Malicious Web Links Detection Using Ensemble Models

Claudia-Ioana Coste[a], Anca-Mirela Andreica[b] and Camelia Chira[c]
Department of Computer Science, Faculty of Mathematics and Computer Science, Babeș-Bolyai University, Mihail Kogalniceanu Street, no. 1, Cluj-Napoca, Romania

Keywords: Malicious Web Links Detection, Machine Learning Algorithms, Ensemble Models, Web-Malware.

Abstract: Malicious links are becoming the main propagating vector for web-malware. They may lead to serious security issues, such as phishing, distribution of fake news and low-quality content, drive-by-downloads, and malicious code running. Malware link detection is a challenging domain because of the dynamics of the online environment, where web links and web content are always changing. Moreover, the detection should be fast and accurate enough that it will contribute to a better online experience. The present paper proposes to drive an experimental analysis on machine learning algorithms used in malicious web links detection. The algorithms chosen for analysis are Logistic Regression, Naïve Bayes, Ada Boost, Gradient Boosted Tree, Linear Discriminant Analysis, Multi-layer Perceptron and Support Vector Machine with different kernel types. Our purpose is twofold. First, we compare these single algorithms run individually and calibrate their parameters. Secondly, we chose 10 models and used them in ensemble models. The results of these experiments show that the ensemble models reach higher metric scores than the individual models, improving the maliciousness prediction up to 96% precision.

1 INTRODUCTION

Web applications are more and more used for data-sensitive services such as banking, e-commerce, e-learning, public administration, and many others. Because of this, web technology has been significantly improved over the years in terms of security, privacy, and development frameworks but the web security domain remains a challenging one. Hackers are always adapting, creating new attacks, especially phishing attacks with malicious links, that trick users into clicking them and launching their malicious behavior. Malware links can point to cloned websites and can download executables without any users' consent. Moreover, malicious URLs can run expensive tasks on the browser's web workers and present spam or fake information that can be deceiving and confusing for users. Bad-intended JavaScript code can be injected into a web page, tracking consumer's private information or performing malicious tasks.

Research provided by (Fortra, 2022) concludes that employees at IT companies who click on a link distributed in phishing emails, are more likely to download the malicious attachment with an overwhelming rate of 84%. Moreover, companies with a medium number of employees are likely to click a phishing link (18%) and download the file (12%) (Fortra, 2022). Shortened URLs are considered a challenging problem. Frequently, they are not blocked by anti-viruses, because the last website destination is not known. Moreover, the top-level domains (TLDs) used in malware URLs do not denote the type of link, because many attackers employ reputable TLDs. Thus, the malicious web links detection problem is a topical domain, extremely relevant for consumers and research contributions are highly needed.

The malicious web links problem can be mathematically formulated as a binary classification with two labels: malicious and benign. The problem can be solved from a preventive perspective, where the detection of malicious links is more important than the prediction of benign web sites. Thus, the False Negative (FN) rate is more important and relevant than the False Positive (FP) one. Still, having too many FPs could involve the overuse of computation resources. For our solution, the input of the problem is defined as a set of lexical and host-based features extracted from the URL. The output is a number: 0 or 1, interpreted as benign and ma-

[a] https://orcid.org/0000-0001-8076-9423
[b] https://orcid.org/0000-0003-2363-5757
[c] https://orcid.org/0000-0002-1949-1298

Coste, C., Andreica, A. and Chira, C.
Malicious Web Links Detection Using Ensemble Models.
DOI: 10.5220/0012180400003584
In *Proceedings of the 19th International Conference on Web Information Systems and Technologies (WEBIST 2023)*, pages 268-275
ISBN: 978-989-758-672-9; ISSN: 2184-3252

licious, respectively. So, $f : D^n \rightarrow 0,1$; where $D^n = (x_1^1, x_1^2, ..., x_1^n), (x_2^1, x_2^2, ..., x_2^n), ..., (x_m^1, x_m^2, ..., x_m^n), n \in \nu$, being the number of features and $m \in \nu$, being the number of records from the used dataset.

The current paper performs a comparative analysis of multiple ML algorithms, such as Adaptive Boosting (ADA), Gradient Boosted Tree (XGB), Logistic Regression (LR), Bernoulli and Gaussian Naive Bayes (BNB and GNB), Linear Discriminant Analysis (LDA), Multi-Layer Perceptron (MLP), Support Vector Machine (SVM) with Radial Basis Function (RBF), linear, polynomial, and sigmoid kernel. The algorithms are calibrated based on the empirical methodology provided by (Coste, 2023). The best 10 configurations are chosen, and ensemble classifiers are built as done by (Pakhare et al., 2021). Computational experiments are performed for the dataset provided by (Urcuqui et al., 2017) and results show that the ensemble models outperform the individual models, which is expected. The best individual algorithm is XGB with 95% precision, and the best ensemble is ADA-XGB-SVM rbf with over 96% precision. Even though our models do not outperform literature models, we propose to investigate the capabilities of voting-majority ensembles into solving malicious web links detection problem. Furthermore, the experiments will be compared with two other approaches, and they will emphasize some improvements we manage to achieve for MLP, LR and BNB models.

The present article is structured in five sections. In 2 we present the related work in the classification of malicious links with multiple ML algorithms. The next section 3 will describe the experimental methodology used. Then, section 4 will detail the results, draw comparisons within our models and with other literature models. Finally, section 5 will accentuate our contributions and propose new ideas for improving our work.

2 RELATED WORK

Many research contributions to the malicious web links detection problem are ML-based, comparing the performance of multiple ML models. The features used in benign / malicious links classification can be extracted from the URL, from the web content of link, including HTML, CSS files and JavaScript (JS) code analysis, from public blacklists and from the graph representation of the malware distribution network.

In some approaches, the experiments are performed in a cross-dataset environment. (Naveen et al., 2019) performs ML techniques (LR, SVM with linear kernel, DT, RF, Categorical Boosting, KNN, NB, K-Means, Feed Forward Neural Network) on 16 datasets. Their experiments relate that in a cross-dataset environment, KNN outperforms all the others. Another cross-dataset approach is developed by (Song et al., 2020), which classifies links based on the JS code, taking into consideration the raw code, the obfuscated and the deobfuscated code. The proposed model is a bidirectional Long Short-Term Memory (BLSTM) and it was compared with NB, SVM, RF, JaST and ClamAV, an anti-virus engine.

Many solutions propose a classification done for multiple labels. (Tung et al., 2022) is proposing a classification model for four classes: benign, spam, malware, and phishing. By comparing DT with RF, RF outperforms the other and the experiments were run on an aggregation of multiple datasets. Likewise, regarding multi-class (spam, defacement, malware, phishing and benign) and binary classification, (Johnson et al., 2020) will propose a ML model comparing: RF, DT, KNN, SVM, LR, ADA, NB, LDA, and two deep learning models developed with Fast.ai and TensorFlow-Keras. One of the purposes of the paper was to analyze how the model is resilient to binary classification and to multi-class classification. It was observed that all models obtained 8-10% higher accuracy for the binary classification than for the multi-class detection. The best performance was achieved by RF and the two deep learning models, but RF was considered the most suitable option.

For a scalable binary classification, (Islam et al., 2019) uses a dataset available in (Urcuqui et al., 2017) and MapReduce. The approach delivers experiments on NN, RF, DT and KNN models. Their results confirmed that RF and DT were the best performing algorithms and the NN model was the most time consuming one. Running experiments on the same dataset, (Vundavalli et al., 2020) compares CNN, LR and NB (Gaussian, Multinomial and Bernoulli), the best results being obtained by Bernoulli NB method. (Janet et al., 2021) compares ML algorithms (SVM, DT, RF, KNN, NB, LR and Stochastic Gradient Descent), the best metrics being reached by the RF model.

(Pakhare et al., 2021) compares multiple ML approaches (LR, SVM with linear, RBF and sigmoid kernels, KNN, DT and RF) and combines three of them into ensembles. From running the algorithms individually, LR performed the best. Regarding the ensembles, KNN-DT-SVM stands out from the rest. Similarly, (Alsaedi et al., 2022) uses three RF models, trained for a different feature category and a MLP meta-classifier that will output the final class. RF was chosen after experiments with other models: NB, CNN, LR, DT and SDL (Sequential deep learning). A

more in-depth analysis on ensemble models was done by (Subasi et al., 2021), which is using multiple classifiers (SVM, KNN, C4.5 DT, CART Tree, NB Tree) with multiple homogeneous ensembles (Multi Boost, Ada Boost, Bagging, Random Subspace). Likewise, (Sajedi, 2019) comes with an ensemble approach with features extracted from HTML and JS files. The system developed with REP Tree and genetic algorithm as meta-classifier is proposed as a solution. Multiple weak classifiers are compared (NB, SVM, KNN, REP Tree) and multiple ensemble methods (Bagging, Ada Boost, Decorate, Rotation Forest).

3 METHODOLOGY

The proposed approach to malicious web links detection is based on the following main steps:

1. Dataset selection;
2. Preprocessing and data normalization;
3. Divide dataset & handle data imbalancement;
4. Running individual ML models;
5. Running ensemble models & Comparisons.

3.1 Dataset Selection

The dataset used is provided by (Urcuqui et al., 2017). It is a free dataset, with 17 host based and two lexical features already extracted. From the attributes provided by the dataset we can recall: URL characters count, special characters count, server, ports opened for connection, number of IPs connected to the server, WHOIS registration data, WHOIS updated date, WHOIS approximate geo-location, DNS query time, TCP packets count, bytes count etc. The dataset contains a total of 216 malicious samples and 1565 benign. When collecting data for the malicious links problem, there is a recurrent problem for the imbalancement of the dataset, because malicious links are hardly identified in real world scenarios. Another problem could be the datasets containing mostly URLs and not any other properties, extracted when the link still performed its malicious behavior. These related properties, mostly host based attributes, need to be retrieved at a specific time when the link is still active. The dataset provided by (Urcuqui et al., 2017) was chosen because of the presence of host-based features already extracted. Moreover, the empirical approach used in (Coste, 2023) and in the present paper involve running many experiments. Therefore, because of limited computation resources, we chose to first experiment on small data until we properly calibrate the algorithms. The dataset was previously integrated in experiments developed by (Islam et al., 2019) and (Vundavalli et al., 2020).

3.2 Preprocessing and Data Normalization

In the preprocessing step of the dataset, data was cleared, categorical features were transformed to numbers based on two different algorithms (supervised ratio algorithm and weight of evidence algorithm). In total, our model had 23 attributes. The normalization process was computed based on two scalers, MinMax Scaler and Standard Scaler, whose implementation is from Sklearn library (Pedregosa et al., 2011).

3.3 Divide Dataset & Handle Data Imbalancement Problem

When running each algorithm, the dataset was split into 20% of data used for testing (357 samples) and 80% of data for training (1424 records), taking into consideration the ratio between the classes for the samples. In addition, our experiments and the methodology specified in (Coste, 2023), includes some other methods for data imbalancement, such as the usage of specific metrics (precision for the unrepresented class, recall, F1 score, and ROC AUC score), and besides the *stratify* argument, setting the class weight argument to *balanced* for the compatible models.

3.4 Running the Individual ML Models & Comparisons

Our experimental approach is based on the steps presented in (Coste, 2023), which involves mainly two stages of experiments. In the first stage, we start the model configuration from the configuration provided by (Islam et al., 2019), for models RF, DT, KNN, MLP and for the rest of the models, we take as starting point the default configuration available in the Sklearn (Pedregosa et al., 2011). The second stage of the experiments includes the selection of relevant parameters according to the results and running all the possible combinations. Lastly, a ranking is made based on the average of the metrics used (precision, recall, F1, ROC-AUC * 100). For the ranking we decided to take into consideration all metrics, all of them are important, and we computed a mean of them. The best 20 configurations are run 100 times, each time with a different dataset split and then these models

Table 1: The number of configurations for each algorithm for each stage of experiments and the parameters varied.

Model	First stage	Second stage	Parameters calibrated in the second stage
XGB	28,012	235,200	scaler, loss, learning rate, n estimators, min samples split, min samples leaf, max leaf nodes, max depth
ADA	248	14,320	scaler, algorithm, learning rate, n estimators
GNB	210	400	scaler, var smoothing
BNB	640	80,000	scaler, alpha, binarize, fir prior
MLP	1248	21,952	scaler, solver, activation, max iter, hidden layer sizes = [i, j, k] where i, j, k $\in$ {20, 40, ..., 140}
LDA	596	1,164,000	scaler, tol, solver, shrinkage, store covariance, n components, store precision, assume centered
LR	194	313,600	scaler, solver, penalty, max iter, intercept scaling
SVM linear	324	365,568	scaler, max iter, shrinking, C, tol, cache size
SVM poly	269,858	1,520,000	scaler, shrinking, degree, cache size, coef0, gamma
SVM sigmoid	14,409	71,400	scaler, shrinking, gamma, cache size, coef0
SVM rbf	664	237,600	scaler, max iter, shrinking, gamma, C, cache size

are ranked again. Finally, the best parameter calibration is chosen. The algorithms we chose to run individually were: XGB, ADA, LR, NB (Bernoulli and Gaussian), LDA and SVM (with RBF, sigmoid, linear, and polynomial kernel). The number of run configurations for the experimental approach and all parameters calibrated are detailed in Table 1.

3.5 Running Ensemble Models & Comparisons

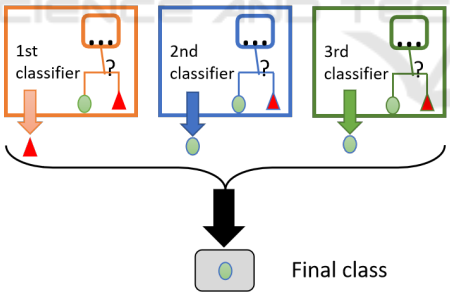


Figure 1: Ensemble model.

The ensemble models are formed out of three models constructed for the best configurations found on the previous models. We included the best configuration for RF, DT and KNN algorithms as computed in (Coste, 2023). For the rest of the algorithms, we chose seven more configurations. Between Bernoulli NB and Gaussian NB, we will choose just the one achieving the highest scores. Moreover, among the four SVM variants, there will be depicted the variant outperforming the rest. The architecture of the ensemble model is presented in Figure 1, where each classifier will output a class and the final annotation is chosen based on the majority of the classifiers. The developed ensemble models for malicious web links detection are based on the approach proposed in (Pakhare et al., 2021). In total, the ensembles will be constructed with 10 ML models, leading to 120 combinations possible.

Finally, we will draw relevant comparisons between our results and the ones obtained by (Islam et al., 2019), (Vundavalli et al., 2020), since all three research studies use the same dataset (Urcuqui et al., 2017). The comparisons are made given the F1 metric.

4 RESULTS AND DISCUSSION

The current section will present the results for each algorithm, the best configuration obtained based on the empirical approach with two stages of experiments. The scores for the best ensembles are presented as well in the present section. Finally, comparisons are drawn, and we discuss our results, placing our solution in the domain literature.

4.1 Computational Experiments

The current subsection will detail the results of each algorithm.

LR manages to reach 74.98% precision score with its best configuration which is parameterized as: *scaler = Min Max, max iter = 154, intercept scaling=100, penalty = l1 and solver = liblinear.*

Results for **NB** algorithms including the **Bernoulli** and **Gaussian** variants can be seen in Table 2. Between these two variants, one can observe that BNB performs better as it is in the case of (Vundavalli et al., 2020). The best configuration from BNB includes

Table 2: Results for the algorithms run individually.

Model	Precision	Recall	F1 score	ROC AUC * 100	Avg. metric
LR - best model	74.99	91.94	82.41	98.57	86.98
BNB - best model	82.75	79.21	80.76	95.3	84.51
GNB - best model	29.49	94.06	43.24	94.72	65.38
ADA - best model	94.19	89.81	91.85	99.45	93.83
XGB - best model	95.07	89.19	91.95	99.41	93.91
LDA - best model	93.93	82.86	87.92	98.47	90.8
SVM rbf - best model	86.67	90.7	88.64	98.56	91.14
SVM linear - best model	75.15	92.55	82.82	98.31	87.21
SVM sigmoid - best model	76.27	88.89	81.97	97.02	86.04
SVM poly - best model	78.06	91.38	84.09	98.04	87.89
MLP - best model	88.73	88.6	88.46	98.62	91.1

scaler= Min Max, fir prior = True, alpha = 1.6 and binarize = 0.42. The best configuration for GNB is obtained with *var smoothing* parameter, *3.93e-09* and Standard scaler. The GNB algorithm is the worst performing algorithm considering our experiments. The explanation for why BNB reaches a better performance than GNB may be because BNB is suitable for large categorical features, while GNB is not recommended in this case according to (Vundavalli et al., 2020).

Ada Boost managed to achieve over 94% precision for its best configuration, being one of the best models (see Figure 3). The *SAMME* algorithm, a *1.01* learning rate and *189* estimators were included in the best parameter configuration.

XGB model obtained the best score metrics, considering precision as it can be observed in Figure 3. The average model is one of the best as seen in Figure 2. The best configuration is obtained with Min-Max Scaler and the rest of the rest of the parameters are *loss=exponential, learning rate=0.82, n estimators=130, subsample=1.0, criterion=friedman mse, min samples split=17, min samples leaf=15, max depth=5, max leaf nodes=15*.

LDA algorithm manages to outperform the others in terms of metric scores (Figure 3). The best model was obtained for *covariance estimator = EmpiricalCovariance(store_precision=False)*, for a *tol = 0.006*, the *lsqr solver*, *None shrinkage*, *0 n components*, *False for store covariance* and scaler = Min Max.

For **SVM**, in table 2 there are all results for SVM variants with **linear**, **sigmoid**, **RBF** and **polynomial** kernels. Considering the best SVM models, SVM with RBF kernel is outperforming the rest, with a 86% precision. Its best configuration resumes to *shrinking = False, cache size = 230, max iter = 1500* reached when normalizing the data with Standard Scaler. Because of its good score metrics, we chose to include the SVM with RBF kernel model in the ensemble models.

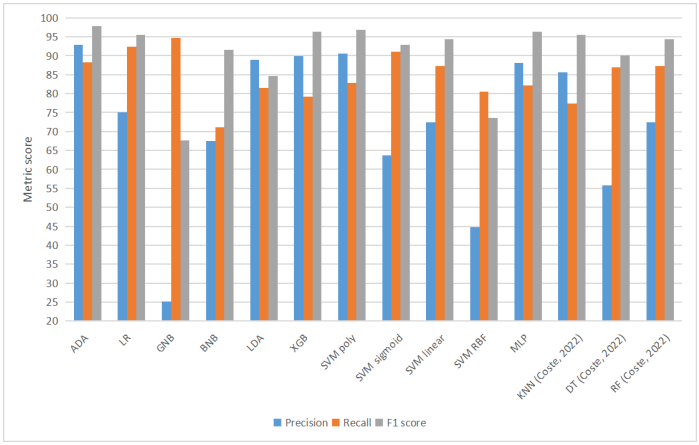
The **MLP** model was the slowest model of all, but its accuracy does not fail, achieving one of the best results. By varying its parameters we have obtained the best configuration with *max iter = 100*, 3 hidden layers with 140, 120, 60 neurons, *activation = identity, solver = tanh* and the rest have default values.

4.2 Comparisons and Discussion

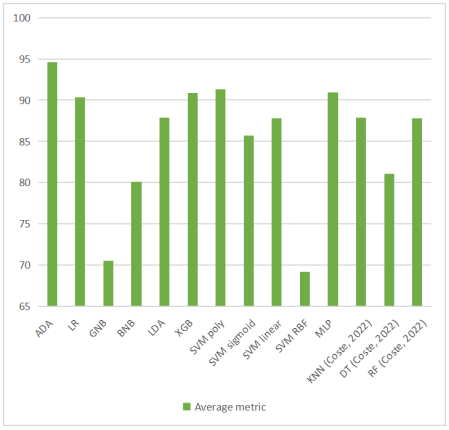
For comparisons we chose to first compare the algorithm individually and then, as ensembles. For comparisons, we depicted the best configurations and how algorithms performed on average. Moreover, our results are compared with other studies that used the same dataset such as (Islam et al., 2019) and (Vundavalli et al., 2020).

In Figure 2, we present the results as the mean of all configurations run of the proposed algorithms for the second stage of experiments, each configuration being run only once. As can be observed in Figure 2b, the best performing algorithm while parameters were calibrated is ADA, with an average metric of 94.56. ADA's performance on average is followed by SVM with polynomial kernel, MLP, XGB and LR.

Figure 3 presents the best configurations obtained. The metrics presented are metrics computed as an average of 100 different dataset splits. Considering the average metric, the best configuration is reached by ADA, XGB and RF developed in (Coste, 2023), all three being promising algorithms. Considering precision, the most precise model is XGB, closely followed by ADA, LDA, MLP, and the RF model from (Coste, 2023). Having a high precision means that the model properly identifies malicious links and has a low rate of FPs. The best recall is observed to be achieved by GNB, considering its average performance and its best one. Unfortunately, GNB has the lowest preci-

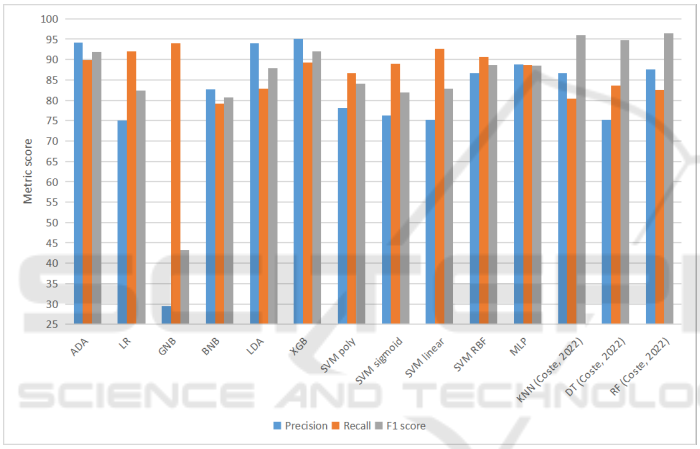


(a) Average models with precision, recall, F1 score.

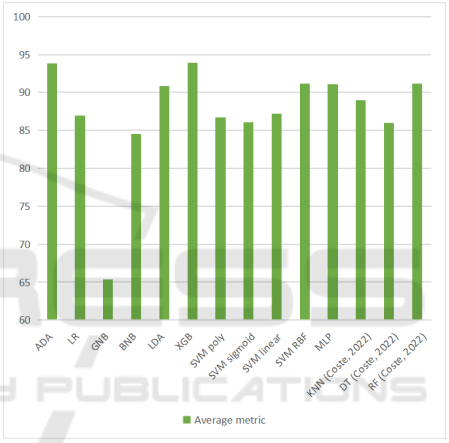


(b) Average models with average metric.

Figure 2: All average configuration results.



(a) Best models with precision, recall, F1 score.



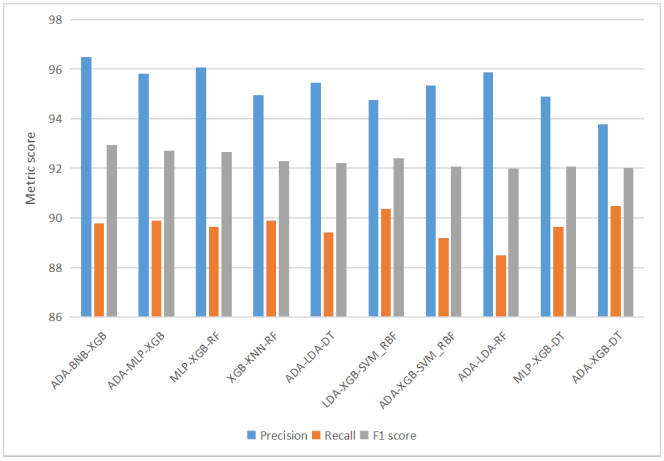
(b) Best models with average metric.

Figure 3: All best configuration results.

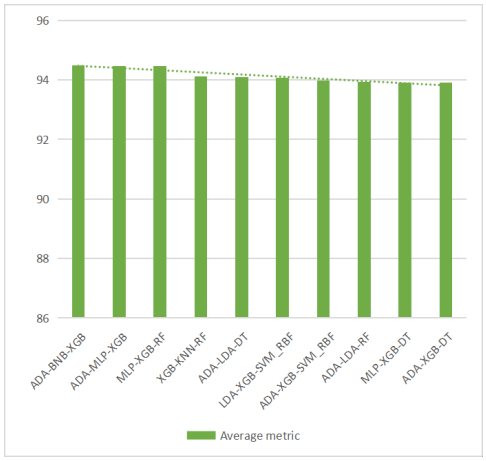
sion score. Having a good recall may denote that there is a low rate of FNs. A high recall can imply a low precision, and this is the case for GNB. Thus, GNB is a good algorithm for predicting benign samples. Since predicting maliciousness of links is a more important problem, GNB is not considered a good model. The best F1 scores were reached by RF, KNN and DT, all models developed in (Coste, 2023). These models are closely followed by XGB, and ADA. A good F1 score indicates good precision and a good recall, by trying to maximize both. The best ROC-AUC scores were obtained by ADA, and XGB, followed by MLP, LR, LDA, RF and SVMs.

By comparing the average performance and the results of the best configuration, one could observe some discrepancies, where for some classifiers the average performance is better than the best performance. This may be due to our running conditions and the way we computed the results. For the average metrics, each parameter configuration was run once, while for the best performance, each configuration was run 100 times.

From NB variants and SVMs models we chose the ones having a better performance (i.e., BNB and SVM with RBF) to further include them in ensembles. The results for the 10 ensemble models are presented in Figure 4. We employ a set of 10 algorithms to construct the heterogeneous model, having three models that participate in the voting system, leading to 120 combinations. Each hybrid configuration was run 100 for each of the scalers used (MinMax or Standard). So, the table presents an average of 200 different data splits. The best performing ensemble was ADA-BNB-XGB and ADA-MLP-XGB, ADA and XGB being the best algorithms when run individually as observed before in Figure 3. In addition, the best configuration when ensembles were run with Standard scaler is ADA-XGB-SVM-rbf model hav-



(a) Top 10 ensemble models with precision, recall, F1 score.



(b) Top 10 ensemble models with average metric.

Figure 4: Top 10 ensemble models (average performance with Min Max and Standard Scaler).

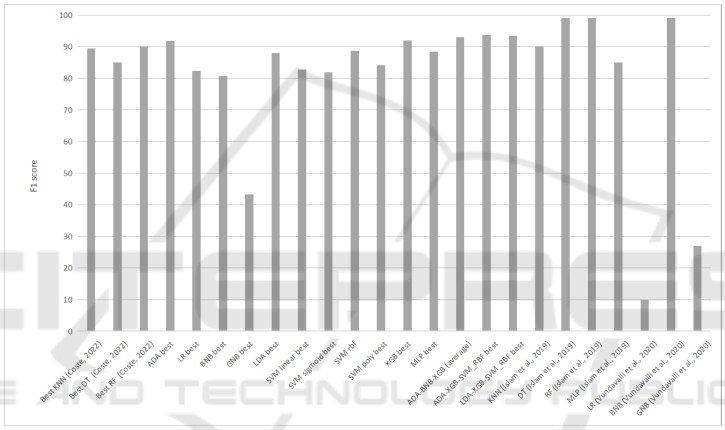


Figure 5: All best models compared with (Islam et al., 2019) and with (Vundavalli et al., 2020).

ing a 95.21 average metric score (96.31% precision, 91.39 % recall, 93.72% F1 score and 0.9941 ROC-AUC score). Considering Min-Max Scaler, the best average metric of 94.91 was obtained by LDA-XGB-SVM-rbf (94.83% precision, 92.09% recall, 93.38% F1 score, 0.9938 ROC-AUC score). For the comparisons, we chose to include three of the best hybrid models: ADA-BNB-XGB, ADA-XGB-SVM-rbf and LDA-XGB-SVM-rbf.

Figure 5 presents a comparison between our models and the models proposed by (Islam et al., 2019) and (Vundavalli et al., 2020). There is a slight improvement between our best model (XGB) and the ensemble models considering the F1 score, which is the common metric between the articles. Comparing the MLP model with the MLP proposed by (Islam et al., 2019), we manage to improve their MLP algorithm with almost 3%. But, even with the best ensemble model (ADA-XGB-SVM rbf) we could not reach their performance for the DT and RF models. These reach 99% F1 score in their experiments, while our best hybrid configuration obtained just 93.73% F1 score. We think that there is more to investigate here and further improve the performance. By comparing our solutions with the ones presented in (Vundavalli et al., 2020), we could observe that LR in our case achieves a considerably higher F1 score. GNB has 27% F1 score in (Vundavalli et al., 2020), while in our case its best configuration has 43.24% F1 score. BNB has an outstanding F1 score of 99% in (Vundavalli et al., 2020) and unfortunately, for our experiments BNB has a modest score of 80.75%, but it is included in one of the best ensembles. The reason for why we have obtained some lower results as opposed to (Islam et al., 2019) and (Vundavalli et al., 2020), may be because we have not used undersampling or oversampling techniques to further counteract the data imbalancement problem. (Islam et al., 2019) implemented k-fold cross validation.

Comparing our hybrid models with the ensembles provided by (Pakhare et al., 2021) is not very relevant since the dataset is different. For the experi-

ments they used: LR, KNN, DT, RF and SVM (linear, sigmoid and RBF kernels). For our experiments we used a larger number of algorithms: LR, LDA, NB (Bernoulli and Gaussian), LDA, XGB, MLP, SVM (with linear, sigmoid, polynomial and RBF kernels) and DT, KNN, RF were taken from (Coste, 2023). Thus, we had 120 different hybrid models, while they used 10 configurations. In their case, the best individual algorithm is not included in the best ensemble model. This is in contrast with our results, which confirm the best model is retrieved in the best ensemble as well.

5 CONCLUSIONS AND FUTURE WORK

Experiments in the malicious web links detection domain are very challenging and complex to develop. We employ the usage of 7 ML algorithms LR, LDA NB (Bernoulli and Gaussian), LDA, XGB, MLP, SVM (with linear, sigmoid, polynomial and RBF kernels), which we calibrate and compare. Moreover, we chose 10 different configurations with which we continue to experiment with hybrid models formed out of three of them. The best ensembles managed to improve the metric scores compared to the single models. Moreover, we observed that the best single model, XGB, was included in the best performing ensembles ADA-BNB-XGB, ADA-XGB-SVM-rbf and LDA-XGB-SVM-rbf. Our best solution has 96.31% precision with ADA-XGB-SVM-rbf hybrid model. In addition, some proposed models such as MLP, LR and GNB manage to improve previous literature results.

Concerning future work, we plan to elaborate further with the detection models and create stacked models constructed from five classifiers and a meta-classifier. Moreover, our methodology can work with multiple datasets, and it would be an idea to experiment in a cross-dataset environment. Moreover, we could elaborate on the data collection process by establishing a centralized dataset with real samples retrieved during a specific period. Additional information regarding links, such as DNS information, WHOIS information, properties about the web server, should be collected in real time.

REFERENCES

Alsaedi, M., Ghaleb, F. A., Saeed, F., Ahmad, J., and Alasli, M. (2022). Cyber threat intelligence-based malicious url detection model using ensemble learning. *Sensors*, 22(9):3373.

Coste, C.-I. (2023). Malicious web links detection - a comparative analysis of machine learning algorithms. *Studia Universitatis Babeș-Bolyai Informatica*, 68(1):21–36.

Fortra (2022). The 2021 gone phishing tournament results: Everything you need to know.

Islam, M., Poudyal, S., Gupta, K. D., et al. (2019). Mapreduce implementation for malicious websites classification. *International Journal of Network Security & Its Applications (IJNSA) Vol*, 11.

Janet, B., Kumar, R. J. A., et al. (2021). Malicious url detection: A comparative study. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, pages 1147–1151, Tamil Nadu, India. IEEE, IEEE.

Johnson, C., Khadka, B., Basnet, R. B., and Doleck, T. (2020). Towards detecting and classifying malicious urls using deep learning. *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, 11(4):31–48.

Naveen, I. N. V. D., Manamohana, K., and Verma, R. (2019). Detection of malicious urls using machine learning techniques. *International Journal of Innovative Technology and Exploring Engineering*, 8(4S2):389–393.

Pakhare, P. S., Krishnan, S., and Charniya, N. N. (2021). Malicious url detection using machine learning and ensemble modeling. In *Computer Networks, Big Data and IoT*, pages 839–850. Springer, Singapore.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sajedi, H. (2019). An ensemble algorithm for discovery of malicious web pages. *International Journal of Information and Computer Security*, 11(3):203–213.

Song, X., Chen, C., Cui, B., and Fu, J. (2020). Malicious javascript detection based on bidirectional lstm model. *Applied Sciences*, 10(10):3440.

Subasi, A., Balfaqih, M., Balfagih, Z., and Alfawwaz, K. (2021). A comparative evaluation of ensemble classifiers for malicious webpage detection. *Procedia Computer Science*, 194:272–279.

Tung, S. P., Wong, K. Y., Kuzminykh, I., Bakhshi, T., and Ghita, B. (2022). Using a machine learning model for malicious url type detection. In Koucheryavy, Y., Balandin, S., and Andreev, S., editors, *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, pages 493–505, Cham. Springer International Publishing.

Urcuqui, C., Navarro, A., Osorio, J., and García, M. (2017). Machine learning classifiers to detect malicious websites. *SSN*, 1950:14–17.

Vundavalli, V., Barsha, F., Masum, M., Shahriar, H., and Haddad, H. (2020). Malicious url detection using supervised machine learning techniques. In *13th International Conference on Security of Information and Networks*, pages 1–6.