Applied Computing and Informatics (2015) **11**, 29–46

Saudi Computer Society, King Saud University

Applied Computing and Informatics

(http://computer.org.sa)
www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

Multi-label rules for phishing classification

Neda Abdelhamid *

Computing and Informatics Department, De Montfort University, Leicester, UK

Received 3 January 2014; revised 13 May 2014; accepted 3 July 2014
Available online 15 July 2014

KEYWORDS

Associative rule;
Associative classification;
Data mining;
Website phishing;
On-line security

Abstract Generating multi-label rules in associative classification (AC) from single label data sets is considered a challenging task making the number of existing algorithms for this task rare. Current AC algorithms produce only the largest frequency class connected with a rule in the training data set and discard all other classes even though these classes have data representation with the rule's body. In this paper, we deal with the above problem by proposing an AC algorithm called Enhanced Multi-label Classifiers based Associative Classification (eMCAC). This algorithm discovers rules associated with a set of classes from single label data that other current AC algorithms are unable to induce. Furthermore, eMCAC minimises the number of extracted rules using a classifier building method. The proposed algorithm has been tested on a real world application data set related to website phishing and the results reveal that eMCAC's accuracy is highly competitive if contrasted with other known AC and classic classification algorithms in data mining. Lastly, the experimental results show that our algorithm is able to derive new rules from the phishing data sets that end-users can exploit in decision making.

* Tel.: +44 (0)116 2 50 60 70.
E-mail address: P09050665@myemail.dmu.ac.uk.
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.aci.2014.07.002

1. Introduction

Generally and according to (Tsoumakas and Katakis, 2007), there are two types of classification problems, these are termed as single label and multi-label. In a single label classification, each training case in the input data is associated with only one class. In cases where the input data set contains just two class labels, the problem is called binary classification. However, if more than two classes are available, the problem is named multi-class classification. The majority of the research works conducted in classification data mining are concerned with single label classification, i.e. (Li et al., 2008; Chien and chen, 2010; Wang et al., 2011). However, domain applications like medical diagnoses, website phishing detection, text categorisation (TC) and bioinformatics may necessitate the production of multiple label rules. This is since there is a class overlapping among the training cases in these applications data. Meaning a set of attribute values in the rule's body may link with more than one class in the data set and thus producing all these classes in the rule's consequent (right hand side). It is advantageous to generate the other classes besides the largest frequency class with the rule's body since they contain valuable information having a sufficient representation in the data set.

In the last few years, a learning strategy which applies the association rule in classification data called associative classification (AC) emerged (Thabtah et al., 2010; Thabtah et al., 2011; Wang et al., 2011). Most AC algorithms like MAC (Abdelhamid et al., 2012), CMAR (Li et al., 2001) and others usually apply an association rule technique to discover the rules, and then filter out the rules to include only those which their consequent is the class attribute. Experimental research works (Jabbar et al., 2013; Jabez, 2011) indicated that AC algorithms frequently build more accurate classifiers than classic classification approaches such as the probabilistic approach (Witten and Frank, 2002), decision tree (Quinlan, 1993) and rule induction (Cohen, 1995). The algorithm proposed in this article is part of the AC family.

Limited research attempts in AC have been conducted to produce rules with more than one class, i.e. Lazy AC (CLAC) (Veloso et al., 2011) and Multi-label Multi-class AC (MMAC) (Thabtah et al., 2004). The rest of the existing AC algorithms is unable to deal with discovering multi-label rules from single label data sets, and normally derive only the largest frequency class connected with the attribute value(s) and ignore all other classes even if these classes have large frequencies with the attribute value(s). For instance, this condition occurs if an attribute value such as $<A>$ is associated with two class labels (cl_1, cl_2) in different places (examples) within the training data set with frequencies equal to 44 and 45 respectively. A typical AC algorithm like CBA will only take on class "cl_2" simply because it has a larger frequency than cl_1 with $<A>$ and ignores class cl_1 even if this class is statistically significant with $<A>$. This surely makes the selection of ($<A>$, cl_2) questionable. In the proposed algorithm we pick the two class labels and construct a multi-label rule rather than removing class cl_1. This enables

the decision maker to obtain knowledge missed by current AC algorithms. The primary motivation of this paper is to deal with the problem of generating multi-label rules via AC from single label data sets. In other words, we intend to discover all class labels associated with the attribute values bringing up novel and useful knowledge normally missed by current algorithms.

In this paper, a new multi-label rule based AC called Enhanced Multi-label Classifiers based Associative Classification (eMCAC) is proposed. This algorithm extracts from data sets not only rules with the most obvious class but rules that are associated with a ranked set of class labels. When an attribute value in a training data set is connected to more than one class in different locations with certain frequencies, the proposed algorithm extracts and sorts all of them in the rule consequent according to their frequencies. Thus, later in the prediction step, there can be more alternatives (classes) when the rule is used in predicting the class for a test case.

The proposed algorithm generates rules from the complete training data set and without performing recursive learning as the MMAC algorithm, which requires learning from parts of the training data set. This means MMAC ends up with several single label classifiers that are then merged in a separate step to make the final classifier. Another main distinction between eMCAC and MMAC is that our algorithm's way of computing the confidence and the support for a multi-label rule is based on the average confidence and support values of all (Items, Classes) contained within the rule, whereas, MMAC assigns the top ranked class confidence and support to the multi-label rule.

The proposed algorithm employs a rule pruning method that considers a rule part of the classifier if its body is contained within the training example. This is done without considering the class similarity of both the evaluated rule and the training case, thus ensuring a high rule coverage with respect to the training cases and consequently a smaller number of extracted rules. Section 4 demonstrates the applicability of the proposed algorithm on real world application data related to phishing that was collected from phishy and legitimate websites (www.phishtank.com) (www.millersmiles.co.uk).

The AC problem, its related basic concepts and relevant literature are given in Section 2. Section 3 surveys common approaches in the literature and the proposed algorithm is presented in Section 4. Experimentations and result analysis are demonstrated in Section 5, and lastly conclusions are given in Section 6.

2. The problem

The general description of the problem besides main definitions has been summarised by (Abdelhamid et al., 2012).

Let T denote the domain of the training cases and C be a list of classes. Each case $t \varepsilon T$ may be given a class $c1, c2, \ldots, c_k$ for $c_i \varepsilon C$, and is represented as a pair $(t, (c_k))$ where c_k is a class from C associated with the case t in the training data.

Let H denote the set of classifiers for $T \rightarrow C$ where each case $t \varepsilon T$ is given a class and the goal is to find a classifier $h \varepsilon H$ that maximises the probability that $h(t) = c$ for each test case. In our algorithm, and in case of multi-label data we assume that it is transformed to a single label data format after applying the copy transformation method (Section 3.1). The proposed algorithm deals with the single label data format only. The multi-label data set is displayed in Table 1. Table 2 denotes the data obtained after applying the copy transformation method in Table 1 and before the mining process starts.

The main definitions related to the AC problem that the proposed algorithm utilises are given below:

For the training data set T with m attributes $A_1, A_2, \ldots, A_m$ and Cl is a set of classes,

Definition 1: An attribute value set (*AttValSet*) 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 2: A rule r is of the form $< AttValSet, cl >$, where $c \varepsilon C$ is the class.

Definition 3: The actual occurrence (*ActOccr*) of r in T is the number of cases in T that match r's antecedent.

Definition 4: The support count (*SuppCount*) of r is the number of cases in T that matches r's antecedent, and belong to a class cl_i.

Definition 5: A rule r passes the user minimum support threshold (*MinSupp*) if for r, the $SuppCount(r)/|T| \geqslant MinSupp$, where $|T|$ is the number of cases in T.

Definition 6: A rule r passes the user minimum confidence threshold (*MinConf*) if $SuppCount(r)/ActOccr(r) \geqslant MinConf$.

Definition 7: A single label rule is represented as: $Antecedent \rightarrow c$, where antecedent is an *AttributeValueSet* and the consequence is a class.

Definition 8: A multi-label rule is represented as: $Antecedent \rightarrow c_{i1} \vee c_{i2} \ldots c_{i3}$, where antecedent is an *AttributeValueSet* and the consequence is a set of class labels.

Table 1 Sample of a multi-label data set.

TID	$Attribute_1$	$Attribute_2$	$Attribute_3$	Class label
1	y	z	b	cl_1, cl_2
2	x	a	b	cl_l, cl_3
3	y	a	d	cl_1
4	x	a	d	cl_3
5	y	a	b	cl_3, cl_1
6	y	a	d	cl_1, cl_2, cl_3
7	x	k	b	cl_3
8	x	k	d	cl_3, cl_1
9	x	k	b	cl_2, cl_3

Table 2 The transformed multi-label data of Table 1 after processing.

$Attribute_1$	$Attribute_2$	$Attribute_3$	Class label
y	z	b	cl_1
y	z	b	cl_2
x	a	b	cl_1
x	a	b	cl_3
y	a	d	cl_1
x	a	d	cl_3
y	a	b	cl_3
y	a	b	cl_1
y	a	d	cl_1
y	a	d	cl_2
y	a	d	cl_3
x	k	b	cl_3
x	k	d	cl_3
x	k	d	cl_1
x	k	b	cl_2
x	k	b	cl_3

3. Literature review

3.1. Data transformation (optional)

Several data transformation methods exist in the literature to convert multi-label data into one or more single label data. To demonstrate these data transformation methods we use the data set of Table 1 which consists of nine training cases that belong to the following class set $\{cl_1, cl_2, cl_3\}$. We summarise some of the common methods from (Tsoumakas and Katakis, 2007) and use our own example to further simplify them.

The first data transformation method simply removes any multi-label case from the training data set. Therefore, from Table 1, cases located in TID (1,2,5,6,8,9) are discarded. Another data transformation method selects one class of each case either arbitrarily or subjectively by the domain expert. So from Table 1 a single associated class for each of the multi-label cases may be picked. Another more realistic method transforms every multi-label case into a single label one by replacing the multi-label case (x_i, Y_i) with $|Y_i|$ cases. After that a number of methods could be applied such as copy-weight which associates a weight of $(1/|Y_i|)$ to each of the transformed cases. Table 2 shows the copy transformation method after applying it against Table 1, which our algorithm employs when the input data are multi-label .

Finally, a common data transformation method used in image classification that derives a single label binary classifier for every class in the class set is called Binary Relevance (BR) (Boutell et al., 2003). It transforms the original multi-label data set into $|L|$ data sets which contain all the cases. This method gives a positive indicator for a class if it is associated with a case in the training data set and a

negative indicator otherwise. For classifying test data, BR outputs the union of all class labels that are predicted by the |L| classifiers.

3.2. Related works

The majority of existing AC mining algorithms use rules learnt from the training data set for constructing a single label classifier which in turn is utilised for predicting test data. Thus, there are limited numbers of research articles related to multi-label rules in AC. Hereunder, we shed the light on two approaches and other techniques related to traditional multi-label classification in data mining. It is worth to note that the traditional classification algorithms surveyed in this section are related to multi-label data sets and they assume each training example to be associated with more than one class. This is unlike the proposed algorithm that assumes each training example to be linked with a single class but produces multi-label rules.

Veloso et al. (2011) proposed a multiple label AC algorithm that adopts the lazy classification approach in which it delays the reasoning process until test data are given. Unlike binary classification which does not consider the correlation among classes, the lazy algorithm takes into account class relationships. Furthermore, it deals with the small disjuncts (rules that cover limited number of training data), this may reduce classification accuracy. This lazy approach has been compared with BoosTexter (Schapire and Singer, 2000) on three medium size data sets from "http://portal.acm.org/dl.cfm" with respect to error rate. The results produced show that this method is competitive to BoosTexter.

Another AC algorithm called MMAC was proposed to find multiple label rules from single label data sets. It has been reported that MMAC was able to generate higher quality classifiers than CBA and decision trees on a number of UCI data sets in regard to classification accuracy. One obvious limitation of the MMAC is that the classifier produced is extracted from parts of the training data set and the requirement of the recursive phase to find the multi-label rules.

Wang et al. (20110 proposed an enhanced Emerging Pattern (EP) algorithm called ADA that constructs rules from both the input training data set as well as the classified resources such as text documents. ADA classifier gets amended on the fly after the classified resources reach a certain amount. The authors have updated the classifier by refining the newly discovered knowledge using the existing rules. Moreover, ADA uses the maximum entropy approach to classify test cases where multiple rules that are applicable to the test case contribute to the prediction decision. Overall, ADA can be considered a semi-incremental algorithm since few training examples or users set of frequent patterns (keywords) are only necessary to build the classifier instead of the complete training examples. Then, the classified examples as well as the rules are employed to update the classifier by adding or removing rules. Limited experiments on four data sets from the UCI data repository (Merz and Murphy, 1996) have been conducted using

ADA, CBA (Liu et al., 1998), CMAR (Li et al., 2001) and C4.5 (Quinlan, 1993). The results showed similarity on the classification accuracy performance of the AC algorithms and superiority over the decision tree approach (C4.5).

In image classification, pictures may belong to multi-labels, i.e. different objects within a view. This problem is called class overlapping where a scene may contain multiple labels. A scene classification method called cross training was developed in (Boutell et al., 2003). This method trains on each available label in an image in turn during the training step in order to consider all available labels. The results produced reveal that the developed scene classification algorithm performs well with respect to classification accuracy.

Tsoumakas and Katakis (2007) surveyed common learning approaches related to multi-label classification in several domains. The authors have firstly presented the multi-label data transformation methods used in the literature, and then surveyed the different multiple label learning approaches including the adaptation and the transformation methods. Experimentation using three different data transformation methods and various learning algorithms with respect to different evaluation measures, has been conducted. The accuracy results of the compared algorithms revealed that the ''PT3'' transformation method (considers each different set of labels that exist in the multi-label data collection as a single label) when used with a learning algorithm generates the highest results in each of the considered data collections.

A kernel based machine learning algorithm from (Quaresma and Rodrigues, 2003) was applied to the problem of categorising legal documents written in Portuguese for the general attorney office. This problem is considered multi-label since a legal document may belong to more than one category. The authors have utilised vector representation based on bag-of-word for document's representation without the usage of semantic and syntactic information. Moreover, each document was processed by applying lemmatisation and stopword elimination. In experimentation, only the top five categories in regard to frequency documents were used and a number of machine learning algorithms including SVM, C4.5 and Naïve Bayes have been contrasted with respect to prediction accuracy and Information Retrieval (IR) measures (Precision, Recall and F1). The results indicated that SVM and C4.5 are more likely to be applicable to the classification of the attorney general documents since they scored the highest prediction rate. The Naïve Bayes scored a low prediction rate and therefore it was excluded from further experiments. The authors have run SVM and C4.5 after reducing the document collection by removing words that had a frequency in less than a specific number of documents, and they used words that appeared at least in 55 documents. The results on reduced document collection showed a slight superiority of C4.5 over SVM with respect to IR measures, i.e. precision, recall, and F1.

4. The eMCAC algorithm

Our algorithm consists of three main steps: Rule discovery, classifier building and class assignment. In the first step, eMCAC iterates over the training data set in which rules are found and extracted. In step (2), redundant rules are discarded which means that rules do not have training data coverage. The outcome of the second step is the classifier which contains rules. The last step involves testing the classifier on the test data set to measure its predictive rate. Details on eMCAC steps are given in the subsequent sections. The proposed algorithm assumes that the input attributes in the training data set are categorical (having distinct values), and for each of these attributes, all possible values are mapped to a set of positive integers. For continuous attributes any discretisation method can be employed before the training phase.

4.1. Pre-processing data

This step is optional and only required when the input data are multi-label. In this case, we copy each training example with each of its connected class labels. So, if there is a training example linked with two classes, this example is repeated with each class. One of the reasons behind our selection of this method is that we would like to treat each class inside the multi-label example equally. This is since class labels are not sorted in the first place within the multi-label training data set for each case and therefore we do not have prior knowledge on the best class for each training example. Later, when the classifier is constructed we will be able to sort classes within each rule generated based on their frequency with the rule's attribute values in the training data set after the transformation. Another reason for selecting this data format method is that we do not want to lose any knowledge that might be useful to the decision maker by ignoring class labels particularly when other data transformation methods are chosen including "largest frequency class", "class random selection", ignore multi-label case", etc.

4.2. Data representation

In the last few years, some scholars (Abdelhamid et al., 2012; Thabtah et al., 2005) have developed AC algorithms which employ vertical data layout. A training data set in the vertical data layout consists of a group of attribute values, where each attribute value is followed by its locations (Tids) in the training data set. Table 3 shows the vertical layout of attribute values "y", "z", and "b" from Table 1. A number of research studies (Thabtah, 2007; Abdelhamid et al., 2012) revealed that the vertical data format is more effective for representing data than the horizontal format since it makes the process of identifying frequent attribute values efficient specifically the task involving the support counting. This is simply because vertical algorithms use simple TIDs intersection among attribute values to accomplish the

Table 3 Sample of vertical data layout for 3 attribute values.

"y"	"z"	"b"
1	1	1
3		2
5		5
6		7
		9

task of support counting. There are few AC algorithms that employ the vertical data layout for mining classification rules such as MAC and MMAC.

In the proposed algorithm, the vertical data format is used to represent the training data set before frequent attribute discovery and rule generation processes begin.

4.3. Rule discovery methodology

In this section, we describe the process of finding and generating rules.

4.3.1. Frequent attribute values discovery

The eMCAC algorithm uses a rule discovery method that utilises fast intersection among attribute values TIDs to discover the rules. The TID of an attribute value holds the locations (row Ids) that contain the attribute values and its associated class labels in the training data set. The learning method of the proposed algorithm discovers the frequent attribute value of size 1 (F1) after scanning the training data set once. In particular, for each attribute value linked with a class, its support is computed from its TIDs list in which the size of the subset of the TIDs list that is associated with the largest frequency class of an attribute value divided by the size of the training data set denotes the attribute value support. In cases where an attribute value is connected with more than one class it will end up with more than one support. This ensures the production of the multiple label rules since the algorithm allows an attribute value to be associated with multiple labels as long as they are frequent.

When F1 is generated, the algorithm simply intersects the TIDs of the disjoint attribute values in F1 to discover the candidate attribute values of size 2, and after finding F2, the possible remaining frequent attribute values of size 3 are obtained from intersecting the TIDs of the disjoint attribute values of F2, and so forth. Since this frequent ruleitems discovery approach iterates over the training data set once, it is highly effective according to several experimental studies in the literature of data mining community especially with regard to processing time and memory usage. More details on the advantage of vertical algorithms over traditional ones are given in (Thabtah, 2007).

4.3.2. Rule production

When frequent attribute values are identified, eMCAC generates any one as a rule when it passes the *MinConf* threshold. This is accomplished in a straightforward manner since all necessary information for calculating the rules confidence values are stored in the attribute value TIDs. Any frequent attribute value that holds a confidence value smaller than the *MinConf* gets discarded.

For any attribute value connected with many classes and which becomes frequent, eMCAC generates a multi-label rule for it when it passes the *MinConf* threshold. For example, the attribute value $<$a$>$ of Table 2 is linked with two class labels, e.g. (cl_1, cl_3) 4 times each in the training data set. Assume that the *MinSupp* and *MinConf* are set to 4/15 and 40% respectively. This means ($<$a$>$, cl_1), and ($<$a$>$, cl_3) have higher support and confidence than the *MinSupp* and *MinConf* thresholds and therefore two rules can be produced in this case: $a \rightarrow cl_3$, and $a \rightarrow cl_1$. For this example, a typical AC algorithm such as CBA only derives the rule that has higher coverage in the training data, meaning any of the above single label rules can be produced. On the other hand, the proposed algorithm does not discard any useful knowledge and for the above example it produces a multi-label rule R: $a \rightarrow cl_1 \vee cl_3$, where normally class labels are ranked based on their count with the attribute values. In the above example the class rank within the rule is random since cl_1 *and* cl_3 have the same count when linked with $<$a$>$ in the training data set.

4.4. Classifier construction

Once the complete set of rules is derived a rule sorting procedure is invoked to ensure that rules with high confidence and support values are given higher priority to be selected during building the classifier. The rule sorting procedure utilised considers different criteria to favour among rules. The criteria order is: rule's confidence, support, length and class frequency. This ordering of rules has been used since it reduces rule random selection in the prediction step when no rules are found to be applicable to the test case which positively affects the classification accuracy of the classifier.

After rules are sorted from which a subset gets chosen to comprise the classifier. Precisely, and for each training case, eMCAC iterates over the complete set of rules discovered (top-down fashion) and marks the first rule that corresponds to the training case to be part of the classifier. A rule gets inputted into the classifier if it covers at least a single training case. The rule coverage does not necessitate the similarity between the rule's class and that of the training case. This results often in more training coverage for each rule since all training data belonging to the rule body are removed during evaluation. This surely reduces overfitting and usually ends up with less number of rules. The same process is repeated until all training

cases are removed (covered) or all rules have been tested. Finally, the algorithm derives the classifier.

Any remaining unmarked rules are discarded by the proposed algorithm since some higher ranked rules have covered their training cases during building the classifier and therefore these unmarked rules become redundant. In cases when there are unclassified cases remaining in the training data set, a default rule for the largest count class linked with the unclassified cases is formed.

4.5. Class assignment of test data

The proposed algorithm fires the first sorted rule in the classifier applicable to the test case and assigns its class to the test case. The rules attribute values must be contained in the test case in order to be chosen for classifying the test case class. When there is no rule fully applicable to the test case then we take on the first rule that partly matches the test case attribute value. Unlike the majority of current prediction procedures in AC mining that takes on the default class when no rules are applicable to the test case our prediction procedure minimises the utilisation of the default rule in class assignment process of test cases which normally improves upon the resulting classifier performance. This is since default rule has been created with high error from the remaining unclassified training data cases while

Table 4 Phishing data selected features.

Feature	Description
IP address	If IP address exists in URL → Phishy else → Legit
Long URL	If URL length < 54 → Legit URL length ⩾ 54 and ⩽ 75 → Suspicious else → Phishy
URL's having @ symbol	If URL has '@' → Phishy else Legit
Adding prefix or suffix	If domain part has '–' → Phishy else → Legit
Misuse of HTTPs protocol	Use of https & trusted issuer & age ⩾ 2 years → Legit using https & issuer is not trusted → Suspicious else → Phishy
Request URL	Request URL < 22% → Legit request URL ⩾ 22% and < 61% → Suspicious else → Phishy
URL of Anchor	URL anchor % < 31% → Legit URL anchor % ⩾ and ⩽ 67% → Suspicious else → Phishy
Server form handler	SFH If 'about:blank' or empty → Phishy SHD redirects to different domain → Suspicious else → Legit
Abnormal URL	No hostname in URL → Phishy else → Legit
Using pop-up window	Right click disabled → Phishy rightClick showing alert → Suspicious else → Legit
Redirect page	Redirect page #s ⩽ 1 → legit redirect page #s > 1 and < 4 → Suspicious else → phishy
DNS record	No DNS record → Phishy else → Legit
Hiding the links	Change of status bar onMouseOver → Phishy no Change → Suspicious else → Legit
Website traffic	webtraffic < 150,000 → Legit webTraffic > 150,000 → Suspicious else → Phishy
Age of domain	Age ⩾ 6 months → Legit else → Phishy

building the classifier and reducing its usage in the prediction step is a definite advantage.

5. Experimental results

In this section, we conduct experimentations on a number classification data set related to website phishing to evaluate the eMCAC algorithm performance. The main evaluation measures used in the experiments are prediction rate (Accuracy), (label-weight, any label) (defined later in this section), and classifier size (number of rules). A number of algorithms have been contrasted with the proposed algorithm with respect to the above mentioned evaluation measures. A number of features (16) related to the process of identifying the type of websites have been collected from different phishing sources, e.g. Phishtank (www.phishtank.com), yahoo directory and Millersmiles archive (www.millersmiles.co.uk). The different features contributing to the classification of the type of the websites have been adopted from (Mohammad et al., 2012). The reason for choosing these features is since the authors of (Mohammad et al., 2012) have applied frequency analysis on over 5000 collected websites (training examples) containing over 27 features as a feature selection metric and found out that 16 features have significant impact on the process of phishing detection. We have collected 1350 websites, details on the data set features are displayed in Table 4. One distinguishing difference between the data we have collected and other scholars is that we have included "Suspicious" (Phishy or Legitimate) class label which converted the phishing problem from binary classification to a multi-label classification. This made the problem harder and more realistic since we have considered the overlapping among the class labels.

5.1. Settings

In all experiments, tenfold cross validation testing method has been employed for fair evaluation of the classifiers derived by the algorithms considered and to reduce overfitting. Furthermore, six dissimilar classification algorithms which utilise a variety of rule learning methodologies have been considered for contrasting purposes with the eMCAC. These algorithms are MMAC (Thabtah et al., 2004), CBA (Liu et al., 1998), PART (Frank and Witten, 1998), MCAR (Thabtah et al., 2005), RIPPER (Cohen, 1995) and C4.5 (Quinlan, 1993).

The experiments were conducted on an I3 machine with 2.3 Ghz. The experiments of C4.5, PART and RIPPER were carried out in Weka software (Witten and Frank, 2002). For AC algorithms, we have selected CBA, and MCAR for single label classifier comparison and MMAC for multi-label classifier. CBA and MMAC source code has been obtained from their prospective authors and the proposed algorithm and MCAR were implemented in Java.

Finally, we have set the *MinSupp* and *MinConf* thresholds for the AC algorithms (CBA, MCAR, MMAC, eMCAC) to 2% and 50% respectively for all experiments. The main reason for giving the *MinSupp* 2% is that previous research works, e.g. (Thabtah et al., 2005), have suggested that *MinSupp* values ranging between 2% and 5% may balance between the number of rules generated and the predictive accuracy of the classifier. On the other hand and for the *MinConf* parameter, it has been set to 40% since it has minor effect on the performance of the classifiers.

5.2. Data collection

Phishing features can be extracted in a number of ways one of which is manual extraction where users derive features and judge their legitimacy. In this method, users have to spend a lot of time studying the up-to-date phishing collection techniques which is an infeasible approach for the majority of the users. The second method employed in extracting phishing features is the automatic extraction. This is accomplished by examining the webpage and extracting a set of patterns related to phishing and legitimate type web pages. This involves examining the webpage properties and all its features. Webpage properties are typically derived from HTML tags, URL address and Javascript source code.

To conduct our experiments a set of phishing websites were collected from Phishtank archive (www.phishtank.com), which is a free community site where users can submit, verify, track and share phishing data. In addition, we utilised the Millersmiles archive (www.millersmiles.co.uk), which is considered a prime source of information about spoof emails and phishing scams. The legitimate websites were collected from the yahoo directory and starting point directory using a PHP script. The PHP script was plugged with a browser and it collected 548 legitimate websites out of 1353 websites. There is 702 phishing URLs, and 103 suspicious URLs. Sample of the phishing data (10 examples) for all features is shown in Table 4. Some of the collected features hold categorical values termed as ''Legitimate'', ''Suspicious'' and ''Phishy'', these values have been replaced with numerical values 1, 0 and −1 instead of ''Legitimate'', ''Suspicious'' and ''Phishy'' respectively.

5.3. Phishing results analysis

Fig. 1 summarises the prediction accuracy produced by the considered algorithms for the phishing problem data. It is obvious from Fig. 1 that the eMCAC algorithm outperformed the other AC algorithms and the traditional one in predicting the type of the websites. In particular, the eMCAC algorithm outperformed RIPPER, C4.5, PART, CBA, and MCAR by 1.86%, 1.24%, 4.46%, 2.56%, 0.8% respectively. Overall, the prediction accuracy obtained from all algorithms is considered acceptable and that reflects goodness of our features in predicting the website class. It

should be noted that eMCAC accuracy was calculated using label-weight (defined below) and only the largest frequency class is utilised for computing the accuracy for the remaining algorithms since they are single label ones.

One main reason for achieving higher predictive accuracy by the eMCAC algorithm is its ability not only to extract one class per rule but also all possible class labels in the form of a disjunctive multiple label rule. This extra useful knowledge is usually missed by the majority of existing AC algorithms and can contribute positively in predictive power as well as serve the need for the end-user. This can be clearly obvious in real world applications such as website phishing. Fig. 2 lists the number of rules generated by all algorithms against the phishing data set. The figure stresses the point that the AC algorithm especially MCAR still generates alarge number of rules if contrasted to decision trees, rule induction or hybrid classification.

For the multi-label classifiers, we contrasted the proposed algorithm with MMAC multiple label AC algorithm. Fig. 3 illustrates the two measures values derived by the eMCAC and the MMAC algorithms named ''Label-weight'' and ''Any-label'' (Thabtah, 2007). Hereunder are the equations for calculating the two evaluation measures:

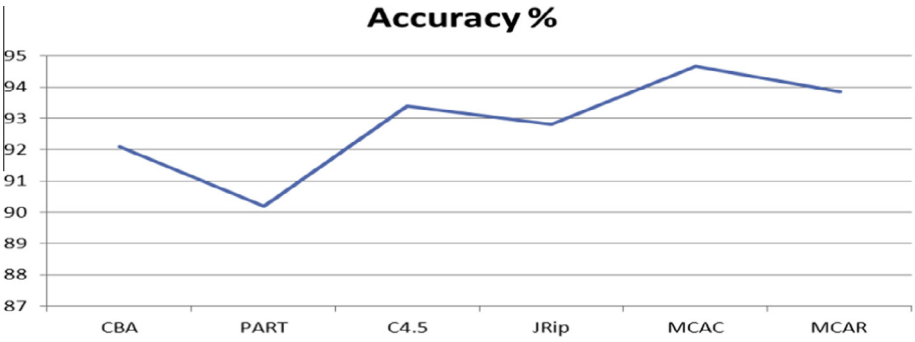

Figure 1 The classification accuracy (%) for the contrasted algorithms derived from the phishing data.

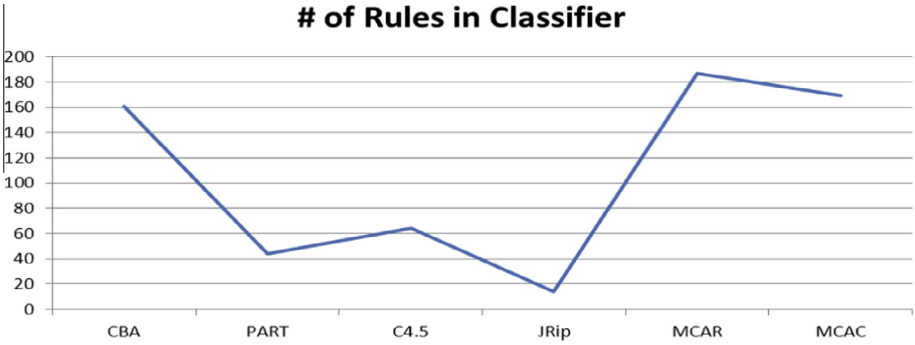

Figure 2 Average number of rules generated by the contrasted algorithms derived from the phishing data problem.

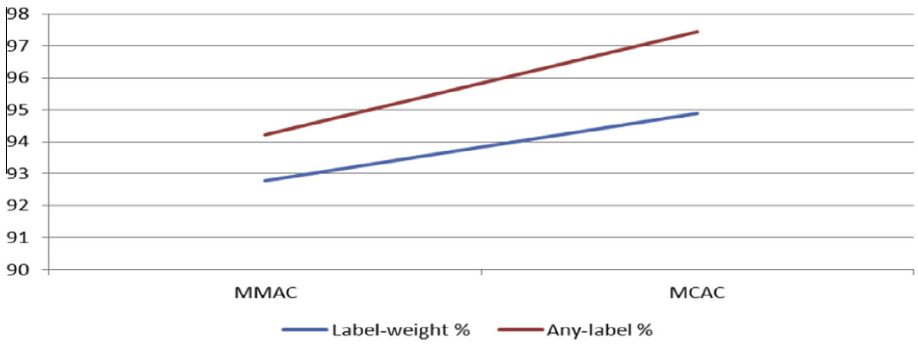

Figure 3 Accuracy % computed using Label weight and any label measures for eMCAC and MMAC algorithms.

$$\sum_{d\in D}\sum_{\{i:h^i(d)=c(d)\}}\left(f^i(d)/\sum_{j=1}^{k(d)}f^j(d)\right) \tag{1}$$

$$\left|\{d\in D:\exists i \text{ with } h^i(d)=c(d)\}\right|/m \tag{2}$$

where D designates a test data set with m data cases $d_1, d_2, \ldots, d_m$, and C represents the set of class labels. Case d has class $c(d)$ in the test data set. A (possibly multi-label) classifier is a multifunction h: $D \rightarrow 2^C$, where for $d \in D$, $h(d) = \langle h^1(d), h^2(d), \ldots, h^{k(d)}(d)\rangle$. The number of times of $h^1(d), h^2(d), \ldots, h^{k(d)}(d)$ in the training data set are given as $f^1(d), f^2(d), \ldots, f^{k(d)}(d)$, respectively.

To further clarify how the label-weight evaluation measure works, consider for case a rule R: $X \wedge Y \rightarrow l_1 \vee l_3$ where attributes value (X, Y) is associated 30 and 20 times with class labels $l_1$and l_3 in the training data respectively. This is why class l_1 precedes class l_3 in R. The label-weight technique assigns the predicted class weight to the test case if the predicted class matches the actual class of the test case. On the other hand, ''Any-label'' evaluation measure considers 100% correct classification when any of the multi-label rule's class matches the test case class. So for the rule R if the test case class is either l_3 or l_1 the test case will be given as ''1''. This explains its higher rate within Fig. 3. In the same figure eMCAC algorithm outperformed the eMMAC algorithm in both label-weight and any-label evaluation measures for the phishing data.

The increase of the predictive accuracy for both evaluation measures for the eMCAC algorithm is due to its ability to reduce the default class usage during prediction step in which if no rules are applicable to the test case the prediction procedure of the eMCAC algorithm takes on the highest ranked partly matching rule and assigns it to the test case. Further, the rule discovery method of the proposed algorithm extracts the multi-label rules early without the need to perform recursive learning which necessitates learning from independent sets of the training data similar to rule induction and greedy approaches. Instead, our algorithm learns the multi-label rules from the whole training data set once by discovering single

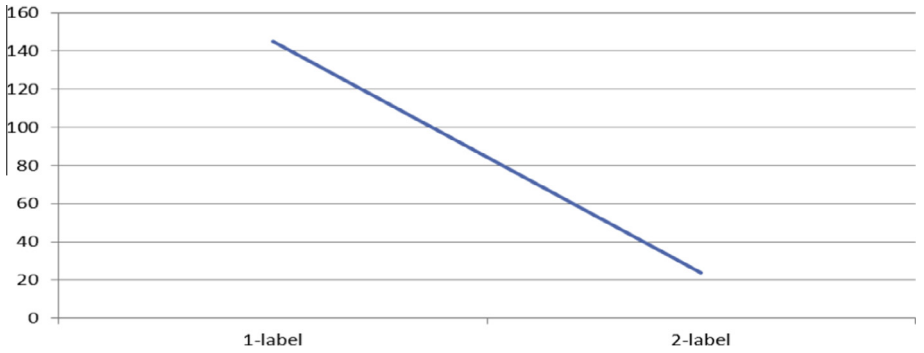

Figure 4 Number of class labels per rule derived by the eMCAC algorithm from the phishing data.

label rules that survive *MinSupp* and *MinConf* early and merge only those that share the antecedent (body) to generate the multi-label rules. This could cause rules to overlap in their training cases but the eMCAC removes the overlapping during constructing the classifier by storing only rules that have training data coverage.

To signify the importance of the additional knowledge produced by the proposed algorithm Fig. 4 displays the number of multi-label rules with respect to their consequent part (class labels on the right hand side). The proposed algorithm was able to extract multiple label rules from the phishing data set solving an important problem in classification data mining regarding rules overlapping class labels. In particular, Figure 10 shows that the eMCAC algorithm generated 24 multiple label rules that represent "Legitimate OR Phishy" website class. This multi-label class is in fact websites that are suspicious and mainly classified by current classification algorithms as "Phishy" since they do not account the class overlapping problem. In other words, the eMCAC algorithm was able to extract rules that current AC algorithms and traditional classification algorithms ignore bringing up interesting useful information for the end-user. The fact that the eMCAC algorithm finds this additional knowledge is an indicator of the ability of the algorithm to discover new data insights most current AC algorithms are unable to detect.

6. Conclusions

In this paper, a new multi-label rule-based classification algorithm based on AC mining called eMCAC has been proposed. The originality of the proposed algorithm is its ability to generate rules with multiple class labels from single data sets and without recursive learning in current AC methods like MMAC. Experimental results against crucial applications named website phishing have been conducted to evaluate the performance of the proposed algorithm in classifying websites. The measures of evaluation are label-weight, any-label, accuracy and number of

rules and the contrasted algorithms are CBA, MCAR, MMAC, PART, C4.5 and RIPPER. The results of the experiments showed that the proposed algorithm outperformed the considered algorithms on the real world phishing data with respect to accuracy. Further, the label-weight and any-label results of the proposed algorithm are better than those of the MMAC algorithm for the same data. The eMCAC algorithm was able to produce multi-label rules from the phishing data where each training example is associated with one class. We have identified a smaller effective feature set for detecting the type of the website after applying Chi-square feature selection method. The results of all considered algorithms other than eMCAC have been consistent in detecting the phishing website. In the near future, we intend to apply the eMCAC algorithm on unstructured data related to text categorisation.

References

Abdelhamid, N., Ayesh, A., Thabtah, F., Ahmadi, S., Hadi, W., 2012. MAC: A multiclass associative classification algorithm. J. Inf. Knowl. Manage. 11 (2), 1250011-1–1250011-10.

Boutell, M., Shen, X., Luo, J., Brown, C., 2003. Multi-label semantic scene classification. Technical report 813, Department of Computer Science, University of Rochester, Rochester, NY 14627 & Electronic Imaging Products R & D, Eastern Kodak Company.

Chien, Y., Chen, Y., 2010. Mining associative classification rules with stock trading data – A GA-based method. Knowl. Based Syst. 23 (2010), 605–614.

Cohen, W., 1995. Fast effective rule induction. Proceedings of the 12th International Conference on Machine Learning, CA, USA, pp. 115–123.

Frank, E., Witten, I., 1998. Generating accurate rule sets without global optimisation. In: Shavlik, J., (ed.), Machine Learning. Proceedings of the Fifteenth International Conference, Madison, Wisconsin. Morgan Kaufmann.

Jabbar, M.A., Deekshatulu, B.L., Chandra, P., 2013. Knowledge discovery using associative classification for heart disease prediction. Adv. Intel. Syst. Comput. 182 (2013), 29–39.

Jabez, C., 2011. A statistical approach for associative classification. Eur. J. Sci. Res. 58 (2), 140–147.

Li, X., Qin, D., Yu, C., 2008. ACCF: Associative Classification Based on Closed Frequent Itemsets. Proceedings of the Fifth International Conference on Fuzzy Systems and Knowledge Discovery – FSKD, pp. 380–384.

Li, W., Han, J., Pei, J., 2001. CMAR: Accurate and efficient classification based on multiple-class association rule. Proceedings of the IEEE International Conference on Data Mining – ICDM, pp. 369–376.

Liu, B., Hsu, W., Ma, Y., 1998. Integrating classification and association rule mining. Proceedings of the KDD, New York, NY, pp. 80–86.

Merz, C., Murphy, P., 1996. UCI Repository of Machine Learning Databases. University of California, Department of Information and Computer Science, Irvine, CA.

Mohammad, R.M., Thabtah, F., McCluskey, L., 2012. An Assessment of Features Related to Phishing Websites using an Automated Technique. ICITST, London.

Quaresma, P., Rodrigues, I., 2003. PGR: Portuguese Attorney General's Office decisions on the web. In: Bartenstein, Geske, Hannebauer, Yoshie, (Eds.), Web Knowledge Management and Decision Support, LNCS/LNAI 2543, Springer-Verlag, pp. 51–61.

Quinlan, J., 1993. C4.5: Programs for machine learning, Morgan Kaufmann, San Mateo, CA.

Schapire, R.E., Singer, Y., 2000. BoosTexter: a boosting-based system for text categorization. Mach. Learning 39 (2/3), 135–168.

Thabtah, F., 2007. Review on Associative Classification Mining, Journal of Knowledge Engineering Review. Cambridge Press 22:1, 37–65.

Thabtah, F., Cowling, P., Peng, Y., 2005. MCAR: Multi-class classification based on association rule approach. Proceeding of the 3rd IEEE International Conference on Computer Systems and Applications. Cairo, Egypt, pp. 1–7.

Thabtah, F., Cowling, P., 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). Brighton, UK, pp. 217–224.

Thabtah, F., Hadi, W., Abdelhamid, N., Issa, A., 2011. Prediction phase in associative classification mining. Int. J. Softw. Eng. Knowl. 21 (06), 855–876.

Thabtah, F., Mahmood, Q., McCluskey, L., Abdel-Jaber, H., 2010. A new classification based on association algorithm. J. Inf. Knowl. Manage. 9 (01), 55–64.

Tsoumakas, G., Katakis, I., 2007. Multi-label classification: an overview. In: David Taniar (Ed.), International Journal of Data Warehousing and Mining, Idea Group Publishing, 3(3), pp. 1–13.

Veloso, A., Meira, W., Zaki, M., Goncalves, M., Mossri, H., 2011. Calibrated lazy associative classification. Inf. Sci. Int. J. 13 (181), 2656–2670.

Wang, X., Yue, K., Niu, W., Shi, Z., 2011. An approach for adaptive associative classification. Expert Syst. Appl. Int. J. 38 (9), 11873–11883.

Witten, I., Frank, E., 2002. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Francisco.