STOR 565 – FINAL REPORT

Team: BEAMING

1. INTRODUCTION

1.1. Motivation and Goal

Many times, you might receive an email or text containing URLs from unfamiliar contacts that could be sent for malicious purposes known as *phishing*. According to the European Union Agency for Cybersecurity (2024), phishing is a means to "persuade potential victims into divulging sensitive information such as credentials, or bank and credit card details." Utilizing a mixture of social engineering and deception, phishing attacks often occur over malicious webpages, e-mails, or instant messages that appear to be originating from a legitimate source. According to a study conducted by Intel, 97% of security experts fail to recognize phishing emails from genuine emails (Business Wire, 2015). Cybercriminals could steal their personal information, including but not limited to their SSN or banking details. Data from the Federal Bureau of Investigation's (FBI) Internet Crimes Report illustrated that in 2022, 300,497 phishing victims lost $52,089,159 just in the U.S. (FBI, 2022).

More dangerously, thanks to the development of new technologies such as deep fake, cybercriminals can use this information to continue defrauding victims' relations by, for example, spreading texts or emails containing phishing URLs or malware, putting victims' relations in emergencies to ask for urgent money transfer, etc. Therefore, it is essential to develop some detectors to identify phishing web pages.

This work can benefit people in different ways. Service providers/web browser developers can utilize these detectors to identify phishing web pages and warn users if they click on these URLs, or even block users from these phishing web pages. On the other hand, authorities, such as cyber police, can base their detection results on tracing where the cybercriminals place their servers and collaborating with local police to catch them. They can also require the hosting provider to close the phishing web pages.

Our goal is to develop binary classification models to categorize if a web page has malicious objectives, i.e. phishing, or if it is, in fact, legitimate. These models will be based on different machine learning methods, specifically we consider logistic regression, Naïve Bayes method, k-nearest neighbors, support vector machines, classification trees, TABNET, gradient boosting trees, random forest, and multilayer perceptron.

1.2. Data Description

Our research project utilizes the free and publicly available dataset of Hannouse and Yahiouche (2021), which includes 11430 URLs with 87 extracted features. The dataset also includes the labels of the URLs: legitimate or phishing and is balanced.

The features of the dataset are constructed based on three categories: 56 URL-based, 24 content-based, and 7 external-based features. URL-based features are obtained by analyzing the text of the URLs. For example: position and presence of URL-based elements, such as the use of 'https' protocol, number of dots or the length of the words etc. Content-based features are extracted by analyzing the HTML contents of the webpages of URLs. They include the number or the nature of the hyperlinks in HTML and information such as empty links or different domain names. External-based features are constructed by querying the external services and search engines, such as page rank and domain age.

2. EXPLANATORY DATA ANALYSIS

2.1. Dimensionality Reduction

From Figure 1 (Appendix), PCA does not capture enough variability in the data as we are only able to explain 15% percent of it using the first two principal components. It hints that the relationship may be complicated and non-linear. More than 52 components were required to explain the variance of at least 90%. Figure 2, which was obtained by t-SNE method, gives us hope as it shows that the legitimate and phishing URLs might be distinguished from each other. It also indicates that there might be some clusters in the data.

2.2. Correlation among Variables

In Figure 3 and Figure 4, the white lines represent not-a-number values, this is due to constant columns (leading to zero variance). These represent those variables that are completely absent in the URL. Legitimate ones had more of these prompting that presence of certain variables might immediately give away if the website is phishing or not.

Finally individual variables were visualized to identify and assess any pattern or distinguishing characteristics between phishing and genuine websites that can be referred to in the Appendix (Figure 5 to 11)

- Legitimate URLs tend to be shorter than the phishing URLs.
- Phishing pages have a higher digit proportion, which could be because these pages are created automatically and hence named by a computer.
- Phishing links seemed to have more hyperlinks in general.

- Phishing URLs are newer than legitimate URLs hence the domain age was larger in the case of legitimate URLs.
- A lower web traffic was observed for phishing pages. Also, at around 6 million, we see an atypical behavior for phishing pages, a lot of pages have the same number which could be caused due to bots that enter the page in order to increase the rank of the page.

In summary, our exploratory data analysis reveals that phishing pages generally align with the expectations we normally had about them. These pages often feature large and unconventional characters, although not universally. Additionally, they may exhibit an increased user flow and a comparatively lower number of hyperlinks compared to legitimate pages.

3. MACHINE LEARNING MODELS AND THEIR OUTCOMES

For every model that was put to test for classifying a URL, we used an 80:20 split for training and test set. The final model was chosen based on the best accuracy rate.

3.1. Linear and Quadratic Discriminant Analysis

The dataset contained a lot of non-numeric variables such as binary yes or no types as well as categorical ones that did not clearly follow a normal distribution. Therefore, these methods were eliminated at the very beginning.

3.2. Logistic Regression

It is a simple yet efficient binary classification method. The accuracy, precision, recall, F1 score, and area under the curve (AUC) of the model on the test data are provided in the table below.

Accuracy	Precision	Recall	F1 score	AUC
0.9367	0.9438	0.929	0.9363	0.9823

In our problem setting, predicting a phishing website as a legitimate website brings more risk than classifying a legitimate website as phishing. Therefore, it would be better to measure recall and precision. The logistic regression achieves 94.38% precision and 92.90% recall with an accuracy of 93.67% on the test data. Moreover, the area under the curve (AUC) is found to be 98.23%.

To understand whether all the predictors are needed and to avoid overfitting, we employ the logistic regression with L-1 penalty (LASSO) and Ridge logistic regression. Fine-tuning the parameter lambda using 5-fold cross validation technique on the training set, LASSO and Ridge models are fitted. LASSO regression reduces the dimension by setting the coefficients of eight predictors to zero. Ridge regression sets the magnitude of the coefficients of the 12

predictors as less than 10^{-3}. Table below summarizes the performance of LASSO and Ridge logistic regressions on the test data.

Method	Accuracy	Precision	Recall	F1 score	AUC
LASSO	0.9239	0.9288	0.9185	0.9236	0.9771
Ridge	0.9303	0.9353	0.9249	0.9301	0.9792

Although Ridge regression performs better than LASSO regression in all categories, logistic regression with no regularization dominates the performance of Ridge and LASSO models. Therefore, we can conclude that the regularization does not improve the performance of the logistic regression on the test data.

3.3. Naïve Bayes Method

Naïve Bayes is a fast and easy-to-implement method. Table below summarizes the performance of the method on the test data.

Accuracy	Precision	Recall	F1 score
0.7579	0.954	0.5431	0.6292

The accuracy of the model is 75.79%, which is the lowest accuracy obtained so far. Although the precision is around 95%, the recall is 54.31%, which shows that around half of the phishing websites are classified as legitimate. In our problem context, classifying phishing websites as legitimate brings more risk than classifying legitimate websites as phishing. Therefore, Naïve Bayes performs poorly and provides a risky classification.

3.4. Decision Trees

Decision trees are one of the widely used methods in machine learning as they do not require any assumptions and are easy to interpret. One of the disadvantages of decision trees is that they might result in overfitting because the tree algorithm stops when the training dataset is perfectly fitted. We test it out by first fitting a classification tree to our training data, and then prune it using cost complexity pruning. Results are provided in the table below.

Accuracy	Precision	Recall	F1 score
0.9358	0.9355	0.9366	0.9360

Based on the results above, the classification tree performs well and achieves similar results as obtained in logistic regression model without regularization.

3.5. K-nearest neighbors (KNN)

This is a popular supervised learning algorithm used for classification and regression tasks. The requirement of no prior knowledge on data distribution or linearity leads us to this method.

Initially k was set to a range of 1 to 40 and leave one out cross validation was performed on all these values. Figure 12 (in appendix) suggests that k=1 might be the best parameter for classification with 3 and 5 being the next best. All 3 values were used to produce the accuracy metrics and k=1 gave the best results. The confusion matrix reveals a balanced performance with high sensitivity and specificity. The model generalizes well to unseen data, as evidenced by comparable performance on both training and testing sets. However, further evaluation and comparison with other machine learning algorithms may provide deeper insights into its performance and suitability for different scenarios.

Metrics	Accuracy	Precision	Recall	F1 score
Validation	0.8454	0.878	0.8331	0.8467
Test	0.8437	0.8551	0.8365	0.8439

3.6. Support Vector Machines (SVM)

We first applied the SVM method with linear kernel and cost (c) parameters set (0.1, 0.5, 1). We found that c = 0.5 gives the lowest error rate, which is equal to 0.073 with 1552 support vectors. Then we applied our best model with c=0.5 on the test set. Although we see the linear boundaries for “nb_hyperlinks” and “length_url” variables in Figures 13 and 15, this separation is not clear for “links_in_tags” and “avg_words_raw” in Figures 14 and 16 (See the appendix). Additionally, we obtained the following performance measurements.

Metrics	Accuracy	Precision	Recall	F1 score
Train	92.88%	92.77%	92.96%	92.90%
Test	90.78%	90.86%	90.76%	90.80%

For c = 0.5 and 1, we also applied the SVM method with polynomial kernel and degree parameters set (2, 3, 4). We found that degree = 3 and c=1 give the lowest error rate, which is equal to 0.063 with 2803 support vectors. Then, we applied our best model with c=1, degree = 3 on the test set. In Figures 17 and 19, we can see the nonlinear boundaries for “nb_hyperlinks” and “length_url” variables although they are different forms. However, this

separation is not clear again for “links_in_tags” and “avg_words_raw” in Figures 18 and 20 (See the appendix). The following table gives the performance results of our polynomial model.

Metrics	Accuracy	Precision	Recall	F1 score
Train	95.50%	93.97%	96.93%	95.50%
Test	93.67%	91.44%	95.73%	93.70%

3.7. Multilayer perceptron (MLP)

Multilayer perceptron is a type of feedforward neural network consisting of fully connected neurons with a nonlinear activation function.

3.7.1. Model Description

The MLP models consist of one input layer, several hidden layers, and one output layer. The input layer has 87 nodes, each corresponding to one feature of the input data. The output layer has a single node for the binary classification problem. We use the ReLU activation function between the layers and the sigmoid activation function at the output layer. Batch data is normalized using batchnorm in between each layer. We also use dropout to randomly drop out a certain percentage of neurons during training (in this case, 20% with p = 0.2). This helps prevent overfitting by forcing the network to learn more robust features that are not dependent on specific neurons. Based on the number of hidden layers, we consider 4 models to examine the effect of models’ complexity on their performance: (1) Model 1: 1 hidden layer with 300 nodes; (2) Model 2: 2 hidden layers with 300 and 100 nodes, respectively; (3) Model 3: 3 hidden layers with 500, 300, and 100 nodes, respectively; (4) Model 4: 4 hidden layers with 500, 300, 100, and 50 nodes, respectively.

3.7.2. Training parameters

For the binary classification problem, we use the Binary Cross-Entropy (BCE) loss function, given by

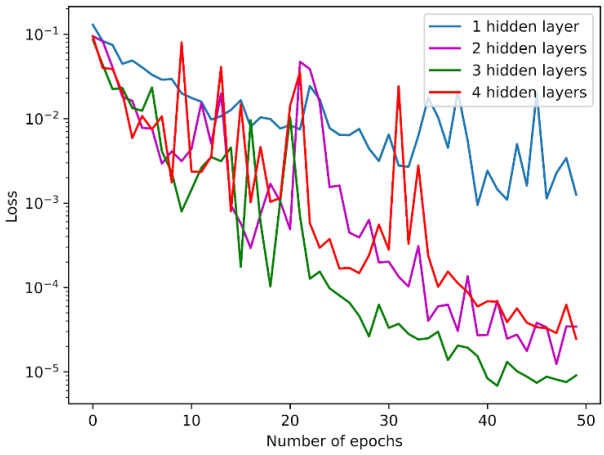
$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(p(y_i)\right) + (1-y_i)\log\left(1-p(y_i)\right)\right],$$

where N is the number of observations in the training set, y_i and $p(y_i)$ are the label and the predicted probability of that label of a training data point, respectively. Among several commonly used optimizers, we choose the Adam optimizer, which works better and is more stable than some other optimizers such as SGD or AdaGrad on this dataset, with a learning rate of 0.001. Each model is trained in 50 epochs (this number of training epochs is enough to avoid expensive computational costs and overfitting).

3.7.3. Training the MLP models

Each training epoch consists of the following steps: (1) forward propagation, (2) loss computation, (3) backpropagation, and (4) updating the parameters. The loss behaviors corresponding to each model are reported in the following table and figure.

Model	Last-iteration loss value
Model 1	0.0012583367060869932
Model 2	$3.4345404856139794 \times 10^{-5}$
Model 3	$9.056506314664148 \times 10^{-6}$
Model 4	$2.4474804376950487 \times 10^{-5}$



Based on the above table and figure, we observe that when the number of hidden layers increases from 1 to 3, the MLP models achieve better loss behaviors (in the long run). However, the 4-hidden layer model does not provide a loss behavior as good as that of the 3-hidden layer models.

3.7.4. Results on the test dataset

Using the trained models, we do the prediction on the test dataset. The results of performance metrics are reported in the following table.

Model	Accuracy	Precision	Recall	AUC
Model 1	95.01%	93.43%	96.85%	95.01%
Model 2	94.90%	94.52%	95.34%	94.90%
Model 3	95.22%	94.55%	95.98%	95.22%
Model 4	95.42%	94.93%	95.98%	95.42%

We see that the model with 4 hidden layers provides the highest accuracy of 95.42%. However, in our context, mislabeling a phishing website as legitimate is more harmful, so recall is also an important measure. Therefore, model 1 is also a good model despite its simple structure, with a recall of 96.85%.

3.8. TABNET

It is well known that tree methods have outperformed most ML algorithms when dealing with tabular data for its flexibility and interpretability, for this section we explore a new alternative to it. We found the following paper **TabNet: Attentive Interpretable Tabular Learning** which is presented as an at least as good interpretable alternative to tree methods for tabular data. TabNet is a Deep Learning architecture proposed in 2020 by Google researchers. It is specially built to deal with tabular data, it performs feature processing inside the architecture selecting the most relevant features using attention layers. It was especially appealing to us for the phishing detection problem as we have around 90 features which TabNet takes care of by itself and also tells which of the features were more important in the classification in a completely analogous way as the results from a random forest.

We used an implementation of TabNet in pythorch. It is a high-level implementation, so we do not have to put our hands into the attention layers, we just specify some parameters such as the width of the decision prediction layer and sparsity in the features. To choose the best parameters we used random search as the training time was long to use cross-validation. Due to the complexity of the problem and running time we only ran 100 searches.

We find the accuracy on the test data as 95.21% and AUC as 98.9%. Table below shows the other performance measures.

Class	Precision	Recall	F1 score
Phishing	96%	94%	95%
Legitimate	94%	96%	95%

One of the good findings about this model is that the precision for the phishing class is slightly higher, which is desirable as false negatives are more costly. Lastly, we investigate the importance of each feature on the model and obtain the 10 most important features (Figure 21 in Appendix). Page rank and google index stay as one of the most important features. We see other variables that did not appear before such as the “number of www”.

3.9. Gradient Boosting Trees

Gradient boosting tree is one of the most promising methods, and it is mostly used in practice to deal with tabular data. We perform a random search to look for the best hyperparameters. As the models are easier to run, we run 1000 searches and find the accuracy of the test data as 96.21%. The AUC is obtained as 99.3%. The rest of the performance measures are provided below.

Class	Precision	Recall	F1 score
Phishing	97%	96%	96%
Legitimate	96%	97%	96%

For the feature importance (Figure 22 in Appendix), we notice that the gradient boosting heavily relies on google index to make predictions as google index has 50% of the importance in the prediction, which is very high compared to the 10% we found in TABNET.

3.10. Random Forest

We also explore random forest, but due to the computational complexity we optimize the hyperparameters in a limited parameter space. Nevertheless, the model performs well on the test data with an accuracy of 96.29% and AUC of 99.2%.

Class	Precision	Recall	F1 score
Phishing	97%	96%	96%
Legitimate	96%	97%	96%

The model gives different results than gradient boosting method in terms of the importance of features. The importance of features is more balanced compared with the gradient boosting (Figure 23 in Appendix). This importance looks more like the one presented from TABNET, google index and page rank being the most important, we also see the number of www playing an important. Overall, this presents a more robust classification as there is not just one leading the classification.

4. CONCLUSION AND LIMITATIONS

Based on the results obtained from all the models, Random Forest had the best overall performance in terms of accuracy in classifying a website. Random Forests are generally known for its flexibility and high power for modelling tabular data which explains its suitability to our dataset.

Although most of the models had high accuracy rates, it cannot be completely justified because of the un-natural type of data. In reality, one might not get a balanced dataset such as the one we used since there exists much more legitimate URLs than phishing ones.

Another limitation that needs to be highlighted is that to put this model to use in a real-world scenario, we would require certain variables which are not easy to find. Most of the algorithms heavily depend on variables from external sources (Google index and page rank).

Nevertheless, this project seeks to spread awareness of malicious acts of online phishing and how machine learning can help produce powerful tools to detect and eliminate fraudulent events.

REFERENCES

Business Wire. 2015. "97% Of People Globally Unable to Correctly Identify Phishing Emails." May 12, 2015. https://www.businesswire.com/news/home/20150512005245/en/97-Of-People-Globally-Unable-to-Correctly-Identify-Phishing-Emails.

ENISA. n.d. "Phishing/Spear Phishing." Page. ENISA. Accessed March 6, 2024. https://www.enisa.europa.eu/topics/incident-response/glossary/phishing-spear-phishing.

Federal Bureau of Investigation. 2022. "Internet Crime Report 2022." https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf.

Hannousse, Abdelhakim, and Salima Yahiouche. 2021. "Web page phishing detection", Mendeley Data, V3, doi: 10.17632/c2gw7fy2j4.3

Hannousse, Abdelhakim, and Salima Yahiouche. 2021a. "Towards Benchmark Datasets for Machine Learning Based Website Phishing Detection: An Experimental Study." Engineering Applications of Artificial Intelligence 104 (September). https://doi.org/10.1016/j.engappai.2021.104347

APPENDIX

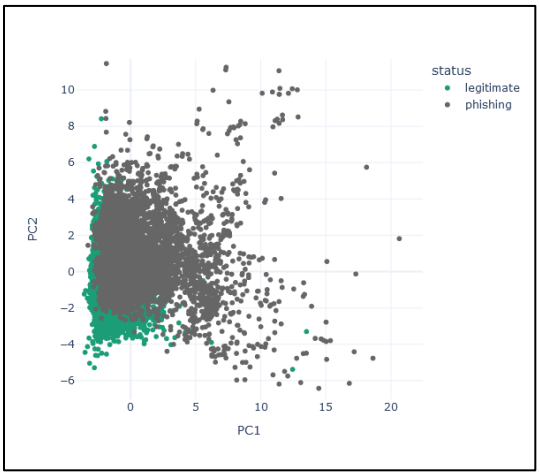


Figure 1: First two components of PCA

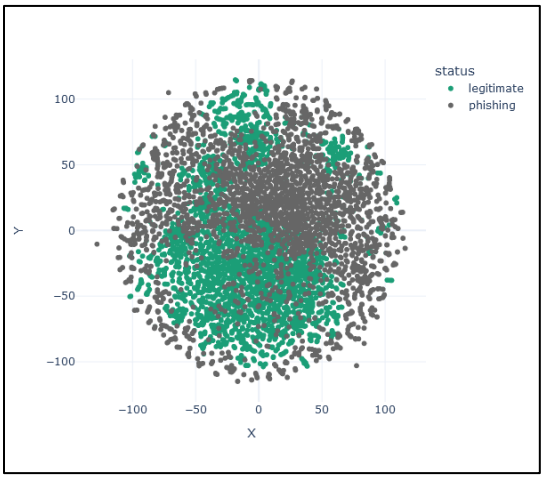


Figure 2: t-SNE with perplexity = 3

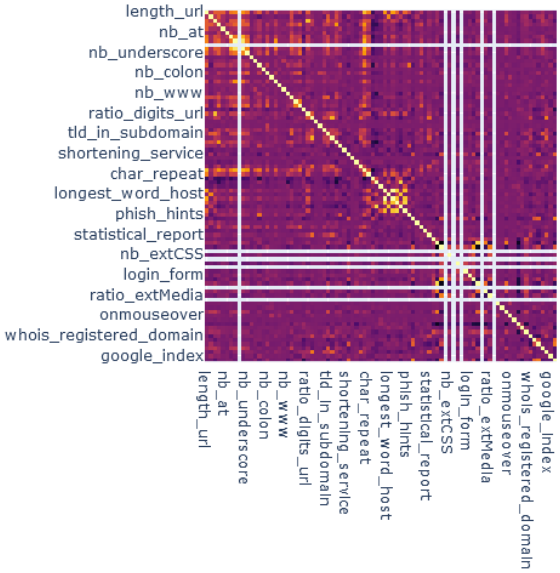


Figure 3: Correlation heatmap for legitimate URLs

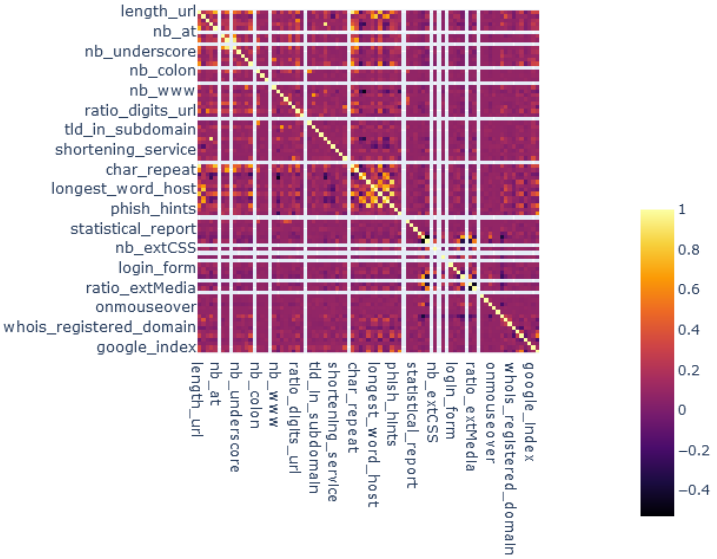


Figure 4: Correlation heatmap for phishing URLs

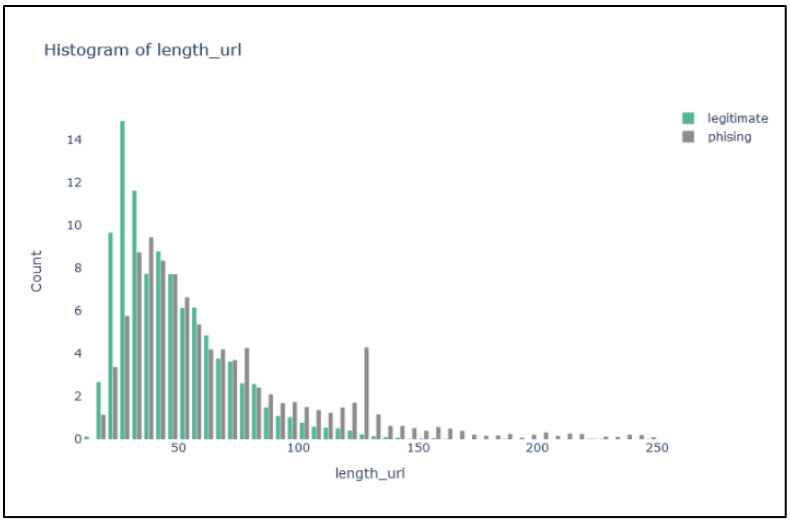


Figure 5: Histogram of URL lengths

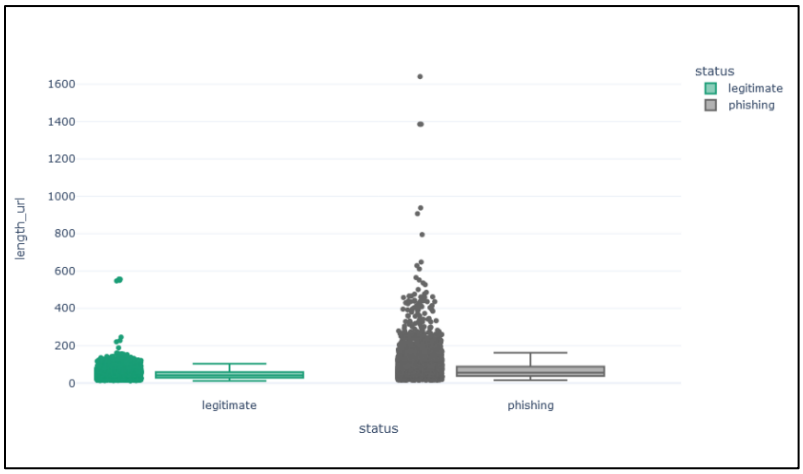


Figure 6: Boxplot of URL lengths

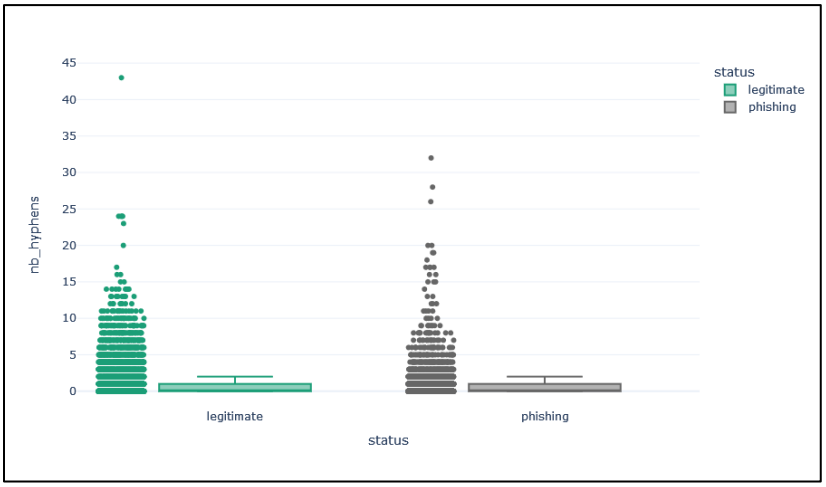


Figure 7: Box plot of the length of the number of hyphens in URLs

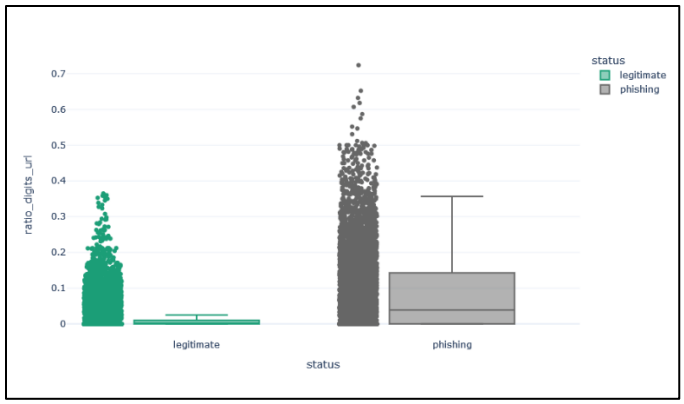


Figure 8: Box plot of the proportion of the digits in URLs

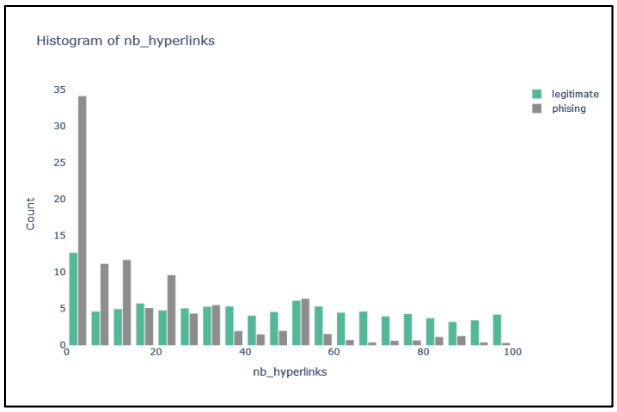


Figure 9: Histogram of the number of hyperlinks

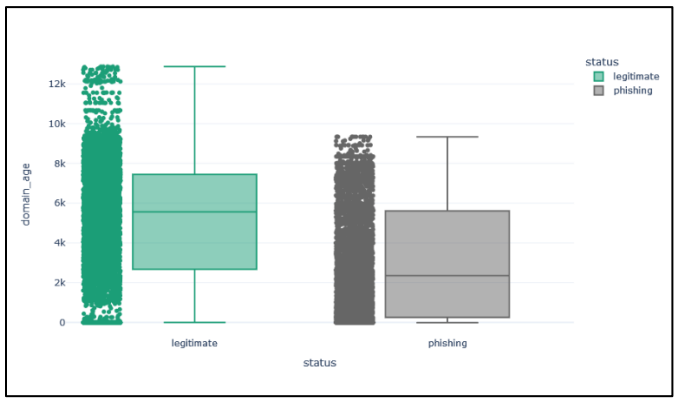


Figure 10: Box plot of the domain age

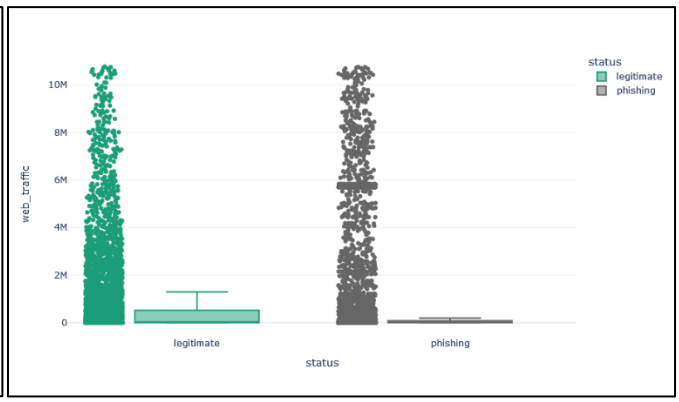


Figure 11: Box plot of the web traffic

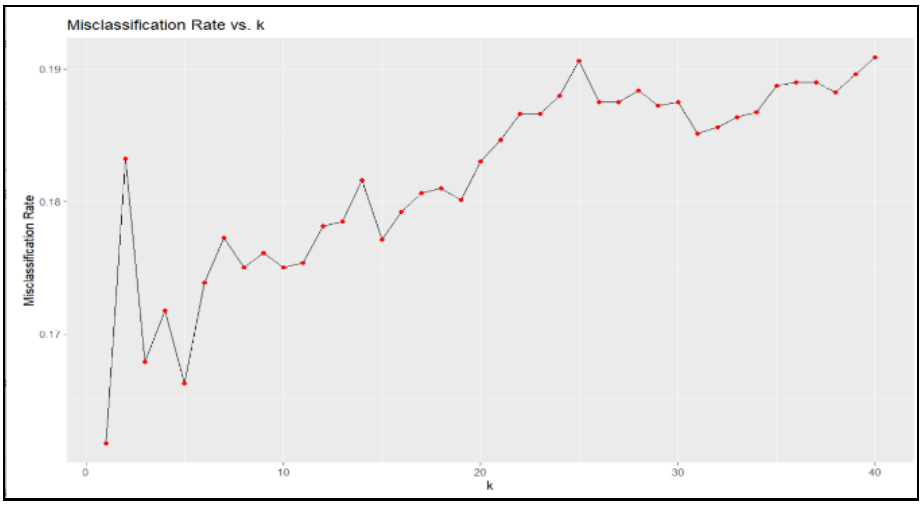


Figure 12: Parameter tuning to select optimal k using misclassification rate.

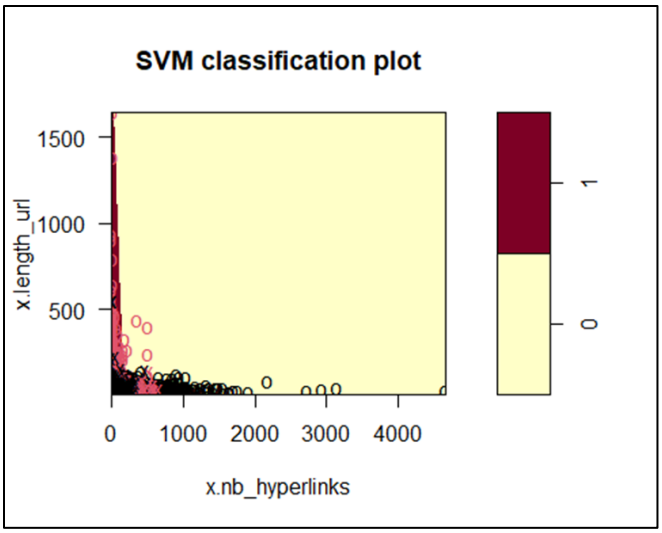


Figure 13: Linear Kernel Plot on Train Set

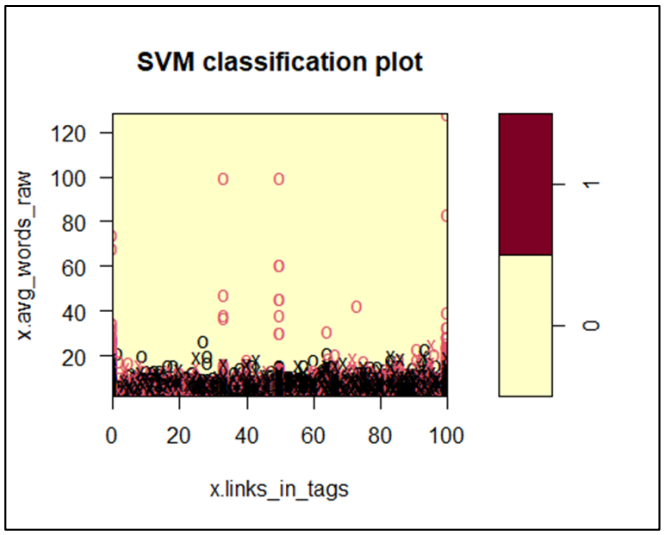


Figure 14: Linear Kernel Plot on Train Set

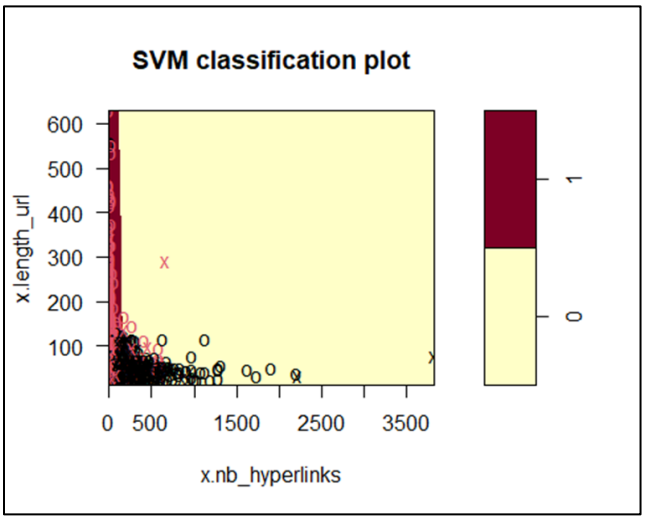


Figure 15: Linear Kernel Plot on Test Set

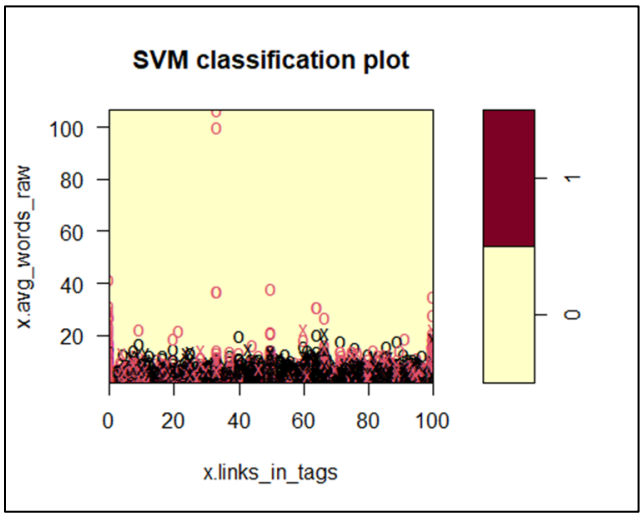


Figure 16: Linear Kernel Plot on Test Set

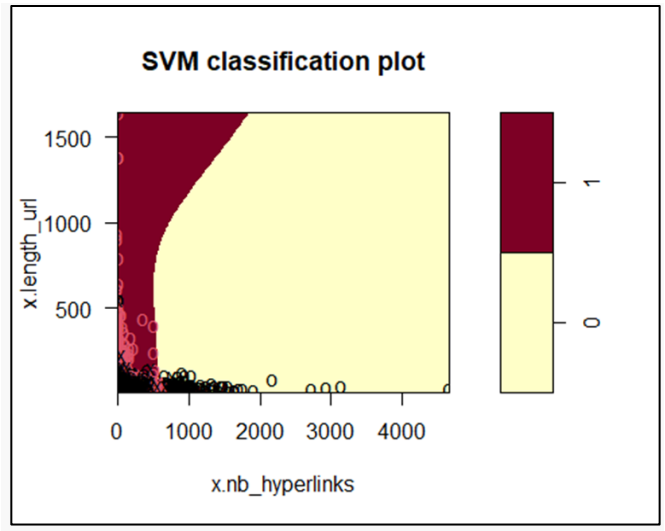


Figure 17: Polynomial Kernel Plot on Train Set

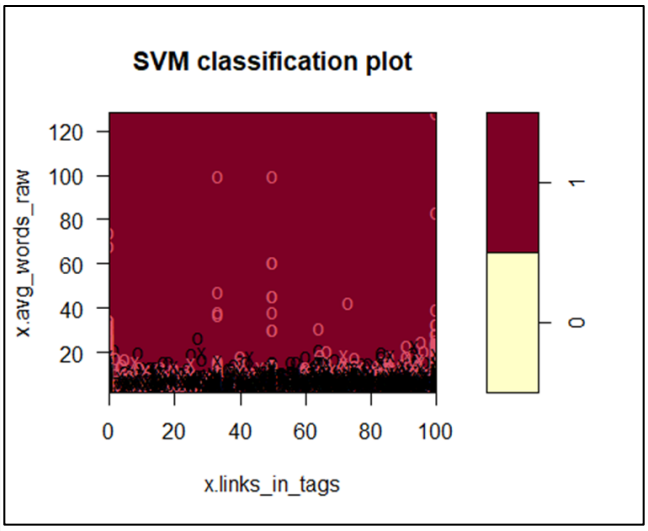


Figure 18: Polynomial Kernel Plot on Train Set

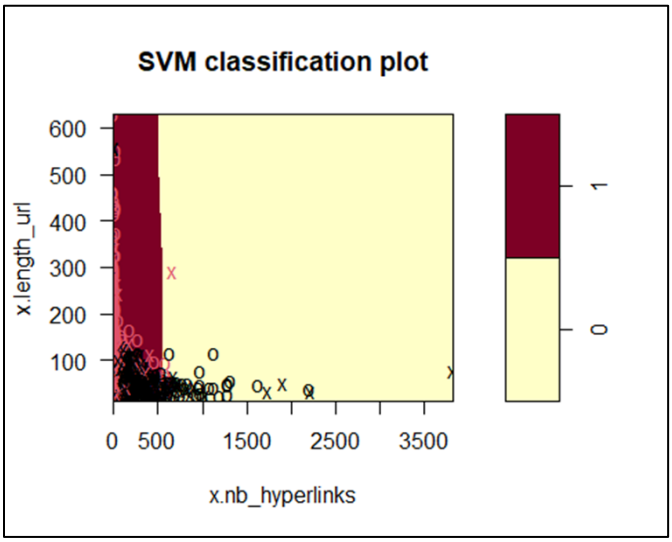


Figure 19: Polynomial Kernel Plot on Test Set

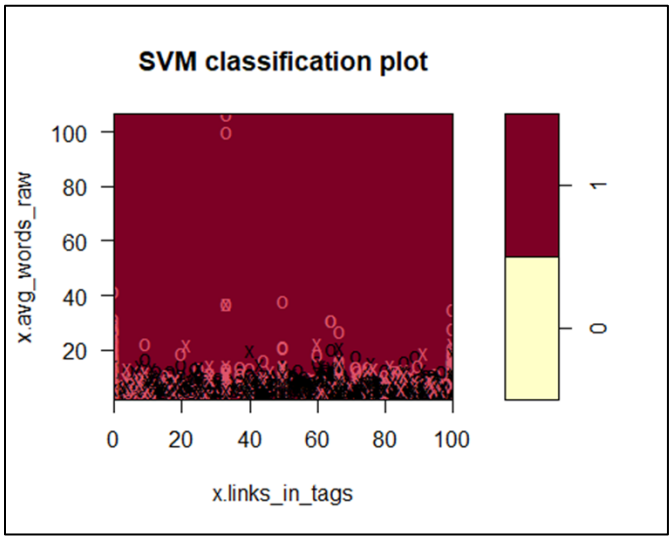


Figure 20: Polynomial Kernel Plot on Test Set

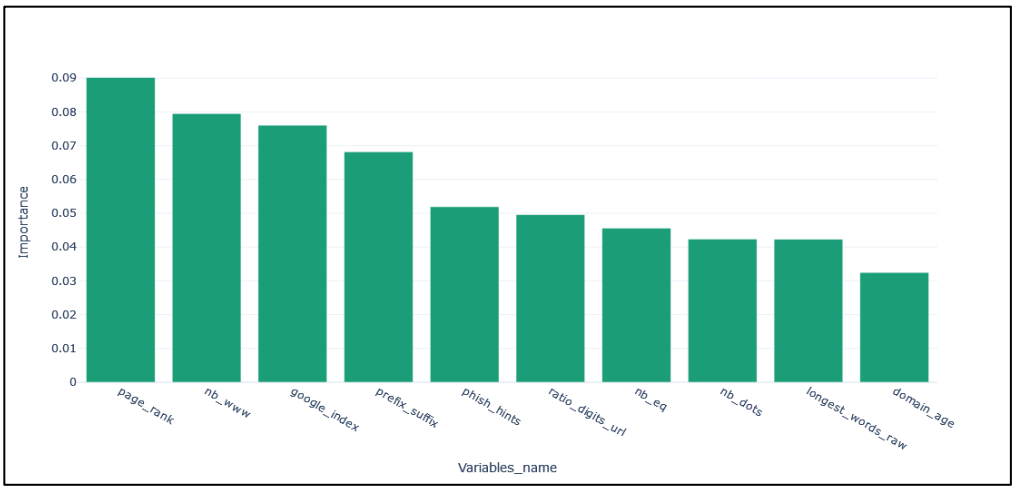


Figure 21: Feature importance plot of TABNET

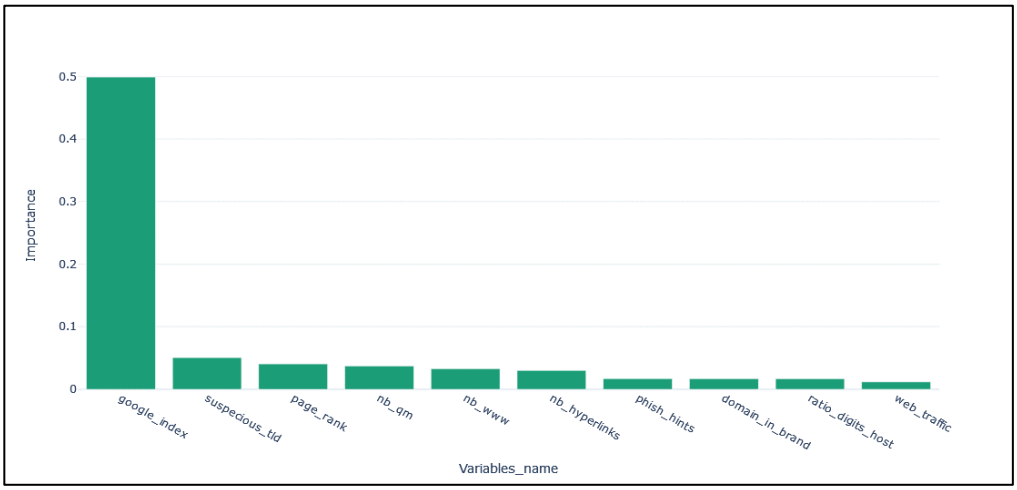


Figure 22: Feature importance plot of XGBoost

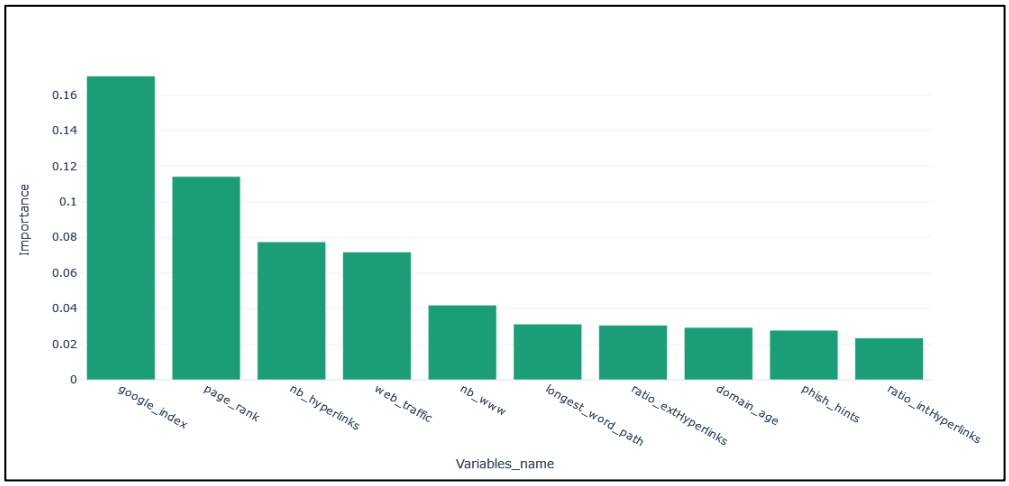


Figure 23: Feature importance plot of random forest