Detection and Prediction of Phishing Websites

Submitted by:

Thomas K John

Batch 30

International School Of Engineering, Bangalore

Table of Contents

1. Introduction

1.1. Problem Statement

Phishing websites are those websites that mimic the legitimate websites and deceive online users in order to steal their sensitive information. Phishing website identification can be defined in machine learning context as a typical classification problem. The aim is to predict the type of the website in an automated manner to either "legitimate" or "phishy" class labels based on a classifier generated from the training dataset.

1.2. Data Description

The dataset is taken from the UCI Machine Learning Repository. The dataset is a collection of 11055 different websites from different sources like PhishTank archive (www.phishtank.com), MillerSmiles archive (www.millersmiles.co.uk) and Google's searching operators. Out of the 11055 websites collected there are 4898 phishing websites and 6157 legitimate websites. The dataset has 30 features and all the collected features hold the categorical values with Legitimate as 1, Suspicious 0 and Phishing as -1.
These 30 features are grouped into 4 groups based on their common properties:-

Address Bar based Features

a. **"having_IP_Address":**

Using IP address as an alternative for the domain name in the URL.

Rule: IF $\begin{cases} \text{If The Domain Part has an IP Address} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$

b. **"URL_Length":**

Phishers can use long URL to hide the doubtful part in the address bar.

Rule: IF $\begin{cases} URL\ length < 54 \rightarrow feature = \text{Legitimate} \\ else\ if\ URL\ length \geq 54\ and\ \leq 75 \rightarrow feature = Suspicious \\ otherwise \rightarrow feature = \text{Phishing} \end{cases}$

c. "Shortening_Service":

URL shortening is a method on the "World Wide Web" in which a URL may be made considerably smaller in length and still lead to the required webpage.

Rule: IF $\begin{cases} \text{TinyURL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$

d. "having_At_Symbol":

Using "@" symbol in the URL leads the browser to ignore everything preceding the "@"symbol and the real address often follows the "@" symbol.

$$\text{Rule: IF} \begin{cases} \text{Url Having @ Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

e. "double_slash_redirecting":

The existence of "//" within the URL path means that the user will be redirected to another website.

$$\text{Rule: IF} \begin{cases} \text{ThePosition of the Last Occurrence of "//" in the URL} > 7 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

f. "Prefix_Suffix":

Phishers tend to add prefixes or suffixes separated by (-) to the domain name.

$$\text{Rule: IF} \begin{cases} \text{Domain Name Part Includes } (-) \text{ Symbol} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

g. "having_Sub_Domain":

Phishers tend to have more than two sub-domains.

$$\text{Rule: IF} \begin{cases} \text{Dots In Domain Part} = 1 \rightarrow \text{Legitimate} \\ \text{Dots In Domain Part} = 2 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

h. "SSLfinal_State":

Using Hyper Text Transfer Protocol with Secure Sockets Layer

$$\text{Rule: IF} \begin{cases} \text{Use https and Issuer Is Trusted and Age of Certificate} \geq 1 \text{ Years} \rightarrow \text{Legitimate} \\ \text{Using https and Issuer Is Not Trusted} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

i. "Domain_registration_length":

It is observed that a phishing website lives only for a short period of time.

$$\text{Rule: IF} \begin{cases} \text{Domains Expires on} \leq 1 \text{ years} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

j. "Favicon":

A favicon is a graphic image (icon) associated with a specific webpage.

$$\text{Rule: IF} \begin{cases} \text{Favicon Loaded From External Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

k. "port":

This feature is useful in validating if a particular service (e.g. HTTP) is up or down on a specific server.

$$\text{Rule: IF} \begin{cases} \text{Port \# is of the Preffered Status} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

l. "HTTPS_token":

The phishers may add the "HTTPS" token to the domain part of a URL in order to trick users.

$$\text{Rule: IF} \begin{cases} \text{Using HTTP Token in Domain Part of The URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Abnormal based Features

a. "Request_URL":

Examines whether the external objects contained within a webpage such as images, videos and sounds are loaded from another domain.

$$\text{Rule: IF} \begin{cases} \%\text{ of Request URL} < 22\% \rightarrow \text{Legitimate} \\ \%\text{of Request URL} \geq 22\% \text{ and } 61\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{feature} = \text{Phishing} \end{cases}$$

b. "URL_of_Anchor":

Examines whether the <a> tags and the website have different domain name. Also verify if the anchor does not link to any webpage.

$$\text{Rule: IF} \begin{cases} \%\text{ of URL Of Anchor} < 31\% \rightarrow \textit{Legitimate} \\ \%\text{ of URL Of Anchor} \geq 31\% \text{ And} \leq 67\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

c. "Links_in_tags":

It is expected that <Meta>, <Script> and <Link> tags are linked to the same domain of the webpage for legitimate websites.

Rule:

$$\text{IF} \begin{cases} \%\text{ of Links in "} < \text{Meta} > \text{","} < \text{Script} > \text{" and "} < \text{Link>"} < 17\% \rightarrow \text{Legitimate} \\ \%\text{ of Links in } < \text{Meta} > \text{","} < \text{Script} > \text{" and "} < \text{Link>"} \geq 17\% \text{ And} \leq 81\% \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

d. "SFH":

Server Form Handlers that contain an empty string or "about:blank" are considered doubtful because an action should be taken upon the submitted information.

$$\text{Rule: IF} \begin{cases} \text{SFH is "about: blank" Or Is Empty} \rightarrow \text{Phishing} \\ \text{SFH Refers To A Different Domain} \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

e. "Submitting_to_email":

Web form allows a user to submit his personal information that is directed to a server for processing. A phisher might redirect the user's information to his personal email.

$$\text{Rule: IF} \begin{cases} \text{Using "mail()" or "mailto:" Function to Submit User details} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

f. "Abnormal_URL":

This feature can be extracted from WHOIS database. For a legitimate website, identity is typically part of its URL.

$$\text{Rule: IF} \begin{cases} \text{The Host Name Is Not Included In URL} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

HTML and JavaScript based Features

a. "Redirect":

Distinguishing phishing websites from legitimate ones based on how many times a website has been redirected.

$$\text{Rule: IF} \begin{cases} \#\text{ofRedirect Page} \leq 1 \rightarrow \text{Legitimate} \\ \#\text{of Redirect Page} \geq 2 \text{ And} < 4 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

b. "on_mouseover":

Phishers may use JavaScript events, particularly the "onMouseOver" event, to show a fake URL in the status bar to users.

$$\text{Rule: IF} \begin{cases} \text{onMouseOver Changes Status Bar} \rightarrow \text{Phishing} \\ \text{It Does't Change Status Bar} \rightarrow \text{Legitimate} \end{cases}$$

c. "RightClick":

Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code.

$$\text{Rule: IF} \begin{cases} \text{Right Click Disabled} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

d. "popUpWindow":

It is unusual to find a legitimate website asking users to submit their personal information through a pop-up window.

$$\text{Rule: IF} \begin{cases} \text{Popoup Window Contains Text Fields} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

e. "Iframe":

IFrame is an HTML tag used to display an additional webpage into one that is currently shown.

$$\text{Rule: IF} \begin{cases} \text{Using iframe} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

Domain based Features

a. "age_of_domain":

Most phishing websites live for a short period of time.

$$\text{Rule: IF} \begin{cases} \text{Age Of Domain} \geq 6 \text{ months} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

b. "DNSRecord":

If the DNS record is empty or not found then the website is classified as "Phishing", otherwise it is classified as "Legitimate".

$$\text{Rule: IF} \begin{cases} \text{no DNS Record For The Domain} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

c. "web_traffic":

This feature measures the popularity of the website by determining the number of visitors and the number of pages they visit. However, since phishing websites live for a short period of time, they may not be recognized by the Alexa database.

$$\text{Rule: IF} \begin{cases} \text{Website Rank} < 100{,}000 \rightarrow \text{Legitimate} \\ \text{Website Rank} > 100{,}000 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Phish} \end{cases}$$

d. "Page_Rank":

PageRank is a value ranging from "0" to "1". PageRank aims to measure how important a webpage is on the Internet.

$$\text{Rule: IF} \begin{cases} \text{PageRank} < 0.2 \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

e. "Google_Index":

This feature examines whether a website is in Google's index or not.

$$\text{Rule: IF} \begin{cases} \text{Webpage Indexed by Google} \rightarrow \text{Legitimate} \\ \text{Otherwise} \rightarrow \text{Phishing} \end{cases}$$

f. "Links_pointing_to_page":

The number of links pointing to the webpage indicates its legitimacy level.

$$\text{Rule: IF} \begin{cases} \#\text{Of Link Pointing to The Webpage} = 0 \rightarrow \text{Phishing} \\ \#\text{Of Link Pointing to The Webpage} > 0 \text{ and} \leq 2 \rightarrow \text{Suspicious} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

g. "Statistical_report":

Several parties such as PhishTank (PhishTank Stats, 2010-2012), and StopBadware (StopBadware, 2010-2012) formulate numerous statistical reports on phishing websites at every given period of time.

$$\text{Rule: IF} \begin{cases} \text{Host Belongs to Top Phishing IPs or Top Phishing Domains} \rightarrow \text{Phishing} \\ \text{Otherwise} \rightarrow \text{Legitimate} \end{cases}$$

2. Models Used

This is a classification task and the aim is to classify the websites to either "legitimate" or "phishy" based on the predictor variables. The dataset has phishing websites and legitimate websites in the ratio 55.6 : 44.3. The dataset is split into training set and testing set by stratified sampling method to maintain the same proportion.

Following are the important features based on the ANOVA F- score:-

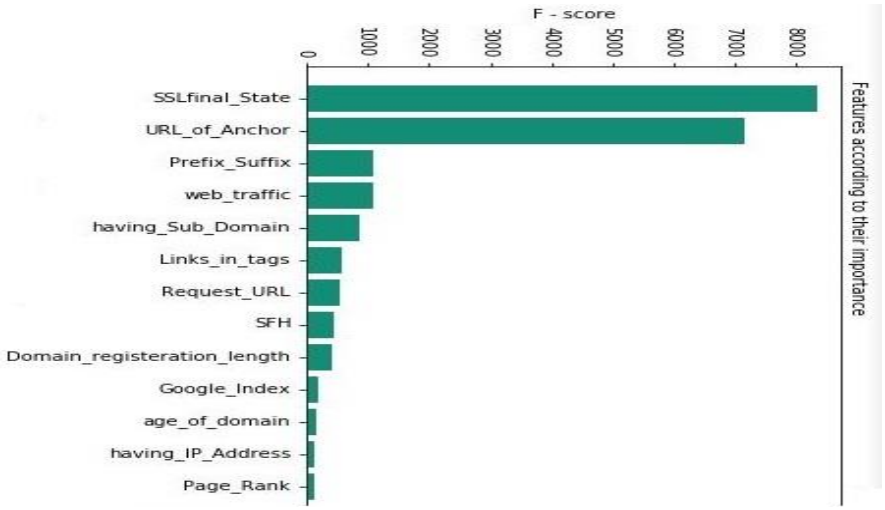

Figure 1: Important Variables from F-score

2.1. Logistic Regression

Logistic regression assumes that there is one smooth linear decision boundary. It finds that linear decision boundary by making assumptions that the P(Y|X) of some form. The logistic regression equation for n features is given as

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 \ldots . + \beta_n * x_n$$

Let F be the linear combination of all the predicators, $F = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 \ldots . + \beta_n * x_n$

Then the above equation can be written as: $p = \frac{1}{1+e^{-F}}$

Logistic regression outperforms other models when the data is having high noise, i.e. if the accuracy of the best model is in the higher range. Similarly logistic regression would give better results when the predictor variables are numerical compared to categorical variables. In phishing dataset the predictor variables are all categorical variables and the accuracy obtained for a simple logistic regression is high and that shows the model has less noise.

Results:

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Logistic Regression (LR)	92.98%	92.99%	92.98%	92.97%	92.52%	92.53%	92.52%	92.51%
LR with k-fold CV	93.03%	93.04%	93.03%	93.02%	92.61%	92.62%	92.61%	`92.60%
LR with L1 regularization	93.02%	93.03%	93.02%	93.01%	92.64%	92.66%	92.64%	92.63%
LR with L2 regularization	92.99%	93.00%	93.00%	92.99%	92.52%	92.53%	92.52%	92.51%
LR with 13 important variables	92.31%	92.32%	92.31%	92.30%	92.10%	92.13%	92.10%	92.08%

Table 1: Logistic Regression Results

Logistic Regression model with K-fold cross validation (k > 3) gave slightly better results compared to normal logistic regression. Logistic Regression with L1 regularization gave better results compared to L2 regularization. Also created a logistic regression model by selecting only the 13 important features and was able to achieve the recall rate which was only 0.7% lesser to the best logistic regression model created with all the 30 features.

2.2. Decision Tree Classifier

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. The leaf node represents a classification or decision. The top most decision node in a tree which corresponds to the best predictor is called the root node. Decision tree can handle both categorical and numerical variables.

Decision tree models are good classifiers when the decision boundaries are parallel to the axes, creating rectangular feature spaces. Decision trees can be applied to situations where there's not just one underlying decision boundary, but many, and will work best if the class labels roughly lie in hyper-rectangular regions.

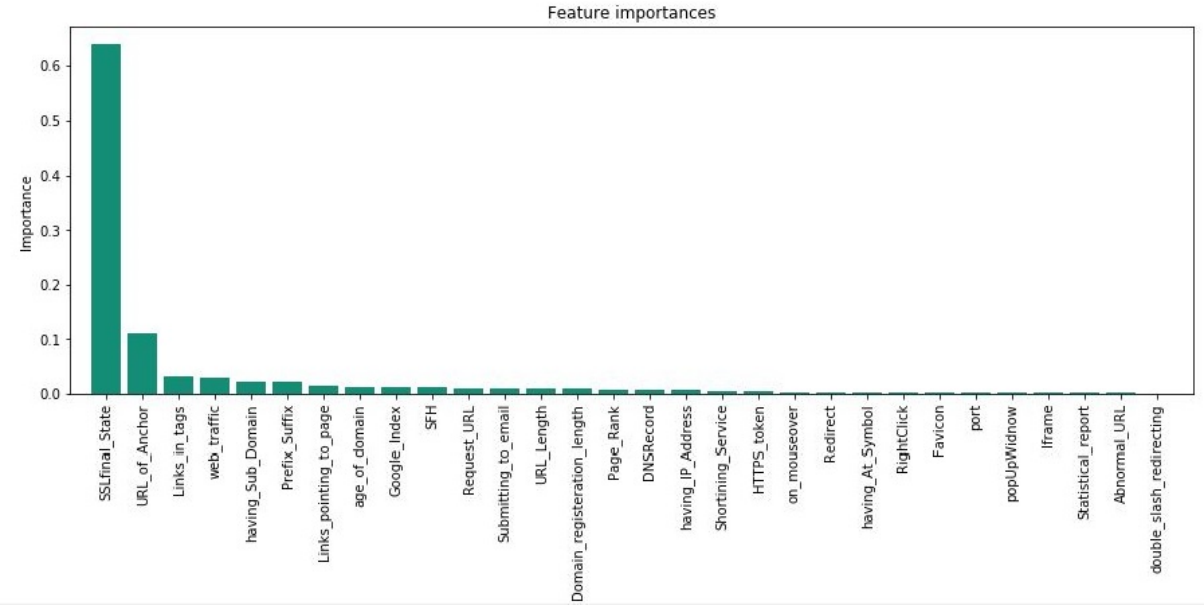

Figure 2: Feature Importance according to decision tree model

Result:

Decision Tree classifier gave a recall of 95.42% in the testing data. Also created a model by selecting only the 13 important features and was able to achieve the recall rate which was only 1 % less compared to the model created with all the 30 features.

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Decision Tree	98.52%	98.53%	98.53%	98.53%	95.42%	95.42%	95.42%	95.42%
Decision Tree with 13 important features	97.47%	97.47%	97.47%	97.47%	94.42%	94.43%	94.42%	94.42%

Table 2: Decision Tree Results

2.3. Random Forest

The random forest algorithm aggregates many decision trees built on bootstrapped samples of the training data in order to reduce the high variance of a single decision tree and improve the prediction accuracy.

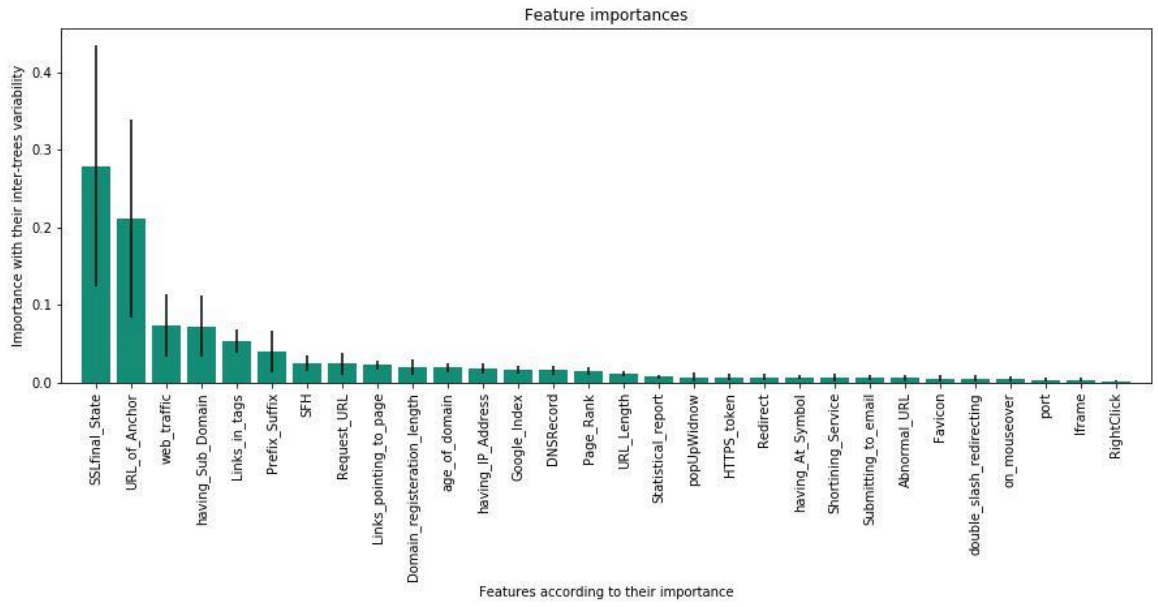

Figure 3: Feature Importance according to Random Forest model

Result:

Random forest classifier gave a recall of 96.90% in the testing data. Also created a model by selecting only the 13 important features and was able to achieve a recall which was only 1 % less compared to the model created with all the 30 features.

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Random Forest	99.03%	99.03%	99.03%	99.03%	96.89%	96.90%	96.89%	96.89%
Random Forest with 13 important features	97.56%	97.56%	97.56%	97.56%	95.60%	95.60%	95.60%	95.60%

Table 3: Random Forest Results

2.4. Naive Bayes

The Naive Bayesian classifier is based on Bayes' theorem with independence assumptions between predictors. Bayes theorem provides a way of calculating the posterior probability, $P(y|x)$, from $P(y)$, $P(x)$, and $P(x|y)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (y) is independent of the values of other predictors. This assumption is called class conditional independence.

$$P(y\,|x) = \frac{p(x|y)*P(y)}{P(x)}$$

$$P(x\,|y) = P(x_1|y) * \ P(x_2|y) * \ldots * \ P(x_n|y) * P(y)$$

- $P(y|x)$ is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
- $P(y)$ is the prior probability of *class*.
- $P(x|y)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

Results:

Naive Bayes gave the worst result (Recall of 61.32%) on test data as compared to other classifiers. In the dataset there are many features which are dependent on each other and hence the assumption of Naive Bayes Classifier is not true.

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
Naive Bayes	59.88%	78.68%	59.89%	54.90%	61.32%	79.25%	61.32%	56.92%
Naïve Bayes with 13 important variables	59.07%	78.73%	59.07%	53.65%	59.93%	78.96%	59.93%	54.91%

Table 4: Naive Bayes Results

2.5. K- Nearest Neighbors

K-Nearest Neighbors is a simple algorithm that stores all available instances and classifies new instances based on a similarity measure (e.g., distance functions). Euclidean distance, Manhattan distance and Minkowski distance are the commonly used functions for continuous variables. In the instance of categorical variables the Hamming distance must be used.

For numerical variables:

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

$$\text{Manhattan distance} = \sum_{i=1}^{k}|x_i - y_i|$$

$$\text{Minkowski distance} = \left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{\frac{1}{q}}$$

For categorical variables:

$$\text{Hamming distance}, D_H = \sum_{i=1}^{k}|x_i - y_i|,$$

$$\text{When, } x = y \Rightarrow D_H = 0 \; ; \; x \neq y \Rightarrow D_H = 1$$

Results:

K-NN Classifier model gave the best result with the hamming distance as the similarity measure and with k = 6. Also created a model by selecting only the 13 important features and with k = 7 and was able to achieve the recall rate which was almost equivalent to the best model created with all the 30 features.

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
KNN with k = 6, Euclidean distance	99.02%	99.02%	99.02%	99.02%	95.18%	95.18%	95.18%	95.18%
KNN with k = 6, Manhattan distance	99.00%	99.01%	99.00%	99.01%	95.86%	95.87%	95.87%	95.87%
KNN with k = 6, Hamming distance	99.18%	99.02%	99.02%	99.02%	96.41%	96.41%	96.41%	96.41%
KNN with k = 7, Hamming distance and 13 important variables	97.53%	97.53%	97.53%	97.53%	95.11%	95.11%	95.11%	95.11%

Table 5: K-NN Results

2.6. Support Vector Machines

A Support Vector Machine (SVM) performs classification by finding the hyperplane that maximizes the margin between the two classes. The vectors (cases) that define the hyperplane are the support vectors. The beauty of SVM is that if the data is linearly separable, there is a unique global minimum value. The simplest way to separate two groups of data is with a straight line (1 dimension), flat plane (2 dimensions) or an N-dimensional hyperplane. However, there are situations where a nonlinear region can separate the groups more efficiently. SVM handles this by using a kernel function (nonlinear) to map the data into a different space where a hyperplane (linear) cannot be used to do the separation. It means a non-linear function is learned by a linear learning machine in a high-dimensional

feature space while the capacity of the system is controlled by a parameter that does not depend on the dimensionality of the space. This is called kernel trick which means the kernel function transform the data into a higher dimensional feature space to make it possible to perform the linear separation. When the predictor variables are overlapping, it's worth to try with SVM to check if the variables are more separable in higher dimensions.

Result:

SVM gave a recall of 96.77% on the testing data. Also created a model by selecting only the 13 important features and was able to achieve the recall which was only 1.78 % less compared to the model created with all the 30 features.

Models	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
SVM	95.15%	95.17%	95.15%	95.15%	94.27%	94.29%	94.27%	94.26%
SVM with cross validation	98.86%	98.87%	98.86%	98.86%	96.77%	96.78%	96.77%	96.77%
SVM with 13 important features and cross validation	97.09%	97.10%	97.09%	97.09%	94.99%	95.00%	95.00%	94.99%

Table 6 : Support Vector Machines Results

3. Conclusion

While classifying websites, it's important to correctly identify the phishing websites. Identifying a phishing website correctly is considered as the true positive scenario and identifying a legitimate website correctly is considered as the true negative scenario. There is no harm caused to the user by misclassifying a legitimate website (false positive), but the consequences of misclassifying a phishing website (false negative) is very disastrous. Hence our models are evaluated based on the True Positive Rate (Recall).

Best Result from each model	Training Results				Testing Results			
	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score
LR with L1 regularization	93.02%	93.03%	93.02%	93.01%	92.64%	92.66%	92.64%	92.63%
Decision Tree	98.52%	98.53%	98.53%	98.53%	95.42%	95.42%	95.42%	95.42%
Random Forest	99.03%	99.03%	99.03%	99.03%	96.89%	96.90%	96.89%	96.89%
Naive Bayes	59.88%	78.68%	59.89%	54.90%	61.32%	79.25%	61.32%	56.92%
KNN with k = 6, Hamming dist.	99.18%	99.02%	99.02%	99.02%	96.41%	96.41%	96.41%	96.41%
SVM with RBF	98.86%	98.87%	98.86%	98.86%	96.77%	96.78%	96.77%	96.77%

Table 7: Aggregated Results

SVM with RBF kernel, K-NN and Random Forest models gave very high recall rate of 96% on the testing dataset. K-NN requires all training data to make predictions and hence it can be very slow for large datasets and requires a lot of space. Random Forest and Decision Tree models were helpful in identifying the important features.

Following are the 13 important features selected from our models: SSLfinal_State, URL_of_Anchor, Prefix_Suffix, web_traffic, Links_in_tags, Request_URL, having_SubDomain, SFH, Domain_registration_length, Google_Index, age_of_domain, having_IP_Address and Page_Rank.