RIAC: A Rule Induction Algorithm Based on Approximate Classification

Howard J. Hamilton, Ning Shan, Nick Cercone

Technical Report CS-96-06
May, 1996

Department of Computer Science
University of Regina
Regina, Saskatchewan, CANADA
S4S 0A2

ISSN 0828-3494
ISBN 0-7731-0321-X

RIAC: A Rule Induction Algorithm Based on Approximate Classification

Howard J. Hamilton, Ning Shan, Nick Cercone

Department of Computer Science, University of Regina

Regina, Saskatchewan, Canada S4S 0A2

E-mail: {hamilton,ning,nick}@cs.uregina.ca

Abstract

We present the RIAC (**R**ule **I**nduction through **A**pproximate **C**lassification) method for inducing rules from examples, based on the theory of rough sets. Imprecise data are generalized using a rough-sets based approximation technique. Positive, boundary and negative regions of the target concept are defined using statistical information. Then, if the concept is learnable, superfluous attributes are eliminated and rules are generated. Each classification rule generated by the RIAC algorithm is tagged with a *certainty factor*, which is the observed probability that an object matching the condition part of the rule belongs to the concept. Our approach treats each rule as a piece of uncertain evidence, which by itself is of little value with respect to classification. Our classification procedure combines as many pieces of evidence (rules) as possible. Experimental results indicate that RIAC produces fewer rules that give more accurate predictions than C4.5.

Keywords: Machine Learning, Knowledge Discovery, Knowledge Representation, Data Mining, Rough Sets.

1 Introduction

As a branch of machine learning, knowledge discovery from databases (KDD) seeks to apply general principles of machine learning to data stored in databases. The central question in knowledge discovery research is how to turn information, expressed in terms of stored data, into knowledge expressed in terms of generalized statements about characteristics of the data.

Machine learning is usually meant as sort of inductive inference, in which a sample is used to draw conclusions about the whole universe. The *learning from examples* inductive-inference technique extracts a description for a concept from a set of individual observations (*examples*). This description can be represented in a high-level language (for example, as *If/Then* rules). Such generalized rules have the advantage of making knowledge easier to understand and communicate to others, and they can also be used as the basis for an experience-based decision support system. Algorithms for learning and refining classification rules from examples include the AQ family (Michalski et al. 1983, 1986), the ID3 family (Quinlan 1993, 1986; Cheng et al. 1988; Fayyad et al. 1993, 1994), and CN2 (Clark and Niblett 1989). The continuing development of rule induction algorithms is motivated by the increasing application of knowledge discovery from database methods (Fayyad et al. 1993; Gemello and Mana 1989; Piatetsky-Shapiro et al. 1991) which apply inductive-inference techniques to large databases.

Sets of real-world training examples are almost invariably imprecise. In such a situation, exact classification rules cannot be derived. The objective of this paper is to suggest a method, based on rough sets, for generating inexact but useful classification rules from imprecise information. Rough sets theory is a mathematical tool for modelling incomplete or imprecise information (Pawlak 1984, 1991). The most important issue addressed in the rough sets theory is the idea of imprecise knowledge. Knowledge is imprecise if it contains any imprecise concept. A concept is *precise* if it can be expressed (defined) in terms of the assumed classification patterns; otherwise the concept is *imprecise* (Pawlak 1991). Applications of rough sets to machine learning area are given in (Chan 1991; Gezymala-Busse 1988; Slowinski 1992; Pawlak 1991).

Learnability of concepts is another important issue in inductive learning. If the learning task is to generate a description of a target concept based on a set of condition attributes, the whether or not it can be done depends on the granularity of the information represented by the data and the target concept's characteristics,

such as the presence or absence of noise, and whether the concept is deterministic or nondeterministic.

In this paper, we present the RIAC (**R**ule **I**nduction through **A**pproximate **C**lassification) method for inducing rules from examples, based on the theory of rough sets. This method determines whether a concept is learnable before generating rules. Statistical information is used to define the positive, boundary, and negative regions of the target concept. Each classification rule generated by the RIAC algorithm is tagged with a *certainty factor*, which is the observed probability that an object matching the condition part of the rule belongs to the concept. The certainty factor represented the degree of certainty associated with any conclusion produced by applying this rule. The algorithm is robust enough to cope with imperfect data. It finds *strong rules*, *i.e.,* rules that cover many cases and make highly accurate predictions (Piatetsky-Shapiro 1991).

The paper begins with a formal description of an attribute-value system. Approximate classification based on rough set theory is introduced in Section 3. In Section 4, our attribute selection is introduced. First, a binarized attribute-value system is created by applying the binarization technique (Quinlan 1986) to replace the original attribute-value system. Examples in the attribute-value system are partitioned by the approximate classification into equivalence classes of examples. Then, an attribute reduction technique is applied to eliminate superfluous binary attributes, each of which corresponds to an attribute-value pair in the original system. The effect is to reform approximate classification by removing irrelevant attribute-value pairs, which accelerates learning and later results in simpler rules. Rule generation is described in Section 5. Our approach uses the strategy of "conquer-without-separating" to generate kernel rules (Rymon 1993). The "dropping conditions" technique (Michalski et al. 1983) could be applied to simplify the rules. Instead, to obtain faster performance, a decision tree is first constructed based on the partitioned approximation space, and a certainty factor is associated with each leaf of the tree. After we collect the set of decision rules from this tree, we apply the same rule generation to generate kernel rules. Our classification procedure is presented in Section 6. We treat each rule as a piece of uncertain evidence, which by itself is of little value with respect to classification, but which jointly with other rules can provide a substantial input to the classification process. Experimental results are given in Section 7. Finally, conclusions are presented in Section 8.

2 Concept Learning in Attribute-Value Systems

We view concept learning as finding a discriminating description of objects with a particular value for a decision attribute in an attribute-value system.

Formally, an attribute-value system S (also called an information system (Pawlak 1991)) is a quadruple $\langle U,\ A,\ V,\ f\rangle$, where $U = \{x_1, x_2, ..., x_n\}$ is a finite set of objects; A is a finite set of attributes; the attributes in A are further classified into two disjoint subsets, the *condition attributes* C and the *decision attributes* D such that $A = C \cup D$ and $C \cap D = \emptyset$; $V = \bigcup_{A_i \in A} V_{A_i}$ is a set of *attribute values* where V_{A_i} is the *domain* of attribute A_i; $f : U \times A \rightarrow V$ is an *information function* which assigns particular values from the domains of attributes to objects such that $f(x_i, A_i) \in V_{A_i}$, for all $x_i \in U$ and $A_i \in A$.

The collection of attribute-value sets describing a group of objects constitutes a set of training examples. Also, typically $D = \{d\}$ is a singleton set with the function $f(x, d) : U \times d \rightarrow V_d$. The decision attribute $d \in A$ corresponds to the *concepts* to be learned. A *concept* is simply a set of domain objects with a particular value $V_{dj} \in V_d$ for attribute d. Here, we consider only one discrete decision attribute.

The goal of concept learning is to find a discriminating description of the subset Y_j of objects with the value of the attribute d equal to V_{dj} that is as simple as possible. In other words, the goal is to learn the description of the set $Y_j = \{x \in U : f(x, d) = V_{dj}, V_{dj} \in V_d\}$. The set Y_j will be referred to as the *target concept* or the set of *positive cases*. The description of the target concept produced by a learning system is usually an expression in predicate logic comprised of symbols (corresponding to attribute names and values) and relational operators (such as $=$, $\leq$ and $\geq$).

Given the task of learning the concepts associated with an k-valued decision attribute, we decompose the task into k subtasks, each requiring the description of the attribute-value system associated with one particular concept Y_j, $\quad 1 \leq j \leq k$. We construct an *associated attribute-value system* for each concept Y_j, with unchanged condition attributes and a new binary decision attribute d_j. The objects included in the set Y_j are called *positive cases*; all other concepts are grouped together to form a complement $\neg Y_j$ of the concept Y_j and the objects belonging to the set $\neg Y_j$ are called *negative cases*. The remainder of the discussion, without loss of generality, is restricted to binary decision attributes, with target concept Y and complement $\neg Y$.

3 Learning by Approximate Classification

3.1 The Equivalence Relation

Since an attribute-value system provides only partial information about the universe, we may not be able to distinguish objects by means of the given attributes and their values. Let B be a subset of A, and let x_i, x_j be members of U. A binary relation $R(B)$, called an *indiscernibility relation* (Pawlak 1984, 1991), is defined as follows:

$$R(B) = \{(x_i, x_j) \in U^2 \mid \forall a \in B,\ f(x_i, a) = f(x_j, a)\}.$$

We say that x_i and x_j are indiscernible by the set of attributes B in S if $f(x_i, a) = f(x_j, a)$ for every $a \in B$. Clearly, $R(B)$ is an equivalence relation on U for every $B \subseteq A$. It classifies the objects in U into a finite, preferably small, number of equivalence classes. Thus, we can define two natural equivalence relations $R(C)$ and $R(\{d\})$ on U for an attribute-value system S. A *concept* Y is an equivalence class of the relation $R(\{d\})$. The collection of the two equivalence classes Y and $\neg Y$ induced by the relation $R(\{d\})$ is defined as: $R^*(\{d\}) = \{Y, U - Y\} = \{Y, \neg Y\}$.

Based on the set of condition attributes C, an object x_i specifies the equivalence class $[x_i]_R$ of the relation $R(C)$:

$$[x_i]_R = \{x_j \in U \mid \forall a \in C,\ f(x_j, a) = f(x_i, a)\}$$

We define conditional probabilities as:

$$P(Y|[x_i]_R) = \frac{P(Y \cap [x_i]_R)}{P([x_i]_R)} = \frac{|Y \cap [x_i]_R|}{|[x_i]_R|}$$

where $P(Y|[x_i]_R)$ is the conditional probability of occurrence of the concept Y conditioned on the occurrence of the equivalence class $[x_i]_R$. That is,

$$P(Y|[x_i]_R) = 1 \qquad \text{iff} \quad [x_i]_R \subseteq Y;$$

$$P(Y|[x_i]_R) > 0 \qquad \text{iff} \quad [x_i]_R \cap Y \neq \emptyset;$$

$$P(Y|[x_i]_R) = 0 \qquad \text{iff} \quad [x_i]_R \cap Y = \emptyset.$$

3.2 β-Approximate Classification

Most learning algorithms assume that each equivalence class fall into a single unique concept (*i.e.*, $P(Y|[x_i]_R) = 1$ or $P(Y|[x_i]_R = 0)$. However, sets of real-world training examples are almost invariably imprecise, so one equivalence class may fall into more than one concepts. In the rough sets, if a particular equivalence class fall into multiple concepts, then usually either all candidate concepts are presented (Pawlak 1991) or the most likely concept is chosen (Ziarko 1993). The most likely concept is chosen based on the conditional probability distributions of positive and negative objects occurring within the equivalence class.

Given an attribute-value system $S = \langle U, C \cup \{d\}, V, f\rangle$ and an equivalence relation $R(C)$ on U, Ziarko (1993) defined a *β-approximation space* AS_P for the condition attributes C as a quadruple $\langle U, R(C), P, \beta\rangle$, where P is the probability measure described above and β is a user-specified real number in the range $[0, 0.5)$. The equivalence classes induced by the relation $R(C)$ are called *elementary sets* in AS_P because they represent the smallest groups of objects which are distinguishable in terms of the attributes and their values.

Let $Y \subseteq U$ be a subset of objects in U representing positive cases, and $R^*(C) = \{X_1, X_2, ..., X_n\} = \{[x_1]_R, [x_2]_R, ..., [x_n]_R\}$ be the collection of equivalence classes induced by the relation $R(C)$. The β-approximation space AS_P can be divided into the following three regions:

1. The β-positive region of the set Y:

$$POS_C(Y) = \bigcup_{P(Y|X_i) \geq 1-\beta} \{X_i \in R^*(C)\}.$$

2. The β-boundary region of the set Y:

$$BND_C(Y) = \bigcup_{\beta < P(Y|X_i) < 1-\beta} \{X_i \in R^*(C)\}.$$

3. The β-negative region of the set Y:

$$NEG_C(Y) = \bigcup_{P(Y|X_i) \leq \beta} \{X_i \in R^*(C)\}.$$

The β-positive region of the set Y corresponds to all elementary sets of U that can be classified into the concept Y with probability $P(Y|X_i)$ greater than or equal to the parameter $1-\beta$. Similarly, the β-negative region of the set Y corresponds to all elementary sets of U that can be classified into the concept Y with probability $P(Y|X_i)$ less than or equal to the parameter β. In other words, all elementary sets of U belonging to the β-negative region of the set Y can be classified into the complementary concept $\neg Y$ with probability $P(\neg Y|X_i)$ greater than or equal to the parameter $1-\beta$. The β-boundary region of the set Y corresponds to all elementary sets of U which cannot be classified into Y or $\neg Y$ according to the user defined parameter β.

Let $x_i \in U$ be an object, and let $POS_C(Y)$ and $NEG_C(Y)$ be the β-positive and β-negative regions of concept Y, respectively. The object x_i is classified as belonging to concept Y if and only if $x_i \in POS_C(Y)$, and it is classified as belonging to the complement $\neg Y$ if and only if $x_i \in NEG_C(Y)$. In concept learning, we must decide whether x_i is in the concept Y on the basis of the set of equivalence classes in AS_P rather than on the basis of Y itself. This means that we deal with $POS_C(Y)$ and $NEG_C(Y)$ instead of the set Y. If $x_i \in U$ is in $POS_C(Y)$, then the probability $P(Y|X_i)$ that it is can be classified into the concept Y is greater than or equal to the parameter $1-\beta$.

3.3 Automatic Selection of β

A significant limitation of the β-approximation classification technique as defined in Ziarko (1993) is the difficulty of choosing an appropriate value for the parameter β. In RIAC, a suitable value for β is determined based on the objects in the attribute-value system (database), instead of asking the user to enter a value. Our approach is based on the probability distribution associated with the classification. Given a database and a specified concept Y, $P(Y)$ is the fraction of objects in the given database that are in concept Y. Based on this information, which can be readily obtained prior to generating any rule, we can classify any unseen object into Y with probability $P(Y)$, or into $\neg Y$ with probability $1 - P(Y|X_i)$. Now, given a database,

a specified concept Y, and an equivalence class X_i as a generated rule, we have $P(Y|X_i)$, which is the conditional probability of the occurrence of the concept Y conditioned on the occurrence of the equivalence class X_i. We can predict that any object which matches the equivalence class X_i belongs to concept Y with the probability $P(Y|X_i)$. The *classification gain* of an equivalence class X_i is defined as

$$CG(X_i) = P(Y|X_i) - P(Y),$$

which measures how much is gained by classifying an unseen object into Y based on the information of the probabilities of the equivalence class X_i and the concept Y. If $CG(X_i) > 0$, then using the equivalence class X_i is helpful for classifying an unseen object into the concept Y, so X_i should be placed in the positive region. Similarly, If $CG(X_i) < 0$, then using X_i is helpful for classifying an unseen object into the concept $\neg Y$, so X_i should be placed in the negative region. When $CG(X_i)$ is equal to 0, then using X_i doesn't add any useful information about classification and X_i is placed in the boundary region. Our redefined β-approximation classification is as follows:

1. The β-positive region of the set Y:

$$POS_C(Y) = \bigcup_{P(Y|X_i) > P(Y)} \{X_i \in R^*(C)\}.$$

2. The β-boundary region of the set Y:

$$BND_C(Y) = \bigcup_{P(Y|X_i) = P(Y)} \{X_i \in R^*(C)\}.$$

3. The β-negative region of the set Y:

$$NEG_C(Y) = \bigcup_{P(Y|X_i) < P(Y)} \{X_i \in R^*(C)\}.$$

This revised version is motivated by the desire to analyze and identify data patterns based on information available in the database rather than according to a user defined parameter.

Learnability of a concept from a set of training examples is an important issue in inductive learning. If the β-positive region does not exist for a particular concept, then no rule can be constructed for this concept.

4 Attribute Selection

Usually there are too many attributes for all to be used by a learning system. Some may be redundant or irrelevant. *Attribute selection* attempts to select a subset of the attributes that is necessary and sufficient to describe a target concept before rule generation. It is also important to accelerate learning and to improve learning quality (Kira and Rendell 1992). Our method does not suffer from the "splintering problem" (Domingos 1995; Holte et al. 1989), because each attribute is selected based on the entire data set.

4.1 Binarization of Attribute-Value Systems

Our method for binarization of an attribute-value system is based on identifying cases where one particular value for an attribute is worth distinguishing but the other possible values are not. Suppose attribute A has values $\{a_1, a_2, a_3, a_4\}$ (i.e., the 4 attribute-value pairs for A are $\{\langle A = a_1\rangle, \langle A = a_2\rangle, \langle A = a_3\rangle, \langle A = a_4\rangle\}$). If only a_3 is relevant for distinguishing a concept, then the domain for A can be simplified to $\{a_3, \neg a_3\}$ by generalizing a_1, a_2, and a_4 to $\neg a_3$.

If an attribute $A_i \in A$ is discrete, then each attribute-value pair $\langle A_i = VAL_j\rangle$ corresponds to a binary valued attribute A_{ij}: $f(x, A_{ij}) = 1$ if $f(x, A_i) \in VAL_j$ and $f(x, A_{ij}) = 0$ if $f(x, A_i) \notin VAL_j$, where VAL_j is a set of values from the domain V_{A_i} of attribute A_i. Creating a binary attribute corresponding to each of m possible values of an attribute expands the search space of the learning process. If an attribute has m possible values, then there are $2^m - 2$ different attribute-value pairs of these values. Since disjunctive values may occur after generalization, a very large search space of attribute-value pairs may result, causing an increase in the cost of finding the best attribute-value pair. Methods for reducing such a search space are reported in (Breiman 1984; Quinlan 1992). To reduce the search space in the learning process, the VAL set of an attribute-value pair for a discrete attribute is restricted to a singleton value in RIAC.

For each continuous attribute A_i, a set of cutpoints $CP_{A_i} = \{VAL_0, VAL_1, VAL_2, ..., VAL_N\}$ is designed, either by using user-supplied discretization formula or by applying automatic discretization al-

gorithms (Catlett 1991; Fayyad & Irani 1992). VAL_0 and VAL_N are the minimum and maximum values of the attribute and $VAL_0 < VAL_1 < VAL_2 < ... < VAL_N$. However, the *natural* interval values $VAL_n = \frac{V_{A_i}^n + V_{A_i}^{n+1}}{2}$ can be used as the cutpoints, where $V_{A_i}^n$ is an element of the set of distinct values $V_{A_i} = \{V_{A_i}^0, V_{A_i}^1, ..., V_{A_i}^n, ..., V_{A_i}^N\}$ of the attribute A_i and $V_{A_i}^0 < V_{A_i}^1 < ... < V_{A_i}^N$. Each attribute-value pair $\langle A_i, VAL_j \rangle$ corresponds to a binary valued attribute A_{ij}: $f(x, A_{ij}) = 1$ if $f(x, A_i) > VAL_j$ and $f(x, A_{ij}) = 0$ if $f(x, A_i) \leq VAL_j$, where VAL_j is j^{th} element of the set of cutpoints CP_{A_i} for the attribute A_i.

We use the *information gain Gain* (Breiman 1984; Quinlan 1986) to assign significance values to each binary attribute to evaluate its ability to discriminate objects in U. Typically, higher significance values for a attribute indicates greater interaction with the decision attribute d. If the binary attribute A_{ij} is irrelevant or weakly relevant to the current binary decision attribute d, we eliminate this attribute while finding classification table for decision attribue d.

Attribute reduction, as described in Section 4.2, can be applied to remove the irrelevant or unnecessary attributes from the binarized attribute-value system. To reduce the search space and remove the weakly relevant attributes, an optional module has been implemented in RIAC system. It is performing internal search looking for attribute-value pairs which maximize the given information gain. Without losing generality, the generalized attribute-value system is created based on all attribute-value pairs whose information gains reach or exceed a threshold. The threshold can be calculated as the product of $MaxGain$ (the maximum information gain of all attribute-value pairs) and the user-specified tolerance level TL $(0 \leq TL \leq 1)$ (Cheng et al. 1988;Fayyad 1994). After generalization, the approximate classification tries to partition it into the β-positive, β-boundary, and β-negative regions. If the β-positive region exists, then we can apply the following steps to generate rules. For the sake of convenience, the generalized attribute-value system is still called an attribute-value system in the following sections.

4.2 Attribute Reduction

It is desirable to remove those condition attributes which do not provide any additional information about the objects in U. A *relative reduct* (Pawlak 1991) is a minimal sufficient subset of a set of attributes which has the same ability to discern concepts as the full set of attributes.

Given an attribute-value system $S = \langle U, C \cup \{d\}, V, f \rangle$, the degree of dependency $K_\beta(C, \{d\})$ in the relationship between the groups attributes C and $\{d\}$ can be defined as

$$K_\beta(C, \{d\}) = \frac{|POS_C(Y)| + |NEG_C(Y)|}{|U|}.$$

The degree of dependency $K_\beta(C, \{d\})$ is the proportion of those objects in the attribute-value system which can be roughly or definitely assigned into the β-positive and the β-negative regions. It characterizes the ability to predict the concept Y and $\neg Y$ from objects in the attribute-value system. An attribute a is said to be *dispensable* in C with respect to $\{d\}$ if $K_\beta(C - \{a\}, \{d\}) = K_\beta(C, \{d\})$; otherwise a is an *indispensable* attribute in C with respect to $\{d\}$. A subset of condition attributes $B \subseteq C$ is said to be a *dependent set* in S with respect to $\{d\}$ if there exists a proper subset $E \subset B$ such that $K_\beta(B, \{d\}) = K_\beta(E, \{d\})$; otherwise B is an *independent set* with respect to $\{d\}$. A *relative reduct* $C^{'}$ of attributes C is a maximal independent subset of condition attributes with respect to $\{d\}$. The intuition behind the relative reduct is the preservation of the concept's certainty regions, *i.e.*, $POS_C(Y)$ and $NEG_C(Y)$, while discarding unnecessary attributes.

More than one relative reduct may exist for a given attribute-value system. Selection of a "best" reduct depends on the optimality criterion associated with the attributes. Here, we base the decision on the significance values computed by the information gain in the previous section 4.1.

To compute a relative reduct the **GENRED** algorithm from (Shan et al. 1995) is adapted here by the addition of handling the imprecise data. With **GENRED**, each condition attribute is tested by removing it temporarily from the set of attributes C. At each step, the degree of dependency is calculated based on the remaining attributes in the set C. If the degree of dependency has been changed (i.e., $K_\beta(C, \{d\}) \neq K_\beta(C^{'}, \{d\})$), then the attribute is restored to the set C; otherwise, the attribute is permanently discarded. The attributes remaining in set C at the end form a relative reduct. The final collection of condition attributes contains only non-redundant attributes and preserves the degree of dependency with the decision $\{d\}$.

5 Rule Generation

Based on the relative reduct obtained as discussed in Section 4, a set of decision rules $RULE^+$ for the target concept Y can be obtained directly. Each rule $r_i^+ \in RULE^+$ has an associated certainty factor $c_i = P(Y|X_i)$. A rule r_i^+ matches an equivalence class $X_i \in POS_{RED}(Y) \subset R^*(RED)$. If an object satisfies the decision rules in $RULE^+$, then the object belongs to Y with some degree of certainty.

Here, we are only interested in constructing a set of decision rules for the positive region (target concept Y). The rules in $RULE^+$ describe this region, but they may contain attributes whose values are irrelevant in determining the target concept. A decision rule obtained by dropping the maximum possible number of conditions is called a *kernel rule* (Rymon 1993).

Rule Induction using the strategy of "conquer-without-separating" (Domingos 1995; Shan et al. 1995) from the data set can alleviate the splintering problem. It can make effective use of statistical measures to combat noise, because each rule is generated taking into consideration the entire data set in a "specific-to-general" fashion. However, this gain may be offset by an increase in computational costs, and this trade-off needs to be explored further. An alternative approach is implemented in RIAC to alleviate the high computational problem. A decision tree is first constructed based on the relative reduct, and a set of probabilistic decision rules $RULE^+$ is collected from this tree. Then using the **GENRULES** algorithm from (Shan et al. 1995), each new rule is evaluated with respect to dropping each condition in that rule. The evaluation function for a condition c_i of a rule $cond \rightarrow dec$ is: $SIG(c_i) = P(cond - \{c_i\})(P(Y|cond - \{c_i\}) - P(Y))$. Conditions with the higher significance values are dropped first. After a condition has been dropped, the significance values of the remaining conditions are updated. This process is repeated until no condition can be dropped.

6 Classification

In general, rules induced from a database are used to classify each unseen object into an appropriate concept. A simple approach, adapted by many systems involves finding the best rule whose conditions are satisfied, directly or indirectly (*i.e.*, after applying some predefined mapping) by the corresponding values of the input

vector. With the exception of special cases, this approach ignores the fact that the rules produced from data are inherently uncertain and the associated decision probabilities (if computed at all) are only crude estimates, rather weakly supported by available data.

Instead of using a single rule to suggest a decision, which tends to minimize the number of rules used, RIAC uses as much input as possible in its classification and tends to maximize the number of rules used. Essentially, our classification procedure combines as many pieces of evidence (rules) as possible. Clearly, there are many possible ways of combining such evidence and the technique implemented in our approach and described below is only one possibility.

In our approach, each rule is associated by a value of rule quality, $Q(r)$, computed from training database. The criterion of rule quality $Q(r)$ is based on the estimation of the following conditional probabilities: $P(cond|Y)$, $P(Y|cond)$, $P(cond|\neg Y)$ and $P(\neg Y|cond)$, where the rule r describes the concept Y. Intuitively, if a rule r and the target concept Y are well connected in statistical sense, then the conditional probabilities $P(cond|Y)$ and $P(Y|cond)$ should yield high values and the conditional probabilities $P(cond|\neg Y)$ and $P(\neg Y|cond)$ should yield low values. Consequently, based on this intuition, the rule quality measure used by our approach is defined as:

$$Q(r) = [P(cond|Y) + P(Y|cond) - P(cond|\neg Y) - P(\neg Y|cond)]/2$$

Clearly, $-1 \leq Q(r) \leq 1$. This measure has two extremes: $Q(r) = 1$ if the rule r matches all of objects belonging to the target concept Y and $Q(r) = -1$ if the rule r matches all of objects belonging to the concept $\neg Y$.

For an unseen object, first complete matching is attempted. The object is classified as belonging to a particular concept Y with the highest decision score $DS(Y)$, which is defined as the following formula:

$$DS(Y) = \sum_i Q(r_i),$$

for $i = 1$ to the number of matching rules in the concept Y.

If complete matching fails, partial matching is considered where some attribute-value pairs of a rule may

match the value corresponding attributes for the unseen object. A partial matching score $PMS(r)$ between a rule r and an unseen object obj is defined as follows:

$$PMS(r) = \frac{m}{n} \times Q(r),$$

where n is the number of attribute-value pairs in the conditions of rule r, and m is the number of attribute-value pairs satisfied by both rule r and object obj. For the partial matching, the decision score $DS(Y)$ for the concept Y is defined as follows:

$$DS(Y) = \sum_i PMS(r_i),$$

for $i = 1$ to the number of partially matching rules in the target concept Y. In partial matching, the unseen object is also classified as the concept with the highest value of the decision score.

7 Experiments

RIAC has been evaluated by comparing its performance with other methods on some standard machine learning problems. To evaluate RIAC, we measured the prediction accuracy of the learned rules on test examples.

7.1 Performance Comparison on the MONK's Problems

The MONK's problems (Thrun et al. 1991; Wu et al. 1995) are three artificially constructed problems and are well-studied. The three Monk's problems, call $MONK_1, MONK_2$ and $MONK_3$, represent three different types of learning tasks with two binary and four nominal attributes. The example space contains 432 ($3 \times 3 \times 2 \times 3 \times 4 \times 2$) possible examples. They differ in the type of the concept to be learned, and in the amount of noise in the training examples. Each problem $MONK_i$ is specified by a logical description of a concept, a training set $MONK\_TRN_i$, and a testing set $MONK\_TST_i$. The learning task is to generalize over the training set and, if the learning technique allows, to derive a simple concept description. The testing set $MONK\_TST_i$ for the problem $MONK_i$ is then used to measure the predictive accuracy of the resulting concept description.

Algorithm	#-Rules			Accuracy		
	$MONK\_TRN_1$	$MONK\_TRN_2$	$MONK\_TRN_3$	$MONK\_TST_1$	$MONK\_TST_2$	$MONK\_TST_3$
C4.5	59	113	27	82.4%	69.7%	90.3%
HCV	**7**	39	**18**	**100.0%**	81.7%	90.3%
RIAC	8	**37**	23	**100.0%**	**91.90%**	**95.83%**

Table 1: Performance Comparison on the MONK's Problems.

A performance comparison of RIAC with C4.5 (Quinlan 1992) and HCV (Wu et al. 1995) on the MONK's problems is shown in Table 1. The results for C4.5 and HCV are from (Wu et al. 1995). " #-Rule" is the number of rules generated by the learning algorithm and "Accuracy" is the classification accuracy when the generated rules set are used to classify the testing set. In all cases, RIAC produces fewer and more accurate rules than C4.5. RIAC produces more accurate results than HCV but produces more rules to do so.

7.2 Performance Comparison on Additional Data Sets

We also compared this accuracy to that for C4.5RULE, the "rule" portion of the output generated by C4.5 (Quinlan 1993). We chose 21 databases from the UC Irvine repository. Each database is stored and processed as a single relation table.

For our experiments, we used "leave-one-out" cross validation to compare RIAC and C4.5RULE (Quinlan 1993) on a given database. *Leave-one-out testing* is an elegant and reliable technique for estimating prediction accuracy. Given a data set containing n objects, leave-one-out testing removes one object, generates rules using the remaining $n-1$ objects as a training sample, and tests these rules using the removed object as a test object. This procedure is repeated n times, using each object in turn. The prediction accuracy is calculated as the number of correctly classified objects divided by n. Leave-one-out testing is computationally expensive, but it gives a more reliable estimate of prediction accuracy than other approaches. Given the same algorithm and data set, leave-one-out testing always reports the same prediction accuracy. On the other hand, *ten-fold testing* randomly partitions the data set into ten subsets and uses each in turn as a training sample. When applied repeatedly to the same algorithm, this method yields varying results because different random subsets are chosen.

Table 2 shows the results from the experiments described above. Boldface is used to indicate better results. For each database and algorithm, we report the prediction accuracy as determined by leave-one-out testing. At the bottom of the table, each column's average is shown. On 3 out of 21 databases, the behavior

Dataset	Size	#Concepts	RIAC (%)	C4.5 (%)
append	106	2	**86.89**	84.91
australian	690	2	**83.19**	82.75
balance-scale	625	3	75.84	**77.28**
breast-cancer(wisconsin)	699	2	94.99	**96.00**
glass	214	7	70.56	**71.50**
heart	270	2	**79.63**	74.44
housing	506	2	**85.38**	83.20
ionosphere	351	2	94.59	**94.87**
iris	150	3	**98.00**	95.33
lenses	24	3	79.17	**83.33**
lung-cancer	32	3	50.00	50.00
monktst1	432	3	100.0	100.0
monktst2	432	3	**100.0**	70.00
monktst3	432	3	100.0	100.0
pima	768	2	73.34	**74.87**
shuttle	15	2	**66.67**	40.00
soybean-small	47	4	**100.0**	97.87
tictac	958	2	**99.90**	99.48
vote	435	2	**96.32**	96.30
wine	178	3	**97.25**	92.70
zoo	101	7	**96.04**	93.06
Average			**87.04**	**83.71**

Table 2: The Leave-one-out Comparisons on 21 Databases.

of RIAC and C4.5RULE is similar. On 12 Out of 21 databases, the RIAC achieves a significant improvement on prediction accuracy over C4.5RULE. The average prediction accuracy of RIAC is higher than C4.5RULE.

8 Conclusions

We have described the RIAC method for generating probabilistic decision rules from imprecise data. This capability is important because in practice we seldom have complete and consistent information for the design of intelligent systems. The salient feature of the proposed method is that it makes use of the statistical information inherent in the attribute-value system. Furthermore, our method can deal with uncertainty, or work with the presence of noise.

Learnability of training set is another important issue in inductive learning. It is associated with the uncertainty that exists in the data set. Such data may result in the failure of a learning algorithm in deriving classification rules. We deal with this problem by using an extension of rough sets theory. The approximate classification partitions the dataset into the β-positive, β-boundary and β-negative regions. If β-positive

region exists, then we can apply rule generation to construct a set of decision rules for the target concept.

Most machine learning approaches to the induction of rules from examples fall into two categories: "divide-and-conquer"(Quinlan 1986) and "separate-and conquer" (Clark and Niblett 1989; Michalski 1983). The former recursively partition the instance space until the remaining small instance space roughly belongs to the uniform concept. The latter induces one rule at a time and removes the instances covered by this rule until no more rule can be generated. Such methods suffer from the "splintering problem", because the size of the available sample dwindles, resulting in decision being made with less and less statistical support. Statistical anomalies become harder to weed out, and noise sensitivity increase. As the result, the overfiting and incorrect rules may be generated and decrease prediction accuracy (Domingos 1995; Holte et. al. 1989). To alleviate the splintering problem, other systems (Domingos 1995; Shan et. al. 1995; Slowinski 1992) using the strategy of "conquer-without-separating" to induce rules from the data set. It can make effective use of statistic measures to combat noise, because each rule is generated taking into consideration the entire data set in a "specific-to-general" fashion. This method handles missing values without filling in with artificial values, because each equivalence class or case is accounted for as a specific rule being generated.

The classification problem consist of assigning an unseen object, described by values of attributes, to one of a set of pre-defined concepts. The classification is performed on the basis of rules derived from previous experience by the learning algorithm. Decision rules in the classification procedure are not equally important or reliable for performing classification even though they have the same predictive accuracy. They are supported by differing numbers of objects from training data. For this reason, each rule in RIAC has a rule quality rating computed from the database when the rule is generated. Here, we did not directly use the number of objects as the rule quality, because probabilistic rules contain both positive and negative cases. Two matching approaches have been proposed based on the rule quality measure.

In addition to the theoretical analysis, experimental results for our method have been given. Experimental studies show that RIAC has a higher average prediction accuracy on 21 databases than C4.5RULE. However, this improvement may be offset by an increase in computational costs. We do not conclude that RIAC is better for all applications than C4.5RULE. Different systems will be suitable for different real-world domains. With the growth of applications of knowledge discovery from databases, a variety of approaches are needed,

and the RIAC approach provides a useful addition.

In future work, we will try different measurements for attribute selection. Rule generation may be improved by allowing the generation of more general rules. Better results might be obtained by relaxing the current requirement that a general rule have the same classification accuracy as a specific rule. Partial matching might be improved by attaching a weight reflecting real-world importance to each attribute-value pair. improved by incorporating the weights of attribute-value pairs.

References

Breiman, L, Friedman, J.H., Olshen, RA. and Stone, C.J. 1984. *Classification and Regression Trees*, Wadsworth, Monterey, CA.

Catlett, J. 1991. "On Changing Continuous Attributes into Order Discrete Attributes," *European Working Session on Learning*. Springer-Verlag. pp. 164-178.

Chan, C.C. 1991. "Incremental Learning of Production Rules from Examples under Uncertainty: A Rough Set Approach," *International Journal of Software Engineering and Knowledge Engineering*, **1**(4):439-461.

Cheng, J., Fayyad, U.M., Irani, K.B. and Qian, Z. 1988. "Improved Decision Trees: A Generalized Version of ID3," *Proc. of the Fifth International Conference on Machine Learning*. Morgan Kaufmann. pp. 100-108.

Clark, P. and Niblett, T. 1989. "The CN2 Induction Algorithm," *Machine Learning*, **3**:261-283.

Domingos, P. 1995. "Rule Induction and Instance-Based Learning: A Unified Approach," *Proc. IJCAI-95*, pp. 1226-1232.

Fayyad, U.M. 1994. "Branching on Attribute Values in Decision Tree Generation," *Proc. of the 12th National Conference on Artificial Intelligence AAAI-94*. Vol. 1. pp. 601-606.

Fayyad, U.M., Weir, N. and Djorgovski, S. 1993. "SKICAT: A Machine Learning System for Automated Cataloging of Large Scale Sky Surveys," *Machine Learning: Proc. of the 10th International Conference*. Morgan Kaufmann. pp. 112-119.

Fayyad, U.M. and Irani, K.B. 1992. "On the Handling of Continuous-Valued Attributes in Decision Tree Generation," *Machine Learning* **8**:87-102.

Grzymala-Busse, J.W. 1988. "Knowledge Acquisition under Uncertainty: A Rough Set Approach," *Journal of Intelligent Robotic Systems*, **1**:3-16.

Holte, R.C., Acker, L.E. and Porter, B.W. 1989. "Concept Learning and the Problem of Small Disjuncts," *Proc. IJCAI-89*, pp. 813-818.

Kira, K. and Rendell, L.A. 1992. "The Feature Selection Problem: Traditional Methods and a New Algorithm," *Proc. AAAI-92*, pp. 129-134.

Michalski, R.S., Mozetic, I., Hong, J. and Lavrac, N. 1986. "The Multi-Purpose Incremental Learning System AQ15 and Its Testing Application To Three Medical Domains," *Proc. of the 5th National Conference on Artificial Intelligence AAAI-86*, pp. 1041-1045.

Michalski, R.S. 1983. "A Theory and Methodology of Inductive Learning," Michalski, R.S., Carbonell, J.G. and Mitchell, T.M. (eds). *Machine Learning: An Artificial Intelligence Approach*, Vol. 1. Morgan Kaufmann. pp. 83-129.

Pawlak, Z. 1991. *Rough Sets: Theoretical Aspects of Reasoning About Data*, Kluwer.

Pawlak, Z. 1984. "Rough Classification," *International Journal of Man-Machine Studies*, **20**:469-483.

Piatetsky-Shapiro, G. 1991. "Discovery, Analysis, and Presentation of Strong Rules," Piatetsky-Shapiro,G. and W.J. Frawley (eds). *Knowledge Discovery in Databases*, AAAI/MIT Press, pp. 229-248.

Piatetsky-Shapiro, G. and Frawley, W.J. (eds). 1991. *Knowledge Discovery in Databases*, AAAI/MIT Press.

Quinlan, J. R. 1992. *C4.5: Programs for Machine Learning*, Morgan Kaufmann.

Quinlan, J. R. 1986. "Induction of Decision Trees," *Machine Learning*, **1**:81-106.

Rymon, R. 1993. "An SE-Tree Based Characterization of the Induction Problem," *Machine Learning: Proc. of the 10th International Conference*. Morgan Kaufmann. pp. 268-275.

Shan, N., Hamilton, H.J. and Cercone, N. 1995. "GRG: Knowledge Discovery Using Information Generalization, Information Reduction, and Rule Generation," *Proc. of the 7th International Conference on Tools with Artificial Intelligence*, Washington, D.C., Nov. 1995. pp. 372-379.

Slowinski, R. 1992. *Intelligent Decision Support: Handbook of Applications and Advances of Rough Sets Theory*, Kluwer.

Thrun, S.B. *et al.*, 1991. "The MONK's Problems: A Performance Comparision of Different Learning Algorithms," Tech. Report CMU-CD-91-197, School of Computer Science, Carnegie-Mellon University.

Wu, X.D., Krisar, J., and Mahlen, P. 1995. "Noise Handling with Extension Matrices," *Proc. of the 7th International Conference on Tools with Artificial Intelligence*, Washington, D.C., Nov. 1995. pp. 190-197.

Ziarko, W. 1993. "Variable Precision Rough Set Model," *Journal of Computer and System Sciences*, **46**(1):39-59.