Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Review

Phishing detection based Associative Classification data mining

Neda Abdelhamid [a,*], Aladdin Ayesh [a], Fadi Thabtah [b]

[a] Computing and Informatics Department, De Montfort University, Leicester, UK
[b] Ebusiness Department, Canadian University of Dubai, Dubai, United Arab Emirates

ARTICLE INFO

Keywords:
Classification
Data mining
Forged websites
Phishing
Internet security

ABSTRACT

Website phishing is considered one of the crucial security challenges for the online community due to the massive numbers of online transactions performed on a daily basis. Website phishing can be described as mimicking a trusted website to obtain sensitive information from online users such as usernames and passwords. Black lists, white lists and the utilisation of search methods are examples of solutions to minimise the risk of this problem. One intelligent approach based on data mining called Associative Classification (AC) seems a potential solution that may effectively detect phishing websites with high accuracy. According to experimental studies, AC often extracts classifiers containing simple "If-Then" rules with a high degree of predictive accuracy. In this paper, we investigate the problem of website phishing using a developed AC method called Multi-label Classifier based Associative Classification (MCAC) to seek its applicability to the phishing problem. We also want to identify features that distinguish phishing websites from legitimate ones. In addition, we survey intelligent approaches used to handle the phishing problem. Experimental results using real data collected from different sources show that AC particularly MCAC detects phishing websites with higher accuracy than other intelligent algorithms. Further, MCAC generates new hidden knowledge (rules) that other algorithms are unable to find and this has improved its classifiers predictive performance.

1. Introduction

The internet is not only important for individual users but also for organisations doing business online. Many of the organisations offer online trading and online sales of services and goods (Liu & Ye, 2001). Nevertheless, internet-users may be vulnerable to different types of online threats that may cause financial damages, identity theft, and loss of private information. Therefore, the internet suitability as a channel for commercial exchanges comes into question.

Phishing is considered a form of online threat that is defined as the art of impersonating a website of an honest firm aiming to acquire user's private information such as usernames, passwords and social security numbers (Dhamija, Tygar, & Hearst, 2006). Phishing websites are created by dishonest individuals to imitate genuine websites. These websites have high level of visual similarities to the legitimate ones in an attempt to defraud honest internet-users. A report published by "Gartner Co." (Gartner Inc., 2011), which is a research and advisory company shows that phishing attacks are increasing rapidly. Gartner estimated that theft through phishing attacks costs U.S. banks and credit card companies $2.8 billion annually. In 2011, the Director of Cisco's security-technology-business-unit issued his concerns that today's main attacks focus on gaining access to corporate accounts that contain valuable financial information.

Social engineering which is the act of manipulating people to obtain sensitive information, can be combined with computerised technical tricks in order to start a phishing attack (Aburrous, Hossain, Dahal, & Thabtah, 2010a). Fig. 1a depicts the general steps conducted in phishing. Phishing websites have become a serious problem not only because of the increased number of these websites but also the intelligent strategies used to design such websites. Therefore, users that have extensive experience and knowledge in computer security and internet might be deceived (Sanglerdsinlapachai & Rungsawa, 2010).

Typically, a phishing attack begins by sending an e-mail that seems to be from an authentic organisation to victims. These emails ask them to update their information by following a URL link within the e-mail. Other methods of distributing phishing URLs include, Black Hat search engine optimization (Black Hat SEO) (Seogod, 2011), Peer-to-peer file sharing, blogs, forums, instant messaging (IM) and Internet Relay Chat (IRC) (Kirda & Kruegel, 2005).

* Corresponding author. Tel.: +44 (0)116 2 50 60 70.
E-mail addresses: P09050665@myemail.dmu.ac.uk (N. Abdelhamid), aayesh@dmu.ac.uk (A. Ayesh), fadi@cud.ac.ae (F. Thabtah).

http://dx.doi.org/10.1016/j.eswa.2014.03.019
0957-4174/© 2014 Elsevier Ltd. All rights reserved.

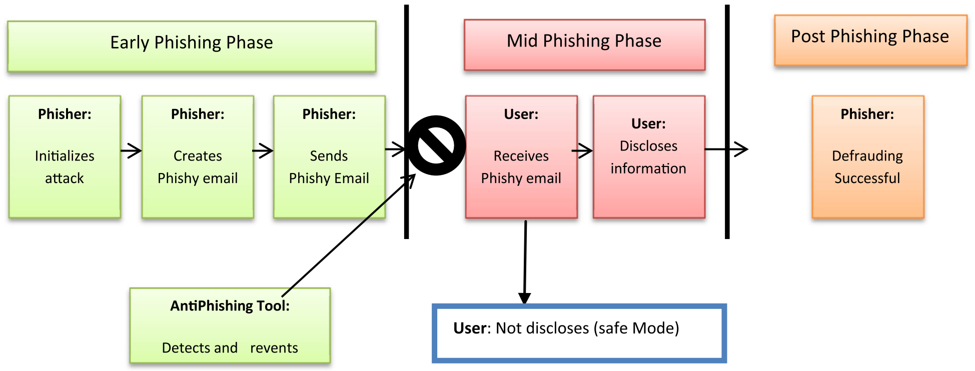
Early Phishing Phase
Phisher: Initializes attack
Phisher: Creates Phishy email
Phisher: Sends Phishy Email
AntiPhishing Tool: Detects and revents
Mid Phishing Phase
User: Receives Phishy email
User: Discloses information
User: Not discloses (safe Mode)
Post Phishing Phase
Phisher: Defrauding Successful

Fig. 1a. Phishing life cycle.

Below, we briefly explain the two most popular approaches in designing technical anti-phishing solutions (Aaron & Manning, 2012; Sadeh, Tomasic, & Fette, 2007).

- Blacklist approach: where the requested URL is compared with a predefined phishing URLs. The drawback of this approach is that the blacklist usually cannot cover all phishing websites since a newly created fraudulent website takes a considerable time before it can be added to the list.
- Search approach: the second approach is based on search/heuristic methods, where several website features are collected and used to identify the type of the website. In contrast to the blacklist approach, the heuristic-based approach can recognise newly created fake websites in real-time (Miyamoto, Hazeyama, & Kadobayashi, 2008).

The numbers of phishing websites are expected to increase over time. Thus, smart solutions are needed to keep pace with the continuous evolution of this problem. Smart solutions are the subject of our interest in this article. They can be combined with the heuristic-based approach as long as historical data exists. In fact, the accuracy of the heuristic-based solution mainly depends on a set of discriminative features extracted from the website. Hence, the way in which those features are processed plays an extensive role in accurately classifying websites. Therefore, an effective intelligent based method when merged with the heuristic method can be essential for making a good decision.

Associative Classification (AC) in data mining is one of the promising approaches that can make use of the features extracted from phishing and legitimate websites to find patterns among them (Costa, Ortale, & Ritacco, 2013; Thabtah, Cowling, & Peng, 2005). This approach normally devises classifiers (set of rules) that are simple yet accurate. The decision-making process becomes reliable because these decisions are made based on rules discovered from historical data by the AC algorithm. Although plenty of applications are available for combating phishing websites few of them make use of AC data mining (Jabbar, Deekshatulu, & Chandra, 2013).

Phishing is a typical classification problem (Abdelhamid, Ayesh, & Thabtah, 2013) in which the goal is to assign a test data (a new website) one of the predefined classes (phishy, legitimate, suspicious, etc.). Once a website is loaded on the browser a set of feature values will be extracted. Those features have a strong influence in determining the website type by applying the rules that have been previously found by the AC algorithm from the historical data (already labelled websites). Then, the chosen rule's class will be assigned to the browsed website and an appropriate action will take place. For instance, a message or an alarm will be fired to alert the user of the risk.

In this paper, the problem of phishing detection is investigated using AC approach in data mining. We primarily test a developed AC algorithm called MCAC and compare it with other AC and rule induction algorithms on phishing data. The phishing data have been collected from the Phishtank archive (PhishTank, 2006), which is a free community site. In contrast, the legitimate websites were collected from yahoo directory. The evaluation measures used in the comparison are accuracy, number of rules, any label, and label-weight (Thabtah, Cowling, & Peng, 2004). More details are given in Section 5.

We show that MCAC is able to extract rules representing correlations among website's features. These rules are then employed to guess the type of the website. The novelty of MCAC is its ability not only to discover one class per rule, but rather a set of classes bringing up the classifier performance in regards to accuracy. This is unlike current AC algorithms that only generate a single class per rule. Thus, the new classes connected with the rules that have been revealed by MCAC correspond to new knowledge missed by the majority of the existing AC algorithms. More details of MCAC processes are given in Section 4.

This paper is divided into different sections where Section 2 surveys common related learning approaches to phishing detection. Section 3 sheds the light on AC as well as its advantages, and MCAC algorithm. The features related to the phishing problem that have been utilised in the experimental section are discussed in Section 4. Section 5 is devoted to experiments where we demonstrate the data collection process, the evaluation measures, the compared algorithms, the results, and the analysis of the results. Lastly, conclusions are given in Section 6.

2. Related works

In this section, we review common intelligent phishing classification approaches from the literature, after shedding the light on the general steps required to solve the phishing problem and its general combating approaches. Further, the section starts by showing the phishing life cycle.

2.1. Phishing lifecycle

Fig. 1a depicts the general steps conducted in the phishing life cycle. According to Fig. 1a, a phishing attack begins by sending an e-mail that seems to be from an authentic organisation to users urging them to change their data by selecting a link within an e-mail. E-mails remain a spreading channel for phishing links since 65% of phishing attacks start by visiting a link received within an e-mail (Kaspersky Lab, 2013).

The main steps that need to be addressed to solve the phishing problem are the following (Horng et al., 2011):

(1) Identification of the required data: for any given problem, we need a set of attributes, which are already measured or preset. These should have some influence on the desired output (classifier). Thus, a set of input and output attributes should be identified.
(2) Training set formation: the training data set consists of pairs of input instances or examples and desired target attribute (class). There are many sources of phishing data such as Phishtank.
(3) Determination of the input feature: the classifier accuracy depends on how the training example is represented and how features have been carefully selected. The feature selection process should discard irrelevant features as possible in order to minimise the dimensionality of the training data set so the learning process can be effectively executed. We show later the ways we assessed the feature before choosing them.
(4) Applying the classification algorithm: the choice of a mining algorithm is a critical step. There are wide ranges of mining methods available in the literature where each of these classification approaches has its own pros and cons. Three main elements in selecting a classification approach are (a) the input data characteristics, (b) the classifier predictive power measured by the accuracy rate, and (c) the simplicity and understandability of the output. Overall, there is no single classifier that works best on all given data, and classifier performance largely relies on the training data set characteristics. For this step, we selected AC since it has many distinguishing features particularly the high predictive accuracy and the understandability of output derived.
(5) Classifier evaluation: the last step is to test the derived classifier performance on test data.

2.2. Non technical approaches to minimise phishing

The known general non technical methods to combat phishing are (James, 2005):

- Legal solutions: followed by many countries where the United States was the first to enact laws against phishing activities and many phishers have been arrested and sued. Phishing has been added to computer crime list in 2004 by Federal Trade Commission (FTC) which is a U.S government agency that aims to promote consumers protection (Kunz & Wilson, 2004). In the years 2005 and 2006, both the Australian and UK governments strengthened its legal arsenal against fraud by prohibiting the development of phishing websites and enacted jail penalties (http://www.finextra.com/news). Lastly, the Australian government also signed a partnership with Microsoft to teach the law enforcement officials how to combat different cyber-crimes (Government of Australia, 2011). Nevertheless, legal solutions do not sufficiently catch phishers since it is very difficult to trace them due to their quick disappearances in the cyber world.
- Education: in combating phishing, consumer's education in order to raise awareness of this online crime is beneficial (Government of Australia, 2011). If internet-users could be convinced to inspect the security indicators within the website the problem is substantially minimised. However, the important advantage for phishers to successfully trick internet-users is that the majority of internet-users lack basic knowledge of current online threats that may target them. Generally speaking, although raising awareness about phishing to users may be seen as a promising direction it is still a hard task to implement. This is because users are required to spend a long time learning phishing methods, and phishers becoming more talented in creating new phishing techniques, which sometimes makes even security experts deceived.

2.3. Technical approaches to handle phishing

Typically, the two most technical methods in fighting phishing attacks are the blacklist and the heuristic-based (Aaron & Manning, 2012; Sadeh et al., 2007). In the blacklist method, the requested URL is compared with a predefined phishing URLs. The downside of this method is that it typically doesn't deal with all phishing websites since a newly launched fake website takes a substantial amount of time before being added to the list. In contrast to the blacklist approach, the heuristic-based approach can recognise newly created fake websites in real-time (Miyamoto et al., 2008). More details on these approaches are given in Section 2.

Weaknesses that appeared when relying on the abovementioned solutions led to the need to innovative solutions. Several solutions are offered these days to handle phishing such as MacAfee. Moreover, some non-profit organisations such as APWG (Aaron & Manning, 2012), PhishTank (PhishTank, 2006) and MillerSmiles (millersmiles, 2011) provide forums of opinions as well as distribution of the best practises against phishing from users' experiences. The success of an anti-phishing technique mainly depends on recognising phishing websites and within an acceptable timescale. Although a number of anti-phishing solutions are developed, most of these solutions were unable to make highly accurate decisions causing a rise of false positive decisions, which means labelling a legitimate website as fake. We focus on technical solutions proposed by scholars in the literature in the following sub-sections for dealing with the phishing problem.

2.3.1. Blacklist-Whitelist based Approach

A blacklist is a list of URL's that are thought to be malicious and have been collected using techniques such as user voting. So, whenever a website is launched the browser refers to the blacklist to check if the launched website exists within the blacklist. If the check result is true the browser warns the users not to submit any sensitive information. Blacklist could be saved either locally on the user's machine or on a server that is queried by the browser for every requested URL.

Sheng et al. (2009) showed that blacklists are usually updated at different frequencies. Precisely, it was estimated that 50–80% of phishy URL's are displayed in the blacklist 12 h after their launch. Other blacklists such as Google's needs on average 7 h to be updated (Dede, 2011). So it is necessary for a decent blacklist to be updated instantly to keep users safe from being victimised.

The blacklist approach has been deployed towards many solutions, one of which is Google Safe Browsing which uses a list of pre-defined phishy URLs to detect fake URLs. Another solution is Microsoft IE9 anti-phishing protection and SiteAdvisor which are basically database based solutions that are primarily created to catch malware attacks such as Spyware and Trojan horses. These contain an automated crawler that browses websites and builds threat ratings based on the visited URLs. Unlike blacklists or database based solutions SiteAdvisor cannot identify newly created threats.

Another anti-phishing tool named VeriSign crawls millions of websites to recognise "clones" in order to distinguish phishing websites. One drawback these approaches might be that the anti-phishing parties will always be in a competition against the attackers. Netcraft is a small program that gets activated upon using a web browser. It relies on a blacklist which consists of fraudulent websites recognised by Netcraft and those submitted by the users and verified by Netcraft. Netcraft displays the location of the server where the webpage is hosted. This is helpful for experienced users of web hosting where for instance a webpage ending with ".ac.uk" is unlikely to be hosted outside the UK.

The opposite term of the blacklist is the whitelist, which is a set of trusted websites, while all other websites are considered untrusted. Chen and Guo (2006) proposed an Automated-Individual-Whitelist (AIWL), which is an anti-phishing tool based on an individual user's whitelist of trusted websites. AIWL traces every login attempt made by the user through the utilisation of a Naive Bayes algorithm. In case a repeated successful login for a specific website is achieved, AIWL prompts the user to add the website to the whitelist.

One other solution that depends on the whitelist was presented in PhishZoo (Afroz & Greenstadt, 2011). PhishZoo builds profiles of trusted websites based on fuzzy hashing technique. A website profile is a combination of several metrics that exclusively identify that website. This approach combines whitelisting with blacklisting and heuristic approach to warn users of attacks. The authors believed that phishing detection should be derived from user's point of view since over 90% of users rely on the website appearance to verify its authenticity.

2.3.2. Fuzzy rule based approaches

One approach employed in Aburrous et al. (2010a) is based on experimentally contrasting few rule based classification algorithms after collecting dissimilar features from a range of websites as revealed in Table 1. Those features varied amongst three uncertain values "Legitimate & Genuine" and "Doubtful". To evaluate the selected features, the authors conducted experiments using the following algorithms in Weka, C4.5, PRISM, PART and JRip (Witten and Frank, 2002; Weka, 2011). The results uncovered a significant association between "Domain Identity" and "URL" features. However, no justifications on the way features have been assessed.

The authors of Aburrous, Hossain, Dahal, and Thabtah (2010b) have used a larger set of features to predict websites type based on fuzzy logic. Although, their developed method gave promising results in accuracy it was not mentioned how the features have been extracted from the website and specifically features related to human factors. Furthermore, the rules used were established based on human experience rather intelligent data mining techniques, which is one of the problems we aim to resolve in this paper. Lastly, the authors classified the websites as very-legitimate, legitimate, suspicious, phishy or very-phishy, but they did not clarify what is the fine line that separates one class from another.

2.3.3. Machine learning approaches

The majority of methods developed to deal with the phishing problem are based on support vector machine or SVM for short. SVM is a known machine learning technique that has been used effectively to solve classification problems (Song, 2009). Its popularity comes from the accurate results it produced particularly from unstructured problems like text categorization. An SVM in general can be seen as a hyper-plane that splits the objects (points) belonging to a class (positive objects) from those that do not belong to that class (negative objects). This split is implemented by the SVM algorithm during the learning step where the hyper-plan is obtained to divide positive and negative objects with maximal margins. The margin denotes the space from hyper-plane to the closest positive and negative object.

A method based on SVM was proposed in Pan and Ding (2006) to discover unusual activities, i.e. phishing, in websites based on two variables. The first one is based on the company's name shown in the domain name, and the second one is called the "page categoriser", which denotes properties related to structural features (Abnormal URL, abnormal DNS record, etc.) that are hard to be duplicated.

Table 1
Phishing criteria from (Aburrous et al., 2010a).

Feature set	Phishing feature indicator
Domain identity and URL	via IP address
	Require URL
	URL of anchor
	DNS details
	Strange URL
Encryption and security	SSL certificate
	Certification authority
	Strange cookie
	Distinguished names certificate (DN)
Java script and source code	Redirect pages
	Straddling attack
	Pharming attack
	Using onMouseOver
	Server form handler
Contents and page style	Spelling mistake
	Replicating a website
	"Submit" button
	Via pop-up windows
	Disabling right-click
Web address bar	Long URL address
	Replacing similar characters for URL
	Adding prefix or suffix
	Using the '@' to confuse
	Using hexadecimal character codes
Social human factor	Much stress on security and response
	Generic welcome
	Buying time to log on accounts

Six different structural features have been chosen, and Vapkin's Support Vector Machine (SVM) algorithm (Cortes & Vapnik, 1995) was used to determine whether the website is phishy or not. Tests on a limited data set consisting of 379 URLs revealed that the "Identity Extractor" is an important feature related phishy URLs. Overall, the accuracy derived by this method was 84%. A solution to increase this method's accuracy would be by employing other features.

In Sadeh et al. (2007), the authors compared some commonly used machine-learning methods including SVM, decision trees, and Naïve Bayes on the problem of email phishing. A random forest algorithm called "Phishing Identification by Learning on Features of email Received" (PILFER) was implemented. A data set consisting of 860 phishy emails and 695 legitimate were used in the experiments. PILFER has good accuracy in identifying phishing emails. The authors used a small number of features for detecting phishing emails those are "IP based URL's, age of domain, non-matching URL's, having a link within the e-mail, HTML emails, number of links within the e-mail, number of domains appears within the e-mail, number of dot's within the links, containing JavaScript and spam filter output". The authors concluded that PILFER can be enhanced towards classifying emails by combining all the 10 features except "Spam filter output" with those shown in Table 2. For assessment; the authors utilised exactly the same data set. The results revealed that PILFER has lowered the false positive rate.

2.3.4. CANTINA based approaches

The method proposed in Guang, Jason, Carolyn, and Lorrie (2011) suggested utilising "Carnegie Mellon Anti-phishing and Network Analysis Tool" (CANTINA) (Zhang, Hong, & Cranor, 2007). This content-based technique often reveals the type of websites using term-frequency-inverse-document-frequency (TF-IDF) (Thabtah, Eljinini, Zamzeer, & Hadi, 2009). CANTINA checks the

Table 2
Features added to PILFER to classify websites.

Phishing factor indicator	Feature clarification
Site in browser history	If a site not in the history list then it is expected to be phishing
Redirected site	Forwarding users to new webpage
TF-IDF	Search key terms on a page then checks whether the current page is present in the result

website content and decides whether it is phishy by using TF-IDF. TF-IDF assesses a document's word importance by assigning weights and counting its frequency. CANTINA calculates the TF-IDF for a given webpage then takes the five highest TF-IDF terms and adds them to the URL to find the lexical signature, which is fed into a search engine.

When the current webpage is among the first 30 results it is considered a legitimate webpage. If not, it is phishy. If the search engine returns zero result, the website is labelled as phishy. To overcome the zero result, the authors combined TF-IDF with some other feature, e.g. (suspicious URL, Age of domain, dots in URL, etc.). A limitation of this method is that some legitimate websites consist of images so using the TF-IDF may not be suitable. In addition, this approach does not deal with hidden texts, which might be effective in detecting the type of the webpage.

Another approach that utilises CANTINA with an additional attributes (Sanglerdsinlapachai & Rungsawang, 2010) have used a small data set having 100 phishy and 100 legitimate websites and eight features, i.e. suspicious link, domain age, TF-IDF, etc. Some changes to the features have been performed during the experiments as follow:

(1) The "Forms" feature is set as a filter to begin the process of deciding the legitimacy of the website since fraudulent sites that may cause the loss of users' information must have "input blocks" within the "Forms".
(2) According to the authors, "Known image" and "Domain age" features are disregarded since they hold no significance to the problem.
(3) The similarity between a fuzzy webpage and top-page of its domain is a newly suggested feature.

The authors have performed three types of experiments against their data set. The first experiment evaluated a reduced CANTINA feature set "IP address dots in URL, suspicious URL, IP address and suspicious link" and the second experiment tested whether the new features "domain top-page similarity" play a significant role in uncovering website type. The third experiment evaluated the results after adding the new suggested feature to the reduced CANTINA features used in the first experiment. By comparing the performance after adding the new feature, a number of classification algorithms showed that the error rate results of the new feature played a key role in detecting the type of the website. Neural network algorithm (NN-BP) performed the best accuracy with an error rate of 7.5% whereas Naïve Bayes was the lowest performing algorithm with 22.5% error rate.

2.3.5. Image based approaches

One approach proposed by Liu, Huang, Xiaoyue, Min, and Deng (2005) detected the type of websites by comparing phishy sites with the non-phishy sites based on visual similarity. This technique breaks down the webpage into block regions depending on "visual cues." The visual similarity between a phishy webpage and non- phishy one is evaluated using three metrics: block level similarity, layout similarity, and style similarity. A webpage is considered phishy if any metric has a value higher than a predefined threshold. The authors collected a limited number of official banking websites, and then carried out the experiments. The results revealed a low error rate.

Lastly, in Dhamija and Tygar (2005), a method called Dynamic Security Skins (DSS) was disseminated. Since the system designer and the phisher rely on the interface to protect or defraud users, this approach used an agreed discrete image that allows a remote server to prove its identity to the user. This technique requires the users verification based on comparing his expected image with an image generated by the server. The authors implemented their method by developing an extension to Mozilla Firefox browser. The main drawback of this method is that the users bear the burden of deciding whether the website is phishy or not, thus users need to be conscious of the phishing and look for signs that the website he is visiting is in fact a spoof website. This approach also suggested a fundamental change to the web infrastructure for both servers and clients, so it can succeed only if the whole online industry supports it.

3. Associative Classification data mining

Merging association rule and classification in data mining had come to surface as a promising research discipline named AC (Abdelhamid, Ayesh, Thabtah, Ahmadi, & Hadi, 2012). In AC, the training phase is about inducing hidden knowledge (rules) using association rule and then a classification model (classifier) is constructed after pruning useless and redundant rules. Many research studies including (Jabbar et al., 2013) revealed that AC usually extracts better classifiers with reference to error rate than other classification approaches like decision tree, and rule induction (Witten and Frank, 2002).

Normally, an AC algorithm operates in three phases. During phase (1), it looks for hidden correlations among the attribute values and the class attribute in the training data set and generates them as "Class Association Rule" (CARs) in "If-Then" format (Thabtah et al., 2004). After the complete set of CARs are found, ranking and pruning procedures (phase 2) start operating where the ranking procedure sorts rules according to certain thresholds such as confidence and support. Further, during pruning, duplicating rules are discarded from the complete set of CARs. The output of phase 2 is the set of CARs which represents the classifier. Lastly, the classifier derived gets evaluated on test data to measure its effectiveness in forecasting the class of this test data. The output of the last phase is the accuracy or error-rate of the classifier.

AC has two distinguishing features over other traditional classification approaches. The first one is that it produces simple rules that can be easily interpreted and manually updated by end-user. Secondly, this approach often finds additional useful hidden information missed by other classification algorithms and therefore the error rate of the resulting classifier is minimised. The main reason behind producing the additional useful information is that AC utilises association rule discovery methods in the training phase (Thabtah, 2007). Though, in some cases the possible numbers of derived rules may become excessive.

There are a number of AC algorithms that have been proposed in the last decade including Multiclass Classification based Association Rules (MCAR) (Thabtah et al., 2005), uncertain CBA (uCBA) (Qin, Zhang, Li, & Wang, 2010), ADA (Wang, Yue, Niu, & Shi, 2011), Multiclass Associative Classification (MAC) (Abdelhamid et al., 2012), X-class (Costa et al., 2013) and others. These algorithms employ different methodologies for knowledge discovery, rule sorting, rule pruning, and class assignment for test data. Hereunder are the related definitions to AC where a training data set *D* with *N* attributes is assumed.

Definition 1. An *AttributeValue* can be described as an attribute name A_i and its value a_i, denoted (A_i, a_i).

Definition 2. The *j*th *row* or a *training case* in *D* can be described as a list of attribute values (A_{j1}, a_{j1}),...,(A_{jk}, a_{jk}), plus a class denoted by c_j.

Definition 3. An *AttributeValueSet* set can be described as a set of disjoint attribute values contained in a training case, denoted $\langle(A_{i1}, a_{i1}),\ldots,(A_{ik}, a_{ik})\rangle$.

Definition 4. A *ruleitem r* is of the form ⟨*antecedent*, *c*⟩, where *antecedent* is an *AttributeValueSet* and *cC* is a class.

Definition 5. The actual occurrence (*actoccr*) of a *ruleitem r in D* is the number of examples in *D* that match *r's antecedent*.

Definition 6. The support count (*suppcount*) of *ruleitem* is the number of examples in *D* that matches *r's antecedent*, and belongs to a class *c*.

Definition 7. A *ruleitem r* passes the *minsupp* if, $suppcount(r)/|D| \geqslant minsupp$. Such a ruleitem is said to be a frequent ruleitem.

Definition 8. A *ruleitem r* passes *minconf* threshold if $suppcount(r)/actoccr(r) \geqslant minconf$.

Definition 9. A single rule is represented as: $Antecedent \to c$, where antecedent is an *AttributeValueSet* and the consequent is a class.

Definition 10. A multi-label rule is represented as: $Antecedent \to c_{i1} \vee c_{i2} \ldots \vee c_{i3}$, where antecedent is an *AttributeValueSet* and the consequent is a set of classes in a disjunction form.

3.1. The MCAC algorithm

The phishing detection process using our model from the user prospective can be explained in the following steps:

(1) The end-user clicks on a link within an email or browses the internet.
(2) He will be directed to a website that could be legitimate or phishy. This website is basically the test data.
(3) A script written in PHP that is embedded within the browser starts processing to extract the features of the test data (current website) and saves them in a data structure.
(4) Now, the intelligent model will be active within the browser to guess the type of the website based on rules learnt from historical websites (previous data collected). The rules of the classifier are utilised to predict the type of the test data based on features similarity.
(5) When the browsed website is identified as legitimate no action will be taken. On the other hand, when the website turned to be phishy, the user will be warned by the intelligent method that he is under risk.

We have implemented steps (1)–(4) in the above model where we utilised MCAC learning strategy to generate the rules. MCAC comprises of three main steps: Rules discovery, classifier building and class assignment. In the first step, MCAC iterates over the training data set (historical websites features) in which rules are found and extracted. In this step, the algorithm also merges any of the resulting rules that have the same antecedent (left hand side) and are linked with different classes to produce the multi-label rules. In addition, redundant rules that have no training data coverage are discarded. The outcome of the second step is the classifier which contains single and multi-label rules. The last step involves testing the classifier on test data set to measure its performance. In predicting a website the rule in the classifier that matches the test data features often fired to guess its type (class).

The general description of the MCAC algorithm is displayed in Fig. 1b.

3.2. MCAC rule learning and prediction example

We show in this section an example on how MCAC generates and produces the rules. Assume that *minsupp* and *minconf* has been set to 20% and 40% respectively. Table 3 displays an initial training data set, and the candidate rules extracted are depicted in Table 4a. While the algorithm is generating the rules, it checks whether there exists a candidate rule that is already extracted with a similar body of the current rule. If this condition is true, the algorithm appends the current rule with the already extracted rule to form a new multi-label rule. For example, in Table 4a, the attribute value $\langle a_1 \rangle$ is connected with two class labels, i.e. (c_2, c_1) with frequencies 4 and 3, respectively. Current AC algorithms will produce only one rule for this attribute value, i.e. $a_1 \to c_2$ and simply discards class (c_1) because (c_1) has more number of occurences in the training

Table 3
Training data set.

Instance number	Att_1	Att_2	Class
1	a_1	b_1	c_2
2	a_1	b_1	c_2
3	a_2	b_1	c_1
4	a_1	b_2	c_1
5	a_3	b_1	c_1
6	a_1	b_1	c_2
7	a_4	b_2	c_1
8	a_1	b_2	c_1
9	a_1	b_3	c_1
10	a_1	b_2	c_2

```
Input: Training data D, minimum confidence (MinConf) and minimum support (MinSupp) thresholds
Output: A classifier
Preprocessing: Discretise continuous attributes if any
Step One:
    • Scan the training data set T to discover the complete set of frequent attribute values
    • Convert any frequent attribute value that passes MinConf to a single label rule
    • Merge any two or more single label rules that have identical body and different class to derive the multi-label rules
Step Two:
    • Sort the rule set according to confidence, support and rule's length
    • Build the classifier by testing rules on the training data and keeping those in Cm that have data coverage
Step Three:
    • Classify test data using rules in Cm
```

Fig. 1b. MCAC algorithm general steps.

Table 4a
Candidate rules produced from the data in Table 3.

Ruleitems		Support count (%)	Confidence (%)
Attribute	Class		
a_1	**c_2**	**40**	**57**
a_1	**c_1**	**30**	**42**
b_1	**c_2**	**30**	**60**
b_1	**c_1**	**20**	**40**
b_2	c_1	30	75
$a_1 \wedge b_1$	c_2	30	100
$a_1 \wedge b_2$	c_1	20	66

Table 4b
Candidate multi-label rules produced from the data in Table 3 via MCAC.

Ruleitems		Support (%)	Confidence (%)
Attribute	Class		
a_1	c_2, c_1	35	50.00
b_1	c_2, c_1	25	50.00

Table 4c
The classifier of MCAC algorithm from the data in Table 3.

Ruleitems		Support count (%)	Confidence (%)
Attribute	Class		
$a_1 \wedge b_1$	c_2	30	100
b_2	c_1	30	75
a_1	c_2, c_1	35	50.00
b_1	c_2, c_1	25	50.00

data set with attribute value $\langle a_1 \rangle$. However, MCAC produces a multi-label rule for $\langle a_1 \rangle$ as $a_1 \to c_2 \vee c_1$. These additional knowledge are vital for many reasons:

(1) The decision makers now have more than one solution for a possible scenario where he can benefit from the additional knowledge in making decisions.
(2) The predictive accuracy may improve since multiple classes are associated with a rule with different possible weights based on their frequencies in the training data set. So when a test data is about to be classified, there will be more than one possible class to allocate it so we can end up with more hits rather wrong classification (using one class approach).

The candidate multi-label rules must pass the *minsupp* and *minconf* in order to be considered while making the classifier and their actual support (frequency) and confidence values are updated when they are formed as displayed in Table 4b. The candidate rules in bold within Table 4a represents the possible candidate multi-label rules shown in Table 4b. Once the rule extraction is finished MCAC sorts all possible candidate rules according to confidence, support, and rule's length. The candidate rules are ready for evaluation against the training data set in order to choose the best ones that can make the classifier. MCAC selects rules that have at least one training data coverage. Table 4c shows the classifier devised by our algorithm that consists of four rules, two of which are multi-label ones, i.e. $a_1 \to c_2 \vee c_1$ and $b_1 \to c_2 \vee c_1$.

4. Website features related to phishing

4.1. Features preparation

There are several features that distinguish phishing websites from other types of websites in the research literature of phishing.

Table 5
The selected features set.

Website feature	Frequency rate
IP address	20.5
Long URL	51.0%
URL's having @ symbol	6.8%
Prefix and suffix	25.4%
Sub-domain (dots)	42.8%
Misuse/fake of HTTPs protocol	89.2%
Request URL	100%
Server form handler	5.7%
URL of anchor	22.3$
Abnormal URL	20.5%
Using pop-up window	14.3%
Redirect page	11.0%
DNS record	7.6%
Hiding the links	21.0%
Website traffic	93.2%
Age of domain	97.4%

We have conducted our feature assessment based on frequency analysis of a number of features collected from the previous researches, (e.g. Miyamoto et al., 2008; Mohammad, Thabtah, & McCluskey, 2012). These features contribute to the classification type of the websites. Particularly, a frequency analysis experiment that counts each feature using over 1350 websites collected from different sources. Phishing websites were collected from Millersmiles and Phishtank data archives, which are free community sites for sharing phishing data. The legitimate websites were collected from yahoo directory using a web script developed in PHP. The script was plugged within a browser and we collected 601 legitimate and 752 phishing websites. The ultimate aim for this initial experiment is to scientifically select the common features that may help in assessing the determination of the website's type accurately.

In our study, sixteen different features plus the class have been identified after performing the frequency analysis against the different collected URLs. The result of the analysis is depicted in Table 5. We can see each feature and its associated frequency rate computed from the gathered data set. For example, "Age of Domain" and "Request URL" (explained shortly in this section) are common features since they constitute high rate. Further, "URL having the @ symbol" constitutes 6.8% which have a relatively low rate, but are always associated with phishy websites, and thus has a high impact on distinguishing this type of websites.

The chosen feature shown in Table 5 taking either a binary or a ternary values where binary features hold either "phishy" or "legitimate" status because the existence or lack of the feature within the website determines the value assigned to it. For ternary value features, the existence of the feature in a specific ratio determines the value assigned for that feature. The features used in our study are explained in Section 4.2. Later in the experimental section, we utilise Chi-Square testing to further assess the selected features set.

4.2. The selected features

In this subsection, we explain the features that have been used in the experiments and their corresponding rules.

(1) **IP address:** Using an IP address in the domain name of the URL is an indicator someone is trying to access the personal information. This trick involves links that may begin with an IP address that most companies do not commonly use any more. In the frequency analysis conducted earlier, 20% of the data contains "IP" address and all of them are associated with phishy websites. An IP address is like http://91.121.10.211/~chems/webscr/verify Sometimes the IP

address is transformed to hexadecimal like http://0x58.0xCC.0xCA.0x62.

Rule : If IP address exists in URL $\rightarrow$ Phishy
else $\rightarrow$ Legit

(2) **Long URL:** Phishers hide the suspicious part of the URL to redirect information's submitted by users or redirect the uploaded page to a suspicious domain. Scientifically, there is no standard reliable length that differentiates between phishing URLs and legitimate ones. (Mohammad et al., 2012) suggested when the URL length is greater than 54 characters the URL can be considered phishy.

Rule : If URL length $< 54 \rightarrow$ Legit
URL length $\geqslant 54$ and $\leqslant 75 \rightarrow$ Suspicious
else $\rightarrow$ Phishy

(3) **URL's having @ symbol:** The "@" symbol leads the browser to ignore everything prior it and redirects the user to the link typed after it.

Rule : If URL has '@' $\rightarrow$ Phishy
else Legit

(4) **Adding prefix and suffix:** Phishers try to scam users by reshaping the suspicious URL so it looks legitimate. One technique used is adding a prefix or suffix to the legitimate URL. Thus, the user may not notice any difference.

Rule : If domain part has '–' $\rightarrow$ Phishy
else $\rightarrow$ Legit

(5) **Sub-domains:** Another technique used by phishers to scam users is by adding a subdomain to the URL so users may believe they are dealing with an authentic website. An example: http://www.paypal.it.ascendancethe atrearts.co.uk.

Rule : If dots in domain $< 3 \rightarrow$ Legit
else if $= 3 \rightarrow$ Suspicious
else $\rightarrow$ Phishy

(6) **Fake HTTPs protocol/SSL final:** The existence of HTTPs protocol every time sensitive information is being transferred reflects that the user is certainly connected with an honest website. However, phishers may use a fake HTTPs protocol so that users may be deceived. So checking that the HTTPs protocol is offered by a trusted issuer such as GeoTrust, GoDaddy, VeriSign, etc., is recommended.

Rule : use of https & trusted issuer & age $\geqslant$ 2 years $\rightarrow$ Legit
using https & issuer is not trusted $\rightarrow$ Suspicious
else $\rightarrow$ Phishy

(7) **Request URL:** A webpage usually consists of text and some objects such as images and videos. Typically, these objects are loaded into the webpage from the same server of the webpage. If the objects are loaded from a domain other than the one typed in the URL address bar, the webpage is potentially suspicious.

Rule : request URL $< 22\% \rightarrow$ Legit
request URL $\geqslant 22\%$ and $< 61\% \rightarrow$ Suspicious
else $\rightarrow$ Phishy

(8) **URL of anchor:** Similar to the URL feature, but here the links within the webpage may point to a domain different from the domain typed in the URL address bar.

Rule : URL anchor $\% < 31\% \rightarrow$ Legit
URL anchor % $\geqslant$ and $\leqslant 67\% \rightarrow$ Suspicious
else $\rightarrow$ Phishy

(9) **Server Form Handler (SFH):** Once the user submitted his information; the webpage will transfer the information to a server so that it can process it. Normally, the information is processed from the same domain where the webpage is being loaded. Phishers resort to make the server form handler either empty or the information is transferred to somewhere different than the legitimate domain.

Rule : SFH If 'about : blank' or empty $\rightarrow$ Phishy
SHD redirects to different domain $\rightarrow$ Suspicious
else $\rightarrow$ Legit

(10) **Abnormal URL:** If the website identity does not match a record in the WHOIS database (WHOIS, 2011) the website is classified as phishy.

Rule : No hostname in URL $\rightarrow$ Phishy
else $\rightarrow$ Legit

(11) **Using Pop-up window:** Usually authenticated sites do not ask users to submit their credentials via a popup window.

Rule : rightClick disabled $\rightarrow$ Phishy
rightClick showing alert $\rightarrow$ Suspicious
else $\rightarrow$ Legit

(12) **Redirect page:** When users clicks on a link they may be unaware that he's redirected to a suspicious webpage. Redirection is commonly used by phishers to hide the real link and lures the users to submit their information to a fake site.

Rule : redirect page #s $\geqslant 1 \rightarrow$ legit
redirect page #s > 1 and $< 4 \rightarrow$ Suspicious
else $\rightarrow$ phishy

(13) **DNS record:** An empty or missing DNS record of a website is classified as phishy. Phishers aim to acquire sensitive information as fast as possible since the phishing webpage often lasts for short period of time and the URL is not valid any more. DNS record provides information about the domain that is still a live at the moment, while the deleted domains are not available on the DNS record.

Rule : No DNS record $\rightarrow$ Phishy
else $\rightarrow$ Legit

(14) **Hiding the links:** Phishers often hide the suspicious link by showing a fake link on the status bar of the browser or by hiding the status bar itself. This can be achieved by tracking the mouse cursor and once the user arrives to the suspicious link the status bar content is changed.

Rule : change of status bar onMouseOver $\rightarrow$ Phishy
no Change $\rightarrow$ Suspicious
else $\rightarrow$ Legit

(15) **Website traffic:** Legitimate websites usually have high traffic since they are being visited regularly. Since phishing websites normally have a relatively short life; they have no web traffic or they have low ranking. It has been recommended that a legitimate webpage has a rank less than or equal to 150,000 in the Alexadatabase (Alexa the Web Information Company, 2011).

$$\boxed{\begin{array}{c}\text{Rule : webTraffic} < 150{,}000 \rightarrow \text{Legit}\\ \text{webTraffic} > 150{,}000 \rightarrow \text{Suspicious}\\ \text{else} \rightarrow \text{Phishy}\end{array}}$$

(16) **Age of domain:** Websites that have an online presence of less than 1 year, can be considered risky.

$$\boxed{\begin{array}{c}\text{Rule : age} \leqslant \text{6 months} \rightarrow \text{Legit}\\ \text{else} \rightarrow \text{Phishy}\end{array}}$$

5. Experimental results

The process of detecting the type of a website is a classification problem where different features are utilised to learn hidden knowledge from them. This knowledge is in fact the classification system that in turn is used to automatically guess the type of the website when a user browses it. We have identified different features discussed earlier related to legitimate and phishy websites and collected over 1350 different websites from various sources. A sample of the phishing data (10 examples) for some features is shown in Table 6. Some of the collected features hold categorical values such as "Legitimate", "Suspicious" and "Phishy". These values have been replaced with the numerical values 1, 0 and −1, respectively.

Normally, there are two classes where a website can be classified into: Legitimate or phishy. The AC algorithm considered in this paper can discover not only rules associated with one class but also two classes, i.e. (legitimate or phishy). Meaning, MCAC algorithm can produce a new type of rule based on a new class label not previously seen in the training data set which we name "Suspicious". When a website is considered suspicious, it can be either phishy or legitimate. Based on the weights assigned to the test data by the rule, the end-user can make a more accurate decision.

Different AC and rule based algorithms have been used to evaluate the performance and applicability of AC in particular the MCAC algorithm on the data set collected. The main algorithms used in the experiments beside MCAC are (CBA, MCAR and MMAC) from AC community, and (C4.5, PART, RIPPER) for rule induction and decision trees. The selection of these algorithms is based on the fact that they are rule based and they use different learning methodologies for fair comparison.

The experiments were conducted on an I3 machine with 2.3 Ghz processor. The experiments of PART, C4.5 and RIPPER were carried out in Weka software (Weka, 2011). Weka stands for "Waikato Environment for Knowledge Analysis" which is a java open source code. It contains many implementations of a number of data mining algorithms performing different tasks like clustering, regression, association rule, classification and feature selection. For AC algorithms, MMAC and CBA source codes have been obtained from their prospective authors and (MCAR, MCAM) were implemented in Java.

Several researchers in association rule discovery and AC, i.e. (Costa et al., 2013; Thabtah et al., 2004), have pointed out that the *minsupp* threshold often controls the numbers of rules generated and the training time consumed during the rule discovery and production steps. Thus, we have followed other data mining scholars in setting the support threshold to 2% in the experiments of CBA, MMAC, MCAR and MCAC. The confidence threshold, however, has less impact on the general performance of AC algorithms and we set it to 50%.

The main measures used for the algorithms results evaluation are:

(1) Classifiers one-error rate (%) or accuracy.
(2) Classifier size (# of rules).
(3) Multi-label rules generation by MCAC.
(4) Reduced features set assessment and its impact on accuracy.

5.1. Results analysis

Fig. 2 summarises the prediction accuracy (%) produced by the considered algorithms for the phishing problem data set. It is obvious from the graph that the MCAC algorithm outperformed the other AC algorithms and the rule based ones in predicting the type of the websites. In particular, the MCAC algorithm outperformed RIPPER, C4.5, PART, CBA, and MCAR with 1.86%, 1.24%, 4.46%, 2.56%, 0.8%, respectively. Overall, the prediction accuracy obtained from all algorithms is acceptable and that reflects features goodness. One main reason for achieving higher predictive accuracy

Table 6
Sample phishing data for ten selected features.

URL anchor	Request URL	SFH	URL length	Having '@'	Prefix suffix	IP	Sub domain	Web traffic	DNS record	Class
0	−1	0	−1	1	1	1	−1	0	1	1
1	1	0	0	1	1	1	0	−1	1	1
1	1	1	0	−1	1	1	0	1	1	−1
0	0	1	−1	1	1	1	1	−1	1	1
−1	−1	0	0	1	1	−1	0	0	−1	−1
1	1	1	0	1	1	1	1	−1	1	1
1	1	1	1	−1	−1	1	1	1	−1	−1
1	1	1	1	1	1	−1	1	1	1	−1

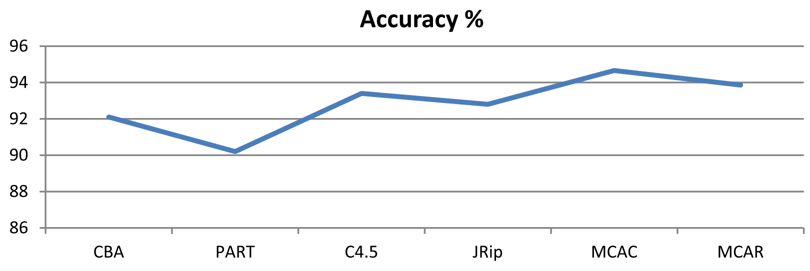

Fig. 2. The classification accuracy (%) for the contrasted algorithms derived from the phishing data.

by the MCAC algorithm is its ability not only to extract one class per rule but also all possible classes in a disjunctive form. These extra rules are usually tossed out by the existing AC algorithms and can contribute positively in predictive power as well as serving the end-users needs. In other words, in some test cases, some rules in MCAC classifier are associated with two class labels. These rules were able to correctly predict the test cases class especially for those websites that are suspicious whereas the other algorithms have misclassified them simply since they assigned these test cases one class, i.e. phishy.

Fig. 3 displays the number of rules generated by all algorithms against the considered data set. The figure stresses that AC algorithms especially MCAR generates a large number of rules if contrasted to decision trees, rule induction or hybrid classification (PART). The main cause of the larger classifiers of the AC algorithms is because they all evaluate every single correlation between the attribute values and each class value during the training phase. This has been inherited from association rule mining and allows a training case to be used more than once in learning rules unlike traditional classification algorithms that allow each training case to be used only once for a particular rule. This explains its smaller size classifiers.In Fig. 4, we evaluate the multi-label rules using two evaluation measures named “Any-label” and “Label-weight”. The “Any-label” considers 100% correct classification, i.e. (1), when any of the multi-label rule’s class matches the test case class. This explains its higher rates within Fig. 4. On the other hand, the “Label-weight” measure gives the test data the true weight of the rule’s class that was given to it. To elaborate, consider for instance 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. The “Label-Weight” assigns the predicted class’s weight, i.e. (30/50 or 20/50), to the test case if the predicted class matches the actual class of the test case. Whereas, “Any-label” method assigns “1” to the test case in the same scenario when any of the rule’s class matches that of the test data. So for R, if the test case class is either l_3 or l_1 the “Any-label” method assigns the test case “1”, whereas, if the test data actual class is l_3, the label-weight assigns 20/50 to it.

For Fig. 4, MCAC algorithm outperformed the MMAC algorithm in both label-weight and any-label evaluation measures on the phishing data. The increase of the predictive accuracy for both evaluation measures for the MCAC algorithm is due to the new extracted knowledge identified. Therefore, rather than classifying websites that are neither (phishy nor legitimate) wrongly, the new rules discovered by MCAC are able to cover these websites by assigning a suspicious class label and improving the predictive performance of the classifiers. Lastly, questions such as “is the website close to the phishy or legitimate class?” and “by how much?” are answered. Another possible contributor to MCACs performance in accuracy is its ability to reduce the default class usage during the prediction step in which if no rules are applicable to the test case the prediction procedure of MCAC takes on the group of rules partly matching the test data and assigns the largest group class to the test case.

To signify the importance of MCAC’s new rules, Fig. 5 displays the number of multi-labels rules with respect to their consequent part (class labels on the right hand side). MCAC was able to extract multiple labels rules from the phishing data set. They are connected with a new class (phishy or legitimate) to deal with test examples that are neither fully phishy or legitimate. This is accomplished using weights associated with the class labels in the discovered rules. In particular, Fig. 5 shows that the MCAC algorithm generated 24 multiple labels rules that represent the “Legitimate Or Phishy” class. These rules are linked to websites that are suspicious and mainly classified by most current classification algorithms as “Phishy”. The fact that MCAC algorithm finds new rules is an indicator on the ability of the algorithm discovering data insights most current AC algorithms are unable to detect.

5.2. Reduced phishing features results

We have assessed the collected features further in order to identify the smallest significant et of features that allow guessing the type of websites. In addition, selecting a small set of features may eliminate the noise in choosing features, which occurs whenever there are irrelevant features presented within the training

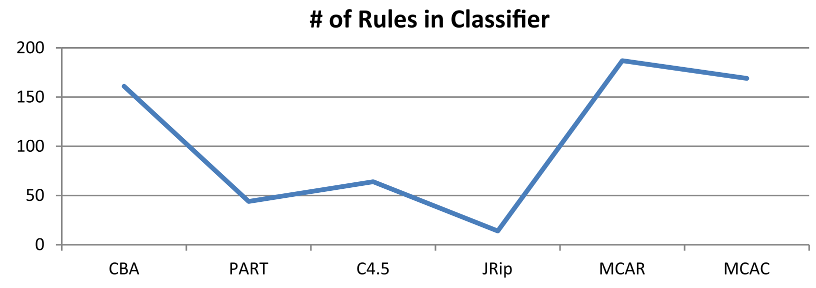

Fig. 3. Average number of rules generated by the contrasted algorithms derived from the phishing data problem.

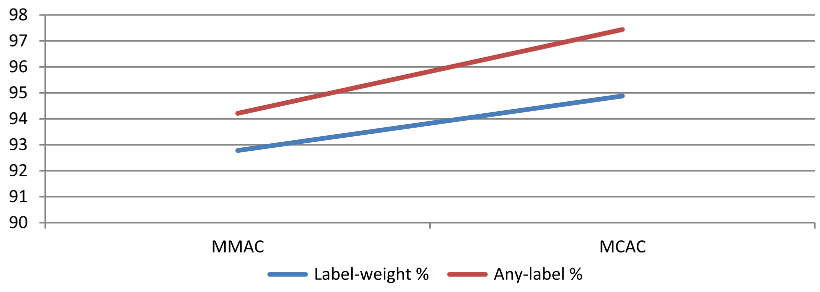

Fig. 4. Accuracy % computed using Label weight and Any label measures for MCAC and MMAC algorithms.

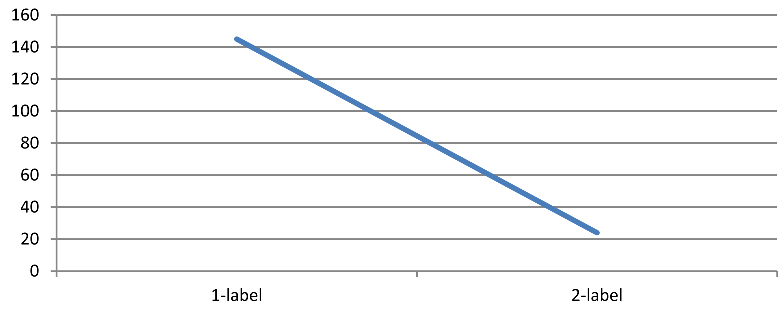

Fig. 5. Number of class labels per rule derived by the MCAC algorithm from the phishing data.

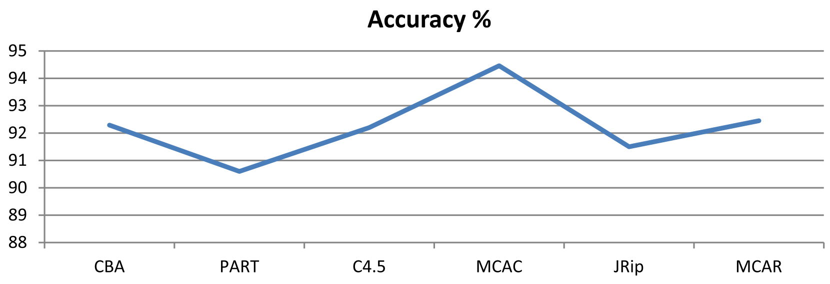

Fig. 6. The accuracy (%) for the contrasted algorithms derived from the reduced features set of the phishing data set.

data set. We applied Chi-Square measure (Witten and Frank, 2002) to further reduce the selected different features that we have collected. The aim of this assessment is to end up with the smallest set of features. We have employed Chi-Square filter in Weka (2011) as a feature selection criterion to accomplish the above task. Chi-square evaluates the relevancy of variables for classification problems. It is a known data hypothesis method from statistics, which evaluates the correlation between two variables and determines whether they are correlated. For our data set, each feature correlation with the class attribute has been evaluated. The test for correlation when applied to a population of objects determines whether they are positively correlated or not.

Chi-Square has been employed to assess the relevancy of attributes in classification data in many practical domains, e.g. (Thabtah et al., 2009). The evaluation of the 16 website features showed that 9 features have correlation with the class attribute values and therefore they may impact on the process of phishing detection. These are: Request URL, Age of Domain, HTTPS and SSL, Website Traffic, Long URL, SFH, Pop-Up window, URL of Anchor, Redirect URL and Using the IP Address.

We assess the accuracy of the same classification algorithms used previously against the reduced features set of the data. Fig. 6 displays the classification accuracy on the reduced number of features of all algorithms considered. We noticed that the classification accuracy was not heavily impacted and on average for all classification algorithms used, the accuracy has been reduced only by 0.6% if contrasted with that derived from all features set. This reflects how good the reduced features set are in classifying the type of the websites.

6. Conclusions

Detecting the phishing websites is one of the crucial problems facing the internet community because of its high impact on the daily online transactions performed. AC is a promising intelligent approach that recently attracted researchers due to its high predictive performance and the simple classifiers it derives. The website phishing problem has been investigated in this article in which we develop an AC data mining method to discover correlations among features and produces them in simple yet effective rules. Unlike other intelligent methods our method discovers new rules that are connected with more than one class giving the user new type of useful information. These rules also enhanced the classification accuracy in detecting phishy websites according to the experimental section. Moreover, we were able to identify significant features related to phishing websites using frequency analysis and Chi-square feature selection method. The performance of our AC method before and after the feature selection process showed consistency in prediction accuracy, which reveals the features high quality. In addition, the paper critically analysed common classic and intelligent phishing detection methods in the literature. Experimental results using real phishy and legitimate websites that have been collected from different sources have been conducted to show the pros and cons of our method. In the experiments, we used known AC and rule based classification methods (CBA, MCAR, MMAC, PART, C4.5). The measures of evaluation are label-weight, any-label, accuracy and number of rules. The results revealed that our method outperformed the considered methods on detecting phishing with respect to accuracy rate. Further, the label-weight and any-label results of MCAC are better than those of the MMAC for the same phishing data. More importantly, MCAC was able to produce multi-label rules from the phishing data generating rules associated with a new class called "Suspicious" that was not originally in the training data set. This has enhanced further its classifiers predictive performance. In near future, we would like to consider content based features to increase the collected features set which we believe will be a potential research direction since it helps in understanding the behaviour of the attackers and could possible improve the performance of our method. Finally, we intend also to utilise test websites once they are classified as training data which will make the phishing model incremental.

References

Aaron, G., & Manning, R. (2012). *APWG phishing reports*. <http://www.antiphishing.org/resources/apwg-reports/>.

Abdelhamid, N., Ayesh, A., & Thabtah, F. (2013) Associative classification mining for website phishing classification. In *Proceedings of the ICAI '2013* (pp. 687–695), USA.

Abdelhamid, N., Ayesh, A., Thabtah, F., Ahmadi, S., & Hadi, W. (2012). MAC: A multiclass associative classification algorithm. *Journal of Information and Knowledge Management, 11*(2), 1250011-1–1250011-10.

Aburrous, M., Hossain, M. A., Dahal, K., & Thabtah, F. (2010a). Predicting phishing websites using classification mining techniques. In *Seventh international conference on information technology; 2010* (pp. 176–181). Las Vegas, Nevada, USA: IEEE.

Aburrous, M., Hossain, M. A., Dahal, K., & Thabtah, F. (2010b). Intelligent phishing detection system for e-banking using fuzzy data mining. *Expert Systems with Applications: An International Journal*, 7913–7921.

Afroz, S., & Greenstadt, R. (2011). PhishZoo: Detecting phishing websites by looking at them. In *Proceedings of the 2011 IEEE fifth international conference on semantic computing (ICSC '11)* (pp. 368–375). Washington, DC, USA: IEEE Computer Society.

Alexa the Web Information Company (2011). *Alexa the Web Information Company*. <http://www.alexa.com/>.

Chen, J., & Guo, C. (2006). Online detection and prevention of phishing attacks. In *IEEE communications and networking, ChinaCom '06* (pp. 1–7).

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297.

Costa, G., Ortale, R., & Ritacco, E. (2013). X-class: Associative classification of xml documents by structure. *ACM Transactions on Information and System Security, 31*(1), 3.

Dede (2011). <http://blog.sucuri.net/tag/blacklisted> (accessed June 25, 2013).

Dhamija, R., Tygar, J. D., & Hearst, M. (2006). Why phishing works. In *Proceedings of the SIGCHI conference on human factors in computing systems; 2006* (pp. 581–590). Cosmopolitan Montréal, Canada: ACM.

Dhamija, R., & Tygar, J. (2005). The battle against phishing dynamic security skins. In *Proceedings of the 1st symposium on usable privacy and security; July, 2005* (pp. 77–85). New York, NY, USA: ACM Press.

Gartner Inc. (2011). <http://www.gartner.com/technology/home.jsp>.

Guang, X., Jason, O., Carolyn, P. R., & Lorrie, C. (2011). CANTINA+: A feature-rich machine learning framework for detecting phishing web sites. In *ACM transactions on information and system security* (pp. 1–28).

Horng, S. J., Fan, P., Khan, M. K., Run, R. S., Lai, J. L., & Chen, R. J. (2011). An efficient phishing webpage detector. *Expert Systems with Applications: An International Journal, 38*(10), 12018–12027.

Jabbar, M. A., Deekshatulu, B. L., & Chandra, P. (2013). Knowledge discovery using associative classification for heart disease prediction. *Advances in Intelligent Systems and Computing, 182*(2013), 29–39.

James, L. (2005). *Phishing Exposed*. s.l.: Syngress Publishing.

Kaspersky Lab (2013). <http://www.kaspersky.com/about/news/spam/2013/>.

Kirda, E., & Kruegel, C. (2005). Protecting users against phishing attacks with antiphish. In *Proceedings of the 29th annual international computer software and applications conference (COMPSAC)* (pp. 517–524), 2005.

Kunz, M., & Wilson, P. (2004). Computer crime and computer fraud report. Submitted to the Montgomery County Criminal Justice Coordinating Commission.

Liu, J., & Ye, Y. (2001). Introduction to e-commerce agents: marketplace solutions, security issues, and supply and demand. In *E-commerce agents, marketplace solutions, security issues, and supply and demand; 2001* (pp. 1–6). London, UK.

Liu, W., Huang, G., Xiaoyue, L., Min, Z., & Deng, X. (2005). Detection of phishing webpages based on visual similarity. In *Proceeding WWW '05 special interest tracks and posters of the 14th international conference on world wide web; 2005* (pp. 1060–1061). New York, NY, USA: ACM.

Millersmiles (2011). *Millersmiles*. <http://www.millersmiles.co.uk/>.

Miyamoto, D., Hazeyama, H., & Kadobayashi, Y. (2008). An evaluation of machine learning-based methods for detection of phishing sites. *Australian Journal of Intelligent Information Processing Systems, 2*, 54–63.

Mohammad, R., Thabtah, F., & McCluskey, L. (2012). An assessment of features related to phishing websites using an automated technique. In *The 7th international conference for internet technology and secured transactions (ICITST-2012)*. London: ICITST.

Pan, Y., & Ding, X. (2006). Anomaly based web phishing page detection. In *ACSAC '06: Proceedings of the 22nd annual computer security applications conference* (pp. 381–392). Washington, DC: IEEE.

PhishTank (2006). *PhishTank*. <http://www.phishtank.com/>.

Qin, X. J., Zhang, Y., Li, X., & Wang, Y. (2010). Associative classifier for uncertain data. In *Web-age information management. 11th international conference on web-age information management (WAIM 2010)* (pp. 692–703), Jiuzhaigou, China, 15–17 July 2010.

Sadeh, N., Tomasic, A., & Fette, I. (2007). Learning to detect phishing emails. In *Proceedings of the 16th international conference on world wide web* (pp. 649–656).

Sanglerdsinlapachai, N., & Rungsawang, A. (2010). Using domain top-page similarity feature in machine learning-based web. In *Third international conference on knowledge discovery and data mining; 2010* (pp. 187–190). Washington, DC: IEEE.

Seogod (2011). *Black Hat SEO*. <http://www.seobesttools.com/black-hat-seo/>.

Sheng, S., Wardman, B., Warner, G., Cranor, L. F., Hong, J., & Zhang, C. (2009). An empirical analysis of phishing blacklists. In *CEAS*.

Song, M. (2009). *Handbook of research on text and web mining technologies*. Information science reference, IGI global.

Thabtah, F., Eljinini, M., Zamzeer, M., & Hadi, W. (2009). Naïve Bayesian based on chi square to categorize Arabic data. In *Proceedings of the 11th international business information management association conference (IBIMA) conference on innovation and knowledge management in twin track economies* (pp. 930–935).

Thabtah, F. (2007). Review on associative classification mining. *Journal of Knowledge Engineering Review, 22*(1), 37–65.

Thabtah, F., Cowling, P., & Peng, Y. (2005). MCAR: Multi-class classification based on association rule approach. In *Proceeding of the 3rd IEEE international conference on computer systems and applications* (pp. 1–7). Cairo, Egypt.

Thabtah, F., Cowling, P., & Peng, Y. (2004). MMAC: A new multi-class, multi-label associative classification approach. In *Proceedings of the fourth IEEE international conference on data mining (ICDM '04)* (pp. 217–224). Brighton, UK.

The Government of Australia (2011). *Hackers, Fraudsters and Botnets: Tackling the problem of cyber crime*. Report on Inquiry into Cyber Crime.

Wang, X., Yue, K., Niu, W., & Shi, Z. (2011). An approach for adaptive associative classification. *Expert Systems with Applications: An International Journal, 38*(9), 11873–11883.

WEKA (2011). Data Mining Software in Java. Retrieved December 15, 2010 from <http://www.cs.waikato.ac.nz/ml/weka>.

Witten, I., & Frank, E. (2002). *Data mining: Practical machine learning tools and techniques with Java implementations*. San Francisco: Morgan Kaufmann.

WhoIS (2011). Available from: <http://who.is/>.

Zhang, Y., Hong, J., & Cranor, L. (2007). CANTINA: A content-based approach to detect phishing web sites. In *Proceedings of the 16th world wide web conference; 2007*, Banff, Alberta, Canada.