# Using Machine Learning on Potentially Harmful Websites for Phishing Detection

Veronica Weiss, Skidmore College

## Abstract

In this paper, we explore how to effectively use rule-based modeling, decision trees, probabilistic modeling, neural networks, and gradient boosting to predict if a website is harmful or not. Our results also highlight the most significant website features to assess when evaluating if a website is malicious or not. After evaluating each model, we assess that gradient boosting outperforms and creates a near perfect model for this dataset.

## Introduction

The 2018 Verizon Data Breach Investigation Report claims that 56% of attacks on private businesses involve personal attacks on employees, and that phishing is one of the most used methods. Cisco defines phishing as "the practice of sending fraudulent communications that appear to come from a reputable source." This paper will draw on Cisco's definition of what phishing is and predict if a website is malicious or noft. A paper released in 2015 states that the two primary methods of phishing detection occurs with either upholding a blacklist or using heuristic-based methods. The blacklist method consists of maintaining and updating a list that contains all the phishing websites for the browser to block. A heuristic approach to phishing website detection is to extract features from the website and assess if that feature leads to an evaluation of if a website is malicious or not.[1] The dataset is comprised of variables that pertain to a website's URL, configuration, deployment, content, and reputation. Further research will expand to classifying if an email is phishing or not.

## Methodology

### Data Collection

The dataset this project is using is taken from University of California Irvine's Website Phishing Dataset located at http://archive.ics.uci.edu/ml/datasets/Website+Phishing#. This data consists of 4898 phishing websites and 6157 legitimate websites. The features' values have been coded as either a 1, 0, or -1 instead of classifying the values for each variable as "Legitimate," "Suspicious," or "Phishing."

A vast majority of the variables are binary categorical variables, with a value of -1 or 1. Few variables are ordinal, with a value of 1, 0, or -1.

This paper uses Repeated Incremental Pruning to Produce Error Reduction, C4.5 Decision Tree, Naïve Bayes, Neural

[1] Intelligent Rule based Phishing Websites Classification. (Mohammad, Thabtah, McCluskey) http://eprints.hud.ac.uk/id/eprint/17994/3/RamiIntelligent_Rule_based_Phishing_Websites_Classification_IET_Journal.pdf

Networks, XGBoost, and Microsoft's Light GBM as for binary classification.

## Classification with Ripper

The first method of classification is with the RIPPER (Repeated Incremental Pruning to Produce Error Reduction) model. The model made 21 rules for all 30 variables to predict the target. When using the five variables that were identified in the prior recursive feature elimination, the model made 12 rules.
This table was computed as the result:

| | |
|---|---|
| Correctly Classified Instances | 6331 |
| | 95.447% |
| Incorrectly Classified Instances | 302 |
| | 4.553% |
| Kappa statistic | 0.9075 |
| Mean absolute error | 0.0818 |
| Root mean squared error | 0.2022 |
| Relative absolute error | 16.5946 % |
| Root relative squared error | 40.7366% |
| Total Number of Instances | 6633 |

RIPPER results with a 95.4% accuracy on the training dataset. This method was then repeated on the testing dataset, which reported an approximately 96% accuracy. The kappa statistic is 0.9075, which means that 90.75% of the data is reliable.

The ruleset created by the model begins with an interesting split. If the website's URL length is greater than 75 characters, then it is automatically classified as phishing. If the website's URL length is less than 75 characters, the next variable assessed is if the website has a HTTPS certificate and if it is from a trusted issuer. This is where the first major split occurs. If the website has a certificate, but not from a trusted issuer, or no certificate at all, the model looks to see if the website contains URLs in its anchor tags. If the website has a trusted HTTPs certificate, the model looks at if the website has an adequate amount of web traffic or not.

## Classification with C4.5

The second method of classification is with the C4.5 Decision Tree. A C4.5 decision tree results with a 96% accuracy on this dataset. When this method was repeated on the testing dataset, the same accuracy was observed. The kappa statistic is also 0.9356, which means that 93.56% of the data is reliable.

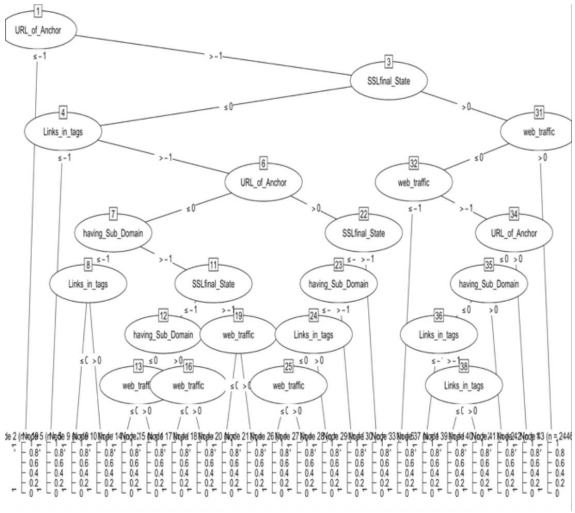


Figure 1: Decision tree produced by C4.5 with the 5 most statistically significant variables

This table was computed as the result:

| | |
|---|---|
| Correctly Classified Instances | 6423 |
| | 96.834% |
| Incorrectly Classified Instances | 210 |
| | 3.166% |
| Kappa statistic | 0.9356 |
| Mean absolute error | 0.0535 |

| | |
|---|---|
| Root mean squared error | 0.1635 |
| Relative absolute error | 10.8514% |
| Root relative squared error | 32.9415% |
| Total Number of Instances | 6633 |

## Recursive Feature Elimination

In order to find the five most statistically significant variables, recursive feature extraction was used. After recursive feature elimination with 25 repetition of bootstraps, we found the top 5 correlated variables as: if the website has a trusted HTTPS certificate, code in links on the website that is used to modify a redirect, if the website has a large amount of web traffic or not, if the URL has multiple sub domains, and if the <Meta> and <Script> tags on a website contained a URL. Recursive feature elimination uses a variable's accuracy and kappa statistic to evaluate its significance. In contrast to forward feature extraction search, recursive feature extraction is more effective in finding confounding variables.

The five most statistically significant variables for determining if a website is harmful or not, that resulted from recursive feature extraction, are somewhat logical in context. A vast majority of legitimate websites do have trusted certificates; however, it is somewhat easy to obtain one. Ideally an attacker would not make a simple oversight mistake of not obtaining a trusted a trusted certificate, so perhaps the websites in the dataset were not crafted by more skilled attackers. Legitimate websites do not need multiple subdomains in their website, as it would be inconvenient for customers and other targeted groups that produce web traffic. Attackers will often use multiple subdomains to have an official sounding identity, for example, such as creating "chase.abc.banking.com" to lure more gullible users to a financial scam. Designing a legitimate website with multiple subdomains would be unnecessary, so the context behind that variable being statistically significant is correct. Web traffic is also a logical statistically significant variable. The main source of a phishing website's web traffic would depend on the popularity and how widespread the phishing campaign grew. A better metric to measure web traffic than the total amount of users could be the rate of how much traffic the website gains. For example, a phishing website might gain as much web traffic as a legitimate one; however, a phishing website will most likely gain a more sporadic amount of high web traffic than a legitimate one. A common attribute of a phishing website is when a website contains code that pertains to dealing with redirecting the user to a different website. The act of redirecting a user, called an open redirect, is a main point of many commonly known application vulnerability attacks such as cross-site scripting and CSRF attacks. The variable that denotes if a website has code that pertains to processing a redirect in some form, such as "#", "#skip", or "JavaScript:: void(0)." It is logical for the recursive feature extraction process to determine this variable as statistically significant, as suspicious ways of dealing with redirecting to a different website is a good indicator of if the website is planning to maliciously

redirect a user. Redirecting can also be accomplished by URLs in the <Script> and <Meta> tags on a website, so it can also be a way to measure if a website uses malicious redirecting or not.

## Classification with Naïve Bayes

Naïve Bayes was used with all 31 variables to test what results are 'out of the box', and an accuracy of 70% with a kappa statistic of .78 was produced. These five variables were then put into a Naïve Bayes model to predict the target variable and reported an accuracy of 91%. Reducing the number of variables from 30 to the five most correlated ones increased the accuracy approximately 21%. The kappa statistic is also 0.81158, which means that 81.2% of the data is reliable.

## Preparation for Neural Networks

To prepare the target value for a neural network, we replace the -1s with 0s because the target values must only contain 0 and 1.

## Classification with Neural Networks

A neural network with a hidden layer of 3 nodes and another hidden layer of 4 nodes, with backpropagation and a learning rate of 0.02, achieves an accuracy of 92%. The kappa statistic is 0.8606, which means that 86% of the data is reliable. The total error is approximately 163 and the total needed number of steps to converge is 94,638.

An improvement over this current paper could be using R's library for Keras instead of its 'neuralnet' library. Keras is an extremely popular API for neural networks. R's implementation of the Keras library uses Tensorflow, a popular open source library that can be used for neural networks, as its back-end foundation.

## Preparation for Gradient Boosting

XGBoost requires a matrix as the input data and numeric vector as the target column. The data frame is coerced to a matrix due to XGBoost's input data conditions. Additionally, the values of -1 in the target column are replaced with a 0 so the target values consist of 0 and 1.

## Classification with Gradient Boosting - XGBoost

Using all 31 variables, XGboost results in a 100% accuracy, with an AUC of 1, and a

0% error on the testing and training datasets. Additionally, the kappa statistic is 1, which means that 100% of the data is reliable. The following confusion matrix was computed using the model.

Confusion Matrix

| | Reference | |
|---|---|---|
| Prediction | 0 | 1 |
| 0 | 1926 | 0 |
| 1 | 0 | 2496 |

As the confusion matrix displays, the model perfectly classifies the true positives and true negatives.A reason for the perfect accuracy can be because of the ease of predicting a binary target value when features also mostly consist of a binary value.

As the tree below shows, the gradient boosted tree in this model is very shallow. The measure Cover is the sum of the second derivative of the training data classified to the leaf, and its high value for in final level helps to support the tree's shallowness. Gain corresponds to the information gain of a split, which shows the importance of the node in the model.[2]

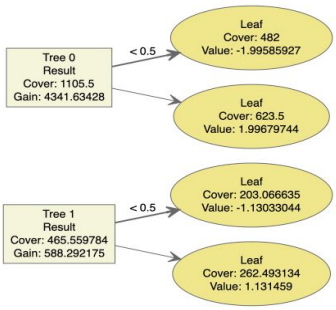


Figure 2: Gradient Boosted Decision Tree produced by XGBoost

## Preparation for Gradient Boosting - Light GBM

Similar to XGBoost, LightGBM requires a matrix as the input data and numeric vector as the target column.

The installation process; however, is much more complex than any other model in this paper. Knowledge of cmake, bash scripting, and setting environment variables is recommended for the installation process.

LightGBM depends upon two algorithmic processes: Gradient-Based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS consists of removing features that are not important to the model, by removing features that have a comparatively smaller information gain. EFB consists of removing features that are mutually exclusive, so variables that usually contain zero.[3]

## Classification with Gradient Boosting – Light GBM

LightGBM was produced by a Microsoft Research team as an incredibly fast gradient boosting algorithm.

Using the five most statistically significant variables found in recursive feature selection, the accuracy from the training set was 97.6%. The testing set achieved a higher accuracy of 98.1%**.** However, the lowest kappa statistic of LightGBM results in the lowest kappa statistic of this paper, which is .1 weighted kappa. This means that only 10% of the data is reliable, so this heavily discredits the value of this model's accuracy.

## Evaluation

After conducting binary classification with rule-based models, classic decision trees, neural networks, and gradient boosted decision trees, it is clear that one model has outperformed the rest. The gradient boosted decision tree model, XGBoost, resulted in a 100% accuracy with 100% of the data being reliable, due to its kappa statistic of 1. However, all of the models did achieve a high accuracy, while not compromising the quality of how reliable the data is. The

[2] https://rdrr.io/cran/xgboost/man/xgb.plot.tree.html

[3] https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

lowest accuracy was 91%, produced by a Naïve Bayes, which is only gives a 9% error from a perfect accuracy. However, Light GBM had the lowest kappa statistic of 0.1. Every model except LightGBM resulted in a kappa statistic range of 8-10, which can be interpreted that data in every model, except LightGBM, is more effective than guessing. This research will also continue looking into the reason why LightGBM gave a much smaller kappa statistic than the other classification models.

## Deployment

The primary goal of this project was to gain insight into how features of a website can be organized into categorical variables, and which model would be the best to classify. Further deployment could include implementing information learned from these classification algorithms into more sophisticated phishing detection software programs.

## Future Work

Instead of simply assessing if a website is harmful or not, we can accelerate a threat model and work on applying machine learning algorithms to determine if an email can be classified as a phishing one or not. We have scraped hundreds of emails from public repositories hosted by different universities, and have obtained a few hundred more from volunteers and Skidmore College's IT department as well. After scraping, a problem arose in what is the best practice to convert the emails into an analytics base table. However, after conducting research on malicious URLs and website configuration, more knowledge has been acquired to further creating a dataset and analytics base table from the scraped phishing emails. These phishing emails would be scraped into 7 categories. The seven categories to classify a phishing email can be observed in Table 1. Non-phishing emails will be taken from the Enron email dataset. An ideal research to determine phishing classification would utilize textual analysis using natural language processing, visual analysis by convolutional neural networks, and continuing classification using the models already implemented in this paper.

Table 1: Categories for Future Phishing Classification Research

| | |
|---|---|
| Category 1 | Bitcoin/Cryptocurrency Extortion Scam |
| Category 2 | Disguised as an Educational Institution |
| Category 3 | Disguised as a Banking Institution |
| Category 4 | Disguised as Application Updates - "Helpdesk" |
| Category 5 | "Advance Fee"/419 |
| Category 6 | Malicious pdfs |
| Category 7 | Non phishing emails |

Works Cited

"Intelligent Rule based Phishing Websites Classification."Mohammad, Thabtah, McCluskey
http://eprints.hud.ac.uk/id/eprint/17994/3/RamiIntelligent_Rule_based_Phishing_Websites_Classification_IET_Journal.pdf

"LightGBM: A Highly Efficient Gradient Boosting Decision Tree" (Ke, Meng, Finley, Wang, Chen, Ma, Ye, Liu)
https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

"Plot a boosted tree model", CRAN.
https://rdrr.io/cran/xgboost/man/xgb.plot.tree.html