RBA: An Integrated Framework for Regression Based on Association Rules*

Aysel Ozgur † Pang-Ning Tan ‡ Vipin Kumar †

Abstract

This paper explores a novel framework for building regression models using association rules. The model consists of an ordered set of IF-THEN rules, where the rule consequent is the predicted value of the target attribute. The approach consist of two steps: (1) extraction of association rules, and (2) construction of the rule-based regression model. We propose a pruning scheme for redundant and insignificant rules in the rule extraction step, and also a number of heuristics for building regression models. This approach allows discovery of global patterns, offers resistance to noise, while building relatively simple models. We perform a comparative study on the performance of RBA against CART and Cubist using 21 real-world data sets. Our experimental results suggest that RBA outperforms Cubist and are equally as good as CART in many data sets, and more importantly, there are situations where RBA is significantly better than CART, especially when the number of noise dimensions in the data is large.

Keywords

quantitative association rules, regression, rule-based learning

1 Introduction

Recent years have witnessed increasing interest in applying association rules [4] to a variety of data mining tasks such as classification [17], clustering [24, 12, 25], and anomaly detection [14]. For instance, techniques such as CBA [17, 18] and CMAR [16] have been developed to incorporate association rules into the construction of rule-based classifiers. Such techniques have been empirically shown to outperform tree-based algorithms such as C4.5 [21] and rule-based algorithms such as Ripper [7] using various benchmark data sets.

Technique	Classification	Regression
Tree-based	C4.5, OC1, etc.	CART, RT, etc.
Rule-based	Ripper, CN2, etc.	Cubist
Association-based	CBA, CMAR, etc.	?

Table 1: Descriptive techniques for predictive modeling.

Regression is another data mining task that can potentially benefit from association rules. Regression can be viewed as a more general form of predictive modelling, where the target variable has continuous values. Currently, there is a wide spectrum of techniques for building regression models from linear to more complex, non-linear techniques such as regression trees, rule-based regression, and artificial neural networks.

Tree-based and rule-based techniques are desirable as they produce descriptive models that can help analysts to better understand the underlying structure and relationships in data. Table 1 presents a taxonomy of descriptive techniques used for classification and regression. Another class of techniques that can produce descriptive models are based on association rules. These techniques are useful as they can efficiently search the entire input space to identify a set of candidate rules for model building. This differs from the approach taken by many tree-based or rule-based techniques, which must grow a tree branch or rule from scratch in a greedy fashion, without the hindsight of knowing whether it will turn out to be a good subtree or rule. Moreover, since the association rules must satisfy certain support criterion and are evaluated over all the instances, models built from these rules are less susceptible to noise.

Table 1 also highlights an important class of techniques still missing from the taxonomy, namely, association-based techniques for regression. In this paper, we present a general framework, Regression Based on Association (RBA), for building regression models using association rules. The model consists of a collec-

*This work was partially supported by NASA grant # NCC 2 1231, NSF Grant IIS-0308264, DOE/LLNL Grant W-7045-ENG-48, and by the Army High Performance Computing Research Center cooperative agreement number DAAD19-01-2-0014. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred. Access to computing facilities was provided by the AHPCRC and the Minnesota Supercomputing Institute.

†Department of Computer Science and Engineering University of Minnesota {aysel, kumar}@cs.umn.edu

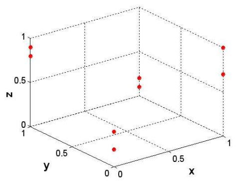
‡Department of Computer Science and Engineering Michigan State University ptan@cse.msu.edu

tion of IF-THEN rules, where the rule consequent contains the predicted value of the target variable. The proposed techniques include single-rule (1-RBA) and multi-rule (weighted k-RBA) schemes. In the single rule scheme, each test example is predicted using a single association rule whereas in the multi-rule scheme prediction is a weighted sum of several association rules.

To illustrate the advantages of RBA, consider the synthetic data set shown in Figure 1. This data set is an example of the well-known, XOR-type problems, where the target variable depends on the input attributes x and y in the following manner:

$$f(x, y) = \begin{cases} \leq 0.5, & \text{if } x = 0,\ y = 0 \\ > 0.5, & \text{if } x = 0,\ y = 1 \\ > 0.5, & \text{if } x = 1,\ y = 0 \\ \leq 0.5, & \text{if } x = 1,\ y = 1 \end{cases} \tag{1.1}$$

x	y	z
0	0	0.2
0	0	0.4
0	1	0.9
0	1	0.8
1	0	1
1	0	0.7
1	1	0.1
1	1	0.2



CUBIST results:

Read 8 cases (3 attributes) from xor.data

Model: Rule 1: [8 cases, mean 0.54, range 0.1 to 1, est err 0.40]
target = 0.54

Evaluation on training data (8 cases):
Average |error| 0.31
Relative |error| 1.00
Correlation coefficient 0.00

Figure 1: An illustrative example: the XOR data set

Figure 1 shows the results obtained using Cubist [20], a rule-based algorithm. As can be seen, Cubist fails to induce meaningful rules and predicts everything to be the mean value of the training examples. This can be attributed to the greedy nature of the algorithm, attempting to grow a rule by choosing the current best attribute at each stage. Since none of the attributes provide a better prediction than the overall mean, Cubist is not able to grow a rule. Similarly, given the limited number of training examples, CART produces a decision stump, i.e., a tree with a single node, that encodes the mean value of the training examples.

The single-rule RBA scheme generates association rules from the input data and then selects the best representative set of rules for building the regression model. The rules generated for the above example are:

x=0 y=1 $\rightarrow$ μ=0.85, σ=0.07, support=0.25

x=1 y=1 $\rightarrow$ μ=0.15, σ=0.07, support=0.25

x=0 y=0 $\rightarrow$ μ=0.3, σ= 0.14, support=0.25

default: μ=0.85, σ=0.55

The above rules, including the default, specify exactly the four regions shown in Equation 1.1. Unlike the greedy strategy employed by CART and Cubist, RBA uses an exhaustive search to identify rules covering regions in the input space where the number of data points is sufficiently large and the target variable has low variance. A low variance rule ensures that its prediction has relatively small expected error.

There are several issues that need to be addressed in order to successfully incorporate association rules into the regression problem. First, standard association rule formulation assumes that the data is binary in nature. Yet, many real-life data sets contain continuous as well as categorical input attributes. This problem can be addressed by discretizing the continuous attributes and replacing each discrete interval or categorical value with a distinct binary attribute[1]. While MDL-based discretization [10] has been found to be very effective in classification, such a method becomes applicable to regression by mapping the continuous values of the target to the closest bin only for the discretization part.

The second issue is that the right hand side of a regular association rule is a class label, not a continuous value. To address this, we apply the quantitative association rule definition proposed by Aumann et al. [5], which captures the descriptive statistics (e.g., mean or variance) of the target variable in the region covered by an association rule. For example, the following quantitative association rule

$\texttt{Age} \in [31, 45) \wedge \texttt{Status} = \texttt{Married} \longrightarrow \texttt{Income}\text{: mean} = 80\text{K}.$

suggests that the average income of a married person, whose age is between 21 and 45, is 80K. The descriptive statistics investigated in this study include measures of central tendency (mean and median), measures of dispersion (variance and mean absolute deviation), and measures of significance (support).

Finally, choosing the appropriate set of rules for building a regression model is non-trivial. There is often a tradeoff between the model complexity and accuracy. Here, the rules are added incrementally as long as the prediction of the overall model is improved. This approach is very similar to the sequential covering approach used by many rule-based classifiers [7].

The major contributions of this paper are summarized below:

- We present a general framework for building regression models using association rules. The proposed framework offers the ability to search the entire input space efficiently to identify a promising set

[1] This approach is similar to the one used by classification based on association rules (CBA) methods [17]

of candidate rules, while offering robustness in the presence of noise.

- We extensively compare the performance of RBA against two leading rule-based regression algorithms (CART and Cubist) using 21 real-world data sets. Experimental results suggest that RBA outperforms Cubist and performs as good as CART in majority of these data sets, provided that discretization is not an issue. More importantly, we were able to determine the characteristics of the data sets for which RBA is significantly better than CART, e.g., when the number of noise dimensions in the data is large.

The remainder of this paper is organized as follows. Section 2 presents the related work in this area while Section 3 describes the problem formulation. The various components of the RBA framework are discussed in Section 4. Section 5 provides the experimental results, and, finally, Section 6 presents the conclusions.

2 Related Work

Various techniques have been developed for building regression models from data. These techniques often have solid mathematical foundations, optimizing certain loss functions. While the produced models can achieve very high accuracy, they are not descriptive enough to be easily interpreted by human analysts, with the exception of rule and tree-based models. A tree-based model, such as CART [6] partitions the input space into smaller, rectangular regions, and assigns the average of the target attribute as predicted value to each region. Another technique, Cubist [20], uses a rules-based approach for partitioning the input space, fitting a linear regression model to each of the regions.

Association rule mining from numeric data has been investigated in many studies [23, 22, 15, 11, 5]. The rules discovered for such data sets are often known as quantitative association rules. However, most of the work in this area is concerned with integrating continuous attributes into the rule antecedent [23, 11, 15, 22], while the rule consequent contains only categorical or discretized numeric attribute(s). With this approach, conventional definitions of support and confidence measures are still applicable. As previously noted, such rules are not directly applicable to regression.

Aumann et al. [5] proposed a quantitative association rule formulation where the rule consequent is the arithmetic mean of the continuous target variable for all instances covered by the rule. For evaluation purposes, they consider a rule to be interesting if its mean significantly differs from the mean for the rest of the population, where the significance tests were performed using Z-test. The authors have also noted that the rule consequent can contain other statistical information such as the variance or median of the target variable. The quantitative association rules used in the RBA framework are based on this formulation. We are specifically interested in the description of rule consequents that include either mean, variance and support, or median, mean absolute deviation and support. We also applied a modified statistical test to determine the significance of a rule (Section 4).

Morishita et al. [19] describe a formulation where the input data may be discrete, but each transaction is associated with a numeric attribute. In this case the goal is to find itemsets that are highly correlated with the numeric factor. They proposed an interestingness measure, "interclass variance", and showed that it possesses certain monotonicity property that can be incorporated directly into the mining algorithm.

Discretization is another important step in the RBA framework. A robust supervised discretization strategy for classification problems was developed by Fayyad et al. in [10]. The method was based on the entropy of class distribution in each discretized interval, and the number of intervals is controlled by using the Minimum Description Length Principle (MDL). Experiments have shown that such a discretization strategy is more tolerant to noise, and outperforms unsupervised discretization schemes [10, 9]. We apply the MDL-based discretization in the RBA framework, with two enhancements, which try to avoid infrequent bins after the discretization, and a loss of an attribute due to "no-split" decision.

Another related work to the proposed approach integrates association rules into classification problems [17, 18]. In fact, our regression framework generalizes the Classification Based on Association (CBA) [17], focusing on the classification problem, which is in general easier than regression. The implementation of main components (discretization, rule extraction, and building the model) are significantly different since we are dealing with continuous-valued target variables. The proposed weighted k-RBA scheme is based on the idea of using multiple association rules for prediction, initially used by Li et al. [16] in classification using associations (instead of an unordered evaluation scheme we use an ordered multi-rule approach, and also our multi-rule weighting schemes are fundamentally different).

3 Preliminaries

Let $D = \{(\mathbf{x}_i, y_i) | i = 1, 2, \cdots, N\}$ be a data set with N instances. Each instance consists of a tuple $(\mathbf{x}, y)$, where $\mathbf{x}$ is the set of input attributes and y is the real-valued target variable. Regression is the task of finding a target function from $\mathbf{x}$ to y, such that $\hat{y} = f(\mathbf{x})$ provides a good estimate of the target variable for any given $\mathbf{x}$. The discrepancy between the estimated and actual value of y can be measured in terms of their mean squared error (MSE) or mean absolute error (MAE).

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y_i - f(\mathbf{x_i})|$$

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} (y_i - f(\mathbf{x_i}))^{\mathbf{2}} \tag{3.2}$$

The input vector $\mathbf{x}$ contains binary, categorical, or continuous attributes. RBA transforms the categorical and continuous attributes into binary attributes through discretization and binarization of attribute values. Let $I = \{i_1, i_2, \cdots, i_d\}$ denote the new set of attributes in binary form. Following the terminology used in association rule literature, a binary attribute is called an item while any subset of I is known as an itemset. Additionally, the number of data points containing the itemset A is denoted as $\sigma(A)$.

A quantitative association rule is an implication expression in the form $A \longrightarrow (m, d, S)$, where m is a measure of central tendency (mean $\hat{\mu}$, or median $\hat{m}$), d is a measure of dispersion (variance $\hat{\sigma}^2$, or mean absolute deviation $\hat{\delta}$), and S is a measure of significance (support s). These measures are computed based on the statistical distribution of instances covered by the rule. Specifically,

$$\hat{\mu} = \frac{\sum_{i \in A} y_i}{|A|}, \quad \hat{\sigma}^2 = \frac{\sum_{i \in S}(y_i - \hat{\mu})^2}{|A| - 1},$$

$$\hat{m} = \text{median}_{i \in A}(y_i), \quad \text{and} \quad \hat{\delta} = \frac{1}{|A|}\sum_{i \in A} |y_i - \hat{m}|.$$

Using the definition given in [5], the support of a quantitative association rule is the fraction of instances that satisfy antecedent of the rule, i.e., $s(A) = \frac{\sigma(A)}{N}$. Support also has an anti-monotone property: $s(A \cup C) \leq s(C)$ for all A, $C \subseteq I$. An association rule is considered to be frequent if its support exceeds a user-specified minimum support threshold. Depending on the choice of the loss function, different statistics may be used to describe the rule consequent. For instance, if the objective is to minimize MSE, then it makes sense to include the mean value of the target variables $\hat{\mu}$ in the rule consequent [8]. On the other hand, if the objective is to minimize MAE, the best estimate of central tendency is given by the sample median.

The regression model constructed by the RBA framework consists of an ordered list of association rules $(r_1, r_2, \cdots, r_n)$. These rules partition the input space into homogeneous, rectangular regions, assigning a predicted value to each region (based on the mean or median of the rule consequent). The regions can be overlapping or disjoint depending on the method used for constructing the model (Section 4). If R_r is the *effective region*[2] of an association rule r, then the model constructed by RBA can be represented as:

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i \times \mu_{r_i} \times I[\mathbf{x} \in R_{r_i}], \tag{3.3}$$

where w_i is a weight factor for rule r_i and $I[\cdot]$ is an indicator function, whose value is 1 when its argument is true and 0 otherwise.

4 Framework for Regression based on Association Rules (RBA)

Figure 2 illustrates the general framework of the regression based on association rules (RBA) scheme. There are four major components in this framework: (1) discretization, (2) association rule generation, (3) regression model building, and (4) model testing.

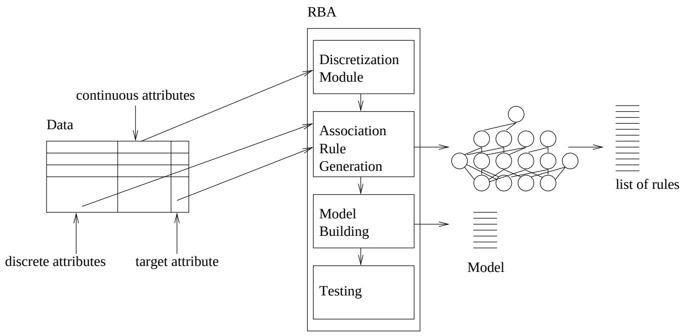


Figure 2: General RBA framework

The discretization step is needed to partition the numeric features of the data set (except for the target variable) into discrete intervals. Each discrete interval and categorical attribute value is then mapped into a binary variable. This step is needed to ensure that existing algorithms such as Apriori [4] or FP-tree [13] can be applied to generate frequent itemsets. Next, each extracted frequent itemset is converted into a rule by computing the descriptive statistics (e.g., mean, variance, or median) of the target variable for all training instances covered by the itemset. A rule pruning step is then applied to systematically eliminate rules that are redundant or statistically insignificant. During model building, a subset of the extracted rules will be selected for constructing the regression model. The induced model is then applied to predict the target value of test instances.

[2] A region that excludes the subspace covered by other, higher precedent rules.

4.1 Discretization We apply the MDL-based supervised discretization method [10] to continuous target variables. This method was originally developed for classification problems, where the range of each continuous input attribute is successively partitioned into smaller discrete intervals until each interval contains instances with relatively homogeneous classes, i.e., their overall entropy is low.

In our approach, the continuous target variable is divided into equally spaced k bins, that are consecutively labelled from 1 to k, where k is specified by the user. Then, discretization of the continuous input attributes is performed according to the new class labelling of the target variable (note that this mapping is preformed only in this section to aid the discretization of input attributes). One potential limitation of this approach is that the entropy measure used in MDL-based discretization scheme will not differentiate between splits containing instances of classes with varying distance from each other (e.g. instances within classes 1, 2, and 3 are closer, while the instances within classes 1, 25, and 50 are farther apart. However, those two cases are penalized equally using the entropy measure.)

We add two constraints to the discretization algorithm: minimum support for each bin, and default equal width split (when MDL-based discretization decides not to partition the input variable).

4.2 Rule Generation Next, we apply a standard frequent itemset generation algorithm (Apriori [4]) to the discretized data set. Each frequent itemset is turned into a rule by computing the needed statistics for the itemset, as described in Section 3. For example, the rule could be of the form $x_k..x_l \rightarrow (\hat{\mu}, \hat{\sigma}^2, support)$ or $x_k..x_l \rightarrow (\hat{m}, \hat{\delta}, support)$, where $\hat{\mu}$ is the mean of the target attribute, $\hat{\sigma}^2$ is the variance, $\hat{m}$ is the median, and $\hat{\delta}$ is the mean absolute error.

Two pruning strategies are applied to eliminate rules that are redundant or statistically insignificant.

- **Redundancy Test:** Let $r : A \longrightarrow \mu_1, \sigma_1^2$ and $r' : A' \longrightarrow \mu_2, \sigma_2^2$ be two association rules. If $A' \subset A$, then r' is said to be a generalization (or ancestor) of r while r is said to be a specialization (or descendent) of r'. A rule r is *redundant* if there exists a generalization of the rule, r', that produces a lower variance. For example, suppose $r : ABC \longrightarrow \mu_1 = 0.1, \sigma_1^2 = 0.4$ and $r' : AC \longrightarrow \mu_2 = 0.2, \sigma_2^2 = 0.25$ are two association rules. In this case, r is redundant since it has a generalization (r') that produces a lower variance (0.25 versus 0.4).

  If the rules are specified using median and mean absolute error, we can perform a similar redundancy test by comparing its mean absolute error to the minimum absolute error of its ancestor rules.

- **Significance Test:** Let $r : A \longrightarrow \mu_1, \sigma_1$ be an association rule. We call $\overline{r} : \overline{A} \longrightarrow \mu_2, \sigma_2$ the complement of r since $\overline{r}$ covers only those instances that do not satisfy the conditions given by the antecedent of r. In [5], the authors perform a Z-test to compare the mean values of a rule r against its complement, $\overline{r}$. The null hypothesis to be tested is $\mu_1 = \mu_2$. If the similarity is statistically significant, then r is pruned. Although such a pruning strategy may be useful from a descriptive data mining perspective, it may not be useful from a predictive perspective. For example, suppose r and $\overline{r}$ have the same mean but the variance for r is much lower than the variance for $\overline{r}$. This suggests that r may still be a useful rule for prediction. In the RBA framework, the rule is kept, if the difference is statistically significant. Otherwise, we compare the variance of r against its complement $\overline{r}$. If r has a larger variance, then the rule is pruned.

Procedure RruleGen(D: set of training instances, $mins$: minimum support)

Let F_1 be the set of frequent 1-itemsets
For each item $i \in F_1$
create the rule $(i \longrightarrow \mu, \sigma^2)$ or $(i \longrightarrow m, \delta)$
for(k=2;$F_{k-1} \neq \emptyset; k = k + 1$)
  C_k=apriori-gen(F_{k-1});
  For each candidate $c \in C_k$,
  compute its mean, variance, and support;
  Let $F_k = \{c | c \in C_k,\ support(c) \geq minsup\}$;
  Prune itemsets in F_k according to
  the redundancy and significance tests;
endfor
return $\forall k\ F_k$;

Figure 3: Rule Generation Procedure

Using the pruning strategies described above, we are left with (1) rules that are more precise than their ancestors and (2) rules that have very different characteristics (mean or median) or lower variance (or mean absolute error) than those describing the rest of the population. The pseudocode for the rule generation step is shown in Figure 3.

4.3 Building a Regression Model Building a regression model requires selection of a smaller, representative set of rules that provides an accurate representation of the training data. More specifically, rules are selected to minimize a certain loss function 3.2. RBA applies several variations of the sequential covering algorithm to generate a regression model. The basic idea of this algorithm is to choose the best remaining rule available in a greedy fashion, add the rule to the model, and then remove instances covered by the rule.

In order to do this, we must first sort the extracted rules according to certain objective criteria. Given a pair of rules, say, r_1 and r_2, let $r_1 \succ r_2$ denote that r_1 has higher precedence than r_2. Naturally, a good rule must have sufficiently high support, i.e. it should cover a large portion of the data, to avoid overfitting. In addition, a good rule must be precise, i.e. the target values of the covered instances must have relatively low variance. Finally, if the support and variance for two rules are the same, the more general rule is preferred.

The following order definition is used to sort the rules: rule r_1 is said to precede r_2 (1) if the variance of r_1 is smaller that variance of r_2, (2) if the variances are the same, but the support of r_1 is greater than that of r_2, or (3) if both variance and support are the same, but condition size of r_1 is smaller than r_2.

There are a number of ways to implement the sequential covering algorithm. This depends on whether each instance should be predicted by a single rule or multiple rules. In the case of a single rule (1-RBA, Section 4.3.1), once a rule has been selected, all instances covered by the rule will be eliminated immediately. If multiple rules are allowed to fire, we need a weighted voting scheme (k-RBA, Section 4.3.2) to determine the predicted target value.

4.3.1 1-RBA 1-RBA adds a rule only if it reduces the prediction errors made by the previous model. The instances covered by the added rule are removed from the data. The rule selection process continues until there are no remaining instances or rules. Each time a new rule is added to the model, the error of the model along with its current default value is recorded. When the stopping criteria is reached, the rules after the index where the minimum error has been recorded are discarded, and the corresponding default value is restored, so that the final model is a list of ordered rules and a default value.

A summary of the 1-RBA algorithm is shown in Figure 4. The algorithm takes a list of sorted rules and the set of training instances as input, and returns a final model M. The getStatistics function is used to compute the mean value of the remaining training instances not covered by the model. The evaluateRule function is used to compute the mean square error or mean absolute error of the training instances covered by the rule r, and the error of the predictions made by the previous model M. evaluateModel function evaluates the model M with default value μ_d over all samples D.

```
Procedure 1-RBA(Rlist: list of sorted rules,
                D: set of training instances)
    Initialize model: M = ∅;
    Initialize error list: e = ∅;
    Let D' = D;
    μ_d = getStatistics(D');
    prev_err=evaluateModel(M, μ_d, D);
    while(Rlist ≠ ∅ and D' ≠ ∅)
        r: remove the top rule from Rlist
        Let S ⊆ D' be the instances in D' covered
        by r;
        if(S ≠ ∅)
            [err,prev_err]=evaluateRule(r,M,μ_d);
            if(err < prev_err)
                M.push(r);
                D' = D' − S;
                μ_d=getStatistics(D');
                e.push(evaluateModel(M,μ_d,D));
            endif
        endif
    endwhile
    find rule index i in M with min error in e;
    remove rules in M after position i;
    return M;
```

Figure 4: 1-RBA Procedure

4.3.2 Weighted k-RBA The main disadvantage of the previous algorithm is that the decision about the target value of an instance is made based on the prediction of a single rule. If the prediction error is high, there is no other way to modify the prediction. Alternatively, we can extend the previous scheme to solicit the opinions of a mixture of experts, i.e., using an ensemble of k rules, while weighting each opinion according to how reliable the rule is. This is the basis for the weighted k-RBA algorithm.

Let μ_r be the predicted value by the rule r, and w_r be the weight of the prediction made by r. Suppose $\{r_1, r_2, \cdots, r_k\}$ are the set of k rules selected by the regression model for predicting the test instance z. The predicted value for z is given by

$$f(z) = \frac{\sum_{i=1}^{k} w_{ri} \times \mu_{ri}}{\sum_{i=1}^{k} w_{ri}}, \tag{4.4}$$

Several weighting schemes are investigated:
(1) For the `precision-k` weighting scheme, each prediction is weighted by the inverse of the variance for the rule. Here the inverse of the variance is evaluated as an

estimation of the precision of the prediction.
(2) For the `probabilistic-k` weighting scheme, we use support as the weight for each prediction.
(3) For the `average-k` weighting scheme, each prediction is weighted equally, i.e., $w_r = 1/k$.

Procedure k-RBA(*Rlist*: list of sorted rules,
    D: set of training instances,
    k: number of experts,
    w_r: weighting scheme)
1. Initialize model: $M = \emptyset$;
2. Let $D' = D$;
3. μ_d = getStatistics(D'); /* default prediction */
4. $\forall z \in D' : z.pred = \mu_d$,
    $z.weight = 0$, $z.count = 0$; /* initialization */
5. **while**($Rlist \neq \emptyset$ and $D' \neq \emptyset$)
6. r: remove the top rule from *Rlist*;
7.   Let $S = \{z | z \in D', z.count < k, r \text{ covers } z\}$;
8. $r_{err} = 0$; $prev_{err} = 0$;
9.   **for**(each instance $z \in$ S)
10. $pred = \frac{z.pred \times z.weight + w_r \times \mu_r}{z.weight + w_r}$;
11.   /* new prediction */
12. $r_{err} = r_{err}$ + Difference($pred$, $z.y$);
13. $prev_{err} = prev_{err}$
            + Difference($z.pred$, $z.y$);
14.   **endfor**
15.   **if**($r_{err} < prev_{err}$)
16. M.push(r);
17.   **for**(each instance $z \in S$)
18. $z.pred = \frac{z.pred \times z.weight + w_r \times \mu_r}{z.weight + w_r}$;
19. $z.weight = z.weight + w_r$;
20. $z.count = z.count + 1$;
21.   **if**($z.count = k$)
22. $D' = D' - z$;
23.   **endif**
24.   **endfor**
25. μ_d = getStatistics(D');
26. $\forall z \in D'$ such that $z.count = 0$,
            assign $z.pred = \mu_d$.
27.   **endif**
28. **endwhile**
29. **if**($D' \neq \emptyset$)
30. μ_d = getStatistics(D');
31. **else** μ_d = getStatistics(D);
32. **endif**
33. return M;

Figure 5: k-RBA Procedure

In order to implement a multi-rule scheme, each training instance z must keep track of several counters:
$z.y$: the actual target value for z.
$z.count$: the present number of rules covering z.
$z.pred$: the current predicted value for z, i.e., $f(z)$.
$z.weight$: the sum of the weights for all the rules covering z, i.e., the denominator of Equation 4.4.

Figure 5 depicts the weighted k-RBA algorithm. Unlike the 1-RBA approach, each instance is not removed until there are k rules covering it. In addition, each rule also keeps track of the amount of error it has committed on the training instances (r_{err}). A rule r is added to the model only if it improves the prediction of instances covered by r (lines 13-14). This implicitly provides resistance to adding rules that are highly correlated/similar to each other, and does not have additional predictive value. The model building procedure terminates when there are no more rules or instances remaining. The default mean is calculated over the remaining instances, if any, otherwise it is assigned to be the mean value of the training set. Once the stoping criteria is reached, as in 1-RBA, the rules after the index where the minimum error has been observed are removed, and the corresponding default value is restored.

5 Experimental Results

Our algorithm was evaluated on both real and synthetic data sets. Synthetic data sets allows us to study the effects of noise on the performance of regression models in a controlled environment, while real data sets are useful to evaluate the relative performance of RBA against other existing algorithms.

5.1 Synthetic Data

The synthetic data is very similar in nature to the XOR example given in Section 1. The target variable depends only on the values of its two "clean" dimensions, x and y. If both x and y have the same values, then the target variable is zero; otherwise, the target variable is one. Note that the truncation to 0 and 1 is to simplify the discussion, and the results presented in this section still hold if the target is a similar continuous valued function on those 4 quadrants. Although attributes, together, contain sufficient information for making the correct prediction, alone, neither of them can predict the target variable correctly.

x	y	n_1	n_2	n_3	n_4	...	target
0	0	1	0	0	1	...	0
0	1	1	1	1	0	...	1
1	0	0	1	1	1	...	1
1	1	0	0	1	0	...	0
...	...	...	...	...	...	...	...

Table 2: Synthetic Data example

5.1.1 Synthetic Data Generator

The synthetic data is parameterized by two parameters: the number of noise dimensions and the size of the training set. Noise dimensions are added to the synthetic data in the following way: first, a random number between 0 and 1 is generated. If the number is greater than 0.5, then the value of the noise dimension is 1, otherwise its value is 0. This process is repeated for every entry in the noise dimensions. Table 2 shows an example of the synthetic data, where each n_i correspond to i^{th} noise dimension.

The above approach is used to generate equal number of training and test data sets. The training set is used for model building, while the test set is used to estimate the MSE. All the MSE results reported in the remainder of this section correspond to the average MSE value of 30 trials.

5.1.2 Effect of Noise on 1-RBA and CART The objective of this experiment is to evaluate the robustness of the regression models in the presence of noise. The following evaluations are conducted:

1. The effect of increasing the training set size on MSE of the induced models, given a fixed number of noise dimensions.

2. We study the number of training samples needed to achieve an acceptable mean square error (MSE) while increasing number of noise dimensions.

3. We study the complexity of the regression model while increasing number of noisy dimensions.

For these experiments, we report the results of 1-RBA and CART. Cubist is excluded, and not investigated further, as it produces only a single default rule-the average value of the target variable for the clean data set as well as with one or two noise dimensions[3].

Figure 6 compares the MSE for 1-RBA against CART as the size of training set increases for a synthetic data set with noise dimensions set to four. It is intuitively clear that the MSE for both methods should decrease with increasing number of training examples, as seen in the Figure 6. This experiment demonstrates that CART needs significantly more data to learn the right concept compared to RBA. If both were provided with only 100 samples for this six-input features problem, RBA would clearly outperform CART.

Figure 7 shows the effective number of training examples needed by both algorithms to learn the correct model (i.e., a model that achieves an MSE $\leq \epsilon$ over the test set, where $\epsilon = 0.01$). Notice that CART requires significantly more training examples in order to learn the correct model compared to 1-RBA. For CART, the size of the training set grows exponentially as the number of noise dimensions increases. This observation suggests that CART is more susceptible to noise, and

[3]Note that the distribution of x and y values are always maintained to be uniform throughout the experiment.

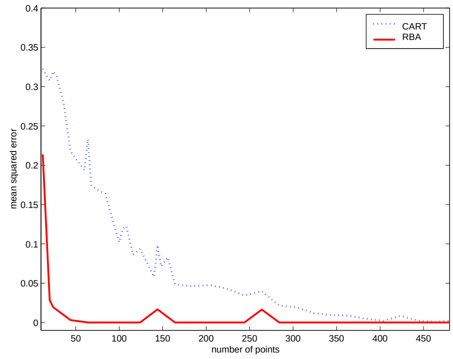


Figure 6: MSE versus number of training samples for 2D XOR with 4 noise dimensions

may produce models containing irrelevant features. 1-RBA is more robust as the needed number of training samples grows less rapidly (at most linearly for these data sets) as the number of noise dimensions increases.

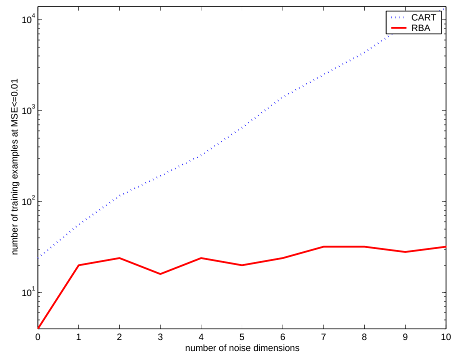


Figure 7: Effective size of training set needed to achieve MSE ≤ 0.01 versus number of noise dimensions

In general, techniques that are susceptible to noise tend to produce models that overfit the training data, producing rules that may contain many irrelevant attributes. There are several ways to capture the complexity of a model. For rule-based methods, model complexity can be expressed in terms of the number of rules, while for tree-based methods, it can be measured in terms of the number of leaf nodes (since each path from the root to the leaf forms a unique rule). As shown in Figure 8, the average size of the rule set for 1-RBA is four, which corresponds to the four distinct (x, y) values for learning the XOR concept. On the other hand, the complexity of models produced by CART is several orders of magnitude higher than 1-RBA and grows exponentially with increasing number of noise dimensions. This observation supports our previous assertion that CART tends to overfit noisy training data by creating unnecessarily complex models. Although tree-based algorithms such as CART may reduce overfitting via tree pruning (beginning from the leaves) or using the early stopping strategy, such approaches do not remove irrelevant attributes appearing at higher levels of the tree.

Another way to capture model complexity is the average number of conditions of the rules, or equivalently, the average depth of the regression tree. The average depth of the regression trees increases linearly with the number of noise dimensions, while remaining fixed at two for RBA.

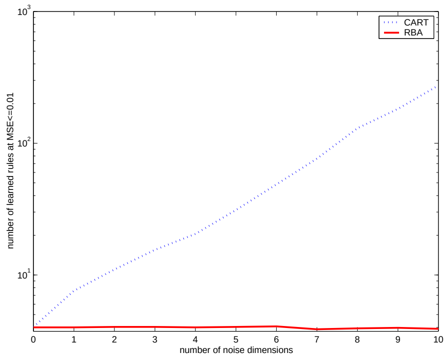


Figure 8: Number of Learned rules vs. number of noise dimensions

Overall, the experiments in this section demonstrate the fundamental advantage of a regression based on association rules scheme: by exploiting global patterns, it avoids irrelevant features in its rules, as well as irrelevant rules in its model. It does not suffer much from the sparsity of the data points in the input space: if there are underlying patterns in the data, it does not need as many points to induce the correct pattern as other techniques do. RBA's success is not limited to the type of the synthetic data presented in this section: the patterns may be much more complex, multi-modal, multi-valued, etc. As long as the discovered global patterns are highly precise, its vision may not be easily obscured by local noise and disturbances.

5.2 Real Data For these experiments, we used 21 real data sets obtained from public-domain sources such as the UCI Machine Learning Repository [3] and CMU StatLib [2], as well as our own in-house data. The experiments were performed on a Pentium 4, 2.8 GHz Linux machine with 1 GB RAM. The RBA algorithms were implemented in C++, and are reasonably optimized.

5.2.1 Methodology As previously noted, current association rule mining algorithms assume that the input data is in binary format. Since most of the real-life data sets are non-binary, some information will be lost during discretization, thus degrading the overall performance of RBA. Other techniques such as CART[4] and Cubist [20] do not suffer from such limitations.

Our goal is to provide a systematic performance study of RBA in comparison to other techniques such as CBA and Cubist. To highlight the strengths and limitations of RBA, we need to isolate the effects of discretization from model building. Therefore, we have conducted two series of experiments, first using the discretized data, and second, using the original data.

Each model is evaluated based on the Pearson correlation coefficient between the predicted value and actual value of the target variables. The correlation coefficients are obtained from evaluation of concatenated results of the test partitions with 10-fold cross validation. The same data partitions in the 10-fold cross validation are used for training and testing for all competing algorithms. Correlation difference test is applied to determine the Win-Loss-Draw counts.

For RBA, the appropriate choice of minimum support threshold depends on a number of factors such as target variable distribution, size of training set, multimodality, etc. We have used the default support threshold (0.01) for all of our experiments. Nevertheless, for large data sets, the support threshold might still be too high to extract a reasonably complete set of association rules, especially if there are many underlying structures in the data. Therefore, in addition to the default value, we have also reported the results using several alternative minimum support thresholds ($3/|D|$, $5/|D|$, and $10/D$). For k-RBA, the number of experts is fixed to be its default value $k = 5$. Cubist and CART are also applied with their default settings.

Table 3 summarizes the characteristics of the data sets and the total run time for both 1-RBA and k-RBA algorithms. The run time for association rule generation is reported in a separate column since it is the same for both 1-RBA and k-RBA. Furthermore, the model building run time and model sizes are very similar for all three versions of multi-rule RBA. In Table 3 the average total run time and average model size for the three variations of the multi-rule scheme is reported.

Table 4 shows the correlation coefficients for the discretized data sets applying CART, Cubist and RBA. The best results obtained for each data set are shown in boldface. For some of the RBA algorithms, we have also reported the best correlation result obtained using alternative support thresholds (beside the default 0.01) as described above. For space considerations, we only show the results of using MSE, as results using MAE were similar.

[4] Matlab's implementation of CART is used [1].

Data set name	Data size	# Orig. Attr.	# items	total 10-cfv Rgen time (s.)	avg. Rgen # rules	total 1-RBA time (s.)	avg # 1-RBA rules	total k-RBA time (s.)	avg # k-RBA rules
autompg	398	7	44	2.97	526.2	0.92	114.1	1.68	192.03
bodyfat	252	13	62	28.82	2586	3.3	115.3	6.53	289.53
bolts	40	6	19	0.11	37.7	0.07	6.5	0.07	7.27
californiahouses	20640	8	63	86.03	40.5	7.71	11	13.33	13
case2002	147	6	14	0.54	91.4	0.2	20.2	0.28	28.3
crabs	200	4	19	0.15	72.7	0.13	24.6	0.15	36.3
housing	506	13	89	107.69	3765.6	6.11	225.3	16.62	534.87
kidney	76	5	16	0.14	44.1	0.1	17.3	0.08	17.87
logis	70	7	16	0.37	119.1	0.1	19.6	0.17	34.07
machine	209	8	48	2	267.7	0.45	53.9	0.57	91.67
mtwashnh6597	12053	8	47	75.41	483.9	23.14	135.6	56.05	131.57
mulimputbvn	100	5	10	0.18	41	0.14	10.1	0.13	13.8
ozone	330	8	20	4.08	622.1	0.89	94.7	1.59	161.33
plasmabcarotene	315	12	40	38.38	3413.9	5.14	167	11.01	284.50
pollen	3848	4	30	3.41	244.5	2.2	108.5	6.03	108.50
pollution	60	15	58	1.1	297.8	0.34	25.2	0.42	59.20
sensory	576	11	36	46.11	4175.6	7.06	219.5	17.19	515.97
servo	167	4	19	0.12	63.8	0.1	36.4	0.13	42.5
socmob	1156	5	43	1.87	222.2	1.23	167.6	2.26	157.53
spacega	3107	6	40	10.54	474.5	5.03	159.5	11.82	195.17
weather	84	7	25	0.45	94	0.17	21.2	0.2	24.47

Table 3: Data set properties, rule generation and model building total timing and average rule sizes

Method	CART	CUBIST	1-RBA	precision-k-RBA	probabilistic-k-RBA	average-k-RBA
autompg	**0.9074**	0.8732	0.8408	**0.9045**	**0.9007**	**0.8992**
bodyfat	0.6897	0.7171	0.4786,0.7142	**0.7396,0.7428**	0.7027,0.7222	**0.7247,0.7320**
bolts	0.9344	0.9026	**0.9672**	0.9059	0.9004	0.9034
californiahouses	**0.7682**	0.5840	0.6768	0.6989	0.7033	0.7033
case2002	0.5538	**0.6314**	0.5231,0.5800	0.5695,0.5904	0.5607,0.5817	0.5681
crabs	**0.9734**	**0.9720**	**0.9720**	**0.9745**	**0.9735**	**0.9742**
housing	0.8797	0.8413	0.6641,0.7174	0.8802	**0.9054**	**0.8921**
kidney	0.3693	0.4128	0.3697,0.3761	0.3996,**0.4641**	**0.4348,0.4645**	**0.4303,0.4761**
logis	0.4811	0.4749	0.4694,**0.5160**	**0.5373**	**0.5972**	**0.5591**
machine	n/a[5]	0.7925	0.6520,0.7947	0.7987,**0.8144**	0.7385,0.7880	0.7923
mtwashnh6597	**0.4967**	0.4002	0.4454	0.4608	0.4765	0.4667
mulimputbvn	0.7656	**0.7739**	0.7601,**0.7806**	**0.7778,0.7813**	0.7505,**0.7841**	0.7549,**0.7812**
ozone	0.7800	0.7803	0.7481,0.7881	0.7822,**0.7988**	0.7885,**0.8065**	0.7894,**0.8022**
plasmabcarotene	0.1113	0.1143	**0.1409**	**0.2256,0.2817**	**0.2480,0.2763**	**0.2526,0.2684**
pollen	**0.7705**	0.5794	0.6882,0.7494	0.7368,**0.7617**	0.7194,0.7583	0.7330,0.7565
pollution	0.4643	0.5260	0.1270,0.4276	0.5129,**0.5823**	0.5233,**0.5905**	**0.5560,0.6306**
sensory	0.4105	0.2498	0.2915	0.4035	**0.4430,0.4718**	0.4079,**0.4653**
servo	**0.9171**	0.9067	0.7945	0.8114	0.8929	0.8907
socmob	**0.8157**	0.5860	0.7298,0.7500	0.7065,0.7244	0.7598,**0.8104**	0.7658,**0.8028**
spacega	**0.7378**	0.6629	0.7157,0.7294	**0.7353,0.7418**	0.7205	**0.7304**
weather	0.5160	0.5476	**0.7464**	**0.7183**	**0.7262**	**0.7180**
average	0.6671	0.6347	0.6571	**0.6942**	**0.7072**	**0.7054**

Table 4: Pearson Correlation coefficients for CART, Cubist, and the proposed four RBA methods

5.2.2 Comparison between RBA and Cubist

Table 5 summarizes the relative performance of RBA against Cubist in terms of the number of wins, losses, and draws. We use two different criteria to decide which method is better: 0.01 and 0.1 difference, which are based on the magnitude of difference in the correlation coefficient between the two competing techniques.

criteria	1-RBA	prec-RBA	prob-RBA	avg-RBA
0.01 difference	**10-6-5**	**16-2-3**	**15-2-4**	**15-2-4**
0.1 difference	**3-2-15**	**6-0-15**	**7-0-14**	**7-0-14**

Table 5: Win-Loss-Draws for the 21 real-life data sets RBA versus Cubist

The results shown in Table 5 suggest that all RBA schemes outperform Cubist for the majority of the data sets. Even when winners are declared with a correlation difference more than 0.10, single-rule scheme wins 3 to 2, and multi-rule schemes are better with a win-loss situation of 6 and 7 to 0.

5.2.3 Comparison between RBA and CART

Comparing the RBA methods against CART (Table 6), using the 0.01 difference criteria shows that k-RBA outperforms CART for more than half of the data sets. In contrast, CART wins only on 5 to 6 different data sets. If the criteria for winning is based on correlation difference greater than 0.1, then CART outperforms only the precision-based k-RBA for one of the 21 data sets, whereas there are at least 3 or 4 data sets where multi-rule RBA predictions correlate better to the target values with 0.10 difference.

criteria	1-RBA	prec-RBA	prob-RBA	avg-RBA
0.01 difference	7-9-4	**9-5-6**	**11-6-3**	**11-6-3**
0.1 difference	1-3-16	**3-1-16**	**4-0-16**	**4-0-16**

Table 6: Win-Loss-Draws for the 21 real-life data sets RBA versus CART

We investigated further the situations where CART does better. `californiahouses` is the data set, where CART performs the best compared to other data sets, with a correlation difference of 0.06. The average number of rules-leaf nodes generated by CART for this data set is 2721. In contrast, the average number of rules for k-RBA is only 13 (selected among the 41 non-redundant and significant rules generated by the rule generation module of RBA). The tradeoff between precision and complexity models suggests another desirable property of RBA methods: they tend to produce simpler models at a reasonable precision tradeoff compared to CART. In fact, the average model size ratio between CART and 1-RBA is 13.39, and 11.36 for CART and k-RBA.

Pruning redundant and significant rules is a very important part of our algorithm, but for this particular case, as part of our investigation, we turned the pruning off, to increase the number of rules available for model building. If we do so for the `californiahouses` data set, RBA will select, on average, 300 rules, increasing the correlation coefficient from 0.67 to 0.73 (for 1-RBA) and from 0.70 to 0.76 (for k-RBA), while still maintaining a much smaller rule set compared to the models produced by CART.

Another interesting observation is that the size of the model produced by CART is found to be highly correlated to the data set size, with a correlation coefficient of 0.93. On the other hand, the correlation between the size of data set and model size is -0.04 for 1-RBA and -0.14 for k-RBA. Indeed, increasing training data size does not always mean introducing new patterns, in a very large data set one may find only few patterns explaining the target, similarly one may mine many patterns from a small data set. This observation supports the previous assertion that RBA tends to produce a compact set of high quality predictive rules from the data, independent of the data size, which is a very useful property especially for noisy data sets.

5.2.4 Discussion on 1-RBA and k-RBA Our experimental results indicate that k-RBA schemes tend to be more effective than 1-RBA for most of the data sets. One possible explanation is that a multi-rule scheme is more tolerant towards errors committed by individual rules, and on average, tends to make better prediction by soliciting the opinions of multiple experts/rules.

The performance of both 1-RBA and k-RBA can be affected by the amount of rule pruning in the rule generation step. On the one hand, too much pruning may reduce the number of rules available for model building, while on the other, pruning may help to eliminate spurious rules. On average, the redundancy and significance tests applied during rule generation result in a 2% gain in correlation produced by the final model. To be more precise, the redundancy test was found to prune 72% of the rules satisfying minimum support while the significance test applied after the redundancy test prunes between 1% to 3% of the remaining rules. In terms of model size, 1-RBA's models are 4% to 8% smaller when pruning is applied (as opposed to no pruning) while the k-RBA models are 35% to 38% smaller. Thus, k-RBA appears to benefit more from rule pruning compared to 1-RBA. Rule pruning not only achieves better correlation, it leads to lower complexity models, and shorter run times for model building, which by itself is a significant gain.

The choice of minimum support threshold have an impact on the performance of the model as it affects the number of rules generated. For large data sets, a minimum support threshold of 0.01 may produce inferior models compared to using a much lower threshold.

5.3 Discussion Given the wide variety of real data sets, it is unreasonable to expect a single regression technique to work well on all data sets. This is partly due to the consensus that no search algorithm works the best for all problems. Thus, it is crucial to understand the characteristics of a data set for which a regression method works better than other competing methods. We have identified one such characteristics in Section 5.1, where RBA-based methods are more resilient to noise compared to CART.

To understand these methods better, we first derived several metrics to characterize the data sets such as skewness, sparseness, size, dimensionality, nearest neighbor error, etc. Next, these metrics are correlated to the difference of RBA and CART's correlation to the output. This will allow us to identify the data characteristics for which RBA performs better or worse. The correlation of data size to RBA-CART performance is -0.36, meaning RBA is more successful compared to CART when the size of the data is small. RBA also does better when the feature space is sparser (the correlation to the ratio of data size to the number of attributes is -0.36), and when the number of attributes is larger (0.45).

Another interesting observation is the correlation between the average prediction error made by 1-nearest neighbor (0.58) and variance of prediction error made by 1-nearest neighbor (0.73) to the relative performance of RBA over CART. These metrics can be used to approximate the amount of noise in the data: if the prediction made by 1-nearest neighbor is unreliable, the data tends to be more noisy. Such an unusually high correlation suggests that RBA indeed does better than CART on noisy data sets, a result that is consistent with the findings in Section 5.1.

There is also positive correlation (0.45) between the skewness of the target variable distribution (normalized difference of the mean and the median) and the success of RBA. We also identified the skewed data sets by visual inspection of their histograms. Out of the nine highly skewed data sets, RBA is doing better in 5 of them, CART on 2, and Cubist on 1, and one was a draw between RBA and CART.

Finally, RBA schemes were found to be very successful when the competitor methods were applied to nominal-valued input variables. However, when the competitors were provided with the original data sets, the RBA schemes become comparable to the competitor methods. The average performance of CART and Cubist, when applied to the original attributes, are 0.69 and 0.71, respectively. In terms of win-loss-draw, using the 0.1 correlation difference criteria, the results for k-RBA versus Cubist is 3-4-14 and k-RBA versus CART is 3-2-15. This suggests that the straightforward MDL-based discretization method does lose quite significant information. We are currently investigating better alternative discretization techniques for regression that will reduce the amount of information loss.

6 Conclusion

Rule based approaches have the benefit of being intuitive, and easily interpretable. In this paper, we proposed a novel rule-based approach for predicting continuous target attribute using association rule mining: Regression Based on Association (RBA). Unlike greedy approaches, association rules analysis explores all possible patterns satisfying the user specified minimum support. Hence, it does not risk missing key patterns that the existing rule-based methods may easily miss, at the expense of runtime in the case of dense data sets. We proposed two pruning strategies for eliminating rules that are redundant and statistically insignificant.

The proposed pruning schemes are found to be very effective at no cost in terms of prediction performance. Four single and multi-rule variations of the proposed method are implemented and compared against two leading rule-based schemes (Cubist, and CART), and found to be superior on majority of real-life data sets. We also provided simple, and illustrative example cases where our approach can successfully find the underlying patterns and predict accurately, while the regression trees fail to do so.

For future work we are considering investigation of different discretization schemes in order to make RBA highly competitive on data sets with continuous input attributes. Dynamic minimum support or anti-monotonic measures are potential directions for future research, which may contribute significantly to the quality of the predictions.

References

[1] Matlab. http://www.mathworks.com.

[2] Statlib. http://lib.stat.cmu.edu.

[3] Uci mlr. www.ics.uci.edu/mlearn/MLRepository.html.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499, 1994.

[5] Y. Aumann and Y. Lindell. A statistical theory for quantitative association rules. In *Proc of Knowledge Discovery and Data Mining*, pages 261–270, 1999.

[6] L. Breiman, J. H. Friedman, R. Olsheb, and C. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[7] W. W. Cohen. Fast effective rule induction. In *Proc. of the 12th Int'l Conf on Machine Learning*, 1995.

[8] M. H. DeGroot and M. J. Schervish. *Probability and Statistics*. Addison-Wesley, 3rd edition, 2002.

[9] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proc of ICML*, 1995.

[10] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-values attributes for classification learning. *IJCAI*, 2:1022–1027, 1993.

[11] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms and visualization. *Proc of ACM SIGMOD*, 1996.

[12] E.-H. Han, G. Karypis, V. Kumar, and B. Mobasher. Clustering based on association rule hypergraphs. In *DMKD'97*, 1997.

[13] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc of ACM SIGMOD*, 2000.

[14] W. Lee, S. J. Stolfo, and K. W. Mok. A data mining framework for building intrusion detection models. In *IEEE Symp on Security and Privacy*, 1999.

[15] B. Lent, A. N. Swami, and J. Widom. Clustering association rules. In *ICDE*, pages 220–231, 1997.

[16] W. Li, J. Han, and J. Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. *ICDM*, pages 369–376, 2001.

[17] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proc of Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[18] B. Liu, Y. Ma, and C.-K. Wong. Classification using association rules: Weaknesses and enhancements, 2001.

[19] S. Morishita and J. Sese. Traversing itemset lattice with statistical metric pruning. In *Symposium on Principles of Database Systems*, pages 226–236, 2000.

[20] R. Quinlan. Cubist. www.rulequest.com.

[21] R. Quinlan. *C4.5: Programs for Machine Learning*. 1994.

[22] R. Rastogi and K. Shim. Mining optimized association rules with categorical and numeric attributes. *Knowledge and Data Engineering*, 14(1):29–50, 2002.

[23] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Proc of ACM SIGMOD*, 1996.

[24] K. Wang, C. Xu, and B. Liu. Clustering transactions using large items. In *CIKM*, pages 483–490, 1999.

[25] C. Yang, U. M. Fayyad, and P. S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. pages 194–203, 2001.

