Cybersecurity Foundations and Analytics

Predicting Phishing URLs using Random Forest and Neural Network Classifiers

Jenő Tóth (158386)

S M Ahasanul Karim (158793)

Character count: 22,459

Page count: 13

Submission date: November 27, 2023

Cybersecurity Foundations and Analytics Final Exam Project

MSc. in Business Administration and Data Science

Introduction

As the world transitions towards more and more advanced technology, security concerns become more and more prominent. Because in many cases, the security aspects are struggling to catch up with the pace of technological advancements. Various system vulnerabilities, spyware, malware, ransomware, phishing and eavesdropping are making 'data breaches' more common. These events lead to drastic consequences like leaks, extortion, blackmailing, espionage, etc. Both businesses and individuals fall victim to these extreme consequences. According to DefiLlama[1] data, cybercriminals have cost crypto businesses and DeFi protocols about $735 million in 69 hacks so far in 2023. The average ransomware demand was expected to be $220,298 in 2021, a 43% rise over 2020. If current attack rates continue, criminals using ransomware will extract $899 million from victims by 2023. According to experts, ransomware will strike a company, customer, or device every two seconds by 2031[2]. However, phishing is the most common method for ransomware delivery.

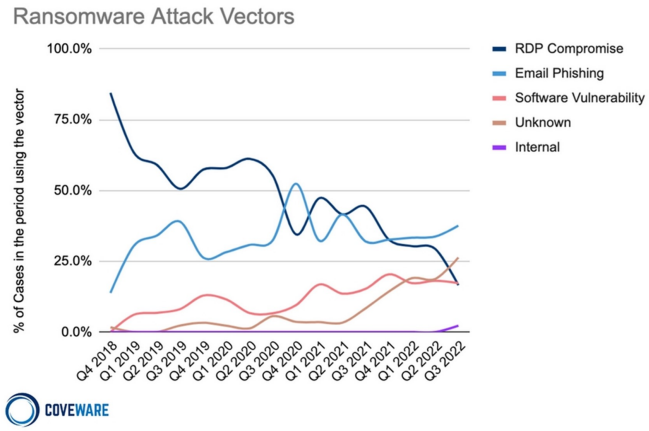

Fig1: Phishing Presiding over Other Data Breach Factors

The term "phishing" originated from the similar activity of "fishing," which involves throwing a line in the hopes of getting bit by the target. The program known as AOHell, created by a teenage developer in Pennsylvania, is credited with coining the term "phishing." It used to

[1] https://defillama.com/hacks
[2] https://www.stationx.net/ransomware-statistics/#:~:text=7.,8.

employ a password-cracking and credit-card-stealing mechanism to interfere with AOL's operations. Additional automated phishing software was produced by this program. “The Warez community” then attributed the first organized phishing attacks targeted AOL users in 1996. They used algorithms to generate random credit card numbers and as soon as the group landed on a valid number, they were able to create real AOL accounts to scam other AOL users.[3] From the description of the scam, it can be inferred that the phishing attackers target a certain audience, deceive them via email, text, social media, or an app; pretending to be someone or some organization, and lure them to take away sensitive personal data like passwords, banking credentials, company secrets or any sort of private documents. The information received can later be used for identity theft and financial loss. There are many types of phishing like ‘email phishing’, ‘smishing’ (SMS Phishing), ‘vishing’ (voice phishing), ‘spear phishing’(targeting individuals), ‘whaling’(targeting CEOs) etc. The companies that have been faked for phishing include Adobe, PayPal, Microsoft, Apple, Google, CIBC, BNP Paribas, Wells Fargo, AT&T etc. However more focus is put into the business sector for impersonating, prevailing by 35% in comparison to the other sectors in 2021.[4] Verizon’s Investigations reported that one-third of all data breaches were the result of phishing emails.[5]

The cost and harm to a company's reputation increase with the size of the data breach. In 2007, Nordea lost over 7 million SEK to phishers who sent emails to customers, luring them to install "haxdoor" Trojan. A breach originated by a spear-phishing email leaked information on three billion Yahoo accounts in 2013 which is considered one of the biggest to date. Names, dates of birth, passwords, and security question answers of all these people could have been used to steal information from other accounts created by the same users. In 2014, Sony went under a huge data leak of 100 Terabytes of confidential company data,which cost them over $100 million.[6] Early in 2015, phishing was used to compromise the data of about 80 million Anthem members. Similar circumstances took place in mid-2013 when 360 million accounts' worth of data were lost on Myspace, an outdated social media mammoth.[4] A man named

[3] https://www.comptia.org/content/articles/what-is-phishing
[4] https://www.idstrong.com/sentinel/what-is-phishing/
[5] https://www.verizon.com/business/resources/reports/dbir/
[6] https://www.hempsteadny.gov/635/Famous-Phishing-Incidents-from-History

Evaldas Rimasauskas, from Lithuania, stole over $100 million from Facebook and Google by faking invoices and contracts with forged email accounts between 2013 to 2015. In December 2015, through a phishing email, the malware "BlackEnergy" was released into Kyivoblenergo, a Ukrainian electricity distribution company, disconnecting 30 substations for three hours, causing 230,000 customers to lose electricity. A San Jose-based technology company, Ubiquity Network and Austrian aerospace parts manufacturer and engineering company FACC became subject to whaling attacks consecutively in 2015 and 2016 causing them to lose around 50 million USD each.[7] Around the same timeline, the same email was sent to multiple Bangladesh Bank employees by a job applicant named Rasel Ahlam, a fictitious character covering the Lazarus Group. It contained a courteous inquiry and an invitation to download his resume and cover letter from a site. Someone in the bank fell for the ruse, downloaded the files, and got the system infected with the viruses concealed therein. The hackers waited for months to plan the perfect escape route and stole around $65 million USD from the central bank of Bangladesh.[8]

However, by utilizing new ways to create phishing attacks, attackers keep refining their attack strategies. For instance, they can create a webpage and phishing URLs that mimic legitimate URLs such as https://www.paypal.com being mimicked by https://www.paypaI.com (small L(l) for I(i)) or https://www.patpal.net. (Prakash et al., 2010). As a result, differentiating between a benign URL and a phishing URL is crucial. Because of this, researchers have recently put forth a number of highly accurate methods to identify phishing attacks, including blacklists (Asiri et al., 2023).

Literature Review

In comparison to other domains, the availability of cybersecurity datasets is limited. As businesses find their cybersecurity data extremely sensitive and private, they typically abstain from publishing or sharing them for research. However, lately, there has been some degree of analytical research in this domain.

[7] https://www.graphus.ai/blog/worst-phishing-attacks-in-history/
[8] https://www.bbc.com/news/stories-57520169

Asiri et al., 2023 surveyed HTML and URL phishing attacks and methods with deep learning models to detect URL-based and hybrid-based phishing attacks. Rugangazi and Okeyo (2023) achieved a remarkable accuracy of 98.85% with the Random Forest algorithm using the ISCXURL-2016 dataset with feature selection processes. Without the need for a human to choose the features based on domain expertise and experience, their research automatically extracts the features from the data. Deshpande et al. (2021) explained phishing domain characteristics, distinguishing features, and how to detect them using machine learning and natural language processing techniques. Saraswathi et al. (2023) used ANN, SVM, Random Forests, and K-NN to detect phishing websites from publicly available phishing websites collected from the UC Irvine ML repository. Singh et al. (2023) came up with a novel approach that combines Particle Swarm Optimization (PSO) with feature selection techniques, including correlation and mutual information, and tree-based feature selection to accurately differentiate between legitimate and phishing websites by identifying relevant features in their research. S. and K. (2023) used a RoBERTa transformer for feature extraction and the LSTM for classification and their approach differentiates between benign and phishing URLs with an accuracy of 97.14% in an extensive 3,00,000 URLs dataset.

Tan et al. (2016) proposed a phishing detection technique based on the difference between the target and actual identities of a webpage. Their approach called ‘PhishWHO’ extracts identity keywords from the textual contents of the website, where they proposed a novel weighted URL tokens system based on the N-gram model. Then the target domain name is found by using a search engine, and the target domain name is selected based on their identity-relevant features. Afterwards, they proposed a 3-tier identity matching system for finding out the legitimacy of the query webpage. They claimed their proposed system outperforms the other conventional phishing detection methods. Hanus et al. (2021) conducted a study in collaboration with a southwestern U.S. municipality where they categorised phishing messages into regular and spear-phishing types. They employed eight supervised learning methods on collected demographic data to find out that spear phishing is more successful and highlight the efficacy of certain machine learning approaches in predicting phishing victims. Katherine et al.

(2019) discussed the various types of phishing attacks and the recent approaches to prevent them. They proposed a framework to detect and prevent them combining supervised and unsupervised machine learning techniques to detect if attacks are known or unknown.

Methodology

The dataset we used contains 2 columns: a string containing the URL name and an associated label. 428,103 URLs were labeled benign and 94,111 labeled phishing (as well as others with different labels which were omitted in this research) (Siddhartha, 2021). A number of features were extracted from the URL name based on Tan (2018), which were used for training the models later on:

- UrlLength: The length of the URL string.
- NumDots: The number of dots in the URL.
- NumDash: The number of dashes in the URL.
- NumAtSymbol: The number of @ symbols in the URL.
- NumTildeSymbol: The number of ~ symbols in the URL.
- NumUnderscore: The number of underscores in the URL.
- NumPercent: The number of % symbols in the URL.
- NumAmpersand: The number of & symbols in the URL.
- NumHash: The number of # symbols in the URL.
- NumNumericChars: The number of numeric characters in the URL.
- NoHttps: If the URL starts with “http://” than 1, otherwise 0.
- PathLevel: Calculated from the number of “/” symbols after the domain.
- SubdomainLevels: Calculated from the number of dots in the domain.
- NumDashInHostname: The number of dash symbols in the domain.
- NumQueryComponents: The number of ? symbols in the path.

Random forest

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. (Breiman, 2001). One of their main advantages over neural networks is their simplicity. They are quick to train, and require less computational resources, meaning that if a random forest model performs similarly to a neural network, it is a better model due to its deployment features. scikit-learn's RandomForestClassifier class was used to train the first set of models. The following hyperparameters were considered when building the models, with their tested values in parentheses:

- n_estimators (100, 200, 300, 400, 500): The number of decision trees to be used.
- max_depth (10, 20, 30, 50, 100, None): The maximum depth of each decision tree in the ensemble.
- min_samples_split (2, 5, 10): The minimum number of samples required to split an internal node.
- min_samples_leaf (1, 2, 4): The minimum number of samples required to be in a leaf node.
- max_features (auto, sqrt, log2): The number of features to consider. 'auto' means considering all features, 'sqrt' considers a number of features equal to the square root of all features, 'log2' considers the log base 2 number of features.

20% of the data was used for testing, and 80% for training. 5-fold cross validation was conducted in order to generalize the models. Since there is a large imbalance between the classes (82% of the data being benign), a set of random forest classifiers and neural network models were trained on an oversampled subset of the data. Synthetic Minority Over-sampling Technique (SMOTE) was used to achieve this, and the models were trained on the same hyperparameter choices as previously.

Neural networks

Neural networks need more input data and more computational capacity, but oftentimes they may lead to significantly more accurate models when compared to random forests. The train-test split was the same as previously. Similarly to the previous section, the imbalanced data and the SMOTE-sampled data were both used in order to find the best model. The neural networks had 3-8 layers. An input layer with the training data and an output layer with a binary output using a sigmoid activation function were the same across all models. Between the input and output layers, the combination of 1-6 hidden layers were tested, with the number of nodes ranging from 32 to 768 for each layer, and all combinations of Rectified Linear Unit (ReLU) and sigmoid activation functions were used. The adam algorithm was used to optimize learning rates. In total, 538 different neural network models were trained for both the complete data as well as the oversampled data. Each model was trained for 10 epochs, using batch learning with size 32 batches.

Results

For measuring observational error, we looked at both accuracy and recall scores. Accuracy is measured since it is important for the model to be as generalizable as possible. However, as the main task of our model is to detect phishing links, the model should prioritize minimizing the amount of false negatives, meaning phishing URLs which are wrongly classified as benign. The baseline model for our research was considered to be a random forest classifier trained on the whole data with default parameters (n_estimators = 100, max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_features = 'sqrt'). This model achieved 0.897 accuracy. Before experimenting with the hyperparameters, another model was trained on the same parameters, however, only on the SMOTE-sampled data. While it was expected that this model will achieve better results, since it tackles the class imbalance, it only achieved 0.839 accuracy. In the end, we came up with 8 different models, based on whether they use random forest or neural networks, whether they are trained on the whole data or the SMOTE-sampled

data, and whether they are optimized for accuracy or recall. These 8 models are described below, and analyzed qualitatively.

After the hyperparameter optimizations, the four best random forest models have the following properties. The model trained on the whole data, optimized for accuracy has the following parameters: n_estimators = 500, max_depth = 30, min_samples_split = 2, min_samples_leaf = 2, max_features = 'sqrt'. This first model achieved 0.90 accuracy, however, only 0.58 recall. The model trained on the SMOTE-sampled data, optimized for accuracy has the following parameters: n_estimators = 500, max_depth = None, min_samples_split = 5, min_samples_leaf = 1, max_features = 'sqrt'. This second model achieved 0.90 accuracy and 0.59 recall. The third model, trained once again on the full data, but now optimized for recall has the following parameters: n_estimators = 500, max_depth = 50, min_samples_split = 2, min_samples_leaf = 1, max_features = 'sqrt'. This third model achieved 0.31 accuracy and 0.95 recall. The final random forest model, trained on the SMOTE-sampled data, optimized for recall has the following parameters: n_estimators = 500, max_depth = 30, min_samples_split = 5, min_samples_leaf = 2, max_features = 'sqrt'. This fourth model achieved 0.30 accuracy and 0.95 recall.

As we can see, there are large differences in model outcomes depending on whether we are aiming for maximizing accuracy or recall. The two models optimized for accuracy achieved similar results, with the SMOTE-oversampled model having slightly higher recall. Similarly, the models optimized for recall achieved similar scores, however, it came at the cost of sacrificing a lot of accuracy, making these models unreliable. It is also interesting to examine the parameters of the models. We can see that all of them have n_estimators set to the highest possible value, 500. This is not surprising, since it allows the models to be larger and more accurate. Future research may be conducted to see how the models improve as n_estimators further increases, however, this may be futile, since the larger a model is, the more computational resources it requires.

When building neural network models, accuracy was used during model training, and recall was only considered when evaluating. This is because the models were not trained to predict exact binary values, rather certainties ranging from 0 and 1. The binary labels only came into play when predicting. Here, a decision boundary was set up to classify the predictions. When the models were focusing on accuracy, this boundary was set to 0.5, when they were tasked at maximizing recall, it was 0.2. This allowed the second types of models to classify more URLs as phishing, at the cost of slightly lower accuracies. As mentioned earlier, the hidden layers were tested with both ReLU and sigmoid activation functions. Despite this, all four of the best models had only ReLU activation functions in their hidden layers. This was a surprise, but a welcome one, since ReLU functions are less computationally expensive compared to sigmoid functions.

The first neural network model, trained on the full data and optimized for accuracy had 4 hidden layers (768, 384, 192 and 64 neurons respectively). It achieved 0.89 accuracy and 0.56 recall. The second model, trained on the SMOTE-sampled data and optimized for accuracy, had 3 hidden layers (128, 64 and 32 neurons respectively). It achieved 0.78 accuracy and 0.35 recall. The third model, trained on the full data and optimized for recall had 3 hidden layers (192, 192 and 128 neurons respectively). It achieved 0.83 accuracy and 0.86 recall. The final model, trained on the SMOTE-sampled data and optimized for recall had only 1 hidden layer with 64 neurons. It achieved 0.34 accuracy and 0.79 recall.

As we can see, the models trained on the SMOTE-sampled data performed significantly worse. This can be explained by the neural networks' advantage of being able to utilize extremely large datasets. Since the sampled data was smaller, the models were unable to utilize their features well. The confusion matrices for all 8 of the best models can be seen on the next two pages.

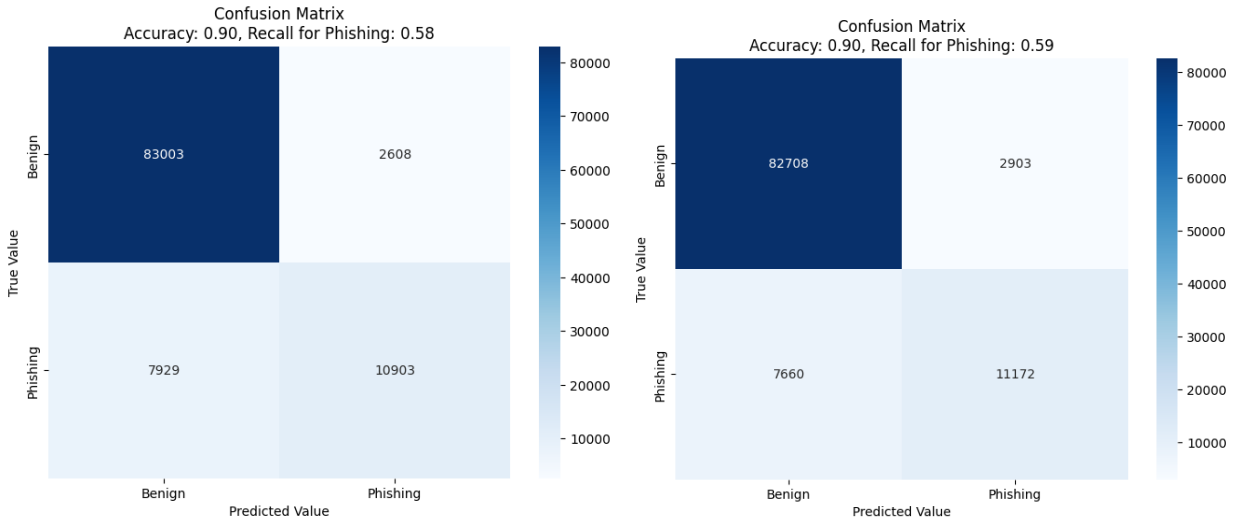

Fig2: Random forest models trained on the full data (left) and the SMOTE-sampled data (right), optimized for accuracy

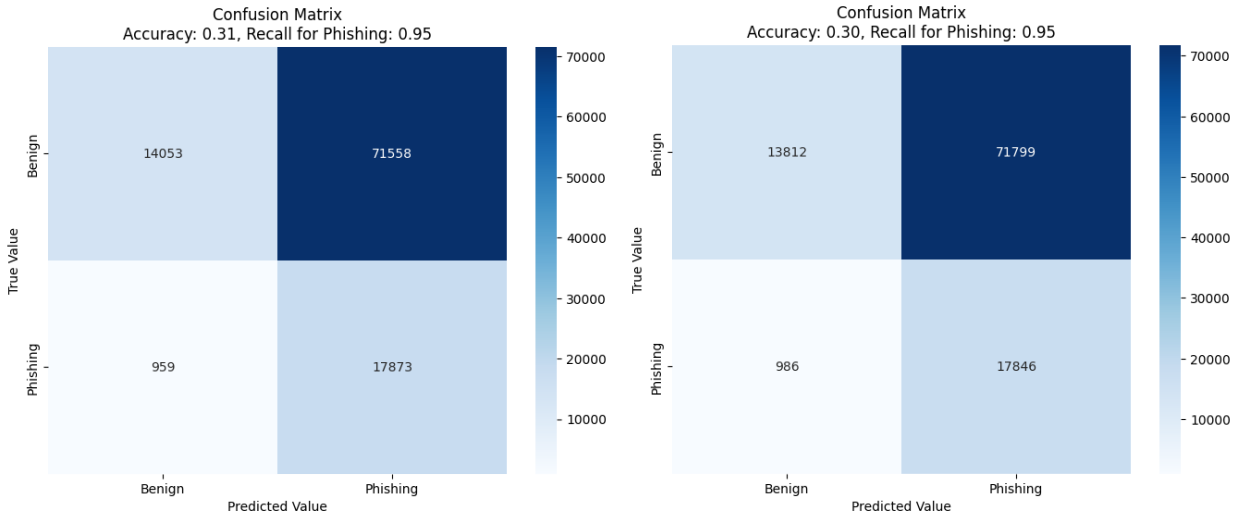

Fig3: Random forest models trained on the full data (left) and the SMOTE-sampled data (right), optimized for recall

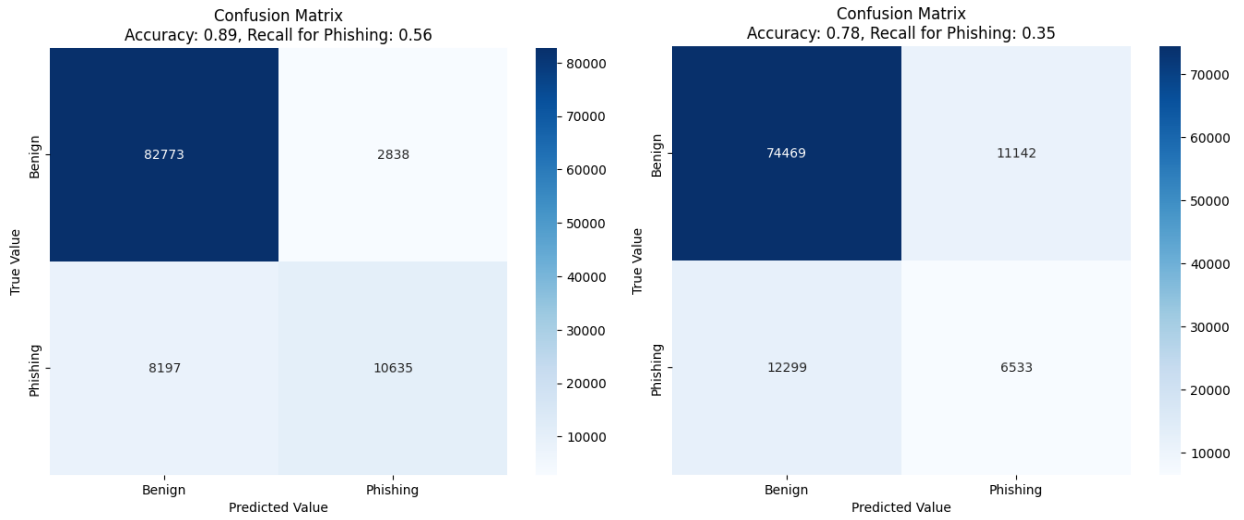

Fig4: Neural network models trained on the full data (left) and the SMOTE-sampled data (right), optimized for accuracy

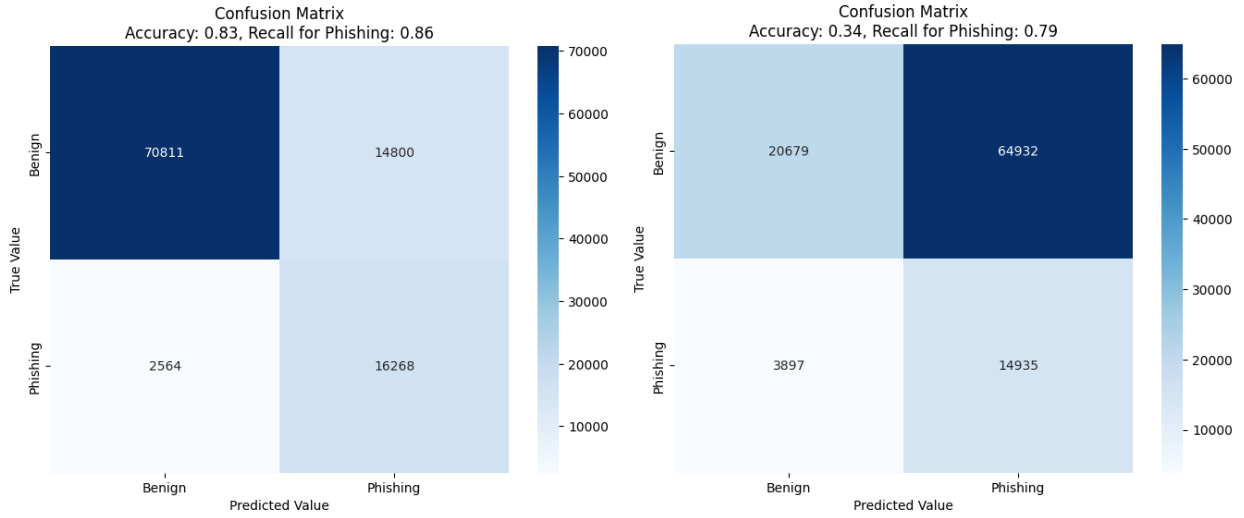

Fig5: Neural network models trained on the full data (left) and the SMOTE-sampled data (right), optimized for recall

Discussion

From the results, we can see that the oversampling models with SMOTE have less overall accuracy than the baseline models for both the Random Forest and Neural Network model variations. However, it is important to ensure that the model doesn't miss many phishing threats, minimizing false negatives. The predicted phishing sites can be always further investigated and marked as safe in case of false alarms. Thus, for detecting the predicting phishing websites accurately the recall scores are also important to consider.

The tradeoff between accuracy and recall was most prominent in the random forest models. When accuracy was the target, the models predicted benign for the majority of the cases. Since this could be due to the class imbalances, SMOTE-oversampling slightly helped this issue, however, recall still remained low. On the other hand, when recall was prioritized, the models predicted phishing overwhelmingly. This is understandable, since they were rewarded for not missing any true phishing URLs.

For our neural network models, this issue was still prominent, but to a lesser extent. Here, the main issue was with the SMOTE oversampling. Since neural networks need large amounts of data, they were unable to build reliable models on the SMOTE data. Despite this concern, the neural network model using the full training data and optimizing for recall managed to achieve high recall, while only sacrificing a few percentage points of accuracy. This model therefore is both useful for predicting phishing URLs, and it is also generalizable for new data.

It should also be considered that these sorts of machine learning models can easily be subject to adversarial attacks as they specifically relate to cybersecurity. Therefore, additional models processed need to be developed to make it more robust against attacks aimed at deceiving the classifiers. Another crucial aspect of deploying these models is their interpretability. While tree-based ensemble modes like Random Forest offer transparency, Neural Networks pose challenges in understanding their decision-making processes. This trade-off between accuracy

and interpretability should be carefully considered when implementing these models in practical cybersecurity applications. The Neural Network model architectures can be studied further for explainability. The computation power should also be evaluated on both of these models in terms of scalability and financial cost.

Conclusion

Phishing is the number one cybersecurity concern for businesses and individuals . This research suggests that phishing attacks are interpretable and detectable with the use of Machine Learning and Deep Neural Networks. We found that deep neural networks perform better than Random Forests in terms of finding malicious websites with oversampling techniques for fixing class imbalance. However, they are less interpretable than Random Forest by nature.

Our study opens avenues for future research in several areas. Further investigations could focus on refining the models by exploring ensemble methods for all the Neural Networks or incorporating additional information into the dataset. We have addressed the limitations in this study, such as the dynamic nature of phishing tactics which can guide the development of more resilient and adaptive models.

In conclusion, this research contributes to the ongoing efforts to confront phishing threats by providing a comparative analysis of machine learning and deep learning models. While acknowledging their respective strengths and weaknesses, the study underscores the importance of a multi-faceted approach to phishing detection. As we keep navigating the landscape of cybersecurity, our findings direct us towards advancements that align with the ever-changing nature of malicious online activities. The results presented in this study can offer valuable insights into the efficiency and convenience of machine learning models in predicting phishing URLs and provide a foundation for further research to fortify online security measures.

References

Aldakheel, E. A., Zakariah, M., Gashgari, G. A., Almarshad, F. A., & Alzahrani, A. I. A. (2023). A Deep Learning-Based Innovative Technique for Phishing Detection in Modern Security with Uniform Resource Locators. In Sensors (Vol. 23, Issue 9, p. 4403). MDPI AG. https://doi.org/10.3390/s23094403

Breiman, L. (2001). In Machine Learning (Vol. 45, Issue 1, pp. 5–32). Springer Science and Business Media LLC. https://doi.org/10.1023/a:1010933404324

Deshpande, A., Pedamkar, O., Chaudhary, N., Borde, S. (2021). Detection of Phishing Websites using Machine Learning. In International Journal of Engineering Research & Technology (Vol. 10, Issue 5).

Hanus, B., Wu, Y. A., & Parrish, J. (2021). Phish Me, Phish Me Not. In Journal of Computer Information Systems (Vol. 62, Issue 3, pp. 516–526). Informa UK Limited. https://doi.org/10.1080/08874417.2020.1858730

Kathrine, G. J. W., Praise, P. M., Rose, A. A., & Kalaivani, E. C. (2019). Variants of phishing attacks and their detection techniques. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). IEEE. https://doi.org/10.1109/icoei.2019.8862697

Prakash, P., Kumar, M., Kompella, R. R., & Gupta, M. (2010). PhishNet: Predictive Blacklisting to Detect Phishing Attacks. In 2010 Proceedings IEEE INFOCOM. IEEE INFOCOM 2010 - IEEE Conference on Computer Communications. IEEE. https://doi.org/10.1109/infcom.2010.5462216

Rugangazi, B., & Okeyo, G. (2023). Detecting Phishing Attacks Using Feature Importance-Based Machine Learning Approach. In 2023 IEEE AFRICON. 2023 IEEE AFRICON. IEEE. https://doi.org/10.1109/africon55910.2023.10293475

S, J. K., & B, A. (2023). Phishing URL detection by leveraging RoBERTa for feature extraction and LSTM for classification. In 2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS). 2023 Second International Conference on

Augmented Intelligence and Sustainable Systems (ICAISS). IEEE. https://doi.org/10.1109/icaiss58487.2023.10250684

Saraswathi, P., Anchitaalagammai, J. V., & Kavitha, R. (2023). A System Review on Fraudulent Website Detection Using Machine Learning Technique. In SN Computer Science (Vol. 4, Issue 6). Springer Science and Business Media LLC. https://doi.org/10.1007/s42979-023-02084-6

Siddhartha, M. (2021). Malicious URLs dataset. Kaggle. https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset

Singh, T., Kumar, M., & Kumar, S. (2023). Enhancing Phishing Website Detection Using Particle Swarm Optimization and Feature Selection Techniques. In 2023 IEEE World Conference on Applied Intelligence and Computing (AIC). 2023 IEEE World Conference on Applied Intelligence and Computing (AIC). IEEE. https://doi.org/10.1109/aic57670.2023.10263814

Tan, C. L. (2018). Phishing Dataset for Machine Learning: Feature Evaluation [dataset]. Mendeley. https://doi.org/10.17632/H3CGNJ8HFT.1

Tan, C. L., Chiew, K. L., Wong, K., & Sze, S. N. (2016). PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder. In Decision Support Systems (Vol. 88, pp. 18–27). Elsevier BV. https://doi.org/10.1016/j.dss.2016.05.005