Detection of Phishing Websites using Machine Learning Approaches

Aditi Sharma Kuchi
College of Sciences
University of New Orleans
New Orleans, USA
askuchi@uno.edu

Suraj Gattani
College of Sciences
University of New Orleans
New Orleans, USA
sggattan@uno.edu

***Abstract*—The goal of this project is to analyze and detect phishing websites from a dataset containing both phishing and non-phishing website data. We tried to use both 10-fold cross validation method as well as independent testing of the dataset. Our methods achieved better accuracy than a paper published in early 2018, called "Artificial Neural Network for Websites Classification with Phishing Characteristics."**

We reached an accuracy of 96.9% without using feature selection, and 97.2% after selecting the best features for 10-fold cross validation. These results further increase with the independent testing. Independent testing resulted in an accuracy of 96.8% without feature selection, and 98.4% with feature selection.

Keywords— Machine Learning, phishing, UCI database, stacking, feature selection, Artificial Intelligence, Pattern Recognition, Social Engineering, Security

I. INTRODUCTION

Modern software development is incremental and evolutionary. This makes attacks on users a lot more commonplace than 10 years ago. Several threats are propagated by malicious websites largely classified as phishing. Phishing is a technique that is used by attackers on the Internet to commit fraud. They try to lure users and steal their personal information.

Phishing is a widely used strategy for spreading malware such as viruses and Trojans. The attackers often use social engineering tactics to address the victims, causing their social networking accounts to be infected and used to spread the coup. The most common method of spreading malicious software is through sending spam emails with links to websites that use social engineering to trick the users into giving up their information. These links direct the user to contaminated sites. Over time, the scams diversified, and even used real events to take advantage of the curiosity of the unsuspecting Internet users.

Its function is to obtain important information of users with the intention of using them for criminal practice. For example, obtaining data of bank accounts, passwords, number of credit cards among others confidential information of individuals or companies, which are then subsequently used fraudulently.

Although there exist some solutions to detect phishing attacks, there is still a lack of accuracy in state-of-the-art phishing detection systems. Novel machine learning techniques can be applied to detect these attacks with greater accuracy.

The objective of this study is to use data from the UCI Machine Learning Repository to train a model to recognize phishing and non-phishing data.

II. THEORETICAL BACKGROUND

Phishing and its methods:

Vulnerable users are often tricked into sharing their personal details such as their bank information, credit card information, email addresses and passwords etc. Phishing is an online fraud technique used by criminals in the cyber world to steal bank information and other personal information in order to use them to commit fraud.

It is a play on the English word "fishing" – Criminals use this technique to "fish" for information from users who "bite the hook."

A phishing attempt can happen through websites or fake emails, (which in turn contain links to websites that are "phishy," which mimic the image of a famous and trusted company to be able to catch the attention of the victims. Typically, website content or phishing emails promise extravagant promotions for the user or ask them to update their bank details, avoiding account cancellation, etc.

The motivations behind phishing attacks are:

1. Financial gain: Attackers can use stolen account details to commit identity theft.
2. Hidden identity: Attackers use these stolen identities by selling them to other criminals who might want to hide their illegal activities.
3. Fame and notoriety

III. RESEARCH METHODS

MACHINE LEARNING TECHNIQUES USED:

Logistic Regression:

This machine learning technique is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. We implemented LOGREG (Hastie, et al., 2009; Szilágyi and Skolnick, 2006) with L2 regularization as another classifier to be used in staking framework. It measures the relationship between the dependent categorical variable (in our case: a

carbohydrate-binding or non-carbohydrate-binding) and one or more independent variables by generating an estimation probability using logistic regression. The parameter, C which controls the regularization strength is optimized to achieve the best 10-fold CV balanced accuracy using grid search (Bergstra and Bengio, 2012).

K-Nearest Neighbours:

This machine learning technique is used in pattern recognition. The k-nearest neighbours algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression: In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour. In k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbours. KNN (Altman, 1992) operates by learning from the *K* number of training samples closest in distance to the target point in the feature space. The classification decision is computed from the majority votes coming from the neighbors. In this work, the value of *K* was set to 9 and all the neighbors were weighted uniformly.

Gradient Boosting:

This machine learning technique, GBC (Friedman, 2001) involves three elements: (a) a loss function to be optimized, (b) a weak learner to make predictions and (c) an additive model to add weak learners to minimize the loss function. The objective of GBC is to minimize the loss of the model by adding weak learners in a stage-wise fashion using a procedure similar to gradient descent. The existing weak learners in the model are remained unchanged while adding new weak learner. The output from the new learner is added to the output of the existing sequence of learners in an effort to correct or improve the final output of the model. Here, we used 1,000 bosting stages where a regression tree was fit on the negative gradient of the deviance loss function.

Support Vector Machine:

We employed SVM (Vapnik, 1999) with the radial basis function (RBF) kernel as one of the classifiers to be used in stacking framework. SVM classifies by maximizing the separating hyperplane between two classes and penalizes the instances on the wrong side of the decision boundary. The performance of SVM with the RBF kernel relies on two parameters *C*, and γ. The RBF kernel parameter γ and the cost parameter *C* are optimized to achieve the best 10-fold CV balanced accuracy using a grid search (Bergstra and Bengio, 2012) technique.

IV. DATASET

For this study we used a data set that was readily available from the University of California's Machine Learning and Intelligent Systems Learning Center. the Phishing Websites Data Set contains 11,055 records, with 30 attributes each. Our Final dataset contains 31 columns and 11055 rows – representing 30 features and the class label (phishing / non-phishing). The table below shows a list of all the attributes of the database of phishing websites.

Attribute Name	Representation
having_IP_Address	{ -1,1 }
URL_Length	{ 1,0,-1 }
Shortining_Service	{ 1,-1 }
having_At_Symbol	{ 1,-1 }
double_slash_redirecting	{ -1,1 }
Prefix_Suffix	{ -1,1 }
having_Sub_Domain	{ -1,0,1 }
SSLfinal_State	{ -1,1,0 }
Domain_registeration_length	{ -1,1 }
Favicon	{ 1,-1 }
port	{ 1,-1 }
HTTPS_token	{ -1,1 }
Request_URL	{ 1,-1 }
URL_of_Anchor	{ -1,0,1 }
Links_in_tags	{ 1,-1,0 }
SFH	{ -1,1,0 }
Submitting_to_email	{ -1,1 }
Abnormal_URL	{ -1,1 }
Redirect	{ 0,1 }
on_mouseover	{ 1,-1 }
RightClick	{ 1,-1 }
popUpWidnow	{ 1,-1 }
Iframe	{ 1,-1 }
age_of_domain	{ -1,1 }
DNSRecord	{ -1,1 }
web_traffic	{ -1,0,1 }
Page_Rank	{ -1,1 }
Google_Index	{ 1,-1 }
Links_pointing_to_page	{ 1,0,-1 }
Statistical_report	{ -1,1 }
Result	{ -1,1 }

Table 1: Attributes from dataset

V. METHODOLOGY

For our purpose, we first used Weka to select the most important features. For this, we used an inbuilt Weka functionality called attribute selection. It functions on the basis of information gain. Attribute selection by Weka lists out each attribute from the .arff file in order of the information gained from each of these.

After this selection procedure, we discarded the features that had an information gain of 0 (zero). There were three such features. This process brought our total number of features down to 27.

Next, we used state-of-the-art machine learning processes and selected the best ones to include in our stacking algorithm. Some of the methods we implemented are:
Extra-Tree classifier, Bagging Classifier, Gradient Boosting Classifier, Logistic Regression, K-Nearest Neighbors, Support Vector Machine, and Random Forest Classifier.
For the Support Vector Machine, we used the RBF kernel by searching for the best parameter values.
SVM yielded a high accuracy of 97.114% for 10-fold cross validation.

After obtaining the accuracies and collection the probabilities of each of the classifiers mentioned above, we apply the stacking technique. Stacking operates on the concept that the collection of probabilities of each of the classifiers, whether weak or strong in the base layer are fed to a classifier in the meta layer. This helps increase the accuracy by a lot because each method in the base layer learns differently. All these characteristics are used by the classifier in the meta layer to predict the results with higher accuracy.

In our implementation, using a machine with 4 GB RAM, Intel core i3 processor, the process ran for almost 2.5 days.

In our stacking technique, we used Logistic Regression, K-Nearest Neighbors, Gradient Boosting Classifier, and Support Vector Machine in our base layer. Our meta layer classifier was Support Vector Machine.

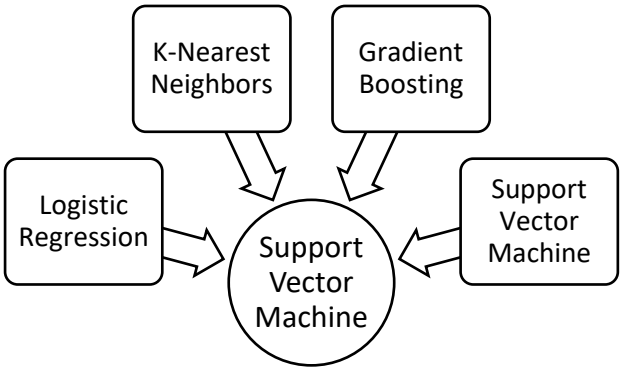

Figure 1: Stacking approach

VI. Results

Using our novel, state-of-the-art stacking technique for machine learning, we were able to surpass the results of a published paper in 2018 as referenced.

Our methods see an increase of almost 11% in training, and approximately 0.5% increase in testing.

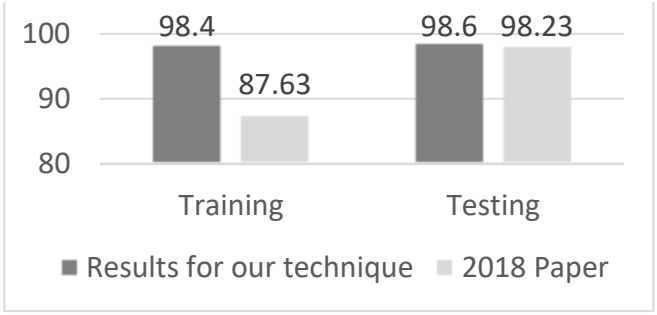

Figure 2: Chart showing our increased performance.

Threats to Validity

The major threat to validity in our project arise from the fact that the dataset itself is quite old. Even though our methods successfully detect these phishing websites, it is not certain whether they will be able to detect newer and more advanced versions of phishing.

Summary

The study resulted in a superior accuracy than that of one of the latest papers published in 2018.

We applied machine learning on a dataset of phishing websites and achieved very high accuracy for detection of phishing. 98.4%, 98.6% for training and testing respectively.

Acknowledgments

We would like to thank Dr. Minhaz Zibran for his guidance and support throughout the course.

We would also like to thank our peers, from whom we received excellent feedback and invaluable knowledge.

Also, the authors of the research papers from where we derived our project. The authors of the original paper who collected the dataset and made it possible for us to implement these methods.

References

[1] https://www.statisticssolutions.com/what-is-logistic-regression/
[2] https://file.scirp.org/pdf/SN_2018042615292358.pdf
[3] https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
[4] https://archive.ics.uci.edu/ml/datasets/phishing+websites.