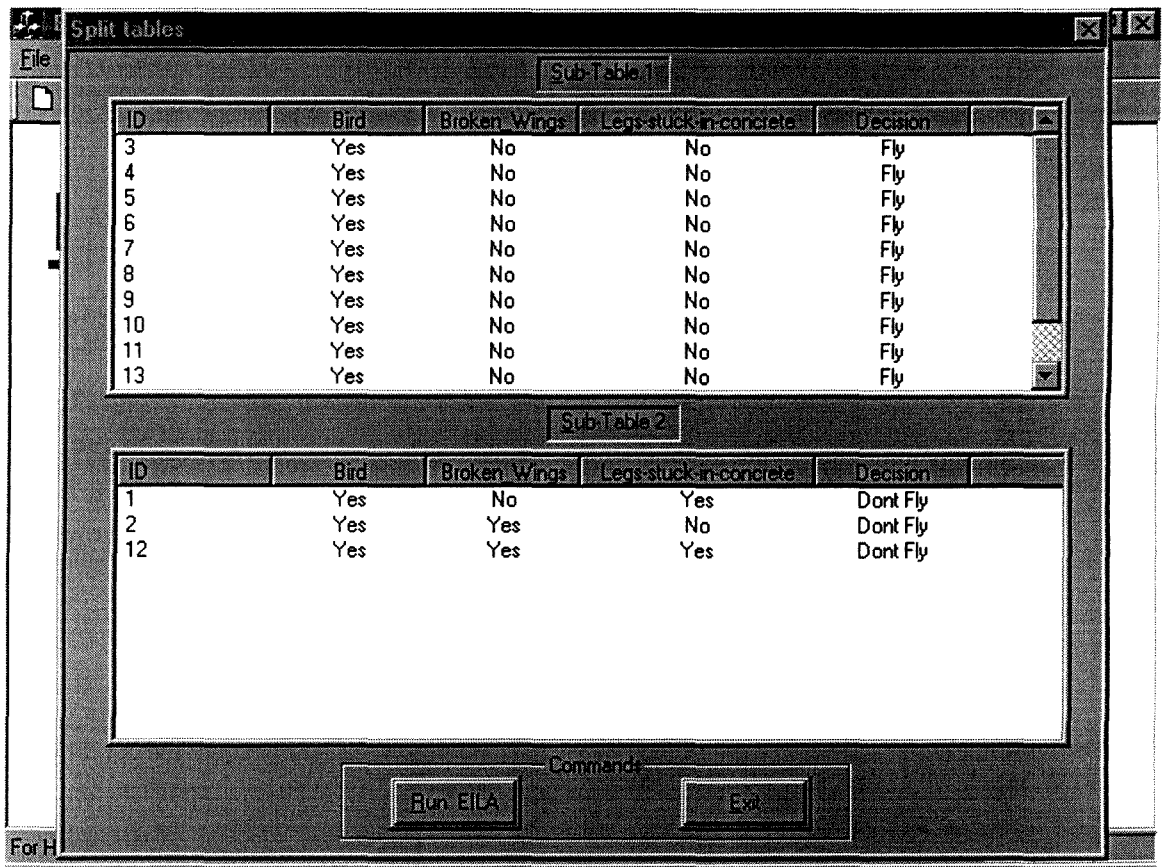
TABLE 4.4- Object Classification Training Set

ID	Bird	Broken-Wings	Legs- stuck-in-concrete	Decision
1	Yes	No	Yes	Dont Fly
2	Yes	Yes	No	Dont Fly
3	Yes	No	No	Fly
4	Yes	No	No	Fly
5	Yes	No	No	Fly
6	Yes	No	No	Fly
7	Yes	No	No	Fly
8	Yes	No	No	Fly
9	Yes	No	No	Fly
10	Yes	No	No	Fly
11	Yes	No	No	Fly
12	Yes	Yes	Yes	Dont Fly
13	Yes	No	No	Fly
14	Yes	No	No	Fly
15	Yes	No	No	Fly

When we run the EILA program the display of input data, split into Sub-Table1 and Sub-Table2, number of rows in each sub-table and the final display for construction of Censored Production Rule (CPR) are shown in the following screens (4-17), (4-18), (4-19) and (4-20) respectively.

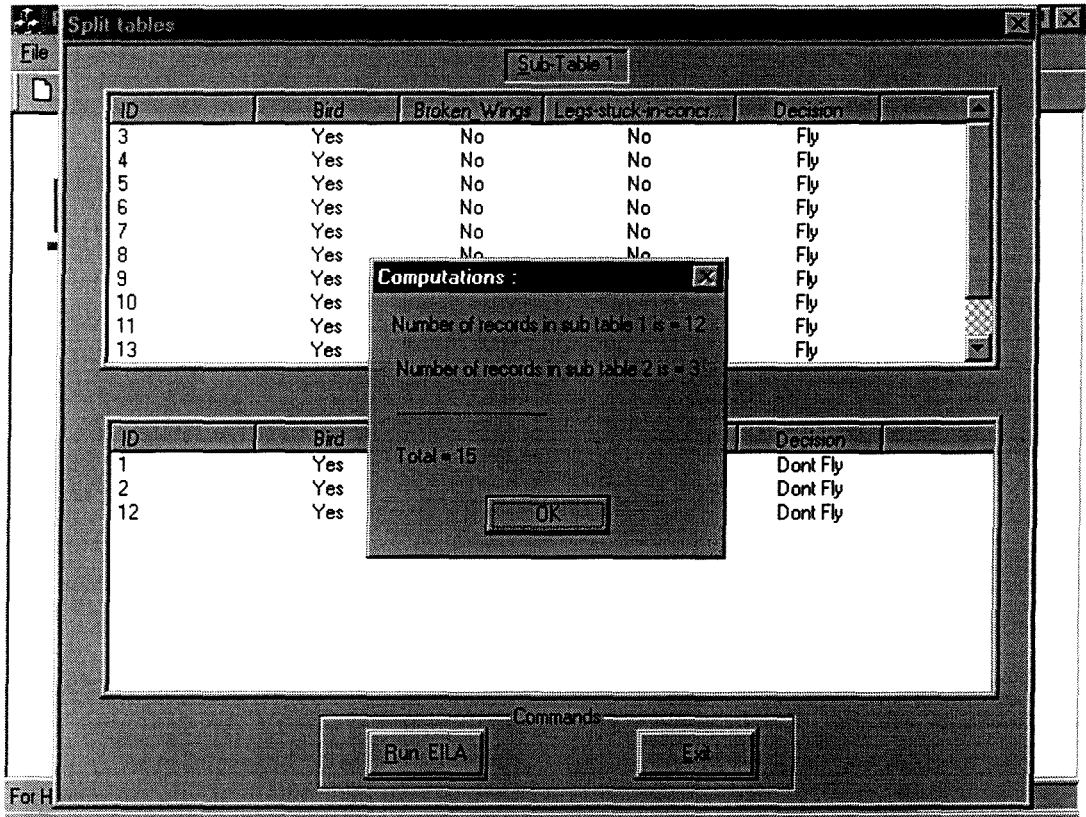


Screen (4-17)

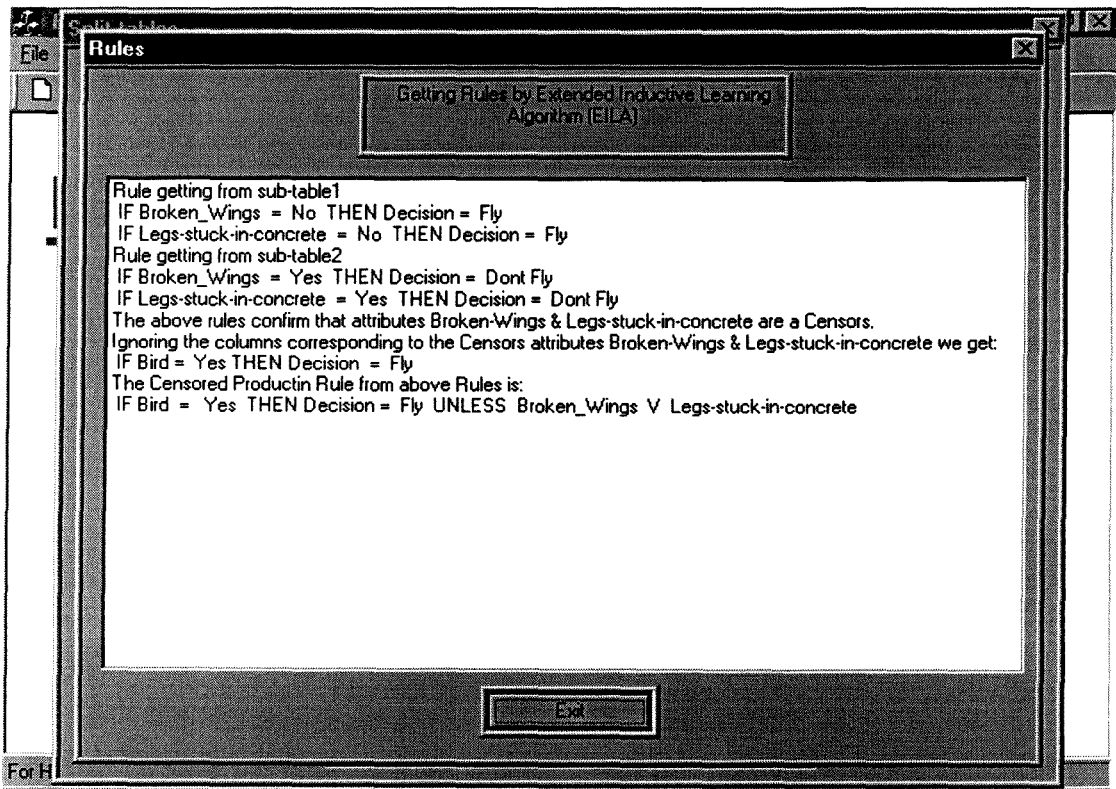


Screen (4-18)





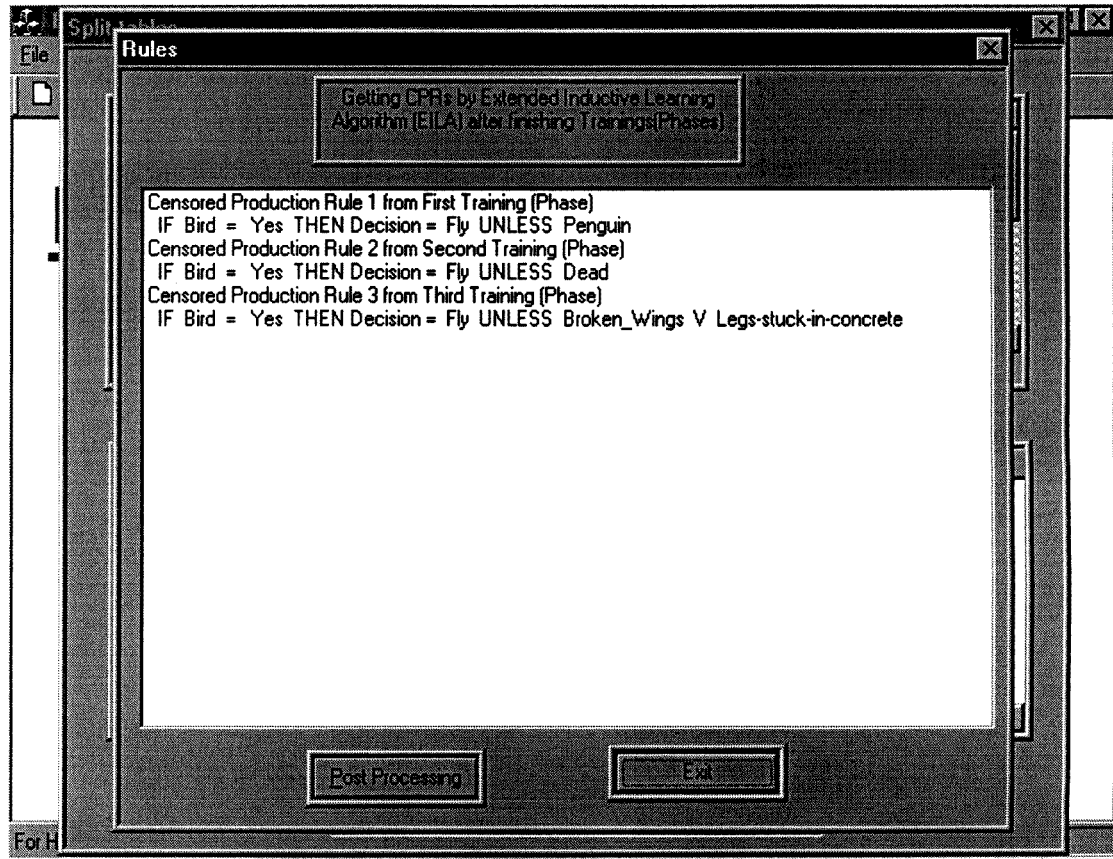
Screen (4-19)



Screen (4-20)

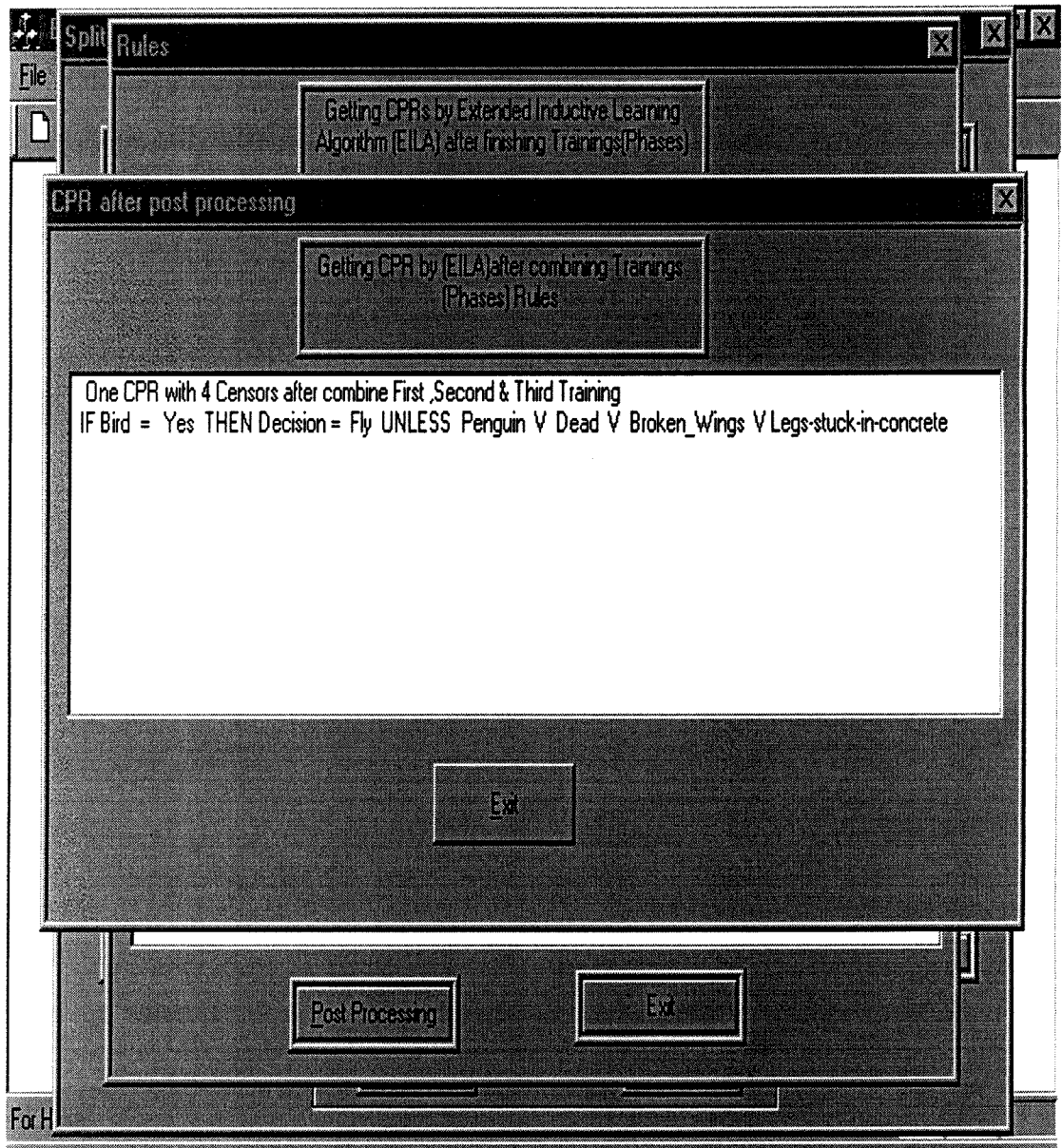
## 4.5 Post Processing

The screen (4-21) shown below contains all Censored Production Rules (CPRs) that discovered from first, second and third training sets.



Screen (4-21)

The incremental learning then combines the Censored Production Rules (CPRs) in Screen (4-21) to produce a single compact Censored Production Rule (CPR) as shown below in screen (4-22).



Screen (4-22)

## CHAPTER-5

### CONCLUSION

In the present work an Inductive Learning Algorithm has been developed as an extension of ILA [8] for the discovers Censored Production Rules that can handle uncertain ,incomplete and imprecise knowledge with resource constraint.

Implementation details are included and experimental results are presented to demonstrate the performance of the proposed algorithm.

Also an extension of the proposed algorithm for incremental learning of new censors is presented and it is shown how related discovered CPRs can be combined to discover single compact CPR.

Further extension of the algorithm for mining data sets with multiple censor conditions is being explored.

One of the most important future research direction would be the discovery of Hierarchical Censored Production Rules (HCPRs) [1] from large databases.

## **BIBLIOGRAPHY**

1. Bharadwaj, K.K., and Jain, N.K., Hierarchical Censored Production Rules (HCPRs) System, Data Knowledge Engineering, North-Holland, 8,19-34, (1992).
2. Bharadwaj, K.K., and Jain, N.K., and Norian Merranghello, Extended Hierarchical Censored Production Rules (EHCPRs) system, An Approach Toward Generalized Knowledge Representation, Journal of Intelligent System, UK, vol 9, No. 3 –4 (1999).
3. Han and Kamber, Data Mining: Concepts and Techniques, Academic Press, (2001).
4. Quinlan, J.R., Induction of Decision Trees, Machine Learning, 81-106 (1986).
5. Michalski and Winston P.H., Variable Precision Logic, Artificial Intelligence, 29, 121-146 (1986).
6. Pieter and Dolf, Data Mining, Addison Wesley (1999).
7. The Data Mine – Data Mining and KDD Information:  
[www.andpryke.com/university/TheDataMine.html](http://www.andpryke.com/university/TheDataMine.html)
8. Tolun and Abu-Soud, ILA: An Inductive Learning Algorithm for Rule Extraction , Expert System with Application ,14(3),April (1998).