A Novel Algorithm for Associative Classification

Gourab Kundu[1], Sirajum Munir[1], Md. Faizul Bari[1], Md. Monirul Islam[1,2,⋆], and Kazuyuki Murase[2,3]

[1] Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh
[2] Department of Human and Artificial Intelligence Systems, Graduate School of Engineering, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
[3] Research and Education Program for Life Science, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
monirul@synapse.his.fukui-u.ac.jp

Abstract. Associative classifiers have been the subject of intense research for the last few years. Experiments have shown that they generally result in higher accuracy than decision tree classifiers. In this paper, we introduce a novel algorithm for associative classification "**C**lassification based on **A**ssociation **R**ules **G**enerated in a **Bi**directional **A**pporach" (CARGBA). It generates rules in two steps. At first, it generates a set of high confidence rules of smaller length with support pruning and then augments this set with some high confidence rules of higher length with support below minimum support. Experiments on 6 datasets show that our approach achieves better accuracy than other state-of-the-art associative classification algorithms.

Keywords: Association rules, Data mining, Knowledge discovery, Classification, rule sorting.

1 Introduction

Building accurate classifiers is one of the essential tasks of data mining and machine learning research. Given a set of training instances with known class labels, classifiers aim to predict the target classes for a set of test instances for which the class labels are not known. At first, a classification model is developed from training data and then it is used to classify unseen instances. There are various methods for building classifiers such as decision trees [1], naïve-Bayesian methods [2], statistical approaches [3], support vector machines [4] etc.

In data mining, association rule mining algorithms are used to discover rules which determine implication or correlation among co-occurring elements within a dataset. Association rule mining algorithms try to answer questions such as "if a customer purchases product A, how likely is he to purchase product B?" or "What products will a customer buy if he buys products C and D?". The actual task is to reduce a potentially huge amount of information into a small, understandable set of statistically supported statements.

⋆ The corresponding author.

M. Ishikawa et al. (Eds.): ICONIP 2007, Part II, LNCS 4985, pp. 453–459, 2008.

Recent works have proposed several techniques to generate high quality *class association rules* from the training data set to build a classifier, with specific thresholds for support and confidence. Such classifiers are CBA (Classification Based on Association) [5], CAEP (Classification based on Aggregating Emerging Patterns) [6] and CMAR (Classification based on Multiple Association Rules) [7]. These approaches have higher accuracy than decision tree classifier due to the fact that decision-tree classifier examines one variable at a time whereas association rules explore highly confident associations among multiple variables at a time. However, these approaches have a severe limitation. All associative classification algorithms use a support threshold to generate association rules. In that way some high quality rules that have higher confidence, but lower support will be missed. Actually the long and specific rules have low support and so they are mostly penalized. But a good classification rule set should contain general as well as specific rules. It should also contain exceptional rules to account for the exceptional instances.

This paper proposes a new algorithm for associative classifier, called CARGBA. It is essentially a bidirectional rule generation approach that generates crisp association rules. It not only tries to generalize the dataset but also tries to provide specific and exceptional rules to account for the specific characteristics and anomalies in the dataset. Although we generate these specific rules, the purpose of this rule generation is not knowledge extraction, rather the only purpose is using these rules for classification to obtain better accuracy. Experiments on 6 datasets show that CARGBA achieves better accuracy than other state-of-the-art associative classifiers.

The rest of the paper is organized as follows. Section 2 describes CARGBA in details. Section 3 presents our experimental results to compare with other state-of-the-art associative classifiers on accuracy. Finally, Section 4 concludes the paper with a brief summary and few remarks.

2 CARGBA

This section describes CARGBA algorithm in details. The algorithm has two main parts. The first part generates rules and it is called **CARGBA Rule Generator**. It generates rules in two steps. First, it generates all rules in Apriori [8] fashion. These rules are as general as possible. They have shorter length and so higher support and provide general knowledge about the training set. This step of our algorithm is similar to other state-of-the-art classification methods. In the second step, we generate rules that are as specific as possible. These rules have higher length and therefore lower support and thus they easily capture the specific characteristics about the data set. That is, if there is a classification pattern that exists over very few instances or there are instances that are exceptions to the general rule, then these instances will be covered by the specific rules. Since these instances are small in number, specific rules are produced without any support pruning. In short, our approach results in a better mixture of class association rules. All the rules generated by CARGBA rule generator

will not be used in the classification. So, the second part builds a classifier with the essential rules and is called **CARGBA Classifier Builder**.

2.1 CARGBA Rule Generator

The key operation of CARGBA Rule Generator algorithm is to find some rules that have *confidence* above or equal to *SatisfactoryConfidence*. Let D be the dataset. Let I be the set of all items in D, and Z be the set of class labels. A rule is of the form: $<$*conditionset*, $z>$ which represents a rule: *conditionset* $\Rightarrow z$ where *conditionset* is a set of items, $z \in Z$ is a class label. The rule has *confidence* equal to (*ruleSupportCount* / *conditionSupportCount*) * 100% where, *conditionSupportCount* is the number of cases in D that contain *conditionset* and *ruleSupportCount* is the number of cases in D that contain *conditionSet* and are labeled with class z. The rule has *support* equal to (*rulesupCount* / $\|D\|$) *100%, where $\|D\|$ is the size of the dataset.

There are two major steps under CARGBA Rule Generator. The steps are summarized as follows:

1. R_1 = { 1-*rules*};
2. Compute *confidence* of each *rule* of R_1;
3. PR_1 ={$r \in R1$ | *confidence* of r >= *SatisfactoryConfidence*
   and *support* of r >= *minSupport*};
4. **for** ($k = 2$; $R_{k-1} \neq \Phi$ **and** k $\leq l$; k++) **do**
5. R_k = Apriori-Gen(R_{k-1});
6. Compute *support*, *confidence* of each *rule* of R_k
7. PR_k ={$r \in R_k$ | *confidence* of r >= *SatisfactoryConfidence*};
8. R_k ={$r \in R_k$ | *support* of r >= *minSupport*};
9. $PR_s = \cup_k PR_k$;

Fig. 1. The first step of CARGBA Rule Generator

Step 1 This step generates all the association rules of the form *1-rules* to *l-rules* that have *confidence* greater than or equal to *SatisfactoryConfidence* under support pruning where *k-rule* denotes a rule whose *conditionset* has k items and l is a parameter of the algorithm. This step is based on Apriori [8] algorithm for finding association rules. The corresponding algorithm is given in fig. 1. At each level of rule generation it prunes away the rules having *support* less than *minSupport*.

R_k denotes the set of *k-rules*. PR_k(Pruned R_k) denotes the set of *k-rules* that have *confidence* greater than or equal to *SatisfactoryConfidence*. *PRs*(Pruned Rules) denotes the set of all rules that have *confidence* greater than or equal to *SatisfactoryConfidence* and *support* greater than or equal to *minSupport*.

```
1. ruleList = Φ;
2. q = Φ;
3. for each record rec ∈ training examples
4.       r = constructRule(rec);
5.       if (confidence of r ≥ satisfactoryConfidence
                 and r ∉ ruleList)
6.                ruleList = ruleList ∪ r;
7.                enqueue (q, r);
8. while (q is not empty)
9.       r = dequeue(q);
10.      for each attribute A ∈ r
11.               r2 = constructRule2(A, r);
12.               if (confidence of r2 ≥ satisfactoryConfidence
                           and r2 ∉ ruleList and number of items
                           at conditionSet of r2 > l)
13.                        ruleList = ruleList ∪ r2;
14.                        enqueue(q, r2);
```

Fig. 2. The second step of CARGBA Rule Generator

Step 2 This step generates all the association rules of the form *(l+1)-rules* to *n-rules* that have *confidence* greater than or equal to *SatisfactoryConfidence* where n is the number of non-class attributes of the data set. This step is based on totally reverse manner of Apriori algorithm [8]. We call this algorithm the "Reverse Rule Generation Algorithm" and is given in figure 2.

ruleList is a list that contains the generated rules and q denotes a queue. *constructRule* function (line 4) constructs a rule r from a record *rec* in the training example. *constructRule* function also calculates the *confidence* of rule r. *constructRule2* function (line 11) constructs a rule *r2* from rule r by removing attribute A. *constructRule2* function also calculates the *confidence* of rule *r2*.

Finally, merging of rules generated from step 1 and 2 is done by:

$$PRs = PRs \bigcup ruleList;$$

2.2 CARGBA Classifier Builder

This section presents the CARGBA Classifier Builder algorithm. *PRs* contains a lot of rules generated by CARGBA Rule Generator. All these rules will not be used to classify test instances. In this step, we select a subset of the rules from *PRs* to cover the dataset. The selected rules are sorted in descending order of *confidence*, *support* and *rule length*. The classifier builder algorithm is given in figure 3.

```
1. finalRuleSet = Φ;
2. dataSet = D;
3. sort(prRs);
4. for each rule r ∈ prRs do
5.       if r correctly classifies at least one training
                  example d ∈ dataset then
6.                remove d from dataset;
7.                insert r at the end of finalRuleSet;
```

Fig. 3. The CARGBA classifier builder

finalRuleSet is a list that will contain rules that will be used in the classifier. *sort* function (line 3) sorts *PRs* in descending order of *confidence*, *support* and rule length. Lines 4-7 take only those rules in the *finalRuleSet* which can correctly classify at least one training example. Note that the insertion in *finalRuleSet* ensures that all the rules of *finalRuleSet* will be sorted in descending order of *confidence*, *support* and *rule length*.

When a new test example is to be classified, the classifier classifies according to the first rule in the *finalRuleSet* that covers the test example. If all the rules of the classifier fail to cover the test example, then the test example will be classified to a *default* class i.e. the class with the maximum number of training examples associated with.

3 Experimental Studies

We have evaluated the accuracy of our algorithm on 6 datasets from UCI ML Repository [9]. The accuracy of each dataset is obtained by 10-fold cross-validations. We use C4.5's shuffle utility to shuffle the data sets. We have calculated the mean and variance of our accuracy based on several runs of our algorithm on each data set. On each run, we have randomly selected the training and test data. Discretization of continuous attributes is done using the same method used in CBA [5].

In the experiments, the parameters of the four methods are set as follows. All C4.5 [1] parameters are set to their default values. We test both C4.5 decision tree method and rule method. Since the rule method has better accuracy, we only present the accuracy for the rule method. For CBA [5], we set *support* threshold to 1% and *confidence* threshold to 50% and disable the limit on the number of rules. Other parameters remain default. For CMAR [7], the *support* and *confidence* thresholds are set as it is in CBA. The *database coverage* threshold is set to 4 and the *confidence difference* threshold is set to 20%. For CARGBA, we investigated parameter sensitivity in details and found that CARGBA is not too sensitive to any particular parameter. So we decided to go with the default parameter values used by other algorithms. Minsupport is set to 1%, *satisfactoryconfidence* is set to 50% and l is set to half of the number of attributes of the

dataset. Maximum no. of rules in a level is set to 30,000 in CARGBA. We have performed pruning using correlation coefficient introduced in [10].

3.1 Results

In this section, we report our experimental results on comparing CARGBA against three other popular classification methods: C4.5 [1], CBA [5] and CMAR [7]. The experimental result is shown in Table 1. For CARGBA, we also present the variance of accuracies obtained for each data set.

Table 1. Comparison of C4.5, CBA, CMAR and CARGBA on accuracy

Dataset	C4.5	CBA	CMAR	CARGBA (Mean)	CARGBA (Variance)
pima	**75.5**	72.9	75.1	73.83	1.1
iris	95.3	94.7	94.0	**95.33**	0.8
heart	80.8	81.9	82.2	**82.22**	1.5
glass	68.7	**73.9**	70.1	73.83	0.7
tic-tac	99.4	**99.6**	99.2	**99.6**	0.3
diabetes	74.2	74.5	75.8	**76.17**	2.2
Average	82.32	82.92	82.73	**83.50**	

The won-loss-tied record of the CARGBA against C4.5 in term of accuracy is 4-1-1. The won-loss-tied record of the CARGBA against CBA and CMAR algorithms in term of accuracy are 4-1-1 and 4-1-1, respectively. The result shows that CARGBA outperforms CBA, C4.5 and CMAR in terms of average accuracy on 6 data sets.

4 Conclusion

Association rules generation algorithms that generate association rules based on Apriori algorithm with low support suffer from a limitation that they miss some high confidence rules with lower support. On the other hand, association rules generation algorithms that generate rules at a reverse order of Apriori algorithm without support pruning suffer from the limitation that the number of support less rules is very large in number and producing the general rules takes a lot of computational time. In this paper we have proposed a novel associative classification method, CARGBA algorithm that overcomes the above two problems successfully. Our experiments on 6 databases in UCI machine learning database repository show that CARGBA is consistent, highly effective at classification of various kinds of databases and has better average classification accuracy in comparison with C4.5, CBA and CMAR.

Acknowledgement. MMI is currently a Visiting Associate Professor at University of Fukui supported by the Fellowship from Japanese Society for Promotion of Science (JSPS). This work was in part supported by grants to KM from JSPS, Yazaki Memorial Foundation for Science and Technology, and University of Fukui.

References

1. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
2. Duda, R., Hart, P.: Pattern Classification and Scene Analysis. John Wiley & Sons, Chichester (1973)
3. Lim, T.S., Loh, W.Y., Shih, Y.S.: A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learning 39 (2000)
4. Cristianini, N., Shawe-Taylor: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge (2000)
5. Liu, B., Hsu, W., Ma, Y.: CBA: Integrating Classification and Association Rule Mining. In: KDD 1998, New York, NY (August 1998)
6. Dong, G., Zhang, X., Wong, L., Li, J.: Caep: Classification by Aggregating Emerging Patterns. In: Arikawa, S., Furukawa, K. (eds.) DS 1999. LNCS (LNAI), vol. 1721, Springer, Heidelberg (1999)
7. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: ICDM 2001, San Jose, CA (November 2001)
8. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: Proc. Of the SIGMOD, Washington, D.C., pp. 207–216 (1993)
9. Blake, C., Merz, C.: UCI repository of machine learning databases, `http://www.ics.uci.edu/~mlearn/MLRepository.html`
10. Antonie, M., Zaïane, O.R.: An Associative Classifier based on Positive and Negative Rules. In: DMKD 2004, Paris, France, June 13 (2004)