A New Class Based Associative Classification Algorithm

Zhonghua Tang and Qin Liao

***Abstract*—applying the association rule into classification can improve the accuracy and obtain some valuable rules and information that cannot be captured by other classification approaches. However, the rule generation procedure is very time-consuming when encountering large data set. Besides, traditional classifier building is organized in several separate phases which may also degrade the efficiency of these approaches. In this paper, a new class based associative classification approach (CACA) is proposed. The class label is taken good advantage of in the rule mining step so as to cut down the searching space. The proposed algorithm also synchronize the rule generation and classifier building phases, shrinking the rule mining space when building the classifier to help speed up the rule generation. Experimental result suggested that CACA is making better performances in accuracy and efficiency in Associative classification approaches.**

***Index Terms*—Classification, Association Rule, Frequent Item, Data mining**

I. Introduction

Classification is one of the most important tasks in data mining. Researchers are focusing on designing classification algorithms to build accurate and efficient classifiers for large data sets. Being a new classification method that integrates association rule mining into classification problems, associative classification achieves high classification accuracy, its rules are interpretable and it provides confidence probability when classifying objects which can be used to solve classification problem of uncertainty. Therefore, it becomes a hot theme in recent year.

The traditional associative classification algorithms basically have 3 phases: Rule Generation, Building Classifier and Classification as shown in Fig.1. Rule Generation employ the association rule mining technique to search for the frequent patterns containing classification rules. Building Classifier phase tries to remove the redundant rules, organize the useful ones in a reasonable order to form the classifier and the unlabeled data will be classified in the third step. Some experiments done over associative classification algorithms such as CBA [1], CMAR [2], MCAR [3] and GARC [4] state that the associative classification methods share the features of being more accurate and providing more classification rules. However, the drawbacks can be generalized as follow:

First, although the associative classification can provide more rules and information, redundant rules may also be included in the classifier which increases the time cost when classifying objects. MCAR determined a redundant rule by checking whether it covers instances in training data set or not [3]. GARC brought in the notion of compact set to shrink the rule set by converting the rule set to a compact one [4]. Since the reduction of redundant rules is lagging of the rule generation, it fails to avoid some meaningless searching.

Second, as we know, the rule generation is based on frequent pattern mining in associative classification, when the size of data set grows, the time cost for frequent pattern mining may increase sharply which may be an inherent limitation of associative classification. W. Li, J. Han and J. Pei mine the frequent patterns with FP Growth technique [5] in CMAR [2] which is proved to be very efficient, but extra time should be considered to compute the support and confidence of rules by scanning data set again. Fadi T., Peter C. and Peng Y. store training data with the form of vertical data representation minimize the time of scanning the data set and simplify the computation of support and confidence. However, the costly problem remains unsolved.

In this research paper, a class based associative classification algorithm is proposed to solve the difficulty aforementioned. Section 2 presents the definition of associative classification problem. Our proposed algorithm is discussed in Section 3. Section 4 demonstrates the experimental result and Section 5 conclude the paper.

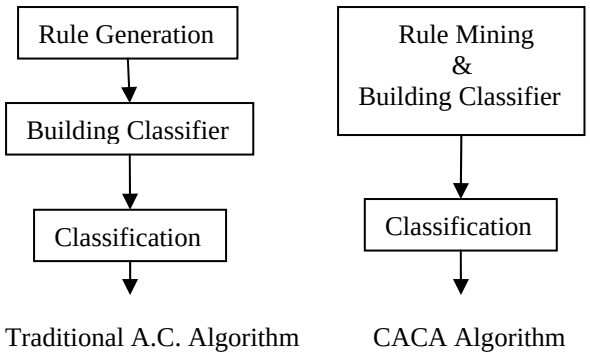

Fig.1 Procedures of A.C. Algorithms

Zhonghua Tang is with the School of Mathematical Science, South China University of Technology, Guangzhou, Guangdong 510640, P. R. China (e-mail: Nelsontang@ sohu.com).

Qin Liao is with the School of Mathematical Science, South China University of Technology, Guangzhou, Guangdong 510640, P. R. China (e-mail: maqliao@scut.edu.cn).

(Advance online publication: 24 May 2007)

II. Associative Classification

Let D be a training data set with m attributes $A_1, A_2, \cdots, A_m$ and $|D|$ instances. C is a class label. Then the values of attribute A_i and class C can be noted as a_i and c, respectively. An item set is a set of several attribute values $< a_{i1}, a_{i2}, \cdots, a_{ir} >, q \le m$ while an instance is $(a_1, a_2, \cdots, a_m, c)$. A classification rule $r =< a_{i1}, a_{i2}, \cdots, a_{il}, c >$, $l \le m$ is a combination of an item set and a class label. $Occr(r)$ is the number of instances in D that match the item set(condition)of r. $Suppcount(r)$ [3] is the numbers of instances in D that match the item set and the class label of r as well. Thus the support of r is $Supp(r) = Suppcount(r) / |D|$ [3] while the confidence of r is $Conf(r) = Suppcount(r)/Occr(r)$ [3]. The $minsupp$ and $minconf$ is the thresholds of support and confidence of rules given by user. Set R is the collection of classification rules where $\forall r \in R$ satisfies $Supp(r) > minsupp$ and $Conf(r) > minconf$. Given $r =< a_{i1}, a_{i2}, \cdots, a_{il}, c > \in R$, it implicates that if the attribute values of an instance match $a_{i1}, a_{i2}, \cdots, a_{iq}$ completely, the confidence of this instance belonging to class c is $Conf(r)$. Associative classification is to collect rules in training data set D, organize them in a certain order to form a classifier. When provided an unlabeled object, the classifier selects the rule in accordance with the order whose condition matches the objects and assigns class labels of the rule to it.

III. Class Based Associative Classification Algorithm

Aiming at solving the difficulty in Section 1, 4 innovations are integrated in CACA: 1, use the class based strategic to cut down the searching space of frequent pattern; 2, design a structure call Ordered Rule-Tree to store the rules and their information which may also prepare for the synchronization of the two steps; 3, redefine the compact set so that the compact classifier is unique and not sensitive to the rule reduction; 4, synchronize the rule generation and building classifier phases(shown in Fig.1).

A. Class based rule mining strategy

Given a training data set D with k classes, the principle idea of class based rule mining is to divide the single attribute value set C_{all} for all classes into k smaller ones for every class, that is, to limit the searching in k low dimensional spaces other than a high dimensional one. An example is given for the further explanation. Table I shows a training data set D with 10 rows, 4 attributes and a class label. The support threshold is 2.

Table . Training Data set

No	A_1	A_2	A_3	A_4	Class
1	a_{11}	a_{21}	a_{32}	a_{41}	c_1
2	a_{11}	a_{22}	a_{32}	a_{42}	c_2
3	a_{11}	a_{21}	a_{31}	a_{43}	c_2
4	a_{11}	a_{22}	a_{33}	a_{41}	c_1
5	a_{12}	a_{21}	a_{31}	a_{42}	c_2
6	a_{12}	a_{21}	a_{33}	a_{41}	c_1
7	a_{12}	a_{23}	a_{33}	a_{43}	c_2
8	a_{11}	a_{21}	a_{32}	a_{41}	c_1
9	a_{12}	a_{24}	a_{32}	a_{42}	c_1
10	a_{13}	a_{21}	a_{32}	a_{41}	c_1

From Table , we see that, a_{13}, a_{23}, a_{24} do not satisfy $minsupp$ and are removed. Instances with the remaining 10 attributes and class label are stored in the form of vertical data representation. Thus the refined data set is shown in Fig.2. Each column stores the NO. of instance which has the corresponding attribute value.

A_1		A_2		A_3			A_4			Class	
a_{11}	a_{12}	a_{21}	a_{22}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}	c_1	c_2
1	5	1	2	3	1	4	1	2	3	1	2
2	6	3	4	5	2	6	4	5	7	4	3
3	7	5			8	7	6	9		6	5
4	9	6			9		8			8	7
8		8			10		9			9	
		10								10	

Fig.2 Refined data of data set D

Intersect the column $C(a_{11}) = \{1,2,3,4,8\}$ of a_{11} and that $C(c_1) = \{1,4,6,8,9,10\}$ of c_1, the intersection is {1,4,8}. Since $|C(a_{11}) \cap C(c_1)| = 3$ satisfies the threshold, a_{11} should be put into the single attribute value set of c1, noted C_1. Intersect every column of attribute value with that of every class, we gain 2 single attribute value sets $C_1 = \{\{a_{11}\},\{a_{21}\},\{a_{32}\},\{a_{33}\},\{a_{41}\}\}$ and $C_2 = \{\{a_{11}\},\{a_{31}\},\{a_{43}\}\}$. Then, the item set (condition) of any rule with label c_1 are the subset of C_1 while that of any rule with label c_2 are the subset of C_2. Rule mining can be limited in these 2 small sets instead of C_{all} with 10 elements. The sum of combinations of elements in the two small sets is $(2^5 - 1) + (2^3 - 1) = 38$ while that of C_{all} is $2^{10} - 1 = 1023$. Obviously, searching limited in these two small sets is more efficient than that in the larger one. Experimental result shows that, when the numbers of attributes and their values are large, this strategic is effective in cutting down the rule mining space which is a key factor to the time cost of associative classification.

B. Ordered Rule Tree Structure (OR-Tree)

To facilitate the synchronization, we design a structure call Ordered Rule Tree under the inspiration of CR-Tree [2] to store and rank rules. The structure of OR-Tree can be view in Fig.3. It is composed with a tree structure and an ordered list. When a rule $r < a_{i1}, a_{i2}, \cdots, a_{il}, c >$ satisfying the support and confidence thresholds is generated, attribute values $a_{i1}, a_{i2}, \cdots, a_{iq}$ will be stored as nodes in this tree according to their frequency in D in descending order. The last node points to an information node storing the rule's information such as class label, support and confidence. Each rule can and only can have one information node. The ordered list is designed to organize all rules in the tree. Each node in the chain points to a certain rule. Nodes pointing to the rules with higher priority are closer to the head node, while those pointing to the rules with lower priority are farther from the head node. When a new non-redundant rule is inserted in the OR-Tree, a new node pointing to this rule will be inserted into a suitable place in the ordered list. With the OR-tree Structure, the classifier can be built and modified easily.

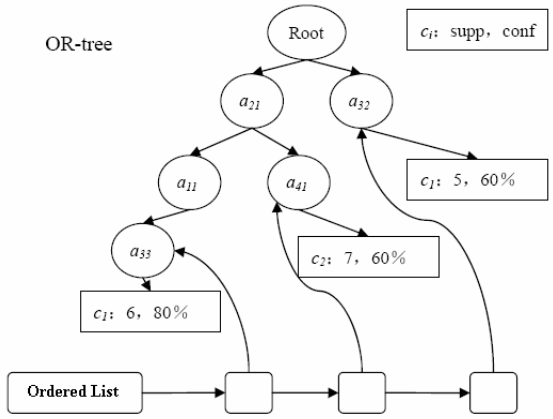

Fig.3 Structure of OR-tree

Given rules r_1 and r_2, the rule ranking standard is:

(1) if $Conf(r_1) > Conf(r_2)$, then r_1 precedes r_2, noted $r_1 \succ r_2$;

(2) if $Conf(r_1) = Conf(r_2)$, but $Supp(r_1) > Supp(r_2)$, then $r_1 \succ r_2$;

(3) if $Conf(r_1) = Conf(r_2)$ and $Supp(r_1) = Supp(r_2)$, but r_1 is more general than r_2, then $r_1 \succ r_2$;

(4) if $Conf(r_1) = Conf(r_2)$, $Supp(r_1) = Supp(r_2)$, and the condition of r_1 is as detailed as r_2, but r_1 is generated before r_2, then $r_1 \succ r_2$.

C. Compact Rule Set Redefinition and Pruning Skill

MCAR judge a redundant rule by check whether it cover at least one instance [3]. On one hand, this strategic can not guarantee the removed rule is redundant to the instances uncovered by training data set; on the other hand, the reduction should be carry out after all rules are generated and ranked. It is impossible to implement the synchronization with this strategic. However, the definition of compact set and redundant rule in [4], can not ensure the compact classifier is unique and with the same accuracy compared with the original one, which means the classifier and the accuracy changes as the order of rule reduction changes. To overcome these problems, the compact set and redundant rule are redefined in this paper.

Definition1 (Redundant Rule): Given $r_1, r_2, r_3 \in R$, r_2 is redundant, if:

(1) $r_1 =< Item_1, c_k >$, and $r_2 =< Item_1, c_p >$, but $r_1 \succ r_2$;

(2) $r_1 =< Item_1, c_k >$, $r_2 =< Item_2, c_p >$, $Item_1 \subset Item_2$, and $r_1 \succ r_2$;

(3) $r_1 =< Item_1, c_k >$, $r_2 =< Item_2, c_k >$, $Item_1 \subset Item_2$, and $r_1 \succ r_2$;

(4) $r_1 =< Item_1, c_k >$, $r_2 =< Item_2, c_k >$, $Item_1 \subset Item_2$, $r_2 \succ r_1$, for $r_3 =< Item_3, c_p >$, $Item_1 \subset Item_3$, $r_3 \prec r_2 \prec r_1$.

Since the transitivity of redundancy is obvious, the proof is ignored here because of the limited space.

Definition2 (Compact Rule Set): For rule set R, if $\hat{R} \subset R$, any redundant rule $r \notin \hat{R}$, and $\hat{R}$ is unique, then $\hat{R}$ is the compact set of R.

Theorem1 indicates that with the redefinition of redundant rule and compact set, the classifier is uniquely equivalent to the original one. Therefore, though the size of the compact set is smaller than the original one, both rule sets should have the same accuracy.

Theorem1 For any rule set R, $\hat{R}$ is the rule set excluding all redundant rules according to the Definition1, then $\hat{R}$ is the compact set of R or $\hat{R}$ is compact over R。

Proof：Since $\hat{R}$ is without redundant rules, we only have to prove to uniqueness of $\hat{R}$。Prove by contradiction，suppose $\hat{R}$ is not unique, so $\exists \hat{R}_1$, $\hat{R} \neq \hat{R}_1$, $\forall r \in R - \hat{R}_1$ is redundant. $\hat{R} \neq \hat{R}_1$, $\exists r_2 \in \hat{R}$, $r_2 \notin \hat{R}_1$, then $r_2 \in R - \hat{R}_1$ is redundant. That is, $\exists r_1 \in \hat{R}_1$, r_2 is redundant when compared to r_1, $r_1 \notin \hat{R}$ (otherwise $r_2 \notin \hat{R}$), $\exists r_0 \in \hat{R}$, r_1 is redundant to r_0, according to the transitivity, r_2 is redundant to r_0, $\therefore r_2 \notin \hat{R}$,contradiction. $\hat{R}_1 = \hat{R}$, $\hat{R}$ is unique。

Although a class based strategic is applied to cut down the rule mining space, some pruning skills may help raise the efficiency of associative classification. Some pruning skills are present in [4] and other papers. Here, a new one is proposed under the synchronization of the 2 phase to help to prevent mining the frequent patterns which won't lead to any not-redundant rule.

For rule $r_i = (item, c_i)$, satisfies:

$$supp(r_i)/conf(r_i)\bullet(1-conf(r_i)) < minsupp\bullet|D| \quad (1)$$

then there won't be any new rule $r_k = (item_k, c_j)$, $item_k \supset item, c_j \neq c_i$, $supp(r_k) >= minsupp$. Thus the new pruning rule is given below.

Pruning: For rule $r_i = (item, c_i)$, if $supp(r_i)/conf(r_i)\bullet(1-conf(r_i)) < minsupp$, stop mining $r_i = (item_k, c_i), item_k \supset item$.

D. CACA Algorithm

Basing on the works of 3.2 and 3.3, CACA technically combined the rule generation and the building classifier phases together. Once a new rule is generated, the algorithm visits the OR-Tree partially to recognize its redundancy, stores it in the OR-Tree and ranks it in the rule set. Not only can the synchronization simplify the procedure of associative classification but also apply the pruning skill to shrink the rule mining space and raise the efficiency. The algorithm is design as follow:

(1) CACA first scans the training data set D, stores data in form of vertical representation, counts the frequency of every attribute value a_{ij} and arrange a_{ij} in descending order by frequency. The a_{ij} which is failed to satisfy the $minsupp$ is filtered in this step.

(2) For the remaining attribute values a_{ij} in step (1), Intersect $C(a_{ij})$ and $C(c_n), n = 1, 2, \cdots k$. Add a_{ij} into C_n if $|D(a_{ij}) \cap D(c_n)| > minsupp$ .Thus we have k single attribute value sets $C_1, C_2, \cdots C_k$.

(3)For class c_n, choose a $a_{i_1 j_1}$ in C_n in accordance with the order, figure out whether rule $r = (a_{i_1 j_1}, c_n)$ can satisfy $minconf$ (all the elements in single attribute value sets satisfy support threshold) and its redundancy. If it satisfies the threshold and is not a redundant one, it would be inserted and ranked in the OR-Tree. Check whether it satisfies the condition of pruning skill. If yes, let $C_n = C_n \setminus a_{i_1 j_1}$ and repeat (3), else go on with the recursive procedure of mining more detailed rules.

(4) Take an $a_{i_2 j_2} \in C_n \setminus a_{i_1 j_1}$, $i_1 \neq i_2$, with respect to the frequency order. Judge the satisfaction of $minsupp$ for $r = (a_{i_1 j_1}, a_{i_2 j_2}, c_n)$. Any dissatisfaction leads to a new selection of element, that is, select $a_{i_3 j_3} \in C_n \setminus \{a_{i_1 j_1}, a_{i_2 j_2}\}$ and go on with the judgment. Otherwise, if $r = (a_{i_1 j_1}, a_{i_2 j_2}, c_n)$ satisfy the $minsupp$, check the confidence threshold and redundancy as in step (3). Insert the satisfactory rule in the OR-Tree (or modify the OR-tree when an old rule should be replaced by a new one or an old rule become redundant), rank it and check whether the pruning can be applied here. If the pruning can be carried out here, go back to the upper layer of the recursion. If not, recursively construct Item sets with more attribute values to obtain new rule. When all rules related with c_n and $a_{i_1 j_1}$ is properly handled, the recursion is finished. Then let $C_n = C_n \setminus a_{i_1 j_1}$, repeat (3), until $C_n = \phi$.

(5) Repeat step (3)-(4) until $C_n = \phi, n = 1, 2, \cdots k$.

(6) Classify the unlabeled data by the obtaining classifier.

IV. Experimental Results

Experiments are carried out on data sets of UCI Data Collection. Comparisons are made over accuracy, time cost and the searching space size among several algorithms. First we compare the accuracy of CBA, MCAR and CACA over 13 data sets. The implementation of CBA is provided by author while MCAR and CACA is implemented with C++6.0.

Table . Accuracy of CBA, MCAR and CACA on 13 datasets

dataset	CBA	MCAR	CACA
Austral	85.32	86.93	87.37
Balance-scale	66.31	76.98	77.20
Breast-w	95.03	96.77	96.59
Cleve	83.11	82.46	83.06
Diabetes	76.01	78.84	79.01
Glass	70.00	69.97	70.86
Heart-s	79.68	81.24	82.92
Iris	94.11	96.33	96.66
Led7	71.80	81.76	81.75
Mushroom	91.89	97.41	97.15
Pima	75.75	78.68	78.73
Tic-tac	100	99.89	99.95
Wine	98.33	97.02	96.97

According to Table , MCAR and CACA are with higher accuracy in most data set compared to CBA. Although CACA and MCAR have similar accuracy, CACA gave better performances over 9 data sets while MCAR achieved higher accuracy on 4. Analysis on the rule sets of both algorithms shows that MCAR's rule sets are equal to or slightly smaller than the CACA's which mean's that a few non-redundant rules are removed in MCAR, and this may also accounts for the CACA's slight improvement on accuracy. Therefore, we concluded that using compact rule set to simplify the classifier is more reasonable.

Since the experimental result in [2] suggested MCAR is more efficient than CBA, here we only compare the Time cost of MCAR and CACA. Algorithms are run on computer P4 2.66Ghz, 512RAM. The time cost test results of both algorithms are presented in Table . Besides, the searching space size are also included in the table.($minconf = 60\%$)

Table Ⅲ. Comparisons between MCAR and CACA on Time cost and Searching Space

Dataset	Minsupp	Time Cost		Searching Space	
		MCAR	CACA	MCAR	CACA
Balance-scale	5%	0.015	0.000	71	22
Breast-w	2%	0.219	0.075	3489	129
Diabetes	5%	0.125	0.047	1201	219
Glass	5%	0.078	0.018	3036	1347
Lymph	10%	1.313	0.151	31939	2610
Primary-tumor	6%	2.730	0.494	49035	5807
zoo	5%	2.421	0.078	151550	1912

Focusing on the time cost of both algorithms, we find CACA has achieved a great improvement. The time cost of CACA is 12.5% of MCAR on average over 7 data sets. The searching spaces of the CACA are smaller than that of MCAR. CACA only search 219 candidate frequent patterns to mining the classification rules but the MCAR search 1209 ones on data set Diabetes with $minsupp = 5\%$. The average search space size of CACA is merely 17% of that of MCAR. Class based frequent pattern mining is proved to be powerful in associative classification.

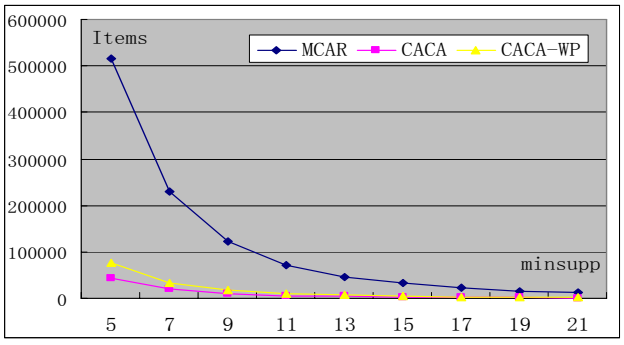

Fig.4 Comparisons over searching space size

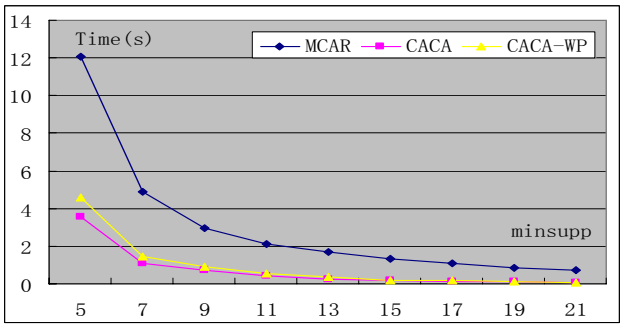

Fig.5 Comparisons over time cost

At last, verify the minimal support count from 5 to 21, Fig.4 shows the comparison of time cost while Fig.5 the searching space size of MCAR, CACA and CACA without Pruning over data set Lymph. The time cost and searching space size of CACA increase slower than those of MCAR, when the *minsupp* becomes smaller. The average time cost and searching space size of CACA are 16.9% and 6.87% of those of MCAR. And the average time cost and searching space size of CACA are 76.9% and 57.8% of those of CACA without pruning.

V. Conclusions

According to the characteristic of associative classification, a new class based frequent pattern mining strategic is designed in CACA to cut down the searching space of frequent pattern. OR-Tree structure enables the synchronization of the traditional phases which may not only simplify the associative classification but help to guide the rule generation and speed up the algorithm. And the redefinition of the redundant rule and compact set guarantee the usage of the compact set to help improve the classification efficiency and rule quality won't affect the accuracy of CACA. Experimental result shows that, CACA is more competitive on accuracy and efficiency among associative algorithms.

References

[1] B. Liu, W. Hsu, &Y. Ma, "*Integrating classification and association rule mining*", Proceeding of KDD'98, 1998, pp. 80-86.

[2] W. Li, J. Han& J. Pei, "*CMAR: Accurate and efficient classification based on multiple-class association rule*", Proceedings of the ICDM'01, 2001, pp. 369-376.

[3] Fadi T.,Peter C. & Peng Y, "*MCAR: Multi-class Classification based on Association Rule*", IEEE International Conference on Computer Systems and Applications ,2005, pp. 127-133.

[4] G. Chen, H. Liu et al, "*A new approach to classification based on association rule mining*", Decision Support Systems 42, 2006, pp. 674-689.

[5] J. Han, J. Pei & Y. Yin, "*Mining frequent patterns without candidate generation*", Proceedings of 2000ACM SIGMOD international conference on management of data,2000, pp. 1-12

[6] G.Dong, X. Zhang,L. Wong, and J.Li, " *Caep: Classification by aggregating emerging patterns*". InDS'99(LNCS 1721), Japan, 1999.

[7] F.Thabtah, P. Cowling and Y. Peng, "*A new multi-class,multi-label associative classification approach*".The 4th International Conference on Data Mining(IVFM'04), Brighton, UK, 2004.

[8] X. Lin and J. Han, "*CPAR: Classification based on predictive association rule*", In SDM2003, San Francisco, CA, 2003.

[9] CBA: http://www.comp.nus.edu.sg/~dm2/p_download.html

[10] R. Agrawal, T. Amielinski, and A. Swami, "*Mining association rule between sets of items in large databases*", In Proceeding of the 1993 ACM SIGMOD International Conference on Management of Data,Washington,DC,1993, pp.207-216.

[11] R. Agrawal and R. Srikant, "*Fast algorithms for mining association rule*", Proceeding of the 20th International Conference on Very Large Data Base,1994, pp. 487-499.