Phishing Website Classification

Ruben Teimas

Universidade de Évora, Évora, Portugal
January 31, 2021
`m47753@alunos.uevora.pt`

Abstract: Phishing is one of the most common cyberattacks. It impersonate emails or websites from legitimate sources in order to obtain personal information from a target.
The goal of this experiment is to evaluate the legitimacy of websites by applying data mining and machine learning techniques/algorithms using a publicly available data set.
This was achieved comparing classification algorithms using cost sensitive classification and class selection. The best result was obtained using *SVM* with only 2 classes (legitimate, phishing).

Keywords: Phishing · Data Mining · Machine Learning

1 Introduction

Since the *Covid-19* pandemic started there has been a rise in cyberattacks with Phishing being one of the most common [2].

Even though it is one of the simplest attacks it is also one of the most effective, specially, but not exclusively, amongst older people [3] which tend not to have much technology literacy (in some countries no literacy at all).

One of the best ways to prevent these attacks from happening is to spread awareness about the subject, but as awareness is being spread the attacks are getting more sophisticated and harder to detect by the common user. That being said, another way to prevent the attacks is to apply data mining techniques / machine learning algorithms to detect and neutralize phishing websites.

In recent years some studies have been conducted in order to answer this need. The study conducted by Aburrous, Hossain, Thabatah, & Dahal [6] focused on applying fuzzy-logic and data mining techniques to 27 features extracted from websites. They got 86% of classification accuracy with 10-folds cross validation. Another study, conducted by Huang, Qian & Wang [4], proposed the use of support vector machines (*SVM*) to detect phishing websites `URL`'s, the features vector contains 23 features and the study got an 99% accuracy.

2 Data

The data set [7] was taken from *UCI Machine Learning repository*, it has 10 attributes and 1353 instances with no missing values.

It was collected and treated in a study [1] conducted by N. Abdelhamid, A. Ayesh & F. Thabtah.

Table 1. Attribute description.

Name	**Considerations**	**Values**
SFH(Server Form Handler)	Take user data. Usually void in phishing sites.	{-1; 0 ;1}
popUpWindow	Phishing sites usually have a lot of pop-ups.	{-1; 0; 1}
SSLfinal_State	SSL is not used in most phishing sites.	{-1; 0; 1}
Request_URL	External objects. Inconsistent with domain in phishing sites.	{-1; 0; 1}
URL_of_Anchor	Similar to *Request_URL*	{-1; 0; 1}
web_traffic	Lots of traffic in short periods of time, in phishing sites.	{-1; 0; 1}
URL_Length	Not always easy to differentiate.	{-1; 0; 1}
age_of_domain	Phishing sites usually have very recent domains.	{-1; 1}
having_IP_Address	Some phishing sites have *IP* in *URL*.	{0; 1}
Result	The class of the classification	{-1; 0; 1}

The attribute *Result* can take the value of the class: Phishing, Suspicious and Legitimate, which are represented by -1, 0 and 1 respectively. The table below (*Table 2*) describes the remaining attributes values.

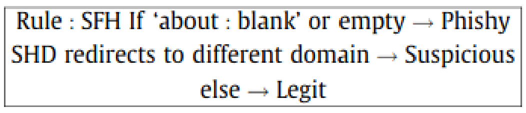
Table 2. Value description per attribute.

Name	**-1**	**0**	**1**
SFH	No SFH	Not known	Has SFH
popUpWindow	No pop-ups	Not known	Has pop-ups
SSLfinal_State	No SSL	Not known	SSL is used
Request_URL	Inconsistent with domain	Not known	Consistent with domain
URL_of_Anchor	Inconsistent with domain	Not known	Consistent with domain
web_traffic	Small traffic in short time	Not known	Large traffic in short time
URL_Length	Big length	Normal length	Small length
age_of_domain	Recent	-	Old
having_IP_Address	-	No *IP* or *URL*	*IP* on *URL*

An attribute's value is one of the classes, therefore it is nominal and discrete.

To obtain the attribute's values for each website they used Multi-label Classifier based Associative Classification (MCAC), which allowed simple rules to be created for multiple classes as it can be seen in **Fig.1**.

Fig. 1. Example rule for SFH


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

3 Proposed solutions

Given the nature of the data set, in which we want to predict a target class, Classification algorithms were used.

Multiple classification techniques were chosen: decision based rule (RIPPER), decision tree based method (C4.5 and Random Forest), Naive Bayes and Support Vector Machines.
These algorithms were used with Cost Sensitive Classification and 2 different Cost Matrix, where the columns represent what the instances were classified as and the rows represent the actual classification. Since we're using 3 classes we'll have a *3*x*3* matrix, with *a*, *b* and *c* on both columns and rows, where a=0, b=1, c=-1.

$$M1 = \begin{array}{c c} \begin{bmatrix} 0 & 1 & 2 \\ 1 & -1 & 2 \\ 1 & 4 & -1 \end{bmatrix} & \begin{array}{c} a \\ b \\ c \end{array} \end{array}$$

$$M2 = \begin{array}{c c} \begin{bmatrix} 0 & 3 & 1 \\ 1 & -1 & 2 \\ 3 & 4 & -1 \end{bmatrix} & \begin{array}{c} a \\ b \\ c \end{array} \end{array}$$

The *M1* matrix enforces great punishment in classifying a Phishing site as a Legitimate one, as it is the worst case possible. It also enforces punishment the other way around, but not as much. It greatly rewards the right classification of Legitimate and Phishing sites. The *M2* matrix is very similar to *M1*, but it enforces greater punishment in misclassifying *suspicious* sites. This approach can be taken in order to protect the user, as a *suspicious* website can be classified as legitimate.

Besides the use of cost matrices, another approach taken is the comparison of using 3 classes (legitimate, suspicious, phishing) with using only 2 classes (legitimate, phishing). This is an interesting approach because suspicious sites might be something we want to avoid, as they give a dubious hint to the user.
M1 and *M2* only differ in the suspicious class, therefore there's no need to use both matrices when using only 2 classes.

A feature selection approach was considered, but given the small number of features and its relevance to the classification it was not applied.

The K-fold cross-validation was used with all the algorithms. It divides the data set into k parts/subsets and it repeats the method k times. Each of the times it repeats, one of the k subsets is used as the test set and the other k-1 subsets are joined in order to make a training set. In these experiment, *K=10*.

To apply these algorithms the ***Weka*** [8] software was used. *Weka* is a machine learning open-source software implemented in Java, it has a great number of built-in tools with focus on data mining.

All of the algorithms were applied using *Weka*'s default parameters.

4 Algorithms

4.1 RIPPER

Rule based algorithm with good results on multiple class classification. It takes all the available classes and rearranges them from the least to the most frequent.

Having a set of classes {C1, C2,..., Cn}, class C1 is the least frequent and Cn is the most frequent, therefore, the default class. The algorithm then derives the rules from the minority to majority class.

In the *Weka* software this algorithm is called ***JRip*** [9].

4.2 C4.5

For each node, this decision tree chooses the attribute of the data that most effectively splits its set of samples into subsets that fits each class.

The splitting criteria is the normalized *information gain.* The attribute with the highest normalized information gain is chosen to make the decision.

In *Weka* the algorithm is called ***J48***. [10]

4.3 Random Forest

The Random Forest [12] algorithm is an expansion of decision tree.

Instead of creating a decision tree it randomly splits the data and creates multiple trees, hence the name random forest.

4.4 Naive Bayes

The Naive Bayes [11] classifier uses estimator classes in order to classify the data and is based on applying the Bayes theorem with strong independence assumptions between attributes. Usually it performs well on multiclass prediction.

4.5 Support Vector Machine (SVM)

The SVM algorithm aims to find a hyper-plane in a N-dimensional space (in which N is the number of attributes, this case: 10). that distinctly classifies the data points.

A variant of this algorithm can be used in *Weka* by selecting *SMO* [13]. The multiclass problems are solved using pairwise classification.

5 Results

By applying the algorithms to the data set a group of confusion matrices were obtained and later used, along with the cost matrices, to calculate the cost of classification.

To automate the cost calculus a small script in python was made.

Table 3. Using *M1* cost matrix with 3 classes

Algorithm	**Correctly Class(%)**	**Cost**
RIPPER	90.61	-797
C4.5	90.61	-837
Random Forest	89.58	-781
Naive Bayes	84.55	-781
SVM	85.80	-763

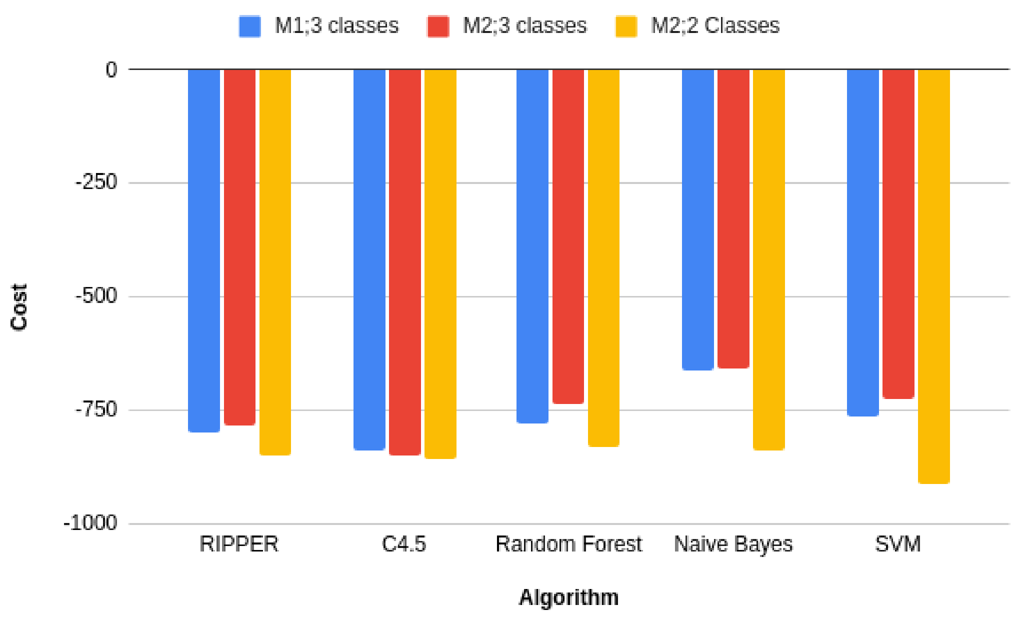
Table 4. Using *M2* cost matrix with 3 classes

Algorithm	Correctly Class(%)	Cost
RIPPER	90.61	-783
C4.5	90.61	-850
Random Forest	89.58	-736
Naive Bayes	84.55	-659
SVM	85.80	-725

Table 5. Using *M2* cost matrix with 2 classes

Algorithm	Correctly Class(%)	Cost
RIPPER	90.61	-849
C4.5	90.61	-856
Random Forest	89.58	-829
Naive Bayes	84.55	-839
SVM	92.8	-912

Fig. 2. Cost for each algorithm with different approaches



In order to better visualize the cost for each algorithm in each approach, a bar chart was created.

Looking at the chart it is possible to observe that, no matter what approach, the *C4.5* algorithm presented some of the best results, with small costs compared to the other algorithms. *RIPPER* also presented good results.

The results of using *M1* cost matrix and *M2* cost matrix with 3 classes were very similar, as it was expected, since the matrix only differed in the suspicious class. Overall the *M2* matrix had slightly worse results than the *M1*, except for the *C4.5* algorithm.

The use of only 2 classes boosted the results of the previous approaches. This can be explained by getting rid of the entropy and confusion that the *suspicious* class caused in the data set. For this approach *C4.5* had the second best score, staying behind the *SVM*.

SVM classifiers were originally used for binary classification, therefore, by getting rid of one class, it allowed this algorithm to perform the best result of this experiment, with the highest percentage of correctly classified instances and the smallest cost. This algorithm had already presented good results in previous work [4] of this subject

6 Conclusions and future work

By applying the classification algorithms to the data set we got a good percentage of correctly classified instances, specially using *C4.5* and *RIPPER* with the 3 original classes and with *SVM* by using only 2 classes.

Nonetheless, there is still a lot to improve, as non of the methods provided a percentage of classified instances above 92%. When it comes to such a sensitive problem, where people loose lots of money, studies must be continued with different classification algorithms and different approaches.

In the future, given the good results of the *C4.5*, ensemble methods could be used, but this is only one of the many experiments that can be made.

References

1. Abdelhamid, N., Ayesh, A., & Thabtah, F.A. (2014). Phishing detection based Associative Classification data mining. Expert Syst. Appl., 41, 5948-5959.
2. Eian, I.C.; Yong, L.K.; Li, M..Y.X.; Qi, Y.H.; Z, F. Cyber Attacks in the Era of COVID-19 and Possible Solution Domains. Preprints 2020, 2020090630 (doi: 10.20944/preprints202009.0630.v1).
3. Oliveira, D., Rocha, H., Yang, H., Ellis, D., Dommaraju, S., Muradoglu, M., Weir, D., Soliman, A., Lin, T., & Ebner, N.C. (2017). Dissecting Spear Phishing Emails for Older vs Young Adults: On the Interplay of Weapons of Influence and Life Domains in Predicting Susceptibility to Phishing (doi:10.1145/3025453.3025831).
4. Huajun Huang, Liang Qian and Yaojun Wang, 2012. A SVM-based Technique to Detect Phishing URLs. Information Technology Journal, 11: 921-925 (doi:10.3923/itj.2012.921.925).
5. Moghimi, Mahmood & Varjani, A.. (2016). New Rule-Based Phishing Detection Method. Expert Systems with Applications. 53. (doi:10.1016/j.eswa.2016.01.028).
6. Aburrous, M., Hossain, M., Thabatah, F., & Dahal, K. (2010). Intelligent detection sys-tem for e-banking phishing websites using fuzzy data mining.Expert Systemswith Applications, 37(12), 7913–7921. (doi:10.1016/j.eswa.2010.04.044).
7. Website Phishing Data Set `https://archive.ics.uci.edu/ml/datasets/Website+Phishing` (Visited on 22/12/2020)
8. Weka. `https://www.cs.waikato.ac.nz/ml/weka/` (Visited on 31/01/2021)
9. Weka. JRip `https://weka.sourceforge.io/doc.stable/weka/classifiers/rules/JRip.html` (Visited on 22/12/2020)
10. Weka. J48 `https://weka.sourceforge.io/doc.dev/weka/classifiers/trees/J48.html` (Visited on 22/12/2020)
11. Weka. Naive Bayes `https://weka.sourceforge.io/doc.dev/weka/classifiers/bayes/NaiveBayes.html` (Visited on 23/12/2020)
12. Weka. Random Forest `https://weka.sourceforge.io/doc.dev/weka/classifiers/trees/RandomForest.html`
13. Weka. SMO `https://weka.sourceforge.io/doc.dev/weka/classifiers/functions/SMO.html`