Proceedings of the Linguistic And Cognitive Approaches To Dialog Agents Symposium, Rafal Rzepka (Ed.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

Associative Text Categorisation Rules Pruning Method

Hussein Abu-Mansour[1], Wa'el Hadi[2], T.L.McCluskey[1] and Fadi Thabtah[2]

Abstract. In this paper, the problem of rule pruning in associative text categorisation is investigated. We propose a new rule pruning method within an existing associative classification algorithm called MCAR. Experimental results against large text collection (Reuters-21578) using the developed pruning method as well as other known existing methods (Database coverage, lazy pruning) are conducted. The bases of the experiments are the classification accuracy and the number of generated rules. The results derived show that the proposed rule pruning method derives higher quality and more scalable classifiers than those produced by lazy and database coverage pruning approaches. In addition, the number of rules generated by the developed pruning procedure is usually less than those of lazy pruning and database coverage heuristics.

Keywords: Association Rule, Classification, Data Mining, Text categorisation, Rule pruning

1. INTRODUCTION

Associative Classification (AC) also known as classification based on association rule, is an integration of two important data mining tasks, association and classification. Association rule and classification are analogous tasks, with the exception that classification's main aim is the prediction of class labels, while association rule describes correlations among items in a database. Several studies (Liu et al.,
1998; Li et al., 2001; Yin and Han, 2003; Thabtah, et al., 2004; Thabtah, et al., 2005; Li et al., 2008; Niu et al., 2009) provide evidence that AC approach is able to derive more accurate classifiers than traditional classification techniques, such as decision trees (Quinlan, 1993; Quinlan, 1998), rule induction (Quinlan and Cameron-Jones, 1993; Cohen, 1995) and probabilistic (Duda and Hart, 1973) approaches. However, this approach suffers from the exponential growth of rules. This is since AC employs association rule during the learning step where all the correlations among the items and the class are discovered in a form of if-then rules. Thus, pruning duplicate and misleading rules becomes essential.

The AC mining approach works as follow. First, all rules satisfying some user-specified parameters (minimum support (minsupp), minimum confidence (minconf)), are generated using an association rule mining algorithm. The number of rules generated might be of the order of several thousands, and many of them are either redundant, i.e. never used, or noisy, i.e. not discriminative among the classes. Second, the redundant and noisy rules are removed using pruning procedure(s). A significant number of rules can be removed at this stage and the rules left are the interesting ones that form a model (classifier) used later to classify new test cases. Each classifier must have a default rule which is applied when no other classifier rule can be used.

AC algorithms normally derive large sets of rules (Topor and Shen, 2001; Li et al., 2001; Kundu et al., 2008; Li et al. , 2008) since (1) classification data sets are typically highly correlated and (2) association rule mining is used in the rule discovery step. As a result, there have been many attempts to reduce the size of classifiers produced by AC approaches, mainly focused on preventing rules that are either redundant or misleading from taking any role in the prediction process of test data cases. The removal of such rules can make the classification process more effective and accurate. It is not always helpful to have a large number of rules when we classify test cases. There is a great chance of having more than one rule contradicting each other in the answer class. We want to have a small number of highly predictive rules.

The aim of this paper is to investigate the rule pruning phase in AC mining in order to reduce the size of the resulting classifiers. Particularly, we develop a new rule pruning method. The developed pruning procedure is implemented within a known AC algorithm called MCAR (Thabtah et al., 2005), and are tested against large text categorisation collection called Reuters-21578 Mod Split (Lewis, 1998). To the best of the authors' knowledge, AC rule pruning on text categorisation has not yet been explored in the literature.

This paper is structured as follows: AC approach and known rule pruning methods are discussed in Section 2. In Section 3, the proposed rule pruning technique is presented. Further, the results that show the impact of pruning on the size of the classifiers and the prediction accuracy are demonstrated in Section 4. Finally, conclusions are given in Section 5.

2. ASSOCIATIVE CLASSIFICATION

2.1 Associative Classification Problem

AC is a special case of association rule in which only the class attribute is considered in the rule's right-hand-side (consequent) (Liu et al., 1998), for example in a rule such as $X \rightarrow Y$, Y must be a class attribute. We follow (Thabtah, et al., 2004) for the definition of the AC problem. A training data set T has m distinct attributes $A_1, A_2, \ldots, A_m$ and C is a list of class labels. The number of rows (cases) in T is denoted $|T|$.

Definition 1: A row or a training case in T can be described as a combination of attributes A_i and values a_{ij}, plus a class denoted by c_j.

[1]Computing and Engineering Dept, Huddersfield University, UK, Email:{H.Y.Abu Mansour, lee}@hud.ac.uk,
[2]MIS Dept, Philadelphia University, Jordan. Email: {whadi, ffayez}@philadelphia.edu.jo

Definition 2: An attribute value can be described as a term name A_i and a value a_i, denoted $<(A_i, a_i)>$.
Definition 3: An *AttributeValueSet* can be described as a set of disjoint attribute values contained in a training case, denoted $<(A_{i1}, a_{i1}), \ldots, (A_{ik}, a_{ik})>$.
Definition 4: A *ruleitem r* is of the form $<$ *AttributeValueSet*, *c*$>$, where $c \in C$ is the class.
Definition 5: The actual occurrence (*actoccr*) of a *ruleitem r* in *T* is the number of rows (cases) in *T* that match the *AttributeValueSet* of *r*.
Definition 6: The support count (*suppcount*) of *ruleitem r* is the number of rows in *T* that match *r's AttributeValueSet*, and belong to the class *c* of *r*.
Definition 7: The occurrence of an *AttributeValueSet i* (*occatt*) in *T* is the number of rows in *T* that match *i*.
Definition 8: A *ruleitem r* passes the *minsupp* threshold if $(suppcount(r)/|T|) \geq minsupp$,
Definition 9: A *ruleitem r* passes the *minconf* threshold if $(suppcount(r)/actoccr(r)) \geq minconf$.
Definition 10: Any *ruleitem r* that passes the *minsupp* threshold is said to be a *frequent ruleitem*.
Definition 11: A class association rule (CAR) is represented in the form: $(A_{i1}, a_{i1}) \wedge \ldots \wedge (A_{ik}, a_{ik}) \rightarrow c$, where the antecedent (rule body/LHS) of the rule is an *AttributeValueSet* and the consequent (RHS) is a class.
Definition 12: We say that a CAR *R* partial matches a test case *t* if R contains at least an item in its antecedent that exists in *t*.
Definition 13: We say that a CAR *R* fully matches a test case *t* if all *R* items are in *t*.

A classifier is a mapping form $H : A \rightarrow Y$, where *A* is the set of *AttributeValueSet* and Y is the set of class labels. The main task of AC is to construct a set of rules (model) that is able to predict the classes of previously unseen data, known as the test data set, as accurately as possible. In other words, the goal is to find a classifier $h \in H$ that maximises the probability that $h(a) = y$ for each test case.

2.2 Current Pruning Methods in AC

Several pruning methods have been used effectively to reduce the size of the classifiers in AC, some of which have been adopted from decision trees, like pessimistic estimation (Quinlan, 1987), others from statistics such as Chi-square testing ($\chi 2$) (Snedecor and Cochran, 1989). These pruning methods are utilised during the construction of the classifier, for instance, a very early pruning step, which eliminates ruleitems that do not pass the support threshold, may occur in the process of finding frequent ruleitems. Another pruning approach such as chi-square testing may take place when generating the rules, and post pruning methods, like database coverage (Liu et al., 1998) may be carried out after discovering all potential rules. Throughout this section, we will discuss pruning methods used in AC. It should be noted that we focus in this paper on pruning methods that are solely developed within AC mining algorithms and not adopted from closely related areas such as statistics and machine learning.

2.2.1 Database Coverage Pruning

The database coverage is a post pruning technique used in AC (Liu et al., 1998), which is usually invoked after rules have been created. If at least one case among all the cases in training data set is fully matched by the rule, the rule is inserted into the classifier and all cases covered are removed from the training data set. The rule insertion stops when either all of the rules are used or no cases are left in the training data set. The majority class among all cases left in the training is selected as default class. The default class is used in case when there are no covering rules. After this process, the first rule which has the least number of errors is identified as the cutoff rule. All the rules after this rule are not included in the final classifier since they often produce errors (Liu et al., 1998). The database coverage method was used first by CBA (Liu et al., 1998) and then latterly by other associative algorithms, including CBA (2) (Liu et al., 2003), CMAR (Li et al., 2001), ARC-BC (Antonie and Zaïane, 2002), CAAR (Xu et al., 2004), ACN (Kundu et al., 2008), Multi-label Classification based on Association Rules (Li et al., 2008), and ACCF (Li et al. , 2008).

2.2.2 Redundant Rule Pruning

In AC, all attribute value combinations are considered in turn as a rule's condition, therefore rules in the resulting classifiers may share training items in their conditions and for this reason there could be several specific rules containing many general rules. Rules redundancy in the classifier is unnecessary and in fact could be a serious problem, especially if the number of discovered rules is extremely large.

A pruning method that discards specific rules with less confidence values than general rules called redundant rule pruning, has been proposed in (Li et al., 2001). Redundant rule pruning method works as follows: Once the rule generation process is finished and rules are sorted, an evaluation step is performed to prune all rules such as $I' \rightarrow c$ from the set of generated rules, where there is some general rule $I \rightarrow c$ of a higher rank and $I \subseteq I'$. This pruning method significantly reduces the size of the resulting classifiers and minimises rules redundancy (Li et al., 2001).

Algorithms, including (Li et al., 2001; Antonie and Zaïane, 2004), have used redundant rule pruning. They perform such pruning immediately after a rule is inserted into the compact data structure, the CR-tree. When a rule is added to the CR-tree, a query is issued to check if the inserted rule can be pruned or some other already inserted rules in the tree can be removed.

2.2.3 Database like Pruning (Lazy Pruning)

Some AC techniques (Baralis and Torino, 2000; Baralis, et al., 2004; Baralis et al. 2008) argue that pruning classification rules

should be limited to only "negative" rules (those that lead to incorrect classification). In addition, they claim that database coverage pruning often discards some useful knowledge, as the ideal support threshold is not known in advance. Due to this, these algorithms have used a late database coverage-like approach, called lazy pruning, which discards rules that incorrectly classify training cases and keeps all others.

Lazy pruning happens after rules have been created and stored, where each training case is taken in turn and the first rule in the set of ranked rules applicable to the case is assigned to it. The training case is then removed and the correctness of the class assigned to the case is checked. Once all training data have been considered, only rules that wrongly classified training data are discarded and their covered data are put into a new cycle and the process is repeated until all training data are correctly classified. The results are two levels of rules; the first level contains rules that correctly classified at least one single training case and the second level contains rules that were never used in the training phase. The main difference between lazy pruning and database coverage pruning is that the second level rules that are held in the memory by the lazy pruning are completely removed by the database coverage during rule discovery step. Furthermore, once a rule is applied to the training data, all cases covered by the rule are removed (negative and positive) by the database coverage method.

Experimental tests reported in (Baralis and Torino, 2000; Baralis, et al., 2004; Baralis, et al., 2008) using 26 different data sets from (Merz and Murphy, 1996) showed that methods that employ lazy pruning such as L3 and L3G may improve classification accuracy on average by 1.63% over other techniques that use database coverage pruning

(Liu et al., 1998; Li et al. 2001). However, lazy pruning may lead to very large classifiers, which makes it difficult for a human to understand or be able to interpret. In addition, the experimental tests indicate that lazy pruning algorithms consume more memory than other AC techniques and, more importantly, they may fail if the support threshold is set to a very low value due to the very large number of potential rules.

3. THE PROPOSED RULE PRUNING METHODS

In this section, we discuss the proposed rule pruning method along with an example to illustrate it. Assume that the eleven rules shown in Table 1 are produced by an AC algorithm called MCAR (Thabtah et al., 2005) using minsupp and minconf of 20% and 40%, respectively from the training data set shown in Table 2. Before pruning kicks in the rules must be ranked according to confidence and support values in descending manner. The rule ranking criterion used in MCAR is as follows: (1) The rule with a higher confidence has a higher rank than the others. (2) If confidence values are the same between two or more rules, then the one with a higher support has a higher rank than the others. (3) If the confidence and support values of two or more rules are the same, then the one with fewer numbers of conditions in the antecedent has a higher rank. (4) If all the above criterions are similar, then the rule with a larger class count in the training data set (majority class) has a higher rank than the others

According to Table 1, Rule-5 is the highest ranked rule since its confidence is the largest one among the rest of the rules. Though Rule-1, Rule-6, Rule-7, Rule-9, Rule-10, and Rule-11 have the same confidence; Rule-1 is ranked higher due to its larger support. Still Rule-6, Rule-7, Rule-9, Rule-10, and Rule-11 have the same confidence and support values; but Rule-6 is ranked higher due to the fact that it has less number of values in its antecedent, and so forth.

3.1 High Precedence Classify Correctly Pruning Method (HPC).

Not all of the rules derived during the learning phase can be used in the prediction step, and therefore some rules will be removed. Figure 1 represents the HPC rule pruning procedure. To describe this method, let's use the example of Tables 1 & 2 in which the first ranked rule (Rule-5) is applied on the training data shown in Table 2. We find that this rule partially matches documents 3 and 4, and the class of this rule is identical to the class labels of these documents. So, this rule is inserted into the classifier, and we remove documents 3 and 4 from the training data set. We proceed to the second ranked rule i.e. Rule-1, and we check the applicability of this rule with the remaining training documents. We find that this rule partially matches

Rule-id	Rule	conf	sup	Class count	Rank
1	real=>sport	0.99	0.286	2	2
2	madrid=>sport	0.67	0.285	2	10
3	stock=>acq	0.75	0.428	3	8
4	share=>acq	0.67	0.285	3	9
5	group=>acq	1	0.285	3	1
6	amman=>general	0.99	0.285	2	3
7	madrid & real=>sport	0.99	0.285	2	6
8	share & stock=>acq	0.67	0.285	3	11
9	group & stock=>acq	0.99	0.285	3	4
10	group & share=>acq	0.99	0.285	3	5
11	group&share&stock=>acq	0.99	0.285	3	7

Table 1: Example of a rule-based model (potential rules)

id	Document	Class	Random Rank
1	real,madrid,stock,loss,share	sport	1
2	real,madrid	sport	5
3	stock,share,group	acq	2
4	stock,share,buy,group	acq	3
5	stock	acq	4
6	iraq,amman,jordan	general	7
7	amman,madrid,real	general	6

Table.2: Example of a training data

documents 1, 2, and 7, but the class label for document 7 is not consistent with that of Rule-1. So, we insert Rule-1 into the classifier since it covers correctly at least one document, and we remove documents 1, and 2. The third ranked rule (Rule-6)

Input: Given a set of generated rules R, and training data set *T*

Output: classifier (*Cl*)
1 *R'* = sort(*R*);
2 For each rule r_i in *R'* Do
3 Find all applicable data cases in *T* that match r_i's condition
4 If r_i correctly classifies a training case in *T*
5 Insert the rule at the end of Cl
6 Remove all cases in *T* covered by r_i
7 end if
8 If r_i cannot correctly cover any training case in *T*
9 Remove r_i from *R*
10 end if
11 end

Figure 1. HPC pruning method

covers correctly two documents 6 and 7, so it gets inserted into the classifier, and documents 6 and 7 are discarded from the training documents set, and we repeat the same steps for the rest of candidate rules until the training set becomes empty. In this example, the classifier contains just four rules, and the remaining candidate rules are deleted.

The difference between this pruning method and the database coverage (Liu et al., 1998) is that in the HPC method a rule gets inserted into the classifier if it partially covers at least one training case. On the other hand, in the database coverage, a rule must fully match the training case antecedent (rule body) in order to be considered.

Category Name	Training set	Testing Set
Acq	1650	719
Crude	389	189
Earn	2877	1087
Grain	433	149
Interest	347	131
Money-FX	538	179
Trade	369	117
Total	6603	2571

Table 3: Number of documents per category (Reuters-21578)

4. EXPERIMENTAL RESULTS

4.1 Text Data Collection

The benchmark used in the experiments is the Reuters-21578 (Lewis, 1998). The Reuters-21578 is the most widely used text data set in the text categorisation research. We used the ModApte version of Reuters-21578. This spilt leads to a corpus of 9,174 documents consisting of 6,603 training and 2,571 testing documents, respectively.

We tested our pruning procedures within the MCAR algorithm on the seven most populated categories with the largest number of documents assigned to them in the training data set. The experiments are conducted on 2.8 Pentium IV machine with 1GB RAM, and the proposed methods and MCAR are implemented using VB.Net programming language with a minsupp and minconf of 2%, and 40%, respectively. The minsupp has been set to 2% since more extensive experiments reported in (Liu, et al., 1998; Li, et al., 2001, Thabtah, et al., 2004, Thabtah, et al., 2005) suggested that it is one of the rates that achieve a good balance between accuracy and the size of the classifiers. The confidence threshold, on the other hand, has a smaller impact on the behaviour of any AC method and it has been set to 40%.

Table 3 represents the number of documents for each category in the Reuters-21578 data set. On these documents we performed stop word elimination but not stemming, and we select the top 1000 features using Chi Square (Snedecor and Cochran, 1989).

In the following section, we show the results of the proposed pruning procedures, the database coverage method, lazy method, and the case of no pruning.

4.2 Experimental Results

We performed extensive experiments on the seven most populated categories of the Reuters-21578 test collection to compare our proposed methods, the database coverage (Liu et al., 1998), lazy pruning (Baralis and Torino, 2000), and the case of no pruning with reference to the number of rules generated and the predictive accuracy.

Table 4 shows the number of rules derived from the Reuters text collection when different pruning approaches are implemented within MCAR algorithm (Thabtah, et al., 2004). It is obvious from the numbers shown in Table 4 that in general HPC rule pruning method outperforms lazy pruning, database coverage, and the case of no pruning.

Moreover, the number of rules generated without pruning method on "Acq" class is 80, whereas the number of rules derived using HPC pruning procedure is 28. The additional fifty two rules produced in the case of no pruning may decrease the classification accuracy and increase the prediction time. It is obvious from the numbers shown in Table 4 that algorithms, which use lazy pruning approach, often generate many more rules than those that employ other approaches. In particular, for all classification data sets we considered, MCAR using lazy pruning method produced more rules than other considered pruning heuristics.

One of the principle reasons for generating large number of rules by lazy pruning algorithms is due to storing rules that do even cover a single training data case in the classifier. For example, while constructing the classifier by the MCAR algorithm using lazy pruning method, every rule is evaluated to validate whether or not it covers a training data case. If a rule covers correctly a training case, it then will be added into the classifier, otherwise it will be removed. However, rules, which have been never tested, are also added as spare rules into the classifier. These spare rules

are may raise many problems such as user understandability and maintenance. Furthermore, the classification time may increase since the classification system must iterate through the rule set, and consequently these problems limit the use of lazy associative algorithms.

Class	no Pruning	HPC	Database Coverage	Lazy
Acq	80	28	27	40
Crude	8	5	4	6
Earn	172	15	17	55
Grain	5	5	5	5
Interest	4	1	2	4
Money -FX	23	12	12	15
Trade	9	4	6	8
Total	**301**	**70**	**73**	**133**

Table 4: MCAR number of rules produced when ifferent pruning approaches are used against Reuter data set

Unlike lazy pruning approach, the database coverage method and our proposed methods eliminate the spare rules and that explains its moderate size classifiers. Specifically, MCAR using our proposed methods and MCAR using database coverage algorithms generate reasonable size classifiers if compared with MCAR using lazy pruning method. This enables domain users to benefit from.

Data	No Pruning	HPC	Database Coverage	Lazy pruning
Acq	80.6	99.9	82.6	82.4
Crude	76.2	96.3	80.4	88.4
Earn	99.6	99.7	99.6	99.6
Grain	90.6	96	91.8	94.8
Interest	3.1	69.5	3.1	3.1
Money-FX	49.7	73.7	49.7	49.7
Trade	93.2	94.9	93.2	93.4
Average	70.4	90.0	71.5	73.1

Table 5: Accuracy per class label of the Reuter data derived by the MCAR algorithms using the different rule pruning methods

Table 5 depicts the classification accuracy (%) derived by MCAR algorithm using the different rule pruning methods on the seven most populated categories of Reuters-21578. We also reported the accuracy derived without pruning in column 2. The accuracy numbers have been generated using a minsupp of 2% and a minconf of 40%. Table 5 indicates that when the HPC pruning is employed, the classification accuracies derived are better than those of database coverage and the lazy pruning methods. Particularly, the won-tied-loss records of HPC records against no pruning, database coverage and lazy pruning are 7-0-0, 7-0-0 and 7-0-0, respectively. In fact, we achieved on average +19.6%, +18.5%, +16.9% higher prediction rates within MCAR algorithm than no pruning, the database coverage and lazy pruning, respectively on the Reuter text collection.

5. CONCLUSION

In this paper, we proposed a new rule pruning method within associative classification mining. We conducted experiments on seven categories selected from the Reuters-21578 text collection using the developed pruning procedures and existing pruning methods in associative classification. The bases of the comparison are the number of produced rules and the accuracy, and we implemented all pruning methods within a known associative algorithm called MCAR. The experimental results revealed that the HPC method outperformed all other pruning techniques with reference to predictive accuracy and number of rules generated. Particularly, it achieved on average +19.6%, +18.5%, +16.9% higher prediction rates within MCAR algorithm than no pruning, the database coverage and lazy pruning, respectively. In near future, we intend to expand our research to include other rule pruning heuristics in the areas of decision trees, statistics, and rule induction.

REFERENCES

[1] Antonie M. and Zaïane O. (2004). An associative classifier based on positive and negative rules. Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (pp. 64 - 69), Paris, France.

[2] Antonie M. and Zaiane O. (2002). Text Document Categorization by Term Association, Proceedings of the IEEE International Conference on Data Mining (ICDM '2002), (pp.19-26), Maebashi City, Japan.

[3] Baralis E., Chiusano S. and Garza P. (2008). A Lazy Approach to Associative Classification. IEEE Trans. Knowl. Data Eng. 20(2): 156-171.

[4] Baralis E., Chiusano S. and Garza P. (2004). On support thresholds in associative classification. Proceedings of the 2004 ACM Symposium on Applied Computing, (pp. 553-558). Nicosia, Cyprus.

[5] Baralis E. and Torino P. (2000). A lazy approach to pruning classification rules. Proceedings of the 2002 IEEE ICDM'02, (pp. 35). Maebashi City, Japan.

[6] Cohen W. (1995). Fast effective rule induction. Proceedings of the 12th International Conference on Machine Learning, (pp. 115-123). CA, USA.

[7] Duda R. and Hart P. (1973). Pattern classification and scene analysis. John Wiley & son.

[8] Kundu G., Islam M., Munir S. and Bari M. (2008). ACN: An Associative Classifier with Negative Rules, Computational Science and Engineering, vol. 0, no. 0, (pp. 369-375), 11th IEEE International Conference on Computational Science and Engineering.

[9] Lewis D. (1998). Reuters 21578 text categorisation test collection.http://www.daviddlewis.com/resources/testcollections/reuters21578.

[10] Li B., Li H., Wu M. and Li P. (2008). Multi-label Classification based on Association Rules with Application to Scene Classification, icycs, (pp.36-41), 2008 The 9th International Conference for Young Computer Scientists.
[11] Li W., Han J. and Pei J. (2001). CMAR: Accurate and efficient classification based on multiple-class association rule. Proceedings of the ICDM'01, (pp. 369-376). San Jose, CA.
[12] Liu B., Hsu W. and Ma Y. (1998). Integrating classification and association rule mining. Proceedings of the KDD, (pp. 80-86). New York, NY.
[13] Liu B., Ma Y., Wong, C-K. and Yu. P. (2003). Scoring the data using association rules. Applied Intelligence, 18(2003): 119-135.
[14] Merz C. and Murphy P. (1996). UCI repository of machine learning databases. Irvine, CA, University of California, Department of Information and Computer Science.
[15] Niu Q., Xia S. and Zhang L. (2009). Association Classification Based on Compactness of Rules, wkdd, (pp.245-247), Second International Workshop on Knowledge Discovery and Data Mining.
[16] Quinlan J. (1998). Data mining tools See5 and C5.0. Technical Report, RuleQuest Research.
[17] Quinlan J. (1993). C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann.
[18] Quinlan J. and Cameron-Jones R. (1993). FOIL: A midterm report. Proceedings of the European Conference on Machine Learning, (pp. 3-20), Vienna, Austria.
[19] Quinlan J. (1987). Simplifying decision trees. International journal of man-machine studies, 27(1987): 221-248.
[20] Snedecor W. and Cochran W. (1989). Statistical Methods, Eighth Edition, Iowa State University Press.
[21] Thabtah, F., Cowling, P., and Peng, Y. (2005) MCAR: Multi-class classification based on association rule approach. Proceeding of the 3rd IEEE International Conference on Computer Systems and Applications (pp. 1-7).Cairo, Egypt.
[22] Thabtah, F., Cowling, P., and Peng, Y. (2004) MMAC: A new multi-class, multi-label associative classification approach. Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM '04), (pp. 217-224). Brighton, UK. (Nominated for the Best paper award).
[23] Topor R. and Shen H. (2001). Construct robust rule sets for classification. Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 564-569). Edmonton, Alberta, Canada.
[24] Xu X., Han G. and Min H. (2004). A novel algorithm for associative classification of images blocks. Proceedings of the fourth IEEE International Conference on Computer and Information Technology, (pp. 46-51). Lian, Shiguo, China.
[25] Yin X. and Han J. (2003). CPAR: Classification based on predictive association rule. Proceedings of the SDM (pp. 369-376). San Francisco, CA.
[26] Yoon Y. and Lee G. (2008). Text Categorization Based on Boosting Association Rules, icsc, pp.136-143, 2008 IEEE International Conference on Semantic Computing.