The Impact of Rule Ranking on the Quality of Associative Classifiers

Fadi Thabtah[1], Peter Cowling[2] and Yonghong Peng[3]
[1]Department of Computing and Engineering, University of Huddersfield
Huddersfield, UK
F.Thabtah@hud.ac.uk
[2] MOSAIC Research Centre, [3]Department of Computing, University of Bradford
Bradford, UK
{P.I.Cowling, Y.H.Peng}@bradford.ac.uk

Abstract

Associative classification is a promising approach that utilises association rule mining to build classifiers. Associative classification techniques such as CBA and CMAR rank rules mainly in terms of their confidence, support and cardinality. We propose a rule sorting method that adds more tie breaking conditions than existing methods in order to reduce rule random selection. In particular, our method looks at the class distribution frequency associated with the tied rules and favours those that are associated with the majority class. We compare the impact of the proposed rule ranking method and two other methods presented in associative classification against 12 highly dense classification data sets. Our results indicate the effectiveness of the proposed rule ranking method on the quality of the resulting classifiers for the majority of the benchmark problems, which we consider. In particular, our method improved the accuracy on average +0.62% and +0.40% for the 12 benchmark problems if compared with (support, confidence) and (support, confidence, lower cardinality) rule ranking approaches, respectively. This provides evidence that adding more appropriate constraints to break ties between rules positively affects the predictive power of the resulting associative classifiers.

1. INTRODUCTION

Association rule discovery and classification are analogous tasks in data mining, with the exception that the ultimate goal for classification is the prediction of classes, while association rule discovery describes associations between attribute values in a database. In recent years, association rule mining has been successfully used to build accurate classification models (classifiers), which resulted in a new approach coming to life, known as associative classification (AC) [1 & 2]. Several studies [2, 3, 4 5 & 13] show that the AC approach is able to extract more accurate classifiers than traditional classification techniques, such as decision trees [6], rule induction [7 & 8] and probabilistic approaches [9]. In contrast to rule induction techniques, which greedily and locally derive rules, AC explores the complete training data set and aims to construct a global classifier that can cover as many instances as possible.

Generally, to build an associative classifier, the complete sort of class association rules (CARs) is first extracted from the training data set and one subset is chosen to form the classifier. The selection of such a subset can be accomplished in many ways, for instance in the CBA [2] and L^3 [10] algorithms, the selection of the classifier is done by evaluating the complete set of CARs on the training data and considering rules that cover at least one training data object. On the other hand, the CPAR algorithm [5] uses a greedy method to choose the classifier. Once the classifier is created, its predictive power is then evaluated on test data objects.

Rule preference is known to be an important concept in classification [11]. Given a set of training data objects, the number of potential classification rules that imply these objects is relatively large, and consequently classification algorithms must have a basis for favouring one rule over another. In the AC approach, rule preference during the ranking process of the rules is important since higher ranked rules are often applied more than lower ranked rules in the prediction step. Thus, it is vital to use appropriate conditions to decide which rule is better.

Many AC techniques have been proposed in recent years, such as CMAR [4], CPAR [5], L^3 [10], Negative-Rules [12] and MCAR [13]. These techniques use several different approaches to discover rules, extract rules, rank rules, store rules, prune redundant or "harmful" rules (Those that lead to incorrect classification) and classify new test objects. The goal of this paper is to study the different approaches used by the state-of-the-art AC techniques for rule ranking and compare their effect on the quality of the resulting classifiers. Particularly, we study the effect of rule ranking procedures proposed in two popular AC algorithms (CBA, CMAR) on the accuracy of the derived classifiers. Furthermore, we present a new rule ranking method that adds upon previous approaches by considering the distribution frequencies of class labels in the training data for each rule in order to minimise randomisation.

The rest of the paper is organised as follows: AC and its main concepts are presented in Section 2. Different methods used to discriminate between rules and our proposed rule ranking method, are given in Section 3. Section 4 is devoted to experimental results and finally, conclusions are presented in Section 5.

2. ASSOCIATIVE CLASSIFICATION

2.1 The PROBLEM

AC is a special case of association rule mining in which only the class attribute is considered in the rule's consequent, for example in a rule such as $X \rightarrow Y$, Y must be the class attribute. Let us define the classification problem in an association rule framework. The training data set T has m distinct attributes A_1, A_2, … , A_m and C is a list of class labels. Attributes could be categorical (meaning they take a value from a finite set of possible values) or continuous (where they are real or integer). In the case of categorical attributes, all possible values may be mapped to a set of positive integers. For continuous attributes, any discretisation method can be used.

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

Definition 2: An item can be described as an attribute name A_i and a value a_{ij}.

Definition 3: An itemset can be described as a set of items contained in a training object.

Definition 4: A rule *r* is of the form <itemset, *c*>, where $c \in C$ is the class.

Definition 5: The actual occurrence (*actoccr*) of a rule *r* in *T* is the number of rows in *T* that match the itemset defined in *r*.

Definition 6: The support count (*suppcount*) of a rule *r* is the number of rows in *T* that match *r's* itemset, and belong to *r* 's class.

Definition 7: a rule *r* passes the *minsupp* threshold if (*suppcount*(*r*)/|*T*|) ≥ *minsupp*, where |*T*| is the number of instances in *T*.

Definition 8: a rule *r* passes *minconf* threshold if (*suppcount*(*r*)/*actoccr*(*r*)) ≥ *minconf*.

Definition 9: An itemset *i* that passes the *minsupp* threshold is said to be a frequent itemset.

Definition 10: An actual class association rule is represented in the form: $(A_i, i_1, a_i, i_1) \wedge \ldots \wedge (A_i, i_k, a_i, i_k) \rightarrow c_i$, where the antecedent of the rule is an itemset and the consequent is a class.

A classifier is of the form $H : I \rightarrow Y$, where I is a set of itemsets and Y is the class. The main task of AC is to construct a set of rules (a 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 instance (a, y).

2.2 SOLUTION STRATEGY

Figure 1 shows the general steps used in an AC approach, in which the generation of the CARs is relatively computational expensive because it is similar to the discovery of frequent itemsets in association rule mining, which is a challenging problem [14, 15, 16 & 17]. Methods that find the complete set of frequent itemsets generally find itemsets that are potentially frequent and then work out their frequencies with classes in the training data. Once all frequent itemsets are identified, for each one of them that passes the *minconf* threshold, a rule such as $X \rightarrow C$ is generated, where C is the largest frequency class associated with itemset X in the training data (Step 2).

The problem of generating classification rules is straightforward, given that all frequent itemsets are already identified as no support counting or scanning of the training data are required. In step 3, a selection of an effective subset of rules is accomplished using various methods and finally the quality of the selected subset of rules (the classifier) is measured on an independent test data set to find the accuracy.

Let us explain the discovery of frequent itemsets and the construction of the classifier in AC using an example. Consider the training data shown in Table 1, which represents three attributes A_1 (a_1, b_1, c_1), A_2 (a_2, b_2, c_2) and A_3 (a_3, b_3, c_3) and two class labels (y_1, y_2). Assuming *minsupp* = 20% and *minconf* = 80%, the frequent one, two and three itemsets for Table 1 are shown in Table 2, along with the relevant supports and confidences. In cases where an itemset is associated with multiple classes, only the class with the largest frequency is considered by AC methods. All frequent itemsets in bold in Table 2 pass the *minconf* threshold, and will be converted into potential rules in the classifier.

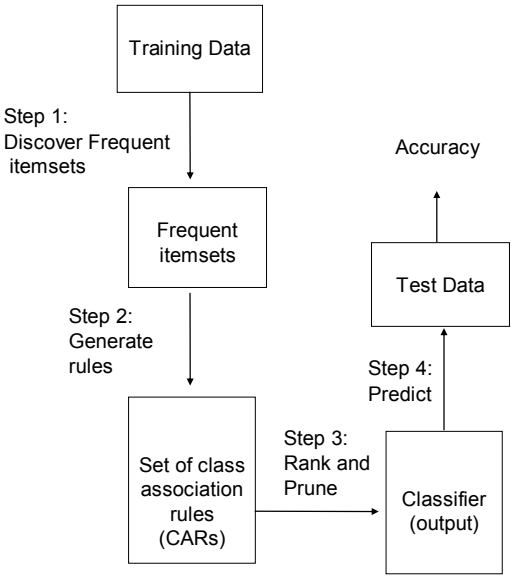

Figure 1 Associative classification steps

Table 1 Training data set

rowids	A_1	A_2	A_3	class
1	a_1	a_2	b_3	y_1
2	a_1	a_2	c_3	y_2
3	a_1	b_2	b_3	y_1
4	a_1	b_2	b_3	y_2
5	b_1	b_2	a_3	y_2
6	b_1	a_2	b_3	y_1
7	a_1	b_2	b_3	y_1
8	a_1	a_2	b_3	y_1
9	c_1	c_2	c_3	y_2
10	a_1	a_2	b_3	y_1

There are many AC algorithms proposed in the literature [2, 3, 4, 10 & 12] where most of which are based on the CBA algorithm [2]. The CBA algorithm uses Apriori candidate generation step [14] to find the rules, where only the subset that leads to the lowest error rate against the training data set is selected to from the classifier. The selection of the subset is accomplished using the database coverage heuristic [2], where each rule is evaluated against the training data in order to determine rules, which cover at least one training data object. The selection of which rule is evaluated first is based on the rule ranking parameters, and therefore, highest order rules are normally evaluated before other potential rules, which give them the opportunity to be used more frequently in predicting test data objects.

Table 2 Potential classifier for Table 1

Frequent *Ruleitems*			
Antecedent	Consequent / class	Supp	Conf
$<a_2, b_3>$	**y_1**	**4/10**	**4/4**
$<a_1, a_2, b_3>$	**y_1**	**3/10**	**3/3**
$<b_3>$	**y_1**	**6/10**	**6/7**
$<a_1, b_3>$	**y_1**	**5/10**	**5/6**
$<a_2>$	**y_1**	**4/10**	**4/5**
$<a_1,a_2>$	y_1	3/10	3/4
$<a_1>$	y_1	5/10	5/7

3. COMMON ASSOCIATIVE CLASSIFICATION RULE RANKING METHODS

Rule ranking before building the classifier plays an important role in the classification process since the majority of AC algorithms such as [2, 4 & 10] utilise rule ranking procedures as the basis for selecting the classifier during pruning. In particular, CBA and CMAR algorithms for example use the database coverage pruning to build their classifiers, where using this pruning, rules are tested according to their ranks. The precedence of the rules is normally determined according to several parameters, including, support, confidence and rule antecedent cardinality. This section highlights the different constraints considered by current algorithms to discriminate between rules in the rule ordering process and also discusses the impact they have on the quality of the resulting classifiers.

3.1 SUPPORT, CONFIDENCE And CARDINALITY METHOD

One of the common rule ranking techniques, which favours rules with large support and confidence values, was introduced in [2], and is shown in Figure 2. The ranking technique employed by CBA considers principally confidence and support to order rules, and when two rules have identical support and confidence, the choice is based on the one generated earlier. This means, CBA selects rules with lower cardinality first since it employs the Apriori step-wise algorithm [14] in its rule generation step. The Apriori algorithm generates rules starting from those that have length 1, then 2 and so on.

Given two rules, r_a and r_b, r_a precedes r_b ($r_a \succ r_b$) if

1. The confidence of r_a is greater than that of r_b.
2. The confidence values of r_a and r_b are the same, but the support of r_a is greater than that of r_b.
3. Confidence and support values of r_a and r_b are the same, but r_a was generated earlier than r_b.

Figure 2 CBA rule ranking method

The CBA sorting method fails to break many ties for highly correlated classification data sets, where the expected number of the produced rules is relatively large. For example, for the "vote" data set downloaded from [18] and using *minsupp* of 2% and *minconf* of 40%, there are 7755 potential rules with identical confidence, of which 6802 have the same support. Also, from the 6802 potential rules with identical confidence and support, there are 5126 that have the same cardinality and only 1113 potential rules from which have different cardinality. These numbers of potential rules have been produced without using any pruning. The remaining rules are ranked randomly if we use the CBA rule sorting method, where many rule ranking decisions may be sub-optimal, reducing the quality of the resulting classifier. Additional tie breaking conditions have the potential to improve classification accuracy over the (support, confidence, cardinality) method.

Another similar rule ranking method to CBA one, which favours specific rules (those with higher antecedent cardinality), was developed in [10]. The main reason for giving specific rules higher ranking than others is to reduce the chance of misclassification and to try specific rules first in the prediction step, then if they fail to cover a test instance, rules with a smaller number of attributes are considered.

The majority of AC algorithms developed after the introduction of CBA, including, [3, 4 & 10] have used the (support, confidence, lower cardinality) ranking method. These algorithms tend to prefer general rules (those with very small numbers of attribute values in their antecedent) since they occur more frequently in the training data. However, such rules may suffer from large error rates. Generally speaking, specific rules (rules with a high cardinality) are supersets of some general rules and cover smaller numbers of training instances. Thus, their chance of misclassification on the training data is usually smaller than that of general rules.

3.2 The PROPOSED RULE RANKING METHOD

Selecting appropriate parameters to favour one rule on another in rule ordering is crucial task since most AC algorithms use rule ranking as the basis to select rules while constructing the classifier. The CBA and CMAR algorithms favour rules principally with reference to confidence, support and lower cardinality. When several rules have identical confidence, support and cardinality, these methods randomly choose one of the rules, which in some cases may degrade accuracy. Since AC approach generates normally large sized classifiers, where rules can be in the order of thousands, so that, there may be several rules with the same support, confidence and cardinality.

Consider for instance the "glass" data set from [18], if it is mined with *minsupp* of 2% and *minconf* of 40% using the CBA algorithm and without using any pruning, there are 759 potential rules that have identical confidence, 624 of these have identical support as well. Also, there are 409 of the 624 potential rules have the same cardinality, leaving no way for CBA or CMAR rule ranking methods to discriminate between them. A more serious case is the "autos" data set, if we mine it using the same support and confidence, there are 2660 potential rules with the same confidence and 2494 of them have identical support too. When the support is lowered further, there may be huge numbers of potential rules with identical support and confidence.

We propose the rule ranking method shown in Figure 3, which adds two tie breaking conditions to the existing methods. Beside confidence, support and cardinality, our method considers the class distribution frequency associated with each rule and favours rules that are associated with the most representative class. For example, if two rules, r_1 and r_2, have the same support, confidence and cardinality, but r_2 is associated with a class label, which has been occurred more frequently in the training data than that of r_1, our method favours r_2 on r_1 in the rule ranking process. In cases where two or more rules have also the same class frequencies, then we select one randomly. Our rule random selection considers the rule's items row ordering in the training data set and prefer rules that have a higher order. We will show in Section 4 the effectiveness of the proposed rule ranking method on the quality of the produced classifiers.

Given two rules, r_a and r_b, r_a precedes r_b ($r_a \rangle r_b$) if:

1. The confidence of r_a is greater than that of r_b.
2. The confidence values of r_a and r_b are the same, but the support of r_a is greater than that of r_b.
3. Confidence and support of r_a and r_b are the same, but r_a has fewer conditions in its left hand side than of r_b.
4. Confidence, support and cardinality of r_a and r_b are the same, but r_a is associated with a a class that occurs more frequently in the training data than that of r_b.
5. All above criteria are identical for r_a and r_b, but r_a generated from rows that occur earlier in the training data than that of r_b.

Figure 3 The proposed rule ranking method

3.3 IMPACT Of RULE RANKING ON CLASSIFIERS

Every rule-based AC technique performs global sorting on the rules in the process of building the classifier. This sorting can be considered a first step toward pruning noisy and redundant rules and explains the reason why these algorithms sort rules before pruning. Sorting aims to give good quality classification rules the chance to be selected first in the process of predicting test data objects, and thus rule ranking can be seen as an important step, which may influence the quality of the resulting classifier. Presented previously in this section, the measures used to discriminate between rules are confidence, support and cardinality. But the question still remains, which rule ranking method is the most effective?

It is the firm belief of the authors that if more effective conditions can be imposed to break ties, random selection will be minimised and better quality classifiers will result. This is because pruning heuristics such as database coverage [2] and lazy pruning [10] consider rules based on their rank when constructing the classifier. Examples demonstrated in Section 3.3 indicate that there is a potential for additional measures to break ties between rules beside support, confidence and cardinality due to the large number of rules extracted using AC approaches. In the next section, we show by experimental results the effectiveness of adding new parameters to discriminate between tied rules on the classification accuracy.

4. EXPERIMENTAL RESULTS

We conducted a number of experiments on 12 highly dense classification data sets from [18] using stratified ten-fold cross validation [19]. The impact of three rule ranking methods: (confidence, support), (confidence, support, lower cardinality) [2 & 4] and our proposed method on the quality of the resulting classifiers from the 12 benchmark problems have been compared. The choice of such rule ranking methods is based on their wide use in AC. We have implemented the three methods in Java within a recent proposed AC algorithm, MCAR [13]. Following experiments conducted in [2, 12 & 13] the *minsupp* was set to 2%. The confidence threshold, on the other hand, has a smaller impact on the behaviour of any AC method, and it

has been set in our experiments to 40%. All experiments were conducted on Pentium IV 1.7 Ghz, 256 MB RAM machine using Java under Windows XP.

To investigate the behaviour of the rule ranking methods, Table 3 shows the number of times each condition does not break tie between rules for the 12 classification problems. Column 2 indicates the number of potential rules with similar confidence, column 3 represents the number of potential rules with the same confidence and support. Column 4 shows the number of potential rules that have identical confidence, support and cardinality and column 5 represents rules from column 4 that share the same class frequency. Column "RowOrd " indicates the number of times the rule's items row ordering condition has been used after trying (support, confidence, cardinality, class frequency). We have used a *minsupp* of 2% and a *minconf* of 40% to produce the potential rules in Table 3.

Values shown in Table 3 represent the potential rules tested by the MCAR algorithm during the ranking process and before building the classifier or performing any pruning and this explains their large numbers. Table 3 shows that support and confidence are not effective in distinguishing between rules in most benchmark problems, we consider. For the "Cleve" data set for instance, there are 17092 potential rules with the same confidence as some other rule, with 16289 rules having identical confidence and support. There are 14647 with the same confidence, support and cardinality as some other rule, where 12942 from the 14647 are associated with classes that have the same frequency in the training data set. The frequent use of the additional conditions in the process of the rule ranking especially class frequency parameter suggests that more discrimination between potential rules positively increases the accuracy of the classifiers.

To

Table 3 Number of times each condition in the proposed rule ranking method does not break tie between potential rules

Data	No. of Rules with the same Conf.	No. of Rules with the same Conf. & Supp.	No. of Rules with the same Conf., Supp. & Cardinality	No. of Rules with the same Conf., Supp., Cardinality &Class Freq.	RowOrd
Autos	2660	2492	2117	1683	181
Glass	759	624	409	245	7
Lymph	11019	10775	10217	9595	2381
Cleve	17092	16289	14647	12942	469
Tic-tac	2297	2047	1796	1541	278
Diabetes	252	91	15	3	0
Breast	3471	2980	2217	1643	75
Vote	7755	6802	5126	4013	207
Heart	4791	4267	3383	2562	145
wine	31012	30486	29730	4500	0
weather	96	85	75	61	37
Pima	252	91	15	3	0

Table 4 Impact of the three rule ranking methods on the accuracy

Data	(Supp, Conf)	(Supp, Conf, Lower cardinality)	The proposed method
Cleve	**82.44**	82.16	81.35
Breast-w	94.61	95.11	**96.32**
Diabetes	76.90	77.05	**77.18**
Glass	67.76	68.79	**69.97**
Pima	77.16	**77.34**	77.11
Tic-Tac	99.76	99.77	**100.00**
Led7	70.95	**71.07**	71.00
Heart-s	81.30	81.87	**82.14**
Lymph	**79.10**	77.13	78.57
Vote	**88.86**	88.17	87.70
zoo	95.38	97.73	**97.78**
Contact-lenses	72.93	73.54	**75.54**
Average	82.26	82.48	**82.88**

show the effectiveness of the rule sorting method on the quality of the classifiers, we conduct a number of experiments to compare the impact on accuracy of the proposed rule ranking method

and two other methods, which are (support, confidence) and (support, confidence, lower cardinality). Each value shown in Table 4 represents an overage over ten cross validation runs each using a different random seed when partitioning the training data set. The figures show a slight improvement of our rule ranking method over that of (support, confidence) and (support, confidence, lower cardinality). In particular, our method achieved on average +0.62% and +0.40% improvements with regards to accuracy on the 12 benchmark problems over (support, confidence) and (support, confidence, lower cardinality) rule ranking approaches, respectively. It appears that the additional constraints imposed to break ties slightly improve the predictive power of the resulting classifiers over the test data sets.

5. CONCLUSIONS

In this paper, the problem of rule ranking in associative classification has been investigated. Particularly, we propose a rule ranking technique, which expands previous rule ranking methods by looking at the class distribution frequencies to discriminate between tied rules. In addition, we use a rule random selection that looks at the rule items ordering in the training data and favour rules that are associated with a higher order. These tie breaking conditions together with existing support, confidence and cardinality approaches have been used to minimise the need for random selection. Empirical Evaluations using 12 highly correlated classification data sets from *Weka* data collection revealed that adding more constraints to discriminate between rules improved the accuracy of the resulting classifiers. Our proposed rule ranking method improved the accuracy for the 12 classification data sets on average +0.62% and +0.40% over (support, confidence) and (support, confidence, lower cardinality) rule ranking approaches, respectively. Moreover, the results show that our additional parameters for breaking ties are used often.

Reference

[1] Agrawal, R. & Srikant, R. Fast algorithms for mining association rule. *Proceedings of the 20th International Conference on Very Large Data Bases* (pp. 487-499), 1994.

[2] Ali, K., Manganaris, S. & Srikant, R. Partial classification using association rules. In Heckerman, D., Mannila, H., Pregibon, D., and Uthurusamy, R., (eds.). *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, (pp. 115-118.), 1997.

[3] Antonie, M. & Zaïane, O. 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, 2004.

[4] Baralis, E. & Torino, P. A lazy approach to pruning classification rules. *Proceedings of the* ICDM'02, (pp. 35), 2000.

[5] Cendrowska, J. PRISM: An algorithm for inducing modular rules. *International Journal of Man-Machine Studies*. Vol.27, No.4, pp.349-370, 1987.

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

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

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

[9] Lim, T., Loh, W. & Shih, Y. A comparison of prediction accuracy, complexity and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40(3): 203--228, 2000.

[10] Lin, J. & Dunham, M. Mining Association Rules: Anti-Skew Algorithms, *Proceedings of the Fourteenth International Conference on Data Engineering*, (pp. 486-493), 1998.
[11] Liu, B., Hsu, W. & Ma, Y. Integrating Classification and association rule mining. *Proceedings of the KDD* (pp. 80-86). New York, NY, 1998.
[12] Michaliski, R. A theory and methodology of inductive learning. Artificial Intelligence 20, 1983, 111-161, 1983.
[13] Park, J., Chen, M., and Yu, P. An Effective Hash-Based Algorithm for Mining Association Rules. *Proceedings of the ACM SIGMOD*, (pp. 175-186). San Jose, CA, 1995.
[14] Quinlan, J. C4.5: *Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.
[15] Thabtah, F., Cowling, P. & Peng, Y. 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, 2005.
[16] Witten, I. & Frank, E. *Data mining: practical Machine learning tools and techniques with Java implementations*. San Francisco: Morgan Kaufmann, 2000.
[17] Yin, X. & Han, J. CPAR: Classification based on predictive association rule. *Proceedings of the SDM (*pp. 369-376). San Francisco, CA, 2003.
[18] Wang, K., Zhou, S. & He, Y. *Growing Decision Tree on Support-less Association Rules*. Proceedings *of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, Boston, Massachusetts, pp. (265 – 269), 2000.
[19] WEKA : Data Mining Software in Java: http://www.cs.waikato.ac.nz/ml/weka.