Detecting Phishing URLs using Machine Learning and Feature Engineering

Marin Mes (149899) & Thor Møldrup (127318) & Ekaterina Rossi (150286)

Copenhagen Business School
Department of Digitalisation
MSc Business Administration and Data Science
Cybersecurity Foundations and Analytics (CDSCV1007E)
Number of pages: 8
Number of characters: 29 479
20-11-2022

Abstract

One of the easiest ways for cybercriminals to get access to someone's data is through phishing. Phishing often comes in the form of an email leading the user to a phishing website. This project explores four different Machine Learning algorithms to classify phishing URLs. Furthermore, it looks at three different methods of feature engineering. Results suggest that the two feature selection methods, Recursive Feature Elimination and The Regression Weight Approach (SFM), performed better than the dimensionality reduction technique, Principal Component Analysis. The two best performing models were the Random Forest classifier with SFM, F1: 93%, and the XGBoost model with Recursive Feature Elimination, F1: 96%. Furthermore, it was found that for this dataset the directory had a big influence on determining the classification of an URL. These findings can help contribute to keeping companies and consumers safer from future phishing attacks.

Keywords: Cybersecurity, Machine learning, phishing, URL, classification, feature selection, PCA, Random Forest, SVM, Naive Bayes, XGBoost

1. Introduction

Cybersecurity is a rapidly growing industry with an expected compounded annual growth rate of 9.51% during the period from 2021 to 2030 (GlobeNewsWire, 2022). With the growth of the industry, the amount of adversaries has also increased, making it more crucial for defenders to possess the right tools so they can protect companies and valuable information. A common way for adversaries to try and penetrate a network is through phishing, where they attempt to steal data or credentials, through fraudulent websites, emails, text messages and more. The targets of these attacks can be anyone: from companies to private entities, from famous to non-famous entities, and from technically savvy to non-technically skilled people. Therefore, the importance of detecting these attacks is high, and is the main motivation behind this paper.

The goal of this paper is to find out which parts of the URL are most decisive in determining whether an URL is phishing or not. In order to achieve this goal, the following research questions should be answered:

1. Which Machine Learning models are best at classifying phishing URLs?
2. Which feature engineering methods will give the best results when classifying phishing URLs?
3. Which features are most important in detecting phishing URLs and to what part of the URL do they belong?

2. Related Work

Since phishing is a term of the 21st century, research on detecting phishing websites is relatively new.

One of the first papers applying machine learning methods to a phishing detection problem is that of Zhang, Hong & Cranor (2007). They introduce CANTINA, a content-based framework to detect phishing sites which is accurate 90% of the time. CANTINA analyses the content of a webpage and utilizes the TF-IDF information retrieval algorithm to extract the five most important terms on this webpage. Then it feeds these terms into a search engine and compares the domain name of the N top search results with the domain name of the webpage under examination. If they match, it is considered a legitimate website, if not, it is considered a phishing site.

In 2013 it was Nguyen et al. that introduced a completely new approach to detect phishing websites. Instead of focusing on the content of a webpage, they utilized the different features of a website's URL to determine whether a website should be classified as

phishing or not. They split URLs into different parts after which for each URL they calculated values for six different heuristics which they then compared to a certain threshold value. Based on this comparison a decision was made whether a URL should be classified as phishing or not. This URL-based approach reached higher accuracy than the content-based approach, namely 97%.

In 2015 the next steps in detecting phishing websites were made. Whereas earlier papers used simple mathematics or comparison methods to detect phishing websites, Amiri et al., used Machine Learning methods. They tested multiple algorithms on a small dataset of phishing and non-phishing URLs. They found that the K-Nearest Neighbor algorithm performs best, reaching an accuracy score of 99.3%.

Research that followed approached the classification problem in similar ways, the only difference being that the size of the datasets used in the research increased. This is a logical development since over the years, more and more phishing websites are being created and deployed. This increase in data resulted in lower accuracy and a different best performing model.

Niakanlahiji, Chu & Shaer (2017) introduce PhishMon, a machine learning framework to detect phishing URLs. They find that the Random Forest Classifier can reach an accuracy of 95.4% when using 15 features from a URL. Desai et al. (2017), also find that the Random Forest Classifier performs best, with an accuracy of 96%, and Tyagi et al. (2017) even reach an accuracy of 98.4% with a Random Forest Classifier.

With this continuous increase in phishing websites, new methods to create and deploy phishing websites arose. This in turn lead to the development of more complex URLs and to an increasing number of features determining whether a website is phishing or not. Consequently, in 2018 researchers started to include feature engineering methods when developing phishing detection models. Karabatack & Mustafa (2018) researched numerous different feature selection algorithms in combination with a Naïve Bayes Classifier. Their results suggest that the Plus-1 Take Away feature selection algorithm resulted in the highest accuracy (93.4%) with a reduced number of features.

Rao and Pais (2018) did feature engineering by conducting a Principal Component Analysis to reduce dimensionality. After dimensionality reduction, they produced a model with an accuracy of 99.3%. Their results suggest that the length of the hostname, search engine title and the presence of anchor links in the HTML body were some of the most important features in detecting a phishing website by studying its URL. Kasim (2021) finds that a combination of a Sparse Autoencoder and Principal Component Analysis methods contributes to successful feature engineering, reaching an accuracy of 99.6% with a Light Gradient Boosted Machine. Almonani et al. (2022) uses the Information Gain Attribute Evaluation to select the best features in the dataset and reaches an accuracy of 99.1% with the Random Forest algorithm.

This research will build upon previously mentioned literature in multiple ways. First of all, the models deployed in this paper are trained on a significantly larger dataset. In previous literature datasets of approximately 23000 URLs of which 5000 phishing were used, whilst this paper uses a dataset of 88,647 total URLs with 30,647 phishing URLs. Secondly, besides accurate classification of phishing websites and successful feature selection, this paper also aims to give insights into which part of the URL is most important in determining whether a URL is classified as phishing or not.

3. Theoretical Background

3.1. Phishing

Phishing can be defined as a type of cyberattack in which an attacker sends an email or an instant message with a fraudulent link on behalf of a trustworthy source to obtain confidential information of a person (Jacobson & Idziorek, 2012). Two types of phishing attacks can be distinguished (Alauthman et al., 2019). In the first case, after opening a phishing URL a fake website will be opened and the user will unknowingly give access to personal data (Ibid.). In the second scenario, after opening the link, malware, which can get access to personal data, will be downloaded to the user's computer (Ibid.). In both situations social engineering is used (Jacobson & Idziorek, 2012). This can be defined as an act that affects a person to make an action against his own interests (Social Engineering Defined, 2022). It is told that the attacker tries to evoke strong emotions, such as anger and fear, in order to influence the decision-making at the moment (Ibid.).

3.2. URL structure

One of the most common things that an internet user will encounter when browsing the web is a Uniform Resource Locator (URL). A URL allows a user to locate a given web page by typing it in the address bar in a browser, making them essential when navigating the internet. Below, the most common parts of a URL will be introduced and visualised in figure 1.

3.2.1. Protocol

The protocol part of the URL describes which protocol the browser should use to access and transfer data to and from the given web page. Two common protocols are the hypertext transfer protocol (HTTP), and hypertext transfer protocol secure (HTTPS). The main

difference between these is that HTTPS uses encryption and verification when transmitting data over the network, making it superior to HTTP with regards to security (Cloudfare, 2022).

3.2.2. Domain

The domain name is the unique reference that identifies a web page. Usually this is a string followed by an extension that defines which country the URL is from. For example, at Copenhagen Business School (CBS) the string "cbs" followed by the extension ".dk" makes up the domain name for CBS and specifies that it is a Danish website.

3.2.3. Directory/Path

Inside a website, there are often sub directories such as "About us", "Contact", and "Products". The sub directories can be specified in the URL, which will let the user navigate there. These are commonly written after the extension and includes a slash followed by the page. This pattern will continue for any sub folders within these directories. Using the same example as before, the graduate degrees on the CBS website can be accessed by writing "/en/study/graduate-programmes" after the domain name. The "en" at first, specifies the language that the website is written in. This gives us a full URL of "https://cbs.dk/en/study/graduate-programmes". Often the prefix WWW, World Wide Web, is used in front of the domain, however it can usually be excluded when navigating to a webpage (Scarpati, 2021).

3.2.4. File

Sometimes, a URL might refer to a specific file that is located on a website. The file can take any format as long as it is allowed to exist on the website. This is indicated in the URL by a filename followed by an extension. A few examples of these are ".csv", ".pdf", and ".xlsx". Usually, the file part of the URL follows the directory which specifies the location of the file.

3.2.5. Parameters

On some websites, the user might notice a question mark followed by one or more values. This is called a parameter and is used in websites when the creator wants to have the capability to perform actions such as tracking, reordering, and filtering. To have multiple parameters an ampersand (&) is used to connect them in the URL.

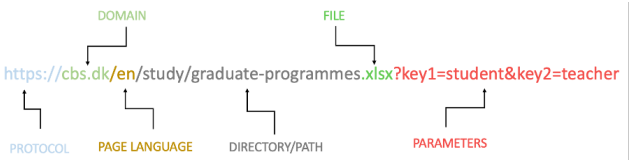


Figure 1: Parts of a URL

3.3. Machine learning in phishing detection

As mentioned in Section 2, in recent years researchers have started to utilize Machine Learning models in order to detect phishing URLs. The benefit of using Machine Learning models is that there is no human intervention needed, it is efficient and it is fast. Moreover, Machine Learning models can improve automatically through experience and by the use of data (Dwivedi, 2022). Therefore, using Machine Learning to detect phishing websites is attractive and quick.

4. Methodology

Figure 2 illustrates the high level methodology used in the research project, namely (1) data collection and Exploratory Data Analysis (EDA), (2) data pre-processing, (3) feature engineering, (4) modeling and evaluation.

4.1. Data collection and EDA

The dataset was collected and prepared by Grega Vrbančiča, Iztok Fister Jr., and Vili Podgoreleca from the University of Maribor in Slovenia. It was created with the purpose of providing a dataset that can help strengthen the knowledge on how a phishing website can be detected through its URL. They collected the data through different sources. First, 30,647 confirmed phishing URLs were collected from the Phishtank website. Further, 58,000 legitimate URLs were obtained from the Alexa ranking website. Figure 3 depicts a visual representation of the classes.

The features of the dataset were extracted from the URLs along with additional features from external sources. These include the number of occurrences of different special characters, both from the entire URL and the following sub sections of the URL: the domain, the file, the directory, and the parameters. Additional features were created based on whether the URL is shortened, and if it is indexed on google. In total, the dataset consists of 111 features with one target variable: 1 = phishing URL, 0 = normal URL. The rest of the dataset contains both numeric and Boolean values. A table of the number of features relating to each part of the URL along a table with a description of each feature can be found in Appendix A.

The EDA process showed us which characters in the URLs occur mostly. In our dataset it is the '%' sign

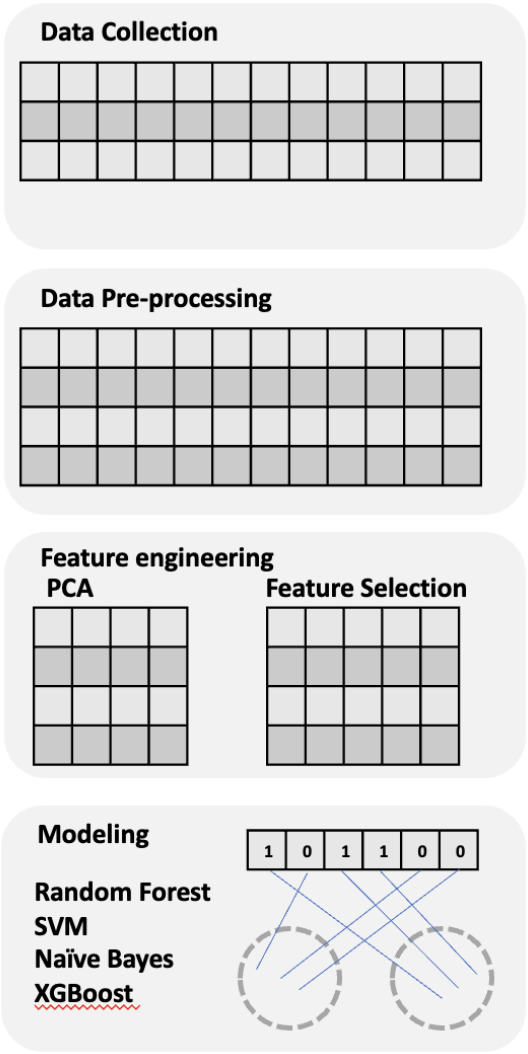


Figure 2: Methodology

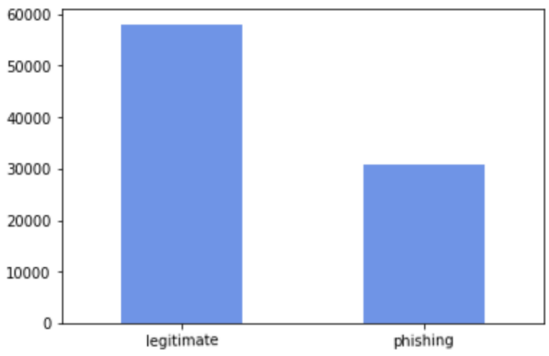


Figure 3: Representation of classes

that is appearing most frequently, especially in the file, directory, and parameter part of the URL. After that, the count of vowels in the domain of an URL, and the count of asterisks in the file and directory part of the URL are the highest. Figure 4 and 5 give an overview of which are the most occurring characters in the URL across the different classes.

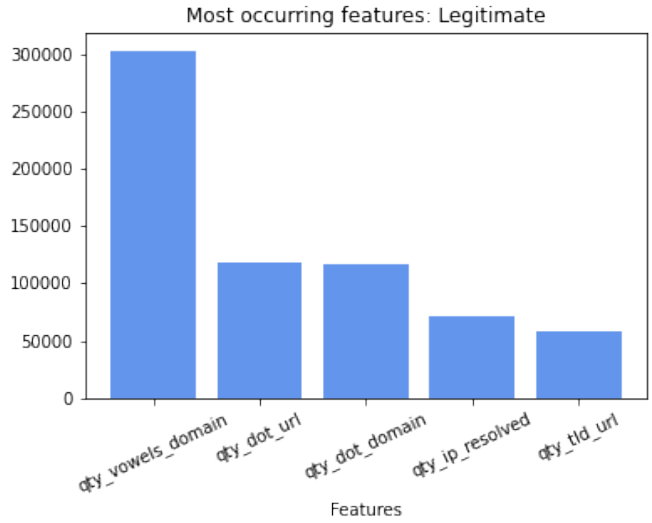


Figure 4: Most occurring characters in legitimate URLs

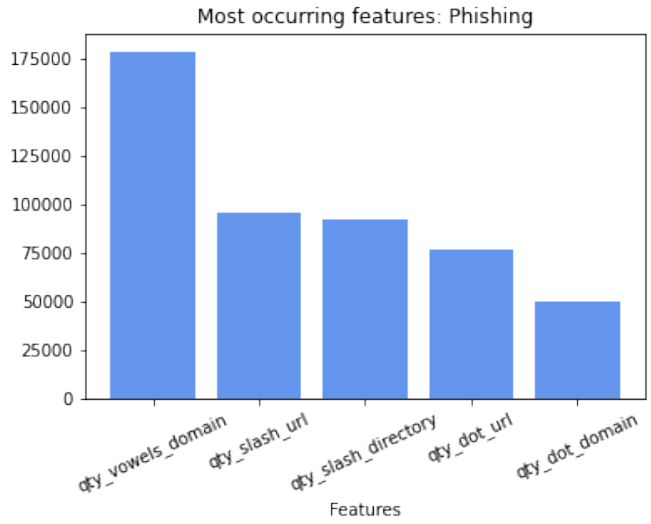


Figure 5: Most occurring characters in phishing URLs

4.2. Data pre-processing

Based on the preliminary analysis of the dataset, it appeared to be readily pre-processed. The dataset does not contain missing values or any other abnormalities. However, it was discovered that there are many features having numerous rows containing a value of –1.

It was concluded that a value of –1 indicates that this part relating to the URL simply does not exist in the total URL. For instance, if there is no directory part in a URL, then the value of all features linked to the directory part of the URL will be –1. This was validated by checking that for the features that relate to the full URL, there are no values of –1.

As can be seen from Figure 3 there is a class imbalance in the dataset. A binary imbalanced classification problem is encountered when one class is overly represented in the dataset compared to another one, majority and minority classes respectively (Johnson & Khoshgoftaar, 2019). It is argued that standard classifiers are biased towards the majority class, which in turn implies sacrificing minority class accuracy (Elreedy & Atiya, 2019). In this paper, the minority class is 35% of all of the dataset. Considering that we are interested in predicting the minority class, the imbalance of the dataset should be handled.

The most used approach is the data level approach (Ibid.), where a dataset is modified by removing instances from the majority class, under-sampling, or adding more instances for the minority class, over-sampling (Ibid.). To maintain the size of the dataset, the decision was made to choose oversampling. ADASYN, adaptive synthetic sampling, an extension of the popular Synthetic Minority Oversampling Technique (SMOTE) oversampling technique, was applied to the dataset. It creates synthetic examples in regions of the feature space where the density of minority examples is low, and fewer synthetic examples where the density is already high (Brownlee, 2020).

The dataset was split into a training and testing dataset, where 70% made up the training set, and 30% made up the testing set. ADASYN was applied on the training set after the split. This balanced out the training set where the share of each class became about 50%, 41,574 - positive class and 40,551 - negative class. To validate whether the oversampling is of added value, a simple RandomForest model was applied to the original data set and the synthetically modified dataset. The classification report of both was compared, and it was found that the simple Random Forest model performed slightly better on the balanced dataset. Hence, we decided to use this dataset for further analyses.

Lastly, the data was scaled using a MinMaxScaler to normalize the data and to make sure the models are not affected by extreme values.

4.3. Feature engineering

Considering the number of features that could be extracted from the URL (111), it is important to determine the most informative features and exclude the rest without loosing any relevant information. For this purpose two feature engineering methods (FEM) were utilised: dimensionality reduction and feature selection. The choice was made to use both techniques and evaluate which method works best when selecting features of a URL.

4.3.1. Dimensionality Reduction

The algorithm used for dimensionality reduction is Principal Component Analysis (PCA). In PCA all data is projected into smaller dimension planes to preserve as much variance from the original dataset as possible (Géron, 2019). PCA identifies n orthogonal components, where n is the number of features in the dataset (Ibid.). We retain 95% of the variance for further analysis.

4.3.2. Feature Selection

In contrast to PCA, feature selection techniques do not combine features to create more useful ones, but they rather select and keep the most useful features from the original dataset (Ibid.). Two approaches were applied for feature selection. The Regression Weight Approach (we will refer to it as 'SFM' since the scikit-learn library uses this terminology) fits data into a logistic model and selects those features which weights are greater than the threshold value. The second approach is recursive feature elimination (RFE), where on each step a logistic regression is build and parameters with lowest weights are eliminated. In this paper a step size of 10 is used.

4.4. Modeling

The dataset used in this paper has been labeled, making the identification of phishing websites a supervised machine learning problem. To classify the URLs the following algorithms are used: Random Forest Classifier (RFC), Support Vector Machine Classifier (SVC), Gradient Boosting (XGB), and Naive Bayes (NB). The choice for these models is based on previous literature where these models have shown promising results. All models will be run in combination with the three previously described Feature Engineering methods, resulting in a total of 12 models to run.

Random Forest is an ensemble of decision trees, where the prediction is obtained as a result of all votes from decision trees (Ibid.). It is a bagging method where all classifiers in the random forest are independent of each other. Besides that, there exist ensemble boosting methods that allow sequential training, where each classifier is trying to correct the previous classifier. An example of such a method is XGBoost, an algorithm consisting of gradient boosted trees where each new tree in the ensemble is fit to the residual errors made by the previous predictor (Ibid.). The Support Vector Machine classifier separates data points in the hyperplane to have as large distance from all data points as possible (Ibid.). This line is a decision boundary for classification for each data point. The Naive Bayes Classifier

is based on Bayes' Theorem and it assumed that each feature is independent from any other feature and contributes equally to the outcome.

4.5. Evaluation Metrics

One of the most used performance metrics for classification problems is accuracy (Johnson & Khoshgoftaar, 2019). It measures the share of the correct predictions over all predictions. Nevertheless, this metric can be misleading with imbalanced datasets (Ibid.). Thus, more informative metrics are precision and recall (Biecek & Burzykowski, 2022). Precision is a metric indicating the share of positive instances among all retrieved, while recall is the share of positive instances that are correctly detected (Geron, 2019). A final evaluation metric is the F-score, which is the harmonic mean of precision and recall (Biecek & Burzykowski, 2022). The F1 score can be used as a balanced accuracy, since it is high only if both precision and recall are high (Ibid.). Therefore, it was decided to use F1-score to evaluate the performance of models since both false negative and false positive are highly undesirable in this case.

5. Results

5.1. General

The test results in a Table 1 show the mean cross validated score of the best estimator for each model. It is seen that with each Feature Engineering Method, the RFC and the XGB perform best. Hence, it was decided to focus on these two classifiers since that will bring the number of models to evaluate down from 12 to 6.

Model	Mean Score	Feature Engineering Method
Random Forest	0.94	RFE
Random Forest	0.93	PCA
Random Forest	0.95	SFM
Support Vector Classifier	0.84	RFE
Support Vector Classifier	0.84	PCA
Support Vector Classifier	0.80	SFM
XGBoost	0.96	RFE
XGBoost	0.93	PCA
XGBoost	0.96	SFM
Naive Bayes	0.90	RFE
Naive Bayes	0.91	PCA
Naive Bayes	0.89	SFM

Table 1: Initial Model Selection

Table 2 shows a more in-depth overview of the evaluation metrics of the 6 selected models. From this table it can be concluded that the models in which feature engineering had been done through PCA performed worse for both the RFC and the XGB classifier. Hence, the decision was made to also filter these two models out before investigating which features are most determinant in classifying a URL as phishing or not.

Model	Accuracy	F1	Feature Engineering Method
Random Forest	0.92	0.92	RFE
Random Forest	0.92	0.92	PCA
Random Forest	0.93	0.93	SFM
XGBoost	0.96	0.96	RFE
XGBoost	0.92	0.92	PCA
XGBoost	0.95	0.95	SFM

Table 2: Performance metrics of models

5.2. Feature Importance

To find out the most important features per model, horizontal bar charts were created for all four models in which the different features against their relative importance are plotted. Figure 6 - Figure 9 show the results. A threshold at 2% for the relative importance was installed to find out the most determinant features.

For the RFC with the RFE Feature Engineering Method (Figure 6), it can be stated that there are many features with a relatively high importance when determining whether a URL is classified as phishing or not. Overall, the graph suggests that it is mostly the directory and the file part of the URL which plays an important role in determining the nature of an URL.

For the RFC with the SFM Feature Engineering Method (Figure 7), there are considerably less features with a high relative importance. For this model it seems to be the content of the the directory having the most power in determining the nature of a URL. It is worth mentioning that the quantity of dots '.', have a relative importance of more than 20%.

For the XGB with the RFE Feature Engineering Method (Figure 8), there are again fewer features reaching the threshold of 2%. In this model it is very clear that it is the contents of the directory mostly determining the nature of the URL. Note that the quantity of asterisks '*' has a relative importance of almost 40%.

For the XBG with the SFM Feature Engineering Method (Figure 9) there are only four features with a high relative importance: the quantity of dots '.' in the directory, the directory length, the quantity of dots '.' in the domain and the time domain activation. The quantity of dots in the directory stands out with a relative importance of more than 70%.

5.3. Error Analysis

The confusion matrices of the four selected models can be found in Appendix B. In the case of this paper, it is important that the percentage of false negatives is low, meaning the models miss as few phishing URLs as possible. For the RFC model, the false positive rates are 10% (RFE) and 8%(SFM). For the XGB model the false positive rates are 4% (RFE) and 6% (SFM). Hence, the XGB RFE model is the best performing model. Due to the lack of raw data (i.e., full URLs), a more detailed error analysis of the misclassified URLs cannot be executed. This is a limitation of this research and at the same time an opportunity for future research to build upon the findings in this paper.

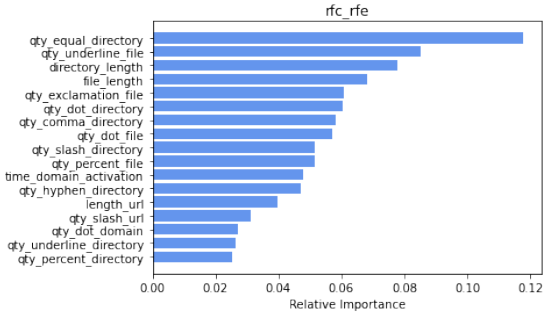


Figure 6: Relative importance of features: RFC/RFE

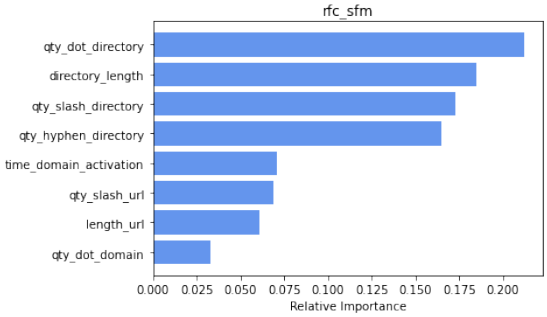


Figure 7: Relative importance of features: RFC/SFM

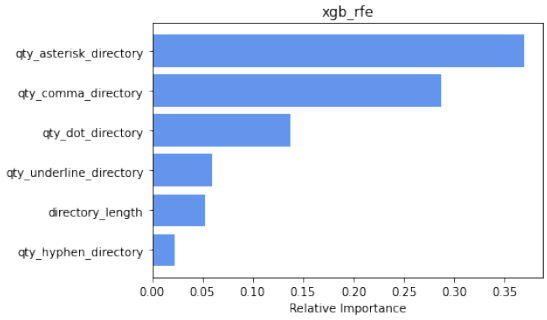


Figure 8: Relative importance of features: XGB/RFE

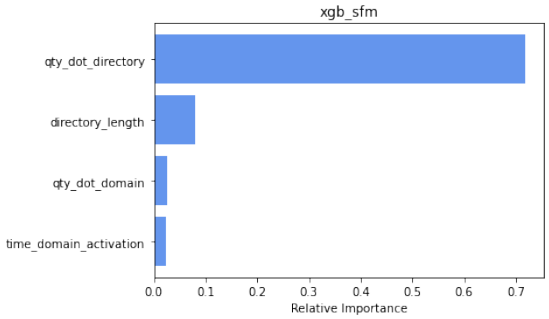


Figure 9: Relative importance of features: XGB/SFM

6. Discussion

Results suggest that in general the Random Forest Classifier and the XGBoost Model perform best when classifying URLs to be phishing or not on this specific dataset. These results align with previous literature (Section 2) where findings also suggest that the Random Forest Classifier and the XGBoost model perform well when classifying URLs. The ultimate best model in this paper was the XGBoost model with a Recursive Feature Extraction feature engineering method, with an F1 score of 96% and a false positive rate of 4%. For this model, the directory seems to be the most determinant part of the URL when it comes to classifying the URL as phishing or not. Specifically, the number of asterisks and commas in the directory appear to be very relevant to determining the target variable, with a combined relative importance of approximately 65%.

When it comes to the Naive Bayes Model, the results from this paper contradict the results of previous literature. Our results suggest the Naive Bayes model is not too accurate on this dataset, whereas related work finds the Naive Bayes Model to perform well. Additionally, whereas previous literature reaches accuracy levels of more than 99%, current research reaches a maximum accuracy of 96%. As mentioned in Section 2, these inconsistencies may be due to the fact that current research includes a considerably larger dataset, not only with regards to number of observations but number of features. Another reason could be the synthetic oversampling that has been done in order to combat the class imbalance in the dataset.

From the analysis in section 5.2, it is derived that many of the most important features in the different models are connected to the directory part of the URL. The directory specifies which part of the website the URL directs to. Because phishing usually does not occur on the front page of a website, it makes sense that the directory contributes a significant part of detecting phishing URLs compared to the rest of the features. It can be seen that directory_length is a main contributor for most of the inspected models. The longer the

length of the directory, the higher the chances are that the URL is a phishing URL. Why exactly it is the directory that is important, remains unanswered in this paper, and is a great opportunity for future research. Moreover, note that these results cannot be generalized and hold only for the dataset used in this paper. Hence, there is much room for future research to investigate whether this also holds for other datasets.

The biggest limitation of this paper is that it is built on a readily pre-processed dataset with no available raw data. This means that there are no records of the full URLs and there is only data on the different features of the URLs. This limits this paper in a sense that there is no possibility for detailed error analysis and there is no check whether the dataset has been cleaned in a way that the authors of this paper would have done themselves. Therefore, the biggest opportunity for a future iteration of this research would be to work with raw data.

7. Conclusion

Throughout this paper, we analyzed how Machine Learning can be utilized in detecting phishing URLs, which would assist cybersecurity professionals in the battle against their adversaries. Further, we looked at the different parts of the URL in an effort to identify which part would be the most beneficial to look at, when trying to detect them. This was done through three different Feature Engineering Methods: Recursive Feature Elimination, Principle Component Analysis, and Regression Weight Approach (Select From Model).

From the four different Machine Learning models applied in this paper, the RFC model and the XGBoost model performs best when classifying phishing URLs on our dataset. We found that the RFC performed best with an SFM feature engineering method (93%), and the XGB performed best with an RFE feature engineering method (96%).

From here, we gathered which features, and therefore which parts of the URL contribute the most to determining the target variable. We found that for all four models, the features with the highest relative importance were part of the directory of the URL. Therefore we can say that in this analysis with this dataset, the directory was the most important part of the URL when classifying phishing vs legitimate URLs.

The fact that this paper does not reach the same accuracy levels as in previous papers can be due to two reasons: difference in size of dataset used (both length and width) and class imbalance reduction methods.

Future research should focus on the following topics: using raw data for the analysis, detailed error analysis and diving into the topic of why exactly the directory could be so important in detecting phishing URLs.

8. References

Alauthman, M., Almomani, A., Alweshah, M., Omoush, W., & Alieyan, K. (2019). Machine Learning for Phishing Detection and Mitigation. Machine Learning for Computer and Cyber Security, 48–74. https://doi.org/10.1201/9780429504044-2

Almomani, A., Alauthman, M., Shatnawi, M. T., Alweshah, M., Alrosan, A., Alomoush, W., & Gupta, B. B. (2022). Phishing Website Detection With Semantic Features Based on Machine Learning Classifiers: A Comparative Study. International Journal on Semantic Web and Information Systems (IJSWIS), 18(1), 1-24.

Amiri, I. S., Akanbi, O. A., & Fazeldehkordi, E. (2014). A machine-learning approach to phishing detection and defense. Syngress.

Biecek, P., & Burzykowski, T. (2022). Explanatory Model Analysis: Explore, Explain, and Examine Predictive Models (Chapman & Hall/CRC Data Science Series) (1st ed.). Chapman and Hall/CRC.

Brownlee, J. (2020, January 24). Tour of Data Sampling Methods for Imbalanced Classification. shorturl.at/awF36

Cloudflare. (2022, November 17. Why is HTTP not secure? https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/

Desai, A., Jatakia, J., Naik, R., & Raul, N. (2017, May). Malicious web content detection using machine leaning. In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 1432-1436). IEEE.

Dwivedi, A. (2022). Lecture 6: Machine Learning for Cyber Security. Copenhagen Business School. Foundations of Cyber Security course.
Elreedy, D., & Atiya, A. F. (2019). A Comprehensive Analysis of Synthetic Minority Oversampling Technique (SMOTE) for handling class imbalance. Information Sciences, 505, 32–64. https://doi.org/10.1016/j.ins.2019.07.070

Feature selection. (n.d.). Scikit-learn. shorturl.at/cjmsz

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (3rd ed.). O'Reilly Media.

GlobeNewsWire. (2022, August 25). With 9.51% CAGR, Global Cyber Security Market Size to Reach USD 478.68 Billion by 2030. shorturl.at/GJMX6

Jacobson, D., & Idziorek, J. (2012). Computer Security Literacy: Staying Safe in a Digital World (1st ed.). Chapman and Hall/CRC. Social Engineering Defined. (2022, February 18).

Johnson, J. M., & Khoshgoftaar, T. M. (2020). The Effects of Data Sampling with Deep Learning and Highly Imbalanced Big Data. Information Systems Frontiers, 22(5), 1113–1131. https://doi.org/10.1007/s10796-020-10022-7

Karabatak, Murat, & Twana Mustafa. "Performance comparison of classifiers on reduced phishing website dataset." 2018 6th International Symposium on Digital Forensic and Security (ISDFS). IEEE, 2018.

Kasim, Ömer. Automatic detection of phishing pages with event-based request processing, deep-hybrid feature extraction and light gradient boosted machine model. Telecommunication Systems 78.1 (2021): 103-115.

Nguyen, L. A. T., To, B. L., Nguyen, H. K., & Nguyen, M. H. (2013, October). Detecting phishing web sites: A heuristic URL-based approach. In 2013 International Conference on Advanced Technologies for Communications (ATC 2013) (pp. 597-602). IEEE.

Niakanlahiji, A., Chu, B. T., & Al-Shaer, E. (2018, November). Phishmon: A machine learning framework for detecting phishing webpages. In 2018 IEEE International Conference on Intelligence and Security Informatics (ISI) (pp. 220-225). IEEE.

Rao, R. S., & Pais, A. R. (2019). Detection of phishing websites using an efficient feature-based machine learning framework. Neural Computing and Applications, 31(8), 3851-3873.

Scarpati, J. (2021, September 1). Uniform Resource Locator. shorturl.at/amvAS

Security Through Education. shorturl.at/lrGKY

Tyagi, I., Shad, J., Sharma, S., Gaur, S., & Kaur, G. (2018, February). A novel machine learning approach to detect phishing websites. In 2018 5th International conference on signal processing and integrated networks (SPIN) (pp. 425-430). IEEE.

Zhang, Y., Hong, J. I., & Cranor, L. F. (2007, May). Cantina: a content-based approach to detecting phishing web sites. In Proceedings of the 16th international conference on World Wide Web (pp. 639-648).

Appendix A. Description of features in the dataset

Appendix A.1. List of all features and descriptions

Table is created by the owners of the dataset.

Table 3: Description of features in the dataset

Feature	Description	Feature	Description
qty_dot_url	count (.) in URL	qty_percent_directory	count (%) in directory
qty_hyphen_url	count (-) in URL	directory_length	directory length
qty_underline_url	count (_) in URL	qty_dot_file	count (.) in file
qty_slash_url	count (/) in URL	qty_hyphen_file	count (-) in file
qty_questionmark_url	count (?) in URL	qty_underline_file	count (_) in file
qty_equal_url	count (=) in URL	qty_slash_file	count (/) in file
qty_at_url	count (@) in URL	qty_questionmark_file	count (?) in file
qty_and_url	count (&) in URL	qty_equal_file	count (=) in file
qty_exclamation_url	count (!) in URL	qty_at_file	count (@) in file
qty_space_url	count ( ) in URL	qty_and_file	count (&) in file
qty_tilde_url	count (∼) in URL	qty_exclamation_file	count (!) in file
qty_comma_url	count (,) in URL	qty_space_file	count ( ) in file
qty_plus_url	count (+) in URL	qty_tilde_file	count (∼) in file
qty_asterisk_url	count (*) in URL	qty_comma_file	count (,) in file
qty_hashtag_url	count (#) in URL	qty_plus_file	count (+) in file
qty_dollar_url	count ($) in URL	qty_asterisk_file	count (*) in file
qty_percent_url	count (%) in URL	qty_hashtag_file	count (#) in file
qty_tld_url	top-level-domain length	qty_dollar_file	count ($) in file
length_url	URL length	qty_percent_file	count (%) in file
qty_dot_domain	count (.) in domain	file_length	file length
qty_hyphen_domain	count (-) in domain	qty_dot_params	count (.) in parameters
qty_underline_domain	count (_) in domain	qty_hyphen_params	count (-) in parameters
qty_slash_domain	count (/) in domain	qty_underline_params	count (_) in parameters
qty_questionmark_domain	count (?) in domain	qty_slash_params	count (/) in parameters
qty_equal_domain	count (=) in domain	qty_questionmark_params	count (?) in parameters
qty_at_domain	count (@) in domain	qty_equal_params	count (=) in parameters
qty_and_domain	count (&) in domain	qty_at_params	count (@) in parameters
qty_exclamation_domain	count (!) in domain	qty_and_params	count (&) in parameters
qty_space_domain	count ( ) in domain	qty_exclamation_params	count (!) in parameters
qty_tilde_domain	count (∼) in domain	qty_space_params	count ( ) in parameters
qty_comma_domain	count (,) in domain	qty_tilde_params	count (∼) in parameters
qty_plus_domain	count (+) in domain	qty_comma_params	count (,) in parameters
qty_asterisk_domain	count (*) in domain	qty_plus_params	count (+) in parameters
qty_hashtag_domain	count (#) in domain	qty_asterisk_params	count (*) in parameters
qty_dollar_domain	count ($) in domain	qty_hashtag_params	count (#) in parameters
qty_percent_domain	count (%) in domain	qty_dollar_params	count ($) in parameters
qty_vowels_domain	count vowels in domain	qty_percent_params	count (%) in parameters
domain_length	domain length	params_length	parameters length
domain_in_ip	URL domain in IP address format	tld_present_params	TLD presence in arguments
server_client_domain	domain contains the keywords "server" or "client"	qty_params	number of parameters
qty_dot_directory	count (.) in directory	email_in_url	email present in URL
qty_hyphen_directory	count (-) in directory	time_response	search time (response) domain (lookup)
qty_underline_directory	count (_) in directory	domain_spf	domain has SPF

Table 3 continued from previous page

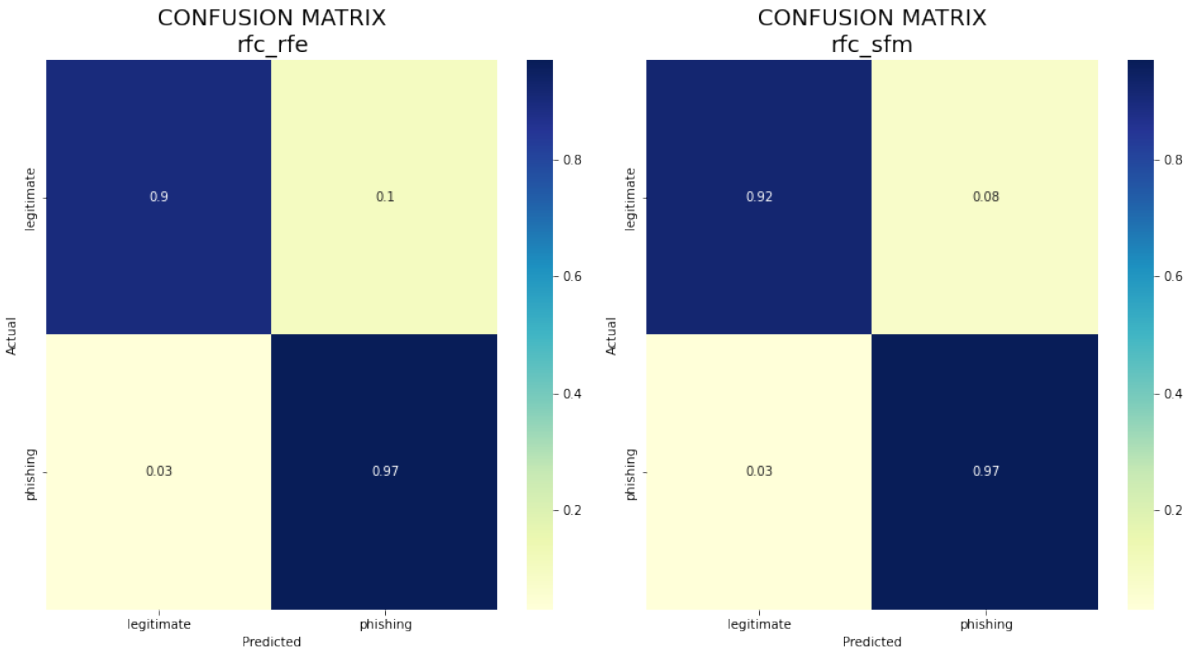
Feature	Description	Feature	Description
qty_slash_directory	count (/) in directory	asn_ip	AS Number (or ASN)
qty_questionmark_directory	count (?) in directory	time_domain_activation	time (in days) of domain activation
qty_equal_directory	count (=) in directory	time_domain_expiration	time (in days) of domain expiration
qty_at_directory	count (@) in directory	qty_ip_resolved	number of resolved IPs
qty_and_directory	count (&) in directory	qty_nameservers	number of resolved name servers (NameServers - NS)
qty_exclamation_directory	count (!) in directory	qty_mx_servers	number of MX Servers
qty_space_directory	count ( ) in directory	ttl_hostname	time-to-live (TTL) value associated with hostname
qty_tilde_directory	count (∼) in directory	tls_ssl_certificate	valid TLS / SSL Certificate
qty_comma_directory	count (,) in directory	qty_redirects	number of redirects
qty_plus_directory	count (+) in directory	url_google_index	check if URL is indexed on Google
qty_asterisk_directory	count (*) in directory	domain_google_index	check if domain is indexed on Google
qty_hashtag_directory	count (#) in directory	url_shortened	check if URL is shortened
qty_dollar_directory	count ($) in directory	phishing	is phishing website

Appendix A.2. Feature count

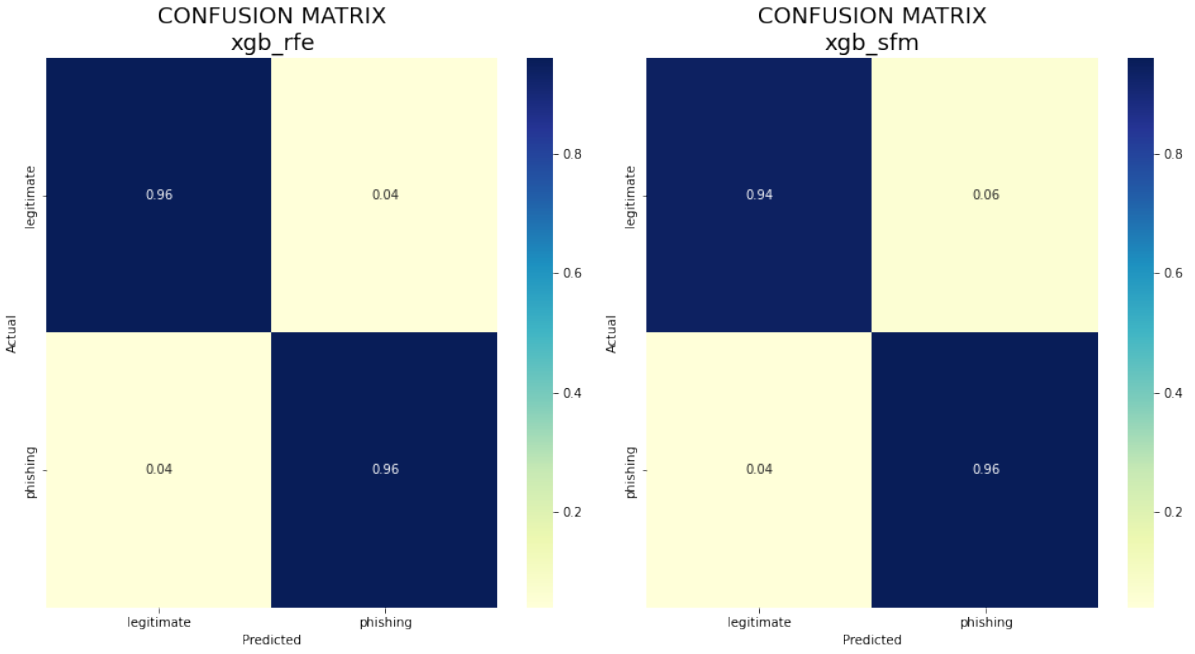
Table 4: Count of features in the dataset

# features in the dataset	Description of features
20	Relating to the entire URL
21	Relating to the domain part of the URL
18	Relating to the directory part of the URL
18	Relating to the file part of the URL
20	Relating to the parameter part of the URL
15	Relating to resolving URLs and external services

Appendix B. Confusion Matrices of selected models



((a)) Random Forest - Recursive Feature Elimination (RFE) **((b))** Random Forest - Regression Weight Approach (SFM)



((c)) XGBoost - Recursive Feature Elimination (RFE) **((d))** XGBoost - Regression Weight Approach (SFM)

Figure 10: Confusion Matrices of Random Forest and XGBoost models after applying feature selection techniques