Improved Classification Association Rule Mining

M.Naresh Kumar
M.Tech (Software Engineering)
J.N.T.University
Anantapur
Andhra Pradesh, India

B.Eswara Reddy
Associate Professor
J.N.T.University
Anantapur
Andhra Pradesh, India
eswarcsejntu@gmail.com

Abstract— **Classification aims to define an abstract model of a set of classes, called classifier, which is built from a set of labeled data, the training set. However, in large or correlated data sets, association rule mining may yield huge rule sets. Hence several pruning techniques have been proposed to select a small subset of high-quality rules. Since the availability of a "rich" rule set may improve the accuracy of the classifier, we argue that rule pruning should be reduced to a minimum. A small subset of high-quality rules is first considered. When this set is not able to classify the data, a larger rule set is exploited. This second set includes rules usually discarded by previous approaches. To cope with the need of mining large rule sets and to efficiently use them for classification, a compact form is proposed to represent a complete rule set in a space-efficient way and without information loss. An extensive experimental evaluation on real and synthetic data sets shows that improves the classification accuracy with respect to previous approache**s.

Keywords- Data mining, associative classification, association rules, condensed representations

I. INTRODUCTION

Classification aims to define an abstract model of a set of limiting rule pruning has already been discussed in classes, called classifier, which is built from a set of labeled data, the training set. The classifier is then used to appropriately classify new data for which the class label is unknown. Different approaches have been proposed to build accurate classifiers, for example, naive Bayes classification [1], decision trees [2], and SVMs [3]. Recently, association rules [2] have become a valuable tool for classification purposes. (for example, CAEP [2], CMAR [1], CBA [2], and ADT [3]).

In associative classification, the rule consequent is a class label, and the classifier is a set of association rules. Since association rules represent the correlation among values of different attributes simultaneously, in general, associative classifiers yield better accuracy than decision trees and rule-based classifiers. The generation of an associative classifier consists of two steps. First, classification rules are extracted from the training data. Then, pruning techniques are applied to select a small subset of high-quality rules and build an accurate model of training data. Usually, a large rule set is mined to allow a wide selection of rules and the generation of accurate classifiers. However, in large or correlated data sets, rule mining may yield a huge number of classification rules. Rule extraction becomes difficult (or at least time consuming), and it becomes hard to optimally exploit the generated rules. Hence, pruning techniques, in particular, support-based pruning, are exploited to reduce the complexity of the extraction task. Most pruning techniques may go too far, by discarding also useful knowledge together with low-quality rules. Since the availability of a large rule model may improve the accuracy of the classifier pruning should be limited to a minimum. The opportunity of limiting rule pruning has already been discussed in [3] and [1]. Liu et al. [1] proposed multiple support thresh-olds to limit the number of extracted rules for frequent classes, without hiding infrequent classes. CMAR [3] proposed a modified database coverage technique, which is the first step toward the reduction of excessive pruning. To address both an excessive rule set size and over-pruning, we propose a new associative classifier that relies on a lazy pruning approach coupled with a compact representation of the rule set We named our classifier L^3, which stands for Live and Let Live[1] (that is, pruning only takes place when strictly necessary). The lazy pruning technique performs a reduced amount of pruning by eliminating only "harmful rules," that is, rules that only misclassify training data.

During classification, L^3 adopts a two-step approach in which high-quality rules (that is, rules used in the classification of training data) are considered first, and "unchecked" rules, that is, rules unused during the training phase, are used next to classify unlabeled data. Rules of the secondtype are only considered when unlabeled datcannot be classified by means of the first type of classification is a well-studied problem (see [2,3] for excellent overviews) and several models have been proposed over the years, which include neural networks [1], statistical models like linear/quadratic discriminates [1], decision trees [2], and genetic algorithms [1]. Among these models, decision trees are particularly suited for data mining. Decision trees can be constructed relatively fast compared to other methods. Another advantage is that decision tree models are simple and easy to understand [1]. And yields improved accuracy over decision trees [2]. As an alternative to decision trees, associative classifiers have been proposed. These methods first mine association rules from the training data, and then build a classifier using these rules. This classifier produces good results and yields improved accuracy over decision trees. Decision trees perform a greedy search for rules by heuristically selecting the most promising features. They start with an empty concept description, and gradually add restrictions to it until there is not enough evidence to continue, or perfect discrimination is achieved. Such greedy (local) search may prune important rules. Associative classifiers, on the other hand, perform a global search for rules satisfying some quality constraints.

This global search, however, may generate a large number of rules, and many of the generated rules may be useless during classification (i.e., they are not used to classify any test instance) In this paper we propose a novel lazy associative classifier, in which the computation is performed on a demand driven basis. We place our associative classifier within an information gain framework that allows us to compare it to decision tree classifiers. Our method can overcome the large rule-set problem of traditional (eager) associative classifiers, by focusing on the features that actually occur within the test instance while generating the rules. We show that the proposed lazy classifier outperforms its eager counter- part, since in the lazy approach only the "useful" portion of the training data is mined for generating the rules applicable to the test instance. Due to this local focus, the lazy classifier can better classify a test instance, for which a global, eager rule-set may not work that well. Simple caching mechanisms are used to avoid work replication during lazy associative classification. First we demonstrate that associative classifiers perform no worse than decision tree classifiers. Then we show that lazy classifiers outperform the corresponding eager classifiers. Our claims are empirically confirmed by an extensive set of experimental results. Timings are also showed in order to evaluate different classifiers with respect to computational complexity of the training data set.

II. Generating Classification Association Rules using TFPC

selected correctly, then the existence of a rule $X \rightarrow c1$ should make it unnecessary to consider any other rules whose antecedent is a superset of X. In practice, however, we may still find a rule $Y \rightarrow c2$, say, where Y is a superset of X, which has higher confidence and to which we would wish to give higher precedence. It remains possible, also, that there will be a further rule $Z \rightarrow c1$, where Z is a superset of Y , with still higher confidence, and so on. This reasoning leads other methods to a process in which all possible rules are first generated and then evaluated.

In this paper we adopt an alternative heuristic: If we can identify a rule $X \rightarrow c$ which meets the required support and confidence thresholds, then it is not necessary to look for other rules whose antecedent is a superset of X and whose consequent is c. It will still of course be necessary to continue to look for rules that select other classes. This heuristic both reduces the number of candidate rules to be considered, and the risk of over fitting. We use a method derived from our TFP (Total From Partial) algorithm to generate a set of CARS. This method, described in [3], first builds a set-enumeration tree structure, the *P-tree* that contains an incomplete summation of support-counts for relevant sets. Using the P-tree, the algorithm uses an Apriori- like procedure to build a second set enumeration tree, the *T-tree*, that finally contains all the frequent sets (i.e. those that meet the required threshold of support), with their support-counts.

The T-tree is built level by level, the first level comprising all the single items (attribute-values) under consideration. In the first pass, the support of these items is counted, and any that fail to meet the required support threshold are removed from the tree. Candidate-pairs are then generated from remaining items, and appended as child nodes. Most work on lazy classification [1] was based on nearest neighbor algorithms [3].

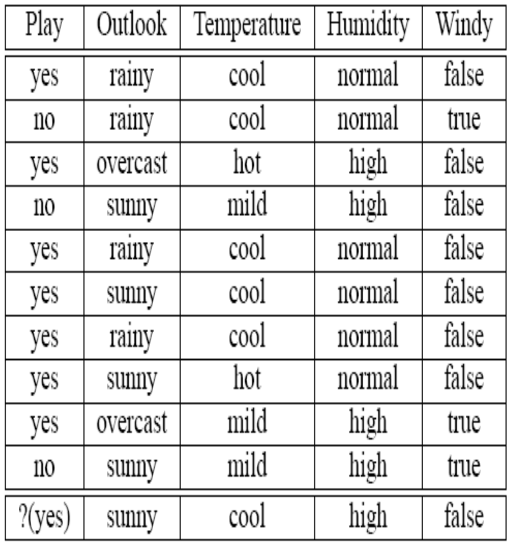
The problem of small disjuncts was first noted in [3], where it was showed that existing classifiers create models that are good for large disjuncts but are far from ideal for small disjuncts (which correctly classifies only few training instances). To investigate the performance of TFPC, we carried out experiments using a number of data sets taken from the UCI Machine Learning Repository. It is hard to assess the accuracy of small disjuncts because they cover few instances, yet removing all of them is unjustified since many of them may be significant and the overall accuracy would be degrade.

III EXPERIMENTAL RESULTS

To investigate the performance of TFPC, we carried out experiments using a number of data sets taken from the UCI Machine Learning Repository. The implementation of TFPC was as a Java program, and for comparison purposes we have used our own (Java) implementations of the published algorithms for

CMAR [3] and CPAR [1]. In the first set of experiments, as in [3] and [2], we have assumed a support threshold of 1% and a confidence threshold of 50%.For the implementation of CPAR, we used the same parameters used in [10],i.e. minimum gain threshold = 0.7, total weight threshold = 0.05, decay factor = 2/3, and similarity ratio 1 : 0.99. We tried to use the same data sets as those used to analyze CMAR and CPAR. However in many cases the data sets that were used in [2] and appear to be no longer available, and in others were found not have identical parameters to those reported. The data sets chosen therefore comprise a subset of those used in [3] and [1], augmented by a further selection from the UCI repository. The choice of additional data sets concentrated on larger/denser data sets (2000+ records) because the majority of the data sets used in the reported analysis of CMAR and CPAR were relatively small (less than 1000 records). Most work on lazy classification [1] was based on nearest neighbor algorithms [3]. The problem of *small disjuncts* was first noted in [1], where it was showed that existing classifiers create models that are good for large disjuncts butare far from ideal for small disjuncts (which correctly classifies only few training instances). It is hard to assess the accuracy of small disjuncts because they cover few instances, yet removing all of them unjustified since many of them.

A lazy decision tree was proposed in [1] and it was shown that the lazy approach is superior than the corresponding eager one (i.e., C4.5). Despite all the improvements obtained by using lazy algorithms, we are not aware of any proposals of lazy associative classification algorithms, as well as an assessment that demonstrates why they perform better than both decision and eager associative classifier. Due to the local focus, the lazy classifier can better classify a test instance, for which a global, eager rule-set may not work that well simple catching mechanism are used to avoid work replication during lazy associative classification. First we demonstrate that associative classifiers to perform to worse than the decision tree classifier. Then we show that lazy classifier outperform the corresponding eager classifier. Our claims are empirically confirmed by an extensive Set of experimental results. Timing are also showed in order to evaluate different classifiers with respect to computational complexity.



Play	Outlook	Temperature	Humidity	Windy
yes	rainy	cool	normal	false
no	rainy	cool	normal	true
yes	overcast	hot	high	false
no	sunny	mild	high	false
yes	rainy	cool	normal	false
yes	sunny	cool	normal	false
yes	rainy	cool	normal	false
yes	sunny	hot	normal	false
yes	overcast	mild	high	true
no	sunny	mild	high	true
?(yes)	sunny	cool	high	false

Figure1.Test instance data set

IV EAGER ASSOCIATIVE CLASSIFIER

In this section we describe eager associative classifiers, and demonstrate why they perform better than decision trees. We start by discussing how decision rules may be generated from decision trees. Then we describe associative classifiers that are based on information gain, so that we may compare them regarding the rules that are generated by each approach.

V. DECISION TREE AND DECISION RULES

Given any subset of training instances S, let Si denote the number of instances with class ci, and let |S| = Pi si be the total number of training instances. Then pi = si |S| denotes the probability of class ci in S. The entropy of S is then given as E(S) = Pi pi log pi. For any partition of S into m subsets Si, with S = [mi=1Si, the resulting split entropy is given as E({Si}) = Pm i=1 |Si||S| E(Si). The information gain for the split is then given as I(S, {Si}) =E(S) − E({Si}).A decision tree is built using a greedy, recursive splitting strategy, where the best split is chosen at each internal node according to the information gain criterion. The splitting at a node stops when all instances are from a single class or if the size of the node falls below a minimum support threshold, called minsup. Figure 1 shows an example of training data, and Figure 2 shows the corresponding decision tree

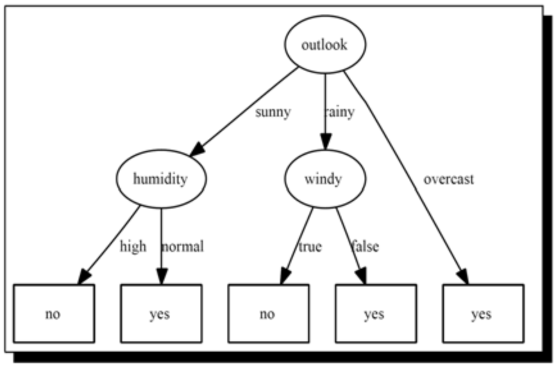


Figure 2. Decision tree

The last row of the table in Figure 1 shows one test instance which is recognized by the decision tree in Figure 2. The decision tree can be considered as a set of disjoint decision rules, with one rule per leaf. In that way, a decision tree can be simulated by a set of decision rules. In this case, the information gain for each decision rule is calculated in the same way as it is calculated for each path of the decision tree. Thus, a decision rule has the same value of information gain of its corresponding path in the decision tree.

VI. ENTROPY-BASED ASSOCIATIVE CLASSIFIER

We denote as Class association rule *(CARs)* those association rules of the form X ! c, where the antecedent (X) is composed of feature variables and the consequent (c) is just a class. CARs may be generated by a slightly modified association rule mining algorithm. Each item set must contain a class and the rule generation also follows a template in which the consequent is just a class. CARs are essentially decision rules, and as in the case of decision trees, CARs are ranked in decreasing order of information gain. Finally, during the testing phase, the associative classifier simply checks whether each CAR matches the test instance; the class associated with the first match is chosen. Note that, seen in the light of CARs, a decision tree is simply a greedy search for CARs, using a level-wise search algorithm that only expands the current best rule with other features. On the other hand, an eager associative classifier mines *all* possible CARs with a given minsup. It is also interesting to note that sorting the final rule-set on information gain, and using the best CAR for classification, is also a greedy strategy. While the greedy approach has its limitations, eager associative classifiers are not limited by the *prefix problem* of decision rules, that is, once the best feature is chosen a teach node, all nodes under that sub tree must contain it.

```
let $\mathcal{D}$ be the set of all $n$ training instances
let $\mathcal{T}$ be the set of all $m$ test instances
  1. let $\mathcal{C}^e$ be the set of all rules $\{\mathcal{X} \rightarrow c\}$ mined from $\mathcal{D}$
  2. sort $\mathcal{C}^e$ according to information gain
  3. for each $t_i \in \mathcal{T}$ do
  4.     pick the first rule $\{\mathcal{X} \rightarrow c\} \in \mathcal{C}^e | \mathcal{X} \subseteq t_i$
  5.     predict class $c$
```

Figure 3. Eager Associative Classifier

.

Figure 3 shows the basic steps of the eager associative classifier. In the initial step, the algorithm mines all frequent CARs, and sorts them in descending order of information gain. Then, for each test instance ti, the first CAR matching ti is used to predict the class. Figure 4 shows an associative classifier built from our example set of Training instances in Figure 1, using the algorithm showed in Figure 3. Three CARs match the test instance of our example (last row of Table 1):

1. {windy=false and temperature=cool play=yes}

2. {outlook=sunny and humidity=high play=no}

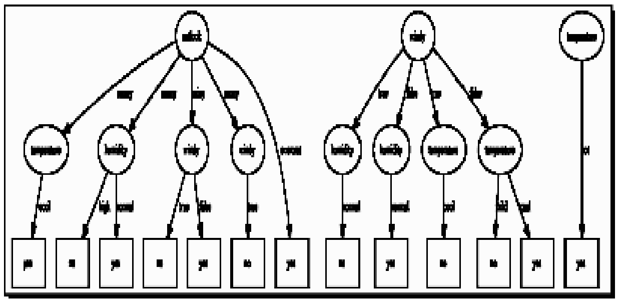
3. {outlook=sunny and temperature=cool play=yes}

Rule {windy=false and temperature=cool play=yes}

Would be selected, since it is the best ranked CAR. By applying this CAR, the test instance will be correctly classified. Intuitively, associative classifiers perform better than decision trees because associative classifiers allow several CARs to cover the same partition of the training data. In our example, the test case is recognized by only one rule in the decision tree, while the same test case is recognized by three CARs in the associative classifier. Selecting the proper CAR to be applied is an issue in associative classification. Next we present a theoretical discussion about the performance of decision trees and eager associative classifiers.

Theorem 1 *The rules derived from a decision tree are a subset of the CARs mined using an eager associative classifier based on information gain.*

Proof 1 Let max E be the maximum entropy of all decision tree rules. Select a set Ce from all CARs that their entropy is at most max E. It is clear that the decision tree rules are a subset of Ce



Theorem 1 states that, for a given minsup, CARs contain (at least) all information of the corresponding decision tree. Since each decision tree rule may be seem as a CAR, and since all possible CARs were enumerated, then the decision tree can be built by choosing the proper CARs

Theorem 2 *CARs perform no worse than decision tree rules, according to the information gain principle.*

Proof 2 *Given an instance to be classified, and, without loss of generality, a decision tree with just pure leaves, the decision tree predicts class* c *for that instance. We analyze two scenarios: first, just one CAR matches the instance, and second, more than one CAR matches. When just one CAR matches, it is the same as the decision tree rule, since the set of CARs subsumes the set of decision rules. In this case, the associative classifier and the decision tree make the same prediction. When more than one CAR matches an instance, the prediction may be either the same class (say* c*) as the matching decision rule or another class. If the associative classifier predicts* c *then the two approaches are equivalent .In case a class other than* c *is predicted, by definition, the best matching CAR provides a better information gain than the decision rule, and thus, according to the information gain principle, the CAR will make a better prediction*

Theorem 2 states that the additional CARs of the associative classifier that are not in the decision tree, cannot degrade CAR is only used if it is better than all decision rules (according to the information gain principle).However, eager associative classifiers generate a large number of CARs, most of which are useless during classification. For instance, from the set of 13 CARs showed in Figure 4, only 3 match the test instance (the remaining 10 CARs are useless). Next, we present a lazy classifier and compare it to the eager version described in this section.

VII LAZY ASSOCIATIVE CLASSIFIER

Unlike the eager associative classifier that extracts a set of ranked CARs from the training data, the lazy associative classifier induces CARs specific to each test instance. The lazy approach *projects* the training data, D, only on those features in the test instance, A. From this projected training data, DA, the CARs are induced and ranked, and the best CAR is used. From the set of all training instances, D, only the instances sharing at least one feature with the test instance A are used to form DA. Then, a rule-set ClAis generated from DA. Since DA contains only features inA, all CARs generated from DA must match A. The lazy associative classifier is presented in Figure 5.

```
let $\mathcal{D}$ be the set of all $n$ training instances
let $\mathcal{T}$ be the set of all $m$ test instances
  1. let $\mathcal{C}^e$ be the set of all rules $\{\mathcal{X} \rightarrow c\}$ mined from $\mathcal{D}$
  2. sort $\mathcal{C}^e$ according to information gain
  3. for each $t_i \in \mathcal{T}$ do
  4.    pick the first rule $\{\mathcal{X} \rightarrow c\} \in \mathcal{C}^e | \mathcal{X} \subseteq t_i$
  5.    predict class $c$
```

Figure 5. Lazy Associative Classifier

Now we demonstrate that the lazy associative classifier produces better results than its eager counterpart. Given a test instance A, and a set of CARs C, we denote by CA those CARs {X →c} in C where X ⊆ A.

Theorem 3 *Let* A *be the set of features in a given test instance.Let* Ce A *be the set of CARs obtained from the eager associative classifier induced by* A, *and* Cl A *be the set of CARs obtained from the lazy associative classifier induced by* A. *For a given* minsup, *we have* $C^e_A \subsetneq C^l_A$

Proof 3 *By definition, both* CeA *and* ClA *are composed of CARs* {X →c} *in which* X →A, *that is, all CARs contain only features in* A. *Also the training instances matching* A *(i.e., the projected training data) are a subset of the set of*

Play	Outlook	Temperature	Humidity	Windy
no	–	–	–	true
yes	overcast	hot	–	–
yes	–	hot	–	–
yes	overcast	–	–	true
no	–	–	–	true
?(yes)	overcast	hot	low	true

All training instances (i.e., DA ⊆D*). Thus, for a given* minsup, *if a rule* {X →c} *is frequent in* D, *and then it must also be frequent in* DA. *Since* C^l_A *is generated from* DA *and* C^e_A *is generated from* D *(and* DA ⊊ D*),* $C^e_A \subsetneq C^l_A$.

The next example illustrates Theorem 3. Figure 6 shows the training data given in Figure 1 (i.e., D), projected by the features in the test instance (i.e., A) showed in the last row in

Figure 6. The projected training data (i.e., DA) is composed of the 5 instances showed in the figure. Suppose minsup is set to 40%. In this case, the set of CARs, Ce,
found by the eager classifier is composed of the two CARs:

1. {windy=false and humidity=normal!play=yes}
2. {windy=false and temperature=cool!play=yes}

None of the two CARs matches the test instance, and thus, C^{e}_{A} =Ǿ;. On the other hand, the projected training data has less instances (DA ⊆ D), and therefore, CARs not frequent in D may be frequent in DA. This is because a frequent CAR must occur at least 4 times in D (since|D|=10), but only 2 times in DA (since |DA|=5). The lazy classifier found two CARs in DA:
1. {outlook=overcast! play=yes}
2. {temperature=hot! Play=yes}
These lazy CARs not only predict the correct class, but also are simpler than the eager CARs. Next we discuss how the lazy CARs perform when compared to the eager CARs
Theorem 4 *Lazy CARs perform no worse than eager CARs, according to the information gain principle.*
Proof 4 *Theorem 3 showed that, for a given* minsup C^{e}_{A} ⊆ C^{l}_{A}. *Let* Re *be the best rule in* CeA *(according to the information gain principle), and let* Rl *be the best rule in* ClA. *Two*
scenarios have to be considered when determining a class for the test instance A. *In the first scenario,* Rl *is identical to* Re *; in this case the same class is predicted by both eager and lazy classifiers. In the second scenario,* Rl *is better than* Re, *and thus* Rl *must provide a better prediction.* _

Theorem 4 states that the CARs added by the lazy classifier do not degrade the classification accuracy. This is because an additional lazy CAR is only used if it is better than all eager CARs (according to the information gain principle).
Intuitively, lazy classifiers perform better than eager classifiers because of two characteristics:

• Missing CARs: Eager classifiers search for CARs in a large search space, which is induced by all features of the training data. While this strategy generates a large rule-set, CARs that are important to some specific test instances may be missed (this is particularly true for skewed/unbalanced distributions). Lazy classifiers, on the other hand, are context-sensitive and focus the search for CARs in a much smaller search space, which is induced by the features of the test instance.
• Highly Disjunctive Spaces: Eager classifiers generate CARs before the test instance is even known. In this case, the difficulty for the classifier is in anticipating all the different directions in which it should attempt to generalize its training examples (i.e., what CARs must be generated). For this reason, eager classifiers often combine small disjuncts in order to generate more general
Predictions (more general CARs should be applicable to more test instances). This can reduce classification performance in highly disjunctive spaces, where single disjuncts may be important to classify specific instances. Lazy classifiers, on the other hand, generalize their training examples exactly as needed to cover the test instance. Thus, lazy classifiers are often most appropriate when the search space is complex, and there are myriad ways to generalize a case.

VIII Experimental Evaluation

In this section we show the experimental results for the evaluation of the proposed classifiers in terms of classification effectiveness and computational performance. Our evaluation is based on a comparison against C4.5 [1] and Lazy DT [1] decision tree classifiers. We also compare our Numbers to some results from other associative classifiers, such as CPAR [2], CMAR [1] and HARMONY [1], and to some results from rule induction classifiers, such as RISE [2], RIPPER [3], and SLIPPER [3]. We used 26 datasets from the UCI Machine Learning Repository to compare

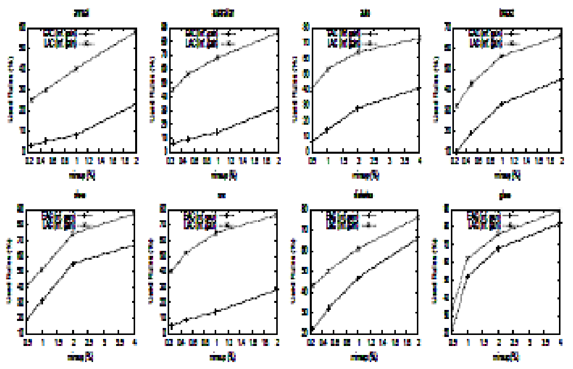
IX Eager and Lazy Associative Classifiers

We continue our analysis by comparing the effectiveness of eager (EAC) and lazy classifiers (LAC). Table 1 shows their corresponding error rates. For very small datasets eager and lazy classifiers perform similarly, since the CARs that were generated by both classifiers were essentially the same for the parameters used. For instance, the result obtained with labor and zoo datasets were exactly the same. Also, we can observe that lazy classifiers perform better when the dataset is sparse (i.e., auto, pima, diabetes, german, and wine datasets). The error reduction in these datasets range from 13.9% to 52.7%. This result is expected, since the small disjuncts problem is more likely to happen in sparse datasets. Further, we can also notice that
the lazy classifiers always outperform the corresponding eager ones, except for the ionosphere dataset. We believe that, for this dataset, the lazy classifiers have over fitted the data.

9.1 Rule Induction and Associative Classifiers

We also compared the proposed eager and lazy associative classifiers against RISE, RIPPER and SLIPPER, using results reported in [3,2,1]. Table 2 shows the relative performance for each classifier (i.e., the accuracy of one classifier divided by the accuracy of the other classifier), when compared to eager associative classifiers. Each number in this table indicates how many times EAC is superior than the corresponding adversary RISE, RIPPER or SLIPPER (in terms of accuracy). The SLIPPER algorithm won in 6 of the 26 datasets (and lost in 1), showing to be most competitive rule induction classifier. The RIPPER classifier won in 4 datasets and the RISE classifier won in only one dataset.
Table 2 also shows the relative performance of rule induction classifiers when compared to the lazy associative classifiers. RISE and RIPPER lost in almost all datasets, and SLIPPER was the most competitive one. Compared toLAC (inf. gain), SLIPPER won in one dataset (and matched

Dataset	Decision Tree Classifiers		Eager Associative Classifiers				Lazy Associative Classifiers		Error Reduction over C4.5(%)	
	C4.5 inf. gain	LazyDT inf. gain	EAC inf. gain	CBA conf	CMAR conf	HARMONY conf	LAC inf.gain	LAC conf	EAC inf. gain	LAC inf. gain
anneal	6.5	4.2	3.9	3.6	2.7	8.5	3.5	3.6	40.0	46.1
australian	13.5	15.2	13.0	13.4	13.9	-	12.7	13.4	3.7	6.2
auto	29.2	24.7	26.5	27.2	21.9	39.0	22.2	22.9	9.2	24.0
breast	3.9	5.1	3.6	4.2	3.6	7.6	3.2	4.1	7.7	17.9
cleve	18.2	17.2	16.1	16.7	17.8	-	15.6	16.7	11.5	14.3
crx	15.9	16.9	15.0	14.1	15.1	-	14.2	14.1	5.7	10.7
diabetes	27.6	24.9	24.6	25.3	24.2	-	19.7	21.3	10.9	28.6
german	29.5	26.1	27.2	26.5	25.1	-	23.4	22.0	7.8	20.7
glass	27.5	26.5	26.8	27.4	29.9	50.2	26.0	27.4	2.5	5.4
heart	18.9	17.7	18.1	18.5	17.8	43.5	16.9	17.2	4.2	10.6
hepatitis	22.6	20.3	17.9	15.1	19.5	22.0	17.1	15.1	20.8	24.3
horse	16.3	17.2	15.4	18.7	17.4	17.5	14.5	17.2	5.5	11.0
hypo	1.2	1.2	1.2	1.7	1.6	-	1.0	1.2	0.0	16.7
ionosphere	8.0	8.0	7.6	8.2	8.5	8.0	7.8	8.5	5.0	2.5
iris	5.3	5.3	4.9	7.1	6.0	6.7	3.2	4.6	7.5	39.6
labor	21.0	20.4	19.1	17.0	10.3	-	19.1	17.0	9.0	9.0
led7	26.5	26.5	24.2	27.8	27.5	25.4	22.1	25.5	8.7	16.6
lymph	21.0	20.1	20.2	19.6	16.9	-	19.1	18.2	3.8	9.0
pima	27.5	25.9	27.5	27.6	24.9	27.6	22.0	21.3	0.0	20.0
sick	2.1	2.1	2.1	2.7	2.5	-	2.0	2.0	0.0	4.8
sonar	27.8	24.6	22.9	21.7	20.6	-	20.5	19.6	17.6	26.3
tic-tac-toe	0.6	0.6	0.6	0.0	0.8	7.7	0.6	0.0	0.0	0.0
vehicle	33.6	31.8	30.8	31.3	31.2	-	28.9	30.0	8.3	14.0
waveform	24.6	22.7	21.3	20.6	16.8	19.5	19.3	18.9	13.4	21.5
wine	7.9	7.9	7.2	8.4	5.0	8.1	3.4	3.9	8.9	57.0
zoo	7.8	7.8	6.6	5.4	2.9	7.0	6.5	5.4	15.4	16.7
Average	17.1	16.2	15.5	15.8	14.8	19.9	14.0	14.3	8.7	18.2



X Conclusions and Future Work

Decision tree classifiers perform a greedy search which may discard relevant information. Based on this observation we present an assessment of associative classification. The generated CARs are ranked based on their information gain, so that we can compare the performance of decision trees and associative classifiers. We present evidence regarding the superiority of associative classifiers. However, it is well known that no classifier can outperform others in all settings [2], and thus there may be certain specific situations Where decision trees outperform associative classifiers.We also propose improvements to associative classification by introducing a novel lazy classifier. The lazy classifier searches a larger hypothesis space than the corresponding eager classifier, because it uses many different local models to form its implicit global approximation to the target function. Eager classifiers commit at training time to a single global approximation. An important feature of the proposed lazy classifier is its ability to deal with the *small disjuncts* problem. Based on this observation, we present evidence showing that a lazy associative classifier outperforms the corresponding eager one. Our claims were con- firmed by empirical comparisons to C4.5 and LazyDT decision tree classifier, using datasets from the UCI data repository. We also compared the proposed Classifiers against other three associative classifiers and three rule induction Classifiers and outperformed them in most of the cases. So far, our classifiers use only the best CAR for sake of classification. In the future we will combine simple and complex CARs in order to enhance classification. Finally, we will explore more realistic application scenarios.

References

[1] Elena Baralis, Silvia Chiusano, and Paolo Garza,” Lazy approach for classification and Association rule mining, ”IEEE Trans. Knowledge and Data Eng., vol. 20,Feb. 2008

[2] W. Li, J. Han, and J. Pei, “CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules,” Proc. IEEE Int’l Conf. Data Mining (ICDM ’01), Nov. 2001

[3] Jiawei Han ,Micheline Kamber, ”Data Mining concepts and Techniques” Second Edition 2008