

Project Report

Ryan Greiner, Grant Gonnerman, Tim Lincoln



Abstract:

With the wide availability and freedom of the internet, there are many people who have dishonest intentions. Phishing is a cyber-crime where phone calls, text messages or emails are sent to scam money from someone. As technology continues to grow, more phishing attempts are made. Using the data, we believe we can flag fraudulent webpages. We will also discuss ways we could employ a successful model to help make sure people do not fall for a scam. There are many classification techniques we have learned in this class that could potentially accomplish this goal.

Table of Contents

[Introduction 2](#_Toc120827041)

[Data Description 2](#_Toc120827042)

[Row and column description 2](#_Toc120827043)

[Visualization 3](#_Toc120827044)

[Modeling 5](#_Toc120827045)

[Importance 6](#_Toc120827046)

[Model Selection 6](#_Toc120827047)

[Parameter Selection 6](#_Toc120827048)

[Optimize the cut-off value 7](#_Toc120827049)

[Final Model 7](#_Toc120827050)

[Conclusion/further study 8](#_Toc120827051)

[References 9](#_Toc120827052)

# Introduction

Website phishing is a type of fraud that aims to steal sensitive information from a target. These goals are accomplished by disguising their phishing website as a legitime one to make the target more comfortable with giving out their sensitive data. Recent increases in technology have given attackers the ability to make these phishing sites look almost identical to legitimate websites. The goal of this project is to build a model using attributes from 10,000 phishing and legitimate websites to accurately predict and flag phishing websites before sensitive information is stolen.

# Data Description

The data set used in this project is called “Phishing Dataset for Machine Learning” and was obtained on Kaggle. The data set consists of one CSV file named “Phishing\_Legitimate\_full.csv”.

## Row and column description

This dataset contains 48 features extracted from 5000 phishing web pages and 5000 legitimate web pages, plus an id and class column depicting phishing or legitimate. These observations were collected from January to May 2015 and from May to June 2017. There are no missing values in this dataset. Each row represents a different web page. A description of each column can be found below.

|  |  |  |
| --- | --- | --- |
| Column name | Data type | Description |
| id | Categorical | Unique identifier |
| NumDots | Numerical | Number of “.” in URL |
| SubdomainLevel | Numerical | Level of subdomains |
| PathLevel | Numerical | Level of path segments |
| UrlLength | Numerical | Length of characters in URL |
| NumDash | Numerical | Number of “-” in URL |
| NumDashInHostname | Numerical | Number of “-” in host name |
| AtSymbol | Categorical | Whether “@” is present in URL |
| TildeSymbol | Categorical | Whether “~” is present in URL |
| NumUnderscore | Numerical | Number of “\_” in URL |
| NumPercent | Numerical | Number of “%” in URL |
| NumQueryComponents | Numerical | Number of query components |
| NumAmpersand | Numerical | Number of “&” in URL |
| NumHash | Numerical | Number of “#” in URL |
| NumNumericChars | Numerical | Number of Numeric characters |
| NoHttps | Categorical | Whether Https is present in URL |
| RandomString | Categorical | Whether random string is present |
| IpAddress | Categorical | Whether an Ip address present |
| DomainInSubdomains | Categorical | Is domain in subdomains |
| DomainInPaths | Categorical | Is domain in paths |
| HttpsInHostname | Categorical | Is https in host name upon request |
| HostnameLength | Numerical | Length of host name |
| PathLength | Numerical | Lenth of path |
| QueryLength | Numerical | Length of query |
| DoubleSlashInPath | Categorical | Is “//” in the path |
| NumSensitiveWords | Numerical | Number of sensitive words |
| EmbeddedBrandName | Categorical | Is brand name embedded |
| PctExtHyperlinks | Numerical | Number of .pct extension hyperlinks |
| PctExtResourceUrls | Numerical | Number of .pct extension Resource URLs |
| ExtFavicon | Categorical | Is a Favicon file extension present |
| InsecureForms | Categorical | Does page have insecure forms |
| RelativeFormAction | Categorical | Does page have relative form action |
| ExtFormAction | Categorical | Does page have form action extension |
| AbnormalFormAction | Categorical | Does page have abnormal form action |
| PctNullSelfRedirectHyperlink | Numerical | Null .pct redirecting hyperlink |
| FrequentDomainNameMismatch | Categorical | Is there Frequent Domain Mismatch |
| FakeLinkInStatusBar | Categorical | Is there a Fake Link in Status Bar |
| RightClickDisabled | Categorical | Is right click disabled |
| PopUpWindow | Categorical | Are there pop-up windows |
| SubmitInfoToEmail | Categorical | Can info be submitted to email |
| IframeOrFrame | Categorical | Is HTML Iframe or frame |
| MissingTitle | Categorical | Is page missing title |
| ImagesOnlyInForm | Categorical | Are images only in form |
| SubdomainLevelRT | Categorical | Standardized Level of subdomains |
| UrlLengthRT | Numerical | Standardized Length of characters in URL |
| PctExtResourceUrlsRT | Numerical | Standardized PctExtResourceUrlsRT |
| AbnormalExtFormActionR | Categorical | Standardized AbnormalExtFormActionR |
| ExtMetaScriptLinkRT | Categorical | Standardized ExtMetaScriptLinkRT |
| PctExtNullSelfRedirectHyperlinksRT | Numerical | Standardized PctExtNullSelfRedirectHyperlinksRT |
| CLASS\_LABEL | Categorical | Phishing or legitimate (0,1) |

## Visualization

Our data set contains a high number of variables. Because of this, we will need to go through and find out which ones are important. Before we do that though, we can visualize at least some of the variables. Below are a few histograms and boxplots we created based on the variables.

Chart, histogram

Description automatically generated*Figure 1:* From the histogram, we can see that a mojority of the web pages fall in the 50-100 length range. The distribution is right skewed with a mean length of 70.

Chart, histogram

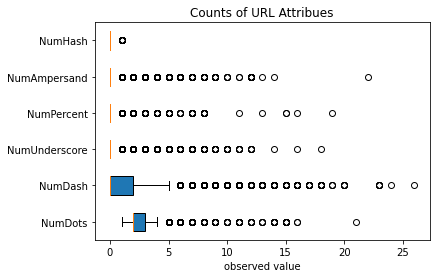
Description automatically generated

*Figure 2:* This graph shows a histogram of the URL path length. A path is defined as a delimited sequence of segments following a”/”. The distribution of path length is right skewed with a mean length of 35.

Chart, histogram

Description automatically generated

*Figure 3:* In the histogram of numeric characters, we can see it is right skewed with most of the values falling in the 0-10 range with many values being outliers.



*Figure 4:* From the boxplot of URL attributes, such as symbols, we can see the data is right skewed with many outliers.

Chart, bar chart

Description automatically generated*Figure 5:*

From this frequency graph, we can see most of the web pages fall into the 1 category meaning most have insecure forms on the page attributing to more phishing attempts.

# Modeling

Our initial exploratory analysis provided us with background information needed to take the next step. It was obvious to us that we had many variables at our disposal and would not need all of these in our model. Our first step is to determine which variables are most important through variable selection. To do this we extracted coefficient importance through a random forest classifier and ran lasso for variable selection to see minimized variables. After we determine which variables play the largest role in detecting phishing websites, we will use these variables to build many different classification models including logistic regression, neural networks, support vector machines, decision trees, random forest, ada boost, and gradient boosting. Next, we will tune these models using appropriate hyperparameters to yield the best and most accurate model. After the best parameters have been set, we will evaluate these models using recall and accuracy scores to determine which model is the best and most accurate at determining a phishing website.

## Importance

To find the most important variables for our phishing model we used a base random forest model with 500 trees and extracted the importance’s over 100 iterations. Next, we averaged these importance’s to find the top variables in terms of importance. To help support these findings we ran lasso as a variable selector and viewed the variables that were minimized to zero over 100 iterations. We settled on the top variables in order of importance with no occurrences of being minimized to zero.

## Model Selection

List all models we plan to use and hyper-parameters we plan to evaluate:

* Random Forest
  + Max\_depth
  + N\_estimators
* Ada Boost
  + Max\_depth
  + N\_estimators
  + Learning\_rate
* Gradient Boosting
  + Max\_depth
  + N\_estimators
  + Learning\_rate
* Support Vector Machine
  + Kernel
* Logistic Regression
* Neural Network
  + Activation functions:
    - Relu
    - Sigmoid
    - Tanh
    - Softmax

## Parameter Selection

To find the best models, we fit every model with many different combinations of hyperparameters and computed the accuracy and recall of each model. We ran this algorithm five times to get an average of the performance over multiple iterations. The algorithm we used to find the best parameters is as follows.

1. Define most important variables from importance for our model to be train with
2. Create data frames to store all combinations of hyperparameters
3. For each possible hyperparameter combination...
   1. Train a logistic regression model with the training dataset, evaluate it on the test dataset, compute accuracy and recall scores.
   2. Train a support vector classifier model with the training dataset, evaluate it on the test dataset, compute accuracy and recall scores.
   3. Train a random forest model with the training dataset, evaluate it on the test dataset, compute accuracy and recall scores.
   4. Train an ada boost model with the training dataset, evaluate it on the test dataset, compute accuracy and recall scores.
   5. Train a gradient boosting model with the training dataset, evaluate it on the test dataset, compute accuracy and recall scores.
4. Average the performance metrics of each model and export as a csv file.

## Optimize the cut-off value

Since we are trying to flag phishing attempts, false negatives are important. Because of this, we are extremely interested in recall. Since a model can have a high recall but still be inaccurate, we decided to weigh these two metrics together. To do this we tested cutoff values from .01 to .5 on our best random forest model. We ran that range of cutoff values on 100 splits of the data so that we had a good average for each metric. We found that using .05 as the cutoff value gave the highest recall, but the tradeoff was an unproportional decrease in accuracy. As we increased the cutoff value, recall would slowly decline, and accuracy would more drastically increase. The specific recall and accuracy for each cutoff value can be seen in the table below.

|  |  |  |
| --- | --- | --- |
| Cutoff Value | Recall | Accuracy |
| .01 | .99966 | .61796 |
| .05 | .99279 | .87831 |
| .10 | .98782 | .92669 |
| .15 | .98294 | .93471 |
| .20 | .9795 | .94291 |
| .25 | .97576 | .94612 |
| .30 | .97263 | .94683 |
| .35 | .96954 | .94714 |
| .4 | .96636 | .94747 |
| .45 | .96156 | .94695 |
| .5 | .95543 | .94576 |

After evaluating the above results and discussing our goal with the dataset, we decided to use .10 as our final cutoff value. This value gives a model that we felt effectively balanced recall and accuracy.

# Final Model

Using our parameter selection algorithm, we described how we were able to pick the hyperparameters and cut-off values for each model that yielded the best results. The performance metrics of the selected best models are listed below.

|  |  |  |
| --- | --- | --- |
| Model | Recall | Accuracy |
| Logistic Regression | 0.95 | 0.79 |
| Support vector classifier | 0.94 | 0.84 |
| Neural network | 0.75 | 0.80 |
| Random Forest | 0.98 | 0.92 |
| Ada boost | 0.98 | 0.95 |
| Gradient boosting | 0.98 | 0.96 |

Since we chose to prioritize recall over accuracy, random forest, ada boost, and gradient boosting were our top three models with the following hyperparameters.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | n\_estimators | max\_depth | learning\_rate |
| Random Forest | 500 | 7 | N/A |
| Ada boost | 100 | 7 | .01 |
| Gradient boosting | 2000 | 7 | .1 |

In an attempt to increase our model's metric scores, we used these three models in an ensemble voting classification model to aggregate the results and get a better more accurate prediction. We chose these three models because they all shared the highest recall value of .98 while also having high accuracy scores. Using this method, we were able to average a recall score of .99 and an accuracy score of .95 gaining a percentage over all three singular models and being similar to our best in terms of accuracy.

# Conclusion/further study

As technology advances, it is becoming increasingly difficult to identify internet fraud attacks like website phishing. The goal of this project was to develop a model that takes the pressure off the individual by identifying phishing websites before sensitive information can be stolen. Using variable selection, we were able to identify the most important website features to identify phishing websites. We were then able to use these variables to identify and select the best possible hyperparameters to give us the highest recall and accuracy scores. With this process, we discovered random forest, ada boost, and gradient boosting were our highest-scoring models. With this information, we were then able to aggregate these models' results in our final ensemble voting classification model which yielded a recall score of .99 and an accuracy score of .95 showing spectacular ability in flagging true positives with very few false negatives along with a great ability of identifying the positive and negative case. These metrics give us great confidence in the ability to identify phishing websites before sensitive information can be stolen from a target.

In further study we could begin applying our models in different ways. If we are to decide that every website our model flags as phishing gets taken down or restricted, we could tune our models to ensure we do not accidentally take down a legitimate sight. We could also tune a different model that we would have warn people about a possible phishing attempt. In that case we are mostly concerned with catching every phishing attempt. These different applications may require us to investigate different metrics and take other parameters into consideration. As the internet continues to grow, so does the need of anti-phishing efforts. We hope models such as these continue to be developed so we can all have better experiences on the internet.

# References

Data set: <https://www.kaggle.com/datasets/shashwatwork/phishing-dataset-for-machine-learning?resource=download>