Detecting Malicious Websites Using Machine Learning

Saeed Ahmad Al Tamimi
aa6764@rit.edu

Follow this and additional works at: https://scholarworks.rit.edu/theses

Recommended Citation

This Master’s Project is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
Detecting Malicious Websites Using Machine Learning

by

Saeed Ahmad Al Tamimi

A Graduate Capstone Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Professional Studies: Data Analytics

Department of Graduate Programs & Research

Rochester Institute of Technology

RIT Dubai

April 20, 2020
RIT

Master of Science in Professional Studies:

Data Analytics

Graduate Capstone Approval

Student Name: Saeed Ahmad Al Tamimi

Graduate Capstone Title: Detecting Malicious Websites Using Machine Learning

Graduate Capstone Committee:

Name: Dr. Sanjay Modak          Date:

Chair of committee

Name: Dr. Ehsan Warriach       Date:

Member of committee
ABSTRACT

The growing use of the internet resulted in emerging of new websites every day (Total number of Websites - Internet Live Stats, 2020). Web surfing has become important for everyone regardless of their occupation, age or location. However, as the use of the internet is increasing so is the vulnerability to malware attacks through malicious websites (Softpedia, 2016). Identifying and dealing with such malicious website has been quite difficult in the past as it is quite challenging to separate good websites from bad websites. However, by using machine learning algorithms on large datasets it is now possible to detect such websites beforehand. Classifiers trained using algorithms such as logistic regression and Support Vector Machine (SVM) can be used to detect malicious websites and the users can be warned about the risk before they visit such sites. This project focuses on using a variety of different classification algorithms to distinguish whether a website is malicious or not using the Kaggle Malicious and Benign Website Dataset. We have showcased that it is possible to detect malicious websites with a reasonable amount of certainty (90% of the 75 malicious websites in the test set were identified) using machine learning models. We have also determined the features that were critical in predicting the likelihood of a website being malicious. Most of our key features are easily available (URL Length, number of Special characters, Country, Age of website).

Keyword: support vector machine, malicious websites, cyber-security, machine learning, logistic Regression.
Contents

ABSTRACT .......................................................................................................................... 3

INTRODUCTION .................................................................................................................. 6

Problem statement ............................................................................................................. 6
Background of the Study ..................................................................................................... 7
Proposed Solution ................................................................................................................ 9

Machine Learning ............................................................................................................. 9

LITERATURE REVIEW ....................................................................................................... 15

METHODOLOGY ................................................................................................................ 17

CRISP-DM (Cross-industry standard process for data mining) ........................................... 17

Business understanding ..................................................................................................... 17
Data understanding ............................................................................................................. 17
Data preparation .................................................................................................................. 20
Modeling ............................................................................................................................. 24
Evaluation ........................................................................................................................... 28
Deployment ......................................................................................................................... 30
Tools .................................................................................................................................. 30

RESULTS ............................................................................................................................ 30

Exploratory Data Analysis ................................................................................................. 30

Number of Special Characters .......................................................................................... 31
Server ................................................................................................................................. 32

Machine Learning Models .................................................................................................. 32

Logistic Regression with All Variables .............................................................................. 32
Logistic Regression with a variable selected according to p-value .................................. 34
Decision Tree ...................................................................................................................... 35
Random Forest ................................................................................................................... 36
Support Vector Machine ..................................................................................................... 37

Model Comparison ............................................................................................................. 37

The Best Model ................................................................................................................... 38

Discussion ............................................................................................................................ 38

Variables important in detecting malicious websites ......................................................... 38

Number of Special Characters .......................................................................................... 38

Site age ................................................................................................................................. 39
INTRODUCTION

Problem statement

Web Security has become very important in recent years as internet connectivity has penetrated more and more regions across the world. While this penetration is great for global connectivity, it also means that more people have access to websites that can potentially attack them using malwares, viruses, and other malicious agents. Thus, it becomes more important than ever to identify and deal with such websites before a normal user has access to them (Jang-Jaccard and Nepal, 2014). Current approaches to deal with this problem have many limitations in terms of effectiveness and efficiency (Eshete, Villafiorita and Weldemariam, 2011). The aim of this study is to detect malicious websites using a group of machine learning algorithms called classifiers, we will try to detect malware on websites.

This will help in safe web surfing and better user experience. By timely reporting malicious websites, the users will be able to avoid any violation and serious privacy breach. The users will also be able to avoid any illegal activities that they can get involved in. Labelling malicious websites will also help to eliminating fraud, as users become victim of attacks that use blackmailing and false information to get monetary advantage of their victim. For example, ransomware attacks are getting quite common. Systems get infected by such viruses through surfing malicious websites.
Background of the Study

The number of websites on the internet is increasing at a rapid rate. In 2018, there were over 1.6 Billion websites on the world wide web (Total number of Websites - Internet Live Stats, 2020). And as the time passes the number is increasing.

Figure 1: The total number of websites over the time (Total number of Websites - Internet Live Stats, 2020)

Figure 1 above shows the number of websites over time. However, as the internet is expanding so is the risk of malware attacks to web services. Corrupt web developers release malware through their websites to hack personal computers and servers and breach privacies for blackmailing, fraud, and theft. These attackers ask for ransom money and can create serious problems for the victims. Attackers can publish private data of their victims, can steal money from their accounts (Jang-Jaccard and Nepal, 2014).

Figure 2 below shows the top ten categories of websites that have malicious content and can potentially harm their user (Softpedia, 2016). As can be observed the list of categories below contain some of the most common websites that can have a lot of utility and can make life of a user easier. When malicious, these can become a nightmare for the user. These websites such as
gambling, shopping and business all prompt users for credit card information. This information can easily get in the hands of the wrong people and can cause financial harm to the users.

![Table showing the top 10 categories of evil websites](image)

*Figure 2: Top 10 Categories of evil websites (Softpedia, 2016)*

Following is a summary of potential impacts of malicious websites on computers:

- Disrupts operations and automated programs that maybe handling some important processes
- Steals sensitive information.
- Provides unauthorized access to system resources to other malicious software’s
- Reduces computer or web browser speeds by increasing dummy processes
- Creates network connectivity issues
- Cause frequent freezing or crashing.
Proposed Solution

All websites whether they are malicious or benign have a lot data associated with them. Historically it has been very difficult to properly analyze this data due to low computing power and primitive analytics techniques. However, with the recent developments in the fields of computer science, data analysis and machine learning, handling, and analyzing large quantities of data is no longer difficult. In this project, I will use supervised machine learning on the data commonly generated by websites across the internet to predict whether a website is malicious or benign.

Machine Learning

Machine learning is a branch of Artificial Intelligence which deals with teaching computers the ability to learn and improve from experience (Kersting, 2018). The primary aim is for the machine to be able to access data and use it to learn and discover patterns in it which can then be used to make predictions, perform categorization and clustering (Kersting, 2018). ML is broadly classified into two types – supervised and unsupervised machine learning:

- Supervised Machine learning uses a labelled set of examples to learn patterns and relationships between the data and the outcome. It then uses these learnings to make predictions for new data (Kersting, 2018).
- Unsupervised Machine Learning tries to uncover the hidden structure from data that is unlabeled. Here it is not about figuring out the right output but instead the focus is on drawing inferences from the datasets (Kersting, 2018).

In this project we will use Supervised Machine Learning to learn patterns that will help us in predicting whether a website is malicious or benign. We will do this using the labelled dataset available at Kaggle. Below I describe a few supervised learning algorithms that we will use to train our machine learning model.

Logistic Regression

LG is a machine learning algorithm used to train classifiers. It is basically the logistic/sigmoid function layered on top a linear regression model. Mathematically it is represented as below:

\[ z = wx + b \]
\[ y = \text{sigmoid}(z) \]
\[ \text{sigmoid}(z) = \frac{1}{1 + e^{-t}} \]

Figure 3: The sigmoid function (Obeid, 2019)

Figure 3 shows that as the value for \( z \) gets larger and larger the value of \( Y \) tends to 1. On the other hand, as the value of \( z \) gets smaller and smaller the value of \( Y \) tends to 0. This means that the output of this model is always in the range of 0 to 1 which basically gives us the probability of an observation being 1 or 0 (Sperandei, 2014).

Logistic regression calculates the probability of a binary outcome and by setting a threshold, it classifies the data points into either outcome. Our data will have a binary outcome as we must decide whether a website is malicious or not.

**Logistic Regression Coefficient Table**

Once the logistic regression model is trained, we can see how each individual predictor variable in the model relates with the target model using the coefficient table (Logistic Regression Essentials in R - Articles - STHDA, 2020; Peng et al., 2002).
Table 1. Example logistic regression coefficient table

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Coefficient</th>
<th>Standard Error</th>
<th>Z_Value</th>
<th>P_Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-3.990</td>
<td>1.140</td>
<td>-3.500</td>
<td>0.000</td>
</tr>
<tr>
<td>gre</td>
<td>0.002</td>
<td>0.001</td>
<td>2.070</td>
<td>0.038</td>
</tr>
<tr>
<td>gpa</td>
<td>0.804</td>
<td>0.332</td>
<td>2.423</td>
<td>0.015</td>
</tr>
<tr>
<td>rank2</td>
<td>-0.675</td>
<td>0.316</td>
<td>-2.134</td>
<td>0.033</td>
</tr>
<tr>
<td>rank3</td>
<td>-1.340</td>
<td>0.345</td>
<td>-3.881</td>
<td>0.000</td>
</tr>
<tr>
<td>rank4</td>
<td>-1.551</td>
<td>0.418</td>
<td>-3.713</td>
<td>0.000</td>
</tr>
</tbody>
</table>

This table can be interpreted as follows:

- For continuous variables such as ‘gre’ and ‘gpa’ the coefficient column tells us the change in the log-odds of admission per unit change in the predictor variable.
- Categorical variables like rank have a slightly different interpretation. Here the categories of the variable are compared to one reference category of the same variable.
- Variables and their coefficients can be deemed statistically significant based on the p-value column according to pre-defined error rate.

**Support Vector Machines**

The objective of a SVM is to find the decision boundary in a n-dimensional space that maximizes the distance between datapoints of target classes. New data points that fall on either side of this decision boundary are then classified into their respective classes. Support vectors are those data points that are closest to the decision boundary. These influence the position and orientation of the boundary itself and are used to optimize the decision boundaries location (Shihong, Ping and Peiyi, 2003). The output of an SVM lies in the range [-1,1].

SVM is not restricted to binary outcomes, though we can use it for our purpose. SVM will use kernel tricks to classify websites into malicious or benign/safe category. Kernel techniques include sigmoid, linear, polynomial and radial. There are several other techniques, but we can focus on these. Figure below shows a typical plot of support vector machine (Shihong, Ping and Peiyi, 2003).

In Figure 4, the red line represents the decision boundary generated using SVM. The dashed lines represent the support vectors. Any data point lying beyond the positive support vector is classified
to the blue class while any data point lying beyond the negative support vector is classified to the green class.

*Figure 4: A 2-dimensional decision boundary generated using SVM (Ashishsubedi, 2020)*

**Decision Tree**

One of the most widely used classification algorithms in machine learning, decision trees are built through an algorithmic approach that sequentially splits the dataset on multiple conditions. The objective of these splits is to obtain the maximum homogeneity in each of the split subsets. An example is shown in Figure 5.
The algorithm starts with the root node (in this case the ‘Outlook’ node). Based on the values present in that variable it splits the dataset into n-subsets (in this case 3). Each of these subsets contains a majority of one the target classes. This followed by further splitting of each subset into nodes. This continues until maximum homogeneity in nodes is achieved (Kotsiantis, 2013; Rokach and Maimon, 2005).

A tree has the following structure: - The root node is the first node where the dataset is split. This node often contains the most important variable in distinguishing the classes present in the dataset. - The terminal node are nodes at the bottom of the tree which contain the outcome. - The branch nodes are intermediate nodes at which the data is split as we move from root node to terminal node(Pandey, 2019).

There are two common measures using which the algorithm decides how to split the dataset at any particular node:

- Gini Index calculates the probability of two items from different subsets of the data belonging to the same class. The lower the probability the better the split.
- Information Gain uses a mathematical formula to calculate the amount of information gained by a particular split. Splits with larger information gain are favored.
The decision tree algorithm outputs a probability value for any observation belonging to a specific class. This type of algorithm forms the foundation behind some of the best performing algorithms in machine learning like Random Forests, Gradient Boosters etc.

**Random Forest**

Random forests are group of machine learning models. This means that they use predictions made by many weak models and combine them to generate the actual prediction. In the case of random forests, the weak models are trained as decision trees. The random forest method also uses the technique of bagging. What this means is that for training each decision tree in the forest, a random sample of size N is sampled from the original training data. Along with sampling from training observations a random sample of features is also sampled. This means that no single tree is trained using all training data and features. This randomness ensures that each decision tree is uncorrelated to other decision trees. All decision trees are trained independently of each other (Breiman, 2001; Fawagreh, Gaber and Elyan, 2014). Once each individual tree has decided, a collective decision is taken using a voting method. The process of bagging is further shown in Figure 6.

![Diagram of Random Forests](image)

*Figure 6: Bagging in Random Forests (Yiu, 2019)*
LITERATURE REVIEW

Several studies have been performed previously that use various aspects of websites to detect whether it contains malicious content. In (Chiba et al., 2012), the authors used machine learning to detect malware in websites using IP address features with an argument that IP address is a much more stable feature of websites as compared to URL and DNS. The goal of their study was to develop a powerful technique to compensate the previous techniques. They used octet, extended octet and bit string-based features to classify websites. The authors argue that malicious websites have a distinct signature for all three feature systems and can be used to differentiate them from benign/safe websites. While the results from this study showed promising accuracy and was able to detect even unknown malicious websites, a limitation was that it misclassified many benign websites as malicious simply because they were hosted on the same hosting service.

In (Ikinci, 2008), the authors claim to provide an internet scale solution to detect malicious websites. Using Monkey-Spider project, authors were able to use subtle attributes of websites to detect malicious websites. Misspelled domain names of popular domains or “typosquatting” was an important indicator for their purpose. A limitation of their study in its current form is that they could only perform crawling of HTML content with data extraction as the focus due to resource constraints. Another limitation is that the crawler recrawls popular websites multiple times leading to inefficiency. The authors demand that after a lot of server security research, there is a need to shift to client security research to make web surfing a good experience for the users and make the internet a safe place for visitors.

In (United States Patent: 8850570, n.d.), the applicants provide a patent to detect malicious websites using a filter score that is calculated using likelihood function. This is done by analyzing the software that is downloaded in result of opening the website and whether the software tries to access sensitive information or not.

“Using website URL features such as textual properties, link structures, webpage contents, DNS information, and network traffic detects malicious websites exploiting state-of-the-art machine learning algorithms”(Choi, Zhu and Lee, 2011). The authors state that most traditional methods are aimed at targeting a single type of malicious attack, while their approach not only detect a malicious attack but also identifies its type. The authors used a SVM to detect malicious URLs
and a Multi-Label k-Nearest Neighbor approach for identifying the type of attack. They achieved >90% accuracy in each task.

In (Heiderich, Frosch and Holz, 2011), the authors provide a novel approach to create a JavaScript tool that detects malicious software’s present online that can harm user’s computers. The algorithm does not only detect but also mitigates by changing suspicious elements, so that the software’s become harmless. This tool was shown to perform its task with very low overheads meaning that it could be deployed on devices with low computing power like smartphones. One limitation for this study was the relatively high false positive rate. The tool also depends on the attacker using native DOM methods and the user using a relatively recent web browser version.

The authors of (Eshete, Villafiorita and Weldemariam, 2011), use practical solution to detect malicious websites. A holistic approach is proposed to effectively and efficiently detect malicious websites. Their major focus is to increase the quality of features extraction techniques.

In summary, there have been a variety of different approaches that have been tried in the literature. Each study has its own set of limitations and one thing that has been common across all of them is that they use fairly simple machine learning models and their feature set doesn’t contain demographic information such as country of registration and dates of registration.
METHODOLOGY

CRISP-DM (Cross-industry standard process for data mining)

Business understanding
In this section the aim is to gain some domain knowledge and learn about how malicious websites are usually identified. The literature review we have performed should help in this section. The characteristics such as length of the URL, number of special characters, operating system from which the website is fetched, number of bytes transferred, and number of IPs connected to the honeypot have been shown to help identify malicious websites.

Data understanding
This data (Malicious and Benign Websites | Kaggle, no date) set contains 1781 websites and 21 columns. These attributes can help to detect the malicious websites. We will also understand how many numerical and categorical variables we have and what their distributions are.
# DATA SET

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>“The anonymous identification URL analyzed in the study” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>URL_LENGTH</td>
<td>“The URL length(characters)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>NUMBER_SPECIAL_CHARACTERS</td>
<td>“The count of the special characters in the URL, for instance (/,%,#,&amp;)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>CHARSET</td>
<td>“The character encoding standard or character set (categorical variable)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>SERVER</td>
<td>“The operative system of the server, from the packet response (categorical variable)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>CONTENT_LENGTH</td>
<td>“The content size of the HTTP header” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>WHOIS_COUNTRY</td>
<td>“The nations we got from the server reaction, explicitly, our content utilized the API of Whois (categorical variable)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>WHOIS_STATEPRO</td>
<td>“The states we got from the server response, specifically, our script used the API of Whois (categorical variable)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>WHOIS_REGDATE</td>
<td>“Server registration date. This variable has date values with format (DD/MM/YYYY HH:MM)” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>WHOIS_UPDATED_DATE</td>
<td>“The last update date from the analyzed server” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>TCP_CONVERSATION_EXCHANGE</td>
<td>“The number of TCP packets exchanged between the server and our honeypot client” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>DIST_REMOTE_TCP_PORT</td>
<td>“The number of the ports detected and different to TCP” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>REMOTE_IPS</td>
<td>“Total number of IPs connected to the honeypot” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>APP_BYTES</td>
<td>“Number of transferred bytes” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>SOURCE_APP_PACKETS</td>
<td>“Packets sent from the honeypot to server” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>REMOTE_APP_PACKETS</td>
<td>“Packets received from server” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>SOURCE_APP_BYTES</td>
<td>“The source of the app bytes” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>REMOTE_APP_BYTES</td>
<td>“The remote app bytes” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>APP_PACKETS</td>
<td>“Complete number of IP created while the correspondence between the honeypot and the server” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>DNS_QUERY_TIMES</td>
<td>“DNS packets generated during the communication between the honeypot and the server” (Malicious and Benign Websites</td>
</tr>
<tr>
<td>TYPE</td>
<td>“Represent the type of web page analyzed (1 is for malicious websites and 0 is for benign websites)” (Malicious and Benign Websites</td>
</tr>
</tbody>
</table>

These attributes can help the classification algorithms defined above to detect the malicious websites.
The dataset selected for this study is called “Malicious and Benign Websites”. It is available on Kaggle (Malicious and Benign Websites | Kaggle, no date). The dataset contains 1781 websites and 21 columns i.e. attributes for each website. The explanation of each attribute shown below:

**Data preparation**

In this section we will clean and prepare the data for further analysis. Handling missing values, outliers and inaccurate data will be the primary focus. The data we get for the project is not fit for analysis and needs to be prepared.

The dataset contains 1781 records of websites and 21 variables associated with each website. There are 4 categorical variables, 15 numerical variables and 2 Date variables. We observed that the “URL” variable has 1781 unique values which means that it is unique for each website in the dataset. Such a variable is of no use for our analysis and hence we remove it from the dataset. We also observed that many columns had a ‘None’ value. This represents the fact that the value for that column is unknown and thus missing. We reloaded the dataset by encoding the value ‘None’ as missing.

**Missing Data Analysis**

Below we check the proportion of missing values in each variable of the dataset. We find the variable “Content Length” has almost half of its values missing. Similarly, the variable “WHOIS_STATEPRO” was discarded due to high amount of missing and unreliable information.

Overall, I found missing values in 2 Date type variables, 4 categorical variables and 1 numerical variable. Out of these we discarded 1 numerical variable and 1 categorical variable due large amount of missing information. For the remaining 3 categorical variable we encoded the missing value to a new category called “Unknown”. For the 2 Date type variable the missing values were encoded to the mean date of the distribution of those 2 Date variables. This meant that now our dataset consisted of 18 variables with no missing values. An additional variable called “APP_PACKETS” was also removed as it contained duplicate values to another variable called “SOURCE_APP_PACKETS”.
Univariate analysis for Outlier Removal and feature simplification

For numerical data, I considered those observations as outliers which lied beyond the 95th percentile in the distribution of the concerned variable. All values for a variable which were greater than the 95th percentile of that variable were made equal to the 95th percentile. Univariate analysis for individual variables is described below.

**URL_LENGTH**

This variable describes the number of characters present in the URL. As seen in Figure 9, the original distribution had a few values greater than 100 which were skewing the distribution. After outlier treatment the distribution looks much better. Similar plots for other numerical variables can be found in the appendix.
Figure 9: Univariate distributions before and after outlier treatment for URL Length

CHARSET
This variable initially had 9 different categories. On inspection we found that many of those categories were different versions of the same category or misspelled versions of the same category. Thus, this variable was simplified the variable “Charset” into fewer categories:

- If the value contains the word ‘iso’ then the charset type is ISO.
- If the value contains the word ‘utf’ then the charset type is UTF.
- If the value contains the word ‘ascii’ then the charset type is ASCII.
- If the value contains the word ‘windows’ then the charset type is Windows.
The variable initially had 239 different categories. On inspection we simplified the variable as follows:

- If the value contains the word ‘apache’ then the server type is Apache
- If the value contains the word ‘nginx’ then the server type is nginx
- If the value contains the word ‘microsoft’ then the server type is Microsoft
- Other non-missing values are represented by the value “others”.

---

**Figure 10: Frequency Distribution for the variable “CHARSET” after simplification**

**Figure 11: Frequency Distribution for the variable “SERVER” after simplification**
WHOIS_COUNTRY
The variable initially had 49 different categories. On inspection we simplified the variable as follows. • If the country is ‘us’ then it is labelled as USA. • All other countries have been labelled into others.

![Figure 12: Frequency Distribution for the variable “WHOIS_COUNTRY” after simplification](image)

WHOIS_REG_DATE & WHOIS_UPDATED_DATE
These variables represented the dates on which a website was first registered and last updated respectively. We converted these variables into numeric variables counting the number of months that have passed since the dates. The new variables are called ‘site_age’ and ‘update_age’ respectively.

After performing the missing value treatment, outlier handling and feature simplification and engineering, the data preparation phase was over. The next step is the modeling phase which is broken into three parts.

**Modeling**
In modeling phase, we will start by performing exploratory analysis on the data. This means that we will use graphs and charts to understand which columns have a relationship with the target column. This will allow us to pick the relevant columns for our model. Once the exploratory
analysis is finished, we will train a classifier using a variety of different machine learning algorithms. Some algorithms that we can try are:

- Logistic Regression
- Decision Trees
- Random Forest
- Support Vector Machines (SVM)

Once all models have been trained, we will choose the best one using the evaluation method.

**Data Partitioning**

In order to evaluate the models we build, we need to separate the dataset into two parts, one big portion which we will use for training our models and another smaller portion which will be used to evaluate the models we build on data that the model has not seen before. For this project I went with a 65:35 split, with 65% of the observations for training and 35% for evaluating the models.

**Exploratory data analysis**

In this section I used charts and plots to understand the variables relationship in the dataset and the target variable for the training data. This helped me to select the variables that I want to use in building the machine learning model. My target variable was a categorical variable representing whether a website was malicious. There are two primary types of relationships that I needed to visualize to understand the relationships between predictor and target variable.

- Relationship between a numerical and a categorical variable
  - To visualize this type of relationship, I use a combination chart as shown below.
  - The bar plot on the top of the combination chart bins the numerical variable into buckets along the distribution and represents the number of observations in each bucket.
  - The stacked bar plot at the bottom of the combination chart then represents each bucket as a 100 percent and shows what proportion of the observation in each bucket were malicious (red) and benign (green)
The way I interpret this chart is that as the URL length increase the propensity of a website being malicious decreases until the length of the URL becomes significantly large. At that point the chances that the website is malicious increases again.

Figure 13: Example visualization used for exploring the relationships between numerical and categorical variables

- Relationship between two categorical variables
  - To visualize such relationships, I use a combination chart as shown below.
  - The bar plot on the top represents the frequency distribution of various categories in the categorical variable.
  - The stacked bar plot at the bottom of the combination chart then represents each category as a 100 percent and shown what proportion of the observation in each category were malicious (red) and benign (green).
  - The way I interpret this chart is that websites whose CHARSET is UTF have the highest propensity to be malicious.
  - While the proportion of malicious websites in the ‘Windows’ category is quite high the actual number of observations in that category is very low.
Similarly, I performed visualization for each variable in the dataset. The full set of charts can be found in the appendix. I present some of the more important charts in the results section.

![Chart 1](image1.png)

**Figure 14: Example visualization used for exploring the relationships between two categorical variables**

**Machine Learning**

**Logistic Regression**

Starting with the simplest model, we built a multivariate binary logistic regression model with the variable ‘Type’ as the target variable and all remaining variables in the dataset as predictor variables. We also used 5-fold cross validation for training to reduce the chances of overfitting. On inspection of the model summary, a second multivariate binary logistic regression model was built using only those predictor variables who were determined to significantly (p-value < 0.05) affect the target variable in the first model. This was done to simplify the model and prevent overfitting.
**Decision Tree**

The decision tree was trained recursively using the CART method deployed in the RPART module of R. The decision tree was trained using only the features that were found significant by the logistic regression model to reduce overfitting and simplify the model.

**Random Forest**

The algorithm was used to train a classification model using the “randomforest” package in R implemented using Caret. This implementation is the same as in the original random forest paper 5-fold cross validation was used in order to reduce overfitting.

**Support Vector Machine**

The algorithm was used to train a classification model using the implementation in the kernLab package in R.

**Evaluation**

Since this is a classification problem, we will use the confusion matrix to evaluate the models we train. A confusion matrix is a 2-dimensional table that puts the actual value of an observation on the rows while the predicted values of an observation are on the columns (Tharwat, 2018). An example of a confusion matrix for binary classification (classification problem with only two possible outcomes) is shown in Figure 12.

![Confusion Matrix](image)

*Figure 12: An example of Confusion Matrix (Introduction To Data Mining | Complete Guide to Data Mining, 2019)*
Below I define key terminology and metrics associated with a confusion matrix (Brown, 2018):

- **True Positive**: These are observations that we predicted as ‘YES’ and were actually ‘YES’ as well. These are present on the bottom right cell of the matrix in Figure 12.
- **True Negative**: These are observations that we predicted as ‘NO’ and were actually ‘NO’ as well. These are present on the top left cell of the matrix in Figure 12.
- **False Positive**: These are observations that we predicted as ‘YES’ and were actually ‘NO’. These are present on the top right cell of the matrix in Figure 12. These are also known as Type I error for a model.
- **False Negative**: These are observations that we predicted as ‘NO’ and were actually ‘YES’. These are present on the bottom left cell of the matrix in Figure 12. These are also known as Type II error for a model.
- **Recall**: This tells us how many of the actual positive observations in our database were we able to predict correctly. This is also known as Sensitivity of a model.

\[
Recall = \frac{TruePositive}{TruePositive + FalseNegative}
\]

- **Precision**: This tells us how many of the observations that we have predicted to be positive are actual positives.

\[
Precision = \frac{TruePositive}{TruePositive + FalsePositive}
\]

- **Specificity**: This tells us how many of the actual negative observations in our dataset we were able to predict correctly.

\[
Specificity = \frac{TrueNegative}{TrueNegative + FalsePositive}
\]

- **Accuracy**: This tells us how many observations we predicted correctly regardless of whether they are negative or positive.

\[
Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}
\]

- **F1-Score**: This is a measure that measure the balance between recall and precision of a model simultaneously. It is an important measure as it can be difficult to compare models with high
recall and low precision to models with low recall and high precision. We will be using the F1 Score to evaluate the models we train.

\[
F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}
\]

For this project, we will need to ensure that our models are able to detect as many of the malicious websites as possible, even if it comes at the cost of predicting some benign websites as malicious. This is because classifying a malicious website as benign is very costly while classifying a benign website as malicious is not that costly.

Each model except the SVM gave the probability of a website being malicious or not. We set a threshold of 20% such that any website whose probability of being malicious was more than 20% was classified as malicious. This low threshold was chosen keeping in mind that it is more important to correctly classify malicious websites as compared to correctly classifying benign websites.

The results and predictions generated by models trained using each algorithm were tabulated as a confusion matrix. The models were compared based on F1-score on the test data, which strikes a balance between precision and recall values.

**Deployment**

The model will be stored in a file for further deployment

**Tools**

R, RStudio and a variety of R libraries were used for this project

**RESULTS**

**Exploratory Data Analysis**

Based on the visualizations we did in the Exploratory data analysis part the following features showed significant relationship with the target variable.

- URL Length
- Number of Special Characters
• Server
• Charset
• Country
• Dist Remote TCP Port
• Remote IPs
• Remote App Packets
• Source App Bytes
• Site Age

**Number of Special Characters**

The trend shows that with an increase in the URL’s special characters, the probability of the website being malicious also increases.

*Figure 15: Exploratory Plot for the variable “NUMBER_SPECIAL_CHARACTERS”*
Server

In this variable we see the server types ‘Apache’ and ‘Nginx’ being associated with higher probabilities of a site being malicious as compared to the other server types.

![Exploratory Plot for the variable “SERVER”](image)

**Figure 16: Exploratory Plot for the variable “SERVER”**

**Machine Learning Models**

**Logistic Regression with All Variables**

**Table 2. Coefficients summary table for logistic regression with all variables. The variables not found to significantly affect the target are highlighted in red**

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Coefficient</th>
<th>Standard_Error</th>
<th>Z_Value</th>
<th>P_Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-3.017</td>
<td>1.342</td>
<td>-2.249</td>
<td>0.025</td>
</tr>
<tr>
<td>URL_LENGTH</td>
<td>-0.156</td>
<td>0.021</td>
<td>-7.423</td>
<td>0.000</td>
</tr>
<tr>
<td>NUMBER_SPECIAL_CHARACTERS</td>
<td>1.101</td>
<td>0.129</td>
<td>8.560</td>
<td>0.000</td>
</tr>
<tr>
<td>CHARSETISO</td>
<td>1.835</td>
<td>1.083</td>
<td>1.694</td>
<td>0.090</td>
</tr>
<tr>
<td>CHARSETUnknown</td>
<td>18.198</td>
<td>17730.372</td>
<td>0.001</td>
<td>0.999</td>
</tr>
<tr>
<td>CHARSETUTF</td>
<td>2.122</td>
<td>1.065</td>
<td>1.993</td>
<td>0.046</td>
</tr>
<tr>
<td>CHARSETWindows</td>
<td>4.765</td>
<td>1.915</td>
<td>2.488</td>
<td>0.013</td>
</tr>
<tr>
<td>SERVERMicrosoft</td>
<td>0.314</td>
<td>0.611</td>
<td>0.513</td>
<td>0.608</td>
</tr>
</tbody>
</table>
Table 2 shows the variables that show a significant relationship with a website being malicious or not. The logistic regression output can be interpreted as follows:

- “URL_LENGTH” has a coefficient of -0.156. The negative sign signifies that an increase in URL length is associated with a decrease in the probability of a website being malicious.
- “NUMBER_SPECIAL_CHARACTERS” has a coefficient of 1.101. The positive sign means that an increase in this variable is associated with an increase in the probability of a website being malicious.
- Any increase in a variable with a p-value < 0.05 and a positive coefficient will increase the probability of the website being malicious.
- Any decrease in a variable with a p-value < 0.05 and a negative coefficient will increase the probability of the website being malicious.

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Coefficient</th>
<th>Standard_Error</th>
<th>Z_Value</th>
<th>P_Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SERVERnginx</td>
<td>-0.116</td>
<td>0.325</td>
<td>-0.358</td>
<td>0.721</td>
</tr>
<tr>
<td>SERVEROthers</td>
<td>-0.912</td>
<td>0.551</td>
<td>-1.655</td>
<td>0.098</td>
</tr>
<tr>
<td>SERVERUnknown</td>
<td>-19.621</td>
<td>921.506</td>
<td>-0.021</td>
<td>0.983</td>
</tr>
<tr>
<td>WHOIS_COUNTRYUSA</td>
<td>-1.301</td>
<td>0.310</td>
<td>-4.193</td>
<td>0.000</td>
</tr>
<tr>
<td>TCP_CONVERSATION_EXCHANGE</td>
<td>7.511</td>
<td>2.720</td>
<td>2.762</td>
<td>0.006</td>
</tr>
<tr>
<td>DIST_REMOTE_TCP_PORT</td>
<td>-0.811</td>
<td>0.139</td>
<td>-5.846</td>
<td>0.000</td>
</tr>
<tr>
<td>REMOTE_IPS</td>
<td>-0.178</td>
<td>0.113</td>
<td>-1.581</td>
<td>0.114</td>
</tr>
<tr>
<td>APP_BYTES</td>
<td>-0.001</td>
<td>0.008</td>
<td>-0.107</td>
<td>0.915</td>
</tr>
<tr>
<td>SOURCE_APP_PACKETS</td>
<td>-7.509</td>
<td>2.711</td>
<td>-2.770</td>
<td>0.006</td>
</tr>
<tr>
<td>REMOTE_APP_PACKETS</td>
<td>0.188</td>
<td>0.061</td>
<td>3.087</td>
<td>0.002</td>
</tr>
<tr>
<td>SOURCE_APP_BYTES</td>
<td>0.000</td>
<td>0.000</td>
<td>-3.805</td>
<td>0.000</td>
</tr>
<tr>
<td>REMOTE_APP_BYTES</td>
<td>0.001</td>
<td>0.008</td>
<td>0.145</td>
<td>0.885</td>
</tr>
<tr>
<td>APP_PACKETS</td>
<td>0.019</td>
<td>0.017</td>
<td>1.125</td>
<td>0.261</td>
</tr>
<tr>
<td>DNS_QUERY_TIMES</td>
<td>7.218</td>
<td>2.558</td>
<td>2.822</td>
<td>0.005</td>
</tr>
<tr>
<td>site_age</td>
<td>-0.012</td>
<td>0.002</td>
<td>-4.739</td>
<td>0.000</td>
</tr>
<tr>
<td>Update_age</td>
<td>-0.047</td>
<td>0.018</td>
<td>-2.647</td>
<td>0.008</td>
</tr>
</tbody>
</table>
Logistic Regression with a variable selected according to p-value

Table 3. Coefficients summary table for logistic regression with a subset of variables. The variables not found to significantly affect the target are highlighted in red

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Coefficient</th>
<th>Standard_Error</th>
<th>Z_Value</th>
<th>P_Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-4.814</td>
<td>1.086</td>
<td>-4.434</td>
<td>0.000</td>
</tr>
<tr>
<td>URL_LENGTH</td>
<td>-0.151</td>
<td>0.019</td>
<td>-7.811</td>
<td>0.000</td>
</tr>
<tr>
<td>NUMBER_SPECIAL_CHARACTERS</td>
<td>1.070</td>
<td>0.119</td>
<td>8.959</td>
<td>0.000</td>
</tr>
<tr>
<td>CHARSETISO</td>
<td>1.963</td>
<td>1.003</td>
<td>1.958</td>
<td>0.050</td>
</tr>
<tr>
<td>CHARSETUnknown</td>
<td>-1.852</td>
<td>10754.013</td>
<td>0.000</td>
<td>1.000</td>
</tr>
<tr>
<td>CHARSETUTF</td>
<td>2.018</td>
<td>0.980</td>
<td>2.060</td>
<td>0.039</td>
</tr>
<tr>
<td>CHARSETWindows</td>
<td>4.284</td>
<td>1.783</td>
<td>2.403</td>
<td>0.016</td>
</tr>
<tr>
<td>SERVERMicrosoft</td>
<td>0.178</td>
<td>0.577</td>
<td>0.308</td>
<td>0.758</td>
</tr>
<tr>
<td>SERVERNginx</td>
<td>-0.123</td>
<td>0.305</td>
<td>-0.403</td>
<td>0.687</td>
</tr>
<tr>
<td>SERVEROthers</td>
<td>-1.056</td>
<td>0.539</td>
<td>-1.958</td>
<td>0.050</td>
</tr>
<tr>
<td>SERVERUnknown</td>
<td>-15.790</td>
<td>810.065</td>
<td>-0.019</td>
<td>0.984</td>
</tr>
<tr>
<td>WHOIS_COUNTRYUSA</td>
<td>-1.101</td>
<td>0.283</td>
<td>-3.895</td>
<td>0.000</td>
</tr>
<tr>
<td>DIST_REMOTE_TCP_PORT</td>
<td>-0.518</td>
<td>0.104</td>
<td>-4.974</td>
<td>0.000</td>
</tr>
<tr>
<td>REMOTE_IPS</td>
<td>-0.160</td>
<td>0.094</td>
<td>-1.694</td>
<td>0.090</td>
</tr>
<tr>
<td>REMOTE_APP_PACKETS</td>
<td>0.150</td>
<td>0.030</td>
<td>5.049</td>
<td>0.000</td>
</tr>
<tr>
<td>SOURCE_APP_BYTES</td>
<td>0.000</td>
<td>0.000</td>
<td>-4.600</td>
<td>0.000</td>
</tr>
<tr>
<td>site_age</td>
<td>-0.013</td>
<td>0.002</td>
<td>-5.523</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Table 3 shows the variables that show a significant relationship with a website being malicious or not. The logistic regression output can be interpreted as follows:

- **“URL_LENGTH”** has a coefficient of -0.151. The negative sign signifies that an increase in URL length is related with decrease in probability of a website being malicious given that all other variables are kept the same.
- **“NUMBER_SPECIAL_CHARACTERS”** has a coefficient of 1.070. The positive sign means that an increase in this variable is related with an increase in the probability of a website being malicious given that all other variables are kept the same.
- Any increase in a variable with a p-value < 0.05 and a positive coefficient will increase the probability of the website being malicious given that all other variables are kept the same.
- Any decrease in a variable with a p-value < 0.05 and a negative coefficient will increase the probability of the website being malicious given that all other variables are kept the same.
Decision Tree

The Figure above shows the decision tree that has been trained for this project. The tree can be interpreted as follows:

- We see that the variable at the root node is “NUMBER_SPECIAL_CHARACTERS”. This means that this model considers this variable as the most important variable.
  - Websites with more than 20 special characters are more likely to be malicious.
  - Websites that have more than 20 special characters and are not originated in the USA have the highest chance of being malicious.

- Other Important variables are:
  - DIST_REMOTE_TCP_PORT: websites with this variable being higher than 2 are less likely to be malicious.
  - REMOTE_APP_PACKETS: websites with this variable being less than 1 are less likely to be malicious.
  - SOURCE_APP_BYTES: websites with this variable being higher than 13,000 are less likely to be malicious.
URL_LENGTH: websites with this variable being less than 44 are less likely to be malicious.
REMOTE_IPS: websites with this variable being higher than 1 are less likely to be malicious.

Random Forest

![Variable importance according to the random forest model](image)

Figure 18: Variable importance according to the random forest model

In the random forest-based model, we see that the variable “site_age” comes out to be the most important feature. This is followed by the feature ‘SOURCE_APP_BYTES’ and “NUMBER_SPECIAL_CHARACTER”. These importance’s are measured using the average decrease in the GINI impurity across all trees.
Support Vector Machine

*Table 4. Variable Importance for predicting website Type according to Support Vector Machine*

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>site_age</td>
<td>73.53</td>
</tr>
<tr>
<td>WHOIS_COUNTRY</td>
<td>72.44</td>
</tr>
<tr>
<td>NUMBER_SPECIAL_CHARACTERS</td>
<td>70.91</td>
</tr>
<tr>
<td>DIST_REMOTE_TCP_PORT</td>
<td>66.47</td>
</tr>
<tr>
<td>SERVER</td>
<td>66.09</td>
</tr>
<tr>
<td>CHARSET</td>
<td>59.47</td>
</tr>
<tr>
<td>SOURCE_APP_BYTES</td>
<td>55.42</td>
</tr>
<tr>
<td>REMOTE_APP_PACKETS</td>
<td>54.51</td>
</tr>
<tr>
<td>URL_LENGTH</td>
<td>54.40</td>
</tr>
<tr>
<td>REMOTE_IPS</td>
<td>54.15</td>
</tr>
</tbody>
</table>

The most important variable according to the SVM based model is the ‘site_age’. This is followed by the features “WHOIS_COUNTRY’ and” NUMBER_SPECIAL_CHARACTERS”.

Model Comparison

*Table 5. Comparison of Model metrics on the training set*

<table>
<thead>
<tr>
<th>Training Algorithm</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Balanced Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic_Full</td>
<td>0.89</td>
<td>0.92</td>
<td>0.60</td>
<td>0.89</td>
<td>0.72</td>
<td>0.91</td>
</tr>
<tr>
<td>Logistic_Partial</td>
<td>0.89</td>
<td>0.91</td>
<td>0.57</td>
<td>0.89</td>
<td>0.69</td>
<td>0.90</td>
</tr>
<tr>
<td>Decision_Tree</td>
<td>0.87</td>
<td>0.91</td>
<td>0.58</td>
<td>0.87</td>
<td>0.69</td>
<td>0.89</td>
</tr>
<tr>
<td>Random_Forest</td>
<td>1.00</td>
<td>0.99</td>
<td>0.95</td>
<td>1.00</td>
<td>0.98</td>
<td>1.00</td>
</tr>
<tr>
<td>Support_Vector_Machine</td>
<td>0.56</td>
<td>0.98</td>
<td>0.81</td>
<td>0.56</td>
<td>0.66</td>
<td>0.77</td>
</tr>
</tbody>
</table>

Here we can see the performance comparison for all models on the training set. It is easy to see that all models except SVM show reasonably high accuracy. When evaluated according to the F1-score, we see that the Random forest-based model outperforms the rest followed by logistic regression.

*Table 6. Comparison of Model metrics on the testing set*

<table>
<thead>
<tr>
<th>Training Algorithm</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Balanced Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic_Full</td>
<td>0.79</td>
<td>0.90</td>
<td>0.53</td>
<td>0.79</td>
<td>0.63</td>
<td>0.85</td>
</tr>
<tr>
<td>Logistic_Partial</td>
<td>0.77</td>
<td>0.92</td>
<td>0.56</td>
<td>0.77</td>
<td>0.65</td>
<td>0.84</td>
</tr>
<tr>
<td>Decision_Tree</td>
<td>0.83</td>
<td>0.91</td>
<td>0.56</td>
<td>0.83</td>
<td>0.67</td>
<td>0.87</td>
</tr>
<tr>
<td>Random_Forest</td>
<td>0.91</td>
<td>0.94</td>
<td>0.68</td>
<td>0.91</td>
<td>0.78</td>
<td>0.92</td>
</tr>
<tr>
<td>Support_Vector_Machine</td>
<td>0.55</td>
<td>0.98</td>
<td>0.82</td>
<td>0.55</td>
<td>0.66</td>
<td>0.77</td>
</tr>
</tbody>
</table>
Here we can see the performance comparison for all models on the testing set. It is easy to see that all models except SVM show reasonably high accuracy. When evaluated according to the F1-score, we see that the Random forest-based model outperforms the rest. The other models all have a very similar performance.

**The Best Model**

![Confusion Matrix](image)

*Figure 19: Confusion Matrix for the Random forest model on the testing dataset*

**Discussion**

**Variables important in detecting malicious websites**

We used a variety of different variables to train classification models based on 4 different algorithms. Across the four different models, we observed a few variables that were considered critical by all. Since these variables are being considered important by each model, it is worth investigating this importance.

**Number of Special Characters**

Represented by the variable ‘NUMBER_SPECIAL_CHARACTERS’, this variable was considered important by each one of our classification models. This makes a lot of sense as special
characters are not typically used in the websites we use often. Previous researches in this space also found the high count of special characters are typically observed in malicious websites (Kolari, Finin and Joshi, 2006; Choi, Zhu and Lee, 2011). One reason for this could be that the special characters make the URL stand out and this could maybe increase clicks. Another reason for this could be because a lot of malicious websites can contain currency symbols, which are again treated as special characters. Overall, a large number of special characters found in the URL of a website does serve as an indicator that the website is likely to be malicious.

Site age
It was seen quite clearly that the age of a website had an effect on the likelihood of a website being malicious or not. Older websites were predicted as more likely to be malicious than newer ones given all other information remained the same. This again makes sense as older website created using legacy code may possess many insecure bits of code that can leave it open to malicious attacks.

CHARSET
We observed that all types of CHARSET except ASCII were heavily associated with increasing the probability of a website being malicious. One reason for this can be the tight constraints used by the ASCII CHARSET as compared to UTF-8 charsets.

COUNTRY
We observed that websites with country as USA were less likely to be malicious as compared to those with other countries. This can be because of better cyber security measures in the USA as compared to the rest of the world.

Machine Learning Model Comparison
We observed that the Random Forest algorithm was the best performing model out of the 5 models we trained. This makes sense due to the following reasons:

• Logistic regression is a linear model as well as heavily sensitive to the distributions of predictor variables. It can be the case that the relationships between the predictors and the target are non-linear in nature (Sperandei, 2014).
• Decision trees are very simplistic and many decision trees come together to form a random forest (Rokach and Maimon, 2005).
• Support Vector machine is also a linear model. Finding one decision boundary with so many dimensions can be difficult (Shihong, Ping and Peiyi, 2003).
• Lastly, Random forests are ensemble methods and have been shown to perform extremely robustly in a variety of classification problems (Fawagreh, Gaber and Elyan, 2014).

The best F1-score we achieved on the test set was equal to 0.78. This is a score while fairly decent can definitely be improved upon. One clear way of improving this will be to use sampling techniques to tackle the problem of class imbalance as only 12% of the websites in our data were malicious. Oversampling of the malicious websites can help in allowing the model to learn patterns that increase the likelihood of a website being malicious much better (Guo et al., 2008). Another way of improving the model performance could be by using newer and more sophisticated training algorithms such as gradient boosters and neural networks.

**Conclusion**

We cleaned and prepared a dataset containing malicious and benign websites and used it to train a classification model which predicts whether a website will be malicious or not based on a selection of features. We have showcased that it is possible to detect malicious websites with a reasonable amount of certainty (90% of the 75 malicious websites in the test set were identified) using machine learning models. We have also determined the features that were critical in predicting the likelihood of a website being malicious. Most of our key features are easily available (URL Length, number of Special characters, Country, Age of website). Further improvements can be made on our results by using sampling techniques to deal with class imbalance and using more sophisticated models.
References


Appendix

Exploratory Plots for Numerical Variables
Exploratory Plots for Categorical Variables
Confusion Matrices

Logistic Regression with All Variables

**Training Set**

![Training Set Confusion Matrix]

**Testing Set**

![Testing Set Confusion Matrix]
Logistic Regression Subset

Training Set

![Confusion Matrix for Training Set](image1)

**Details**

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.887</td>
<td>0.950</td>
<td>0.986</td>
<td>0.897</td>
<td>0.991</td>
</tr>
</tbody>
</table>

Accuracy: 0.903

Kappa: 0.637

Testing Set

![Confusion Matrix for Testing Set](image2)

**Details**

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.773</td>
<td>0.918</td>
<td>0.558</td>
<td>0.773</td>
<td>0.848</td>
</tr>
</tbody>
</table>

Accuracy: 0.899

Kappa: 0.591
Decision Tree

Training Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th>Actual</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted</td>
<td>928</td>
<td>19</td>
</tr>
<tr>
<td>0</td>
<td>90</td>
<td>122</td>
</tr>
</tbody>
</table>

DETAILS

Sensitivity | 0.865
Specificity | 0.912
Precision | 0.575
Recall | 0.865
F1 | 0.691
Accuracy | 0.906
Kappa | 0.638

Testing Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th>Actual</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted</td>
<td>498</td>
<td>13</td>
</tr>
<tr>
<td>0</td>
<td>49</td>
<td>82</td>
</tr>
</tbody>
</table>

DETAILS

Sensitivity | 0.927
Specificity | 0.91
Precision | 0.559
Recall | 0.827
F1 | 0.887
Accuracy | 0.9
Kappa | 0.611
Support Vector Machine

Training Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>999</td>
<td>63</td>
</tr>
<tr>
<td>1</td>
<td>19</td>
<td>78</td>
</tr>
</tbody>
</table>

DETAILS

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
<th>Kappa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.953</td>
<td>0.981</td>
<td>0.904</td>
<td>0.553</td>
<td>0.655</td>
<td>0.929</td>
<td>0.618</td>
</tr>
</tbody>
</table>

Testing Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>538</td>
<td>34</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>41</td>
</tr>
</tbody>
</table>

DETAILS

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
<th>Kappa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.547</td>
<td>0.964</td>
<td>0.82</td>
<td>0.547</td>
<td>0.650</td>
<td>0.931</td>
<td>0.619</td>
</tr>
</tbody>
</table>
Random Forest

Training Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1012</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>141</td>
</tr>
</tbody>
</table>

DETAILS

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0.994</td>
<td>0.959</td>
<td>1</td>
<td>0.979</td>
</tr>
</tbody>
</table>

Accuracy: 0.995
Kappa: 0.976

Testing Set

CONFUSION MATRIX

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>515</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>68</td>
</tr>
</tbody>
</table>

DETAILS

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.907</td>
<td>0.841</td>
<td>0.85</td>
<td>0.907</td>
<td>0.777</td>
</tr>
</tbody>
</table>

Accuracy: 0.937
Kappa: 0.742