ISSN No: 2456-2165 https://doi.org/10.38124/ijisrt/25may1412

Detecting Malicious URLs: A Machine Learning Approach using Feature Engineering and Ensemble Models

Saeed Hubairik Aliyu[1]; Naeem Naseer[2]; Bilal Muhammad[3]

[1,2,3]Muhammad Nawaz Shareef University of Agriculture, Multan

Publication Date: 2025/06/03

Abstract: This work considers the use of machine learning to classify URLs into four categories: benign, defacement, phishing, and malware. In this research, a dataset used contains 651,191 URLs where there are 428,103 benign, 96,457 defacements', 94,111 phishing, and 32,520 malware URLs. For this comparison, three machine learning models were used: Cat Boost classifier, Snapshot Ensemble, and Stacked Ensemble with Snapshots. The Cat Boost classifier was fairly accurate, at about 96%, with previsions ranging from 91% to 97% and recall from 82% to 99%, thus handling class imbalance rather well. Snapshot Ensemble scored an accuracy of about95.83%, thus performing quite great in classification tasks and handling model complexity and generalization effectively. Using Stacked Ensemble with Snapshots resulted in a somewhat-lower accuracy of 91.30% but high-performance variability across the different classes. These results have shown the power of ensemble techniques in enhancing classification performance and solving issues related to class imbalance. Future research should be directed toward the refinement of feature engineering techniques and real-time detection capabilities, focusing on high ethical standards with regard to public, readily available data, further contributing to the development of URL classification and thus to cybersecurity as a whole.

Keywords: *URL Classification, Machine Learning, Cybersecurity, Ensemble Techniques, Phishing Detection, Malware Detection, Class Imbalance, Feature Engineering, Real-Time Detection, Precision and Recall.*

How to Cite: Saeed Hubairik Aliyu; Naeem Naseer; Bilal Muhammad (2025) Detecting Malicious URLs: A Machine Learning Approach using Feature Engineering and Ensemble Models. *International Journal of Innovative Science and Research Technology*,10(5), 2947-2956. https://doi.org/10.38124/ijisrt/25may1412

I. INTRODUCTION

The creation of the Internet, most of the activities in our daily life have become computerized, such as e-commerce, business, social networking, and banking. With this increase of activities carried out online it also brings about the factor of online criminal activities making it even more crucial to protect the WWW. Internet World Stats (Tupsamudre, Singh and Lodha, 2019) stated that there were about 237,418,349 users employing Arabic language on the internet in 2020. Cyber criminals entice the users to click on links which leads to system vulnerability or unauthorized access to privileged information. Therefore, this aspect of online interactions is becoming increasingly important to protect. These protocols and regulations have been put in place to ensure the protection of the link between the client and the server; however, these links are at risk of being attacked by anyone with ill intentions. The term "Malicious" is a general category of different attack such as phishing, spam, malware and etc. URLs are especially perilous because they are employed to obtain unauthorized data and deceive unsuspecting end-users to engage in scams, causing billions of dollars' worth of losses per year.

In the given context, the online security community has devised blacklisting services that enable the detection of dangerous sites. A blacklist is a list of URLs that are considered to be toxic. URL blacklisting has been termed as a success in some instances (Sheng *et al.*, 2009). Nonetheless, the attackers can manipulate this system whereby they make slight changes on one or more units of the URL string and end up not being detected. Therefore, the majority of hostile sites remain undetected due to their novelty, lack of analysis or incorrect assessment. Another method of detecting the malicious site is the heuristic method Heuristic approach is another method of detecting malicious sites. This improved version of the method of list of blacklists uses signatures to match the new URLs with the known URLs of the malicious kind. Although these approaches can detect both the bad and the good URLs, they have their drawbacks. The third approach to detecting malicious sites is based on machine learning (ML) and deep learning (DL) techniques of artificial intelligence. AI methods have been applied in numerous domains such as cybersecurity, healthcare, medical image analysis, e-commerce, and social networks. In general, for the cybersecurity domain, it is possible to develop models that would use previous experience and improve their self-learning ability without human interference. Thus, this property is particularly useful for large organizations, companies, banks and other structures. Furthermore, ML and DL have been proven to be effective in many fields and are often used to identify dangerous sites(Aalla, Dumpala and Eliazer, 2021). 8The advantages of using ML for recognizing the malicious URL consist in its capability to recognize newly developed URLs and the model update. Some current researches have examined DL models that employ methods for identifying newly created URLs and extracting features. From URLs, several features can be obtained and used by ML algorithms to determine whether a given URL is malicious or not. The lexical, content based and network-based features are the most common one's which are extracted from URL's. The literature search involved articles published from 2012 to 2021 that used either ML or DL for the classification of malicious URLs. The contents of the websites were then categorized as either Arabic language or English language. Based on the aspects such as language, URL features, ML techniques and datasets used, we present a categorisation of the studies reviewed for detection of malicious URLs(Aljabri, Altamimi, *et al.*, 2022a).

II. RESEARCH METHOD AND PROCEDURE

Malicious URLs have been an important part of cybersecurity, protecting users against phishing attacks, malware, and other dangers in cyberspace. This chapter describes the research methodology that was adopted to develop a robust and efficient system for the detection of malicious URLs using the machine learning approach. The methodology represents the stages of data collection, feature engineering, model selection, and evaluation. The aim of this research is to enhance the accuracy and reliability of malicious URL detection systems through the integration of feature engineering and ensemble modeling techniques. In what follows, procedures and methods used in this research are presented in detail to obtain the targeted results.

➢ *Data Set Description*

Malicious URLs pose a great threat to cybersecurity, as they host unsolicited content, including spam, phishing attempts, and drive-by downloads. These websites will lure the user into being a victim of scams, leading to financial losses and leak private information into possible malware installation that damages several billion dollars' worth annually. The threat is countered with a developed dataset that can facilitate machine learning models to recognize and block malicious URLs before they do any harm The dataset consists of 651,191 URLs, with four classes: 428,103 are benign, 96,457 defacement URLs, 94,111 phishing URLs, and 32,520 malware URLs. Figure 2 indicates the percentage distribution of these URLs.

➢ *Data Source*

The ISCX-URL-2016 dataset is used in the collection of URLs pertaining to benign, phishing, malware, and defacement sites. The Malware Domain Blacklist dataset is used to complement the dataset with more phishing URLs and malware URLs. Faizan Git Repository provides more URLs, which are benign. Second, Phishtank Dataset contributed more phishing URLs, while the PhishStorm Dataset added further URLs that were related to phishing. The data collection procedure involved the aggregation of URLs from these sources into different data frames, and afterwards merging them to retain only the URLs with their class types. The result will present a standard dataset for training and testing machine learning models used in detecting malicious URLs.

➢ *Data Analysis*

This is a dataset of malicious URLs with 651,191 URLs and four different classes: benign, defacement, phishing, and malware. The concrete composition includes 428,103 benign URLs, 96,457 defacement URLs, 94,111 phishing URLs, and 32,520 malware URLs. This distribution 19reveals the dominant percentage of benign URLs, about 65.7% in the dataset, while the rest include defacement URLs, which account for 14.8%, phishing URLs, accounting for 14.4%, and malware URLs accounting for 5%. The dataset was created, from a number of sources, in a way that is heterogeneous and representative in each category. The ISCX-URL-2016 dataset was used as the base for collecting benign, phishing, malware, and defacement URLs. To make the dataset more robust, more phishing and malware URLs were added from the Malware Domain Blacklist dataset. Additional benign URLs were supplemented with data from our Git repository. In order to focus on the category of phishing in particular, supplemental data were drawn from both Phishtank and PhishStorm datasets. Each URL is labeled, therefore, allowing a model to be trained and subsequently evaluated in a supervised learning approach.

The large size and class balance of malicious URLs increase the utility of the dataset for developing machine learning models that could accurately detect and classify malicious URLs.Analysis of this distribution shows that it is an imbalanced dataset, with a huge majority being benign URLs, thus proper techniques such as oversampling, undersampling, or use of class weights should be employed to ensure that the machine learning model learns effectively to tell the difference between the classes. The variation across sources suggests variations in URL structures and content, hence increasing complexity involved in feature engineering. The completeness of this dataset helps the researcher or practitioner engaged in cybersecurity to form a ground for developing advanced machine learning models that can preemptively identify and mitigate the risks by malicious URLs. The analysis of this dataset shall be in respect to the understanding of characteristics and patterns associated with every URL category, using statistical and machine learning techniques for meaningful features that enable predictive accuracy. This will entail cutting down on the structure and composition of the dataset to enable insight into the prevalence and nature of the various cyber threats that aid in effective and proactive cybersecurity measures. The overall aim is to use machine learning to develop a very strong system for detection, protecting the user from this ever-present malicious URL threat and saving sensitive information to prevent financial losses. With the comprehensive scope and fine-grained labeling of this dataset, it is only the right candidate to go through rigorous analysis and model development for surefire advances in the domain of cyber security threat detection.

➢ *Evaluation of Ensemble Method*

Evaluation is checking on their performance based on how well they are able to combine the advantages of individual models to predict accurately and robustly. Since an ensemble modelbe it bagging, boosting, or stacking essentially aggregates the predictions of many base models to improve performance on tasks that no single model could do, what follows is an end-to-end approach toward the evaluation of ensemble models: In that respect, evaluation of the model would involve a proper look at different performance metrics with an understanding of how well such an ensemble model in classification tasks performs. Performance metrics provide insights concerning different aspects of model effectiveness and can guide improvements or adjustments.

One of the most straightforward metrics is accuracy, which evaluates the ratio of correctly classified instances against the total count of instances. This will give the general feeling of how often the model's predictions actually match the real outcomes. It is an important metric, but it doesn't capture model performance in class imbalance cases explicitly all alone.

Imbalanced classes. Precision provides the proportion of the true positive predictions against all positive predictions taken up by the model. In simple words, it answers the question: Of all instances predicted as positive, how many actually are? This measure is very important when the cost of false positives is high, like in medical diagnosis where false positives are wrongly treated.

- Recall, or sensitivity, measures the proportion of real positives against all positive instances. It measures the model's ability to correctly identify positive instances out of the total number of actual positives. This is useful, especially in domains where missing positives examples may be very costly, such as fraud detection or outbreaks of disease.

- The F1-score is literally a balance of these two metrics in one score. It is the harmonic mean of precision and recall and can be useful when working on datasets with class imbalance and a need to minimize false positives and false negatives at the same time. The F1 score gives a balanced measure of a model's performance by considering both precision and recall in its calculation.

- The most important metric to measure model quality for class differentiation is the ROC-AUC. The latter plots the true positive rate (i.e., recall) against the false positive 28rate at different threshold settings, and AUC measures the area under this curve. Clearly, AUC ranges from 0 to 1, and values closer to 1 indicate better model performance. A higher AUC means that, for any given threshold, the model is more able to discriminate between positive and negative classes.

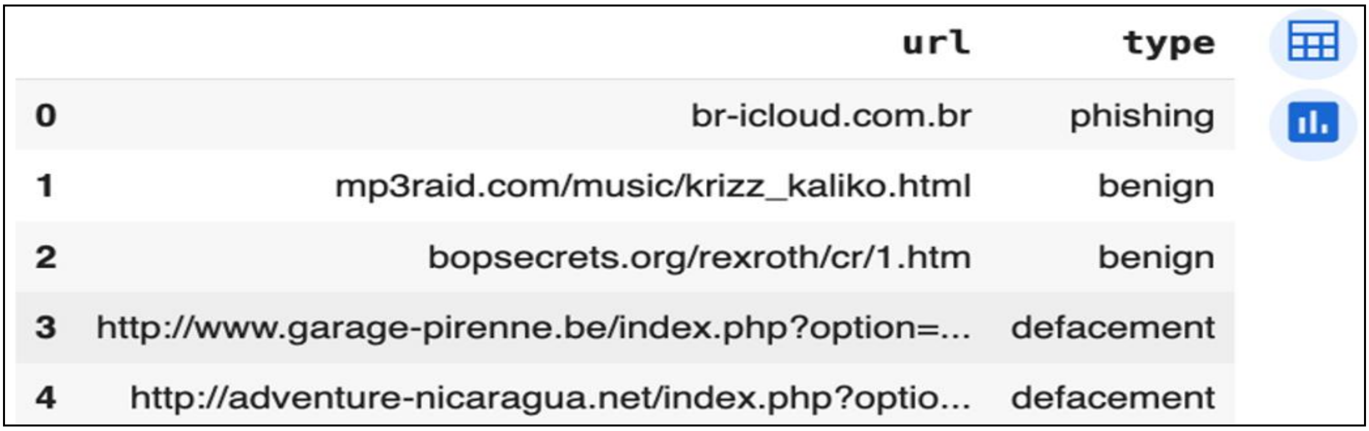
- The Confusion Matrix, finally, provides a detailed record of how the model is performing by showing how many are the true positives, true negatives, false positives, and the false negatives. It gives insights into the type of errors that the model is making and thus helps to know which classes are misclassified. For instance, it may show that the model systematically confuses one class for another, which could be very important while debugging or fine-tuning the model.

These metrics, in their entirety, with regard to accuracy, precision, recall, F1-Score, ROC AUC, and the confusion matrix, are important in forming a full evaluation when considering ensemble models on classification tasks. They give insight into the general correctness of a model but also its effectiveness in many other different aspects, hence giving a nuanced view of performance across scenarios.

III. RESULT AND DISCUSSION

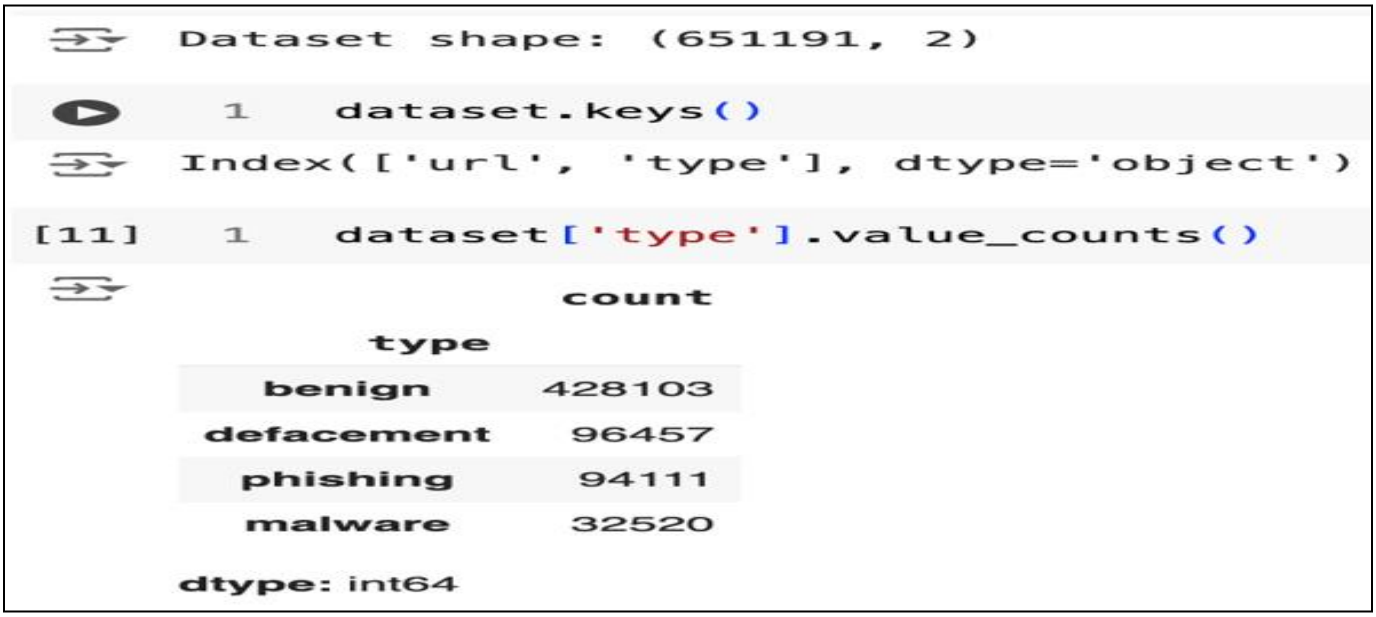
Our experiments were related to evaluating the effectiveness of various features extracted from URL datasets and their impact on classification performance. In this paper, several ensemble techniques such as random forest, gradient boosting, and stacking ensembles have been tested to prove their effectiveness in classifying a given URL as benign or malicious. The results underline the strengths as well as weaknesses of each model, providing valuable knowledge on real-world usage in enhancing web security. Drawing from this work on performance metrics accuracy, precision, recall, and F1 score we detail how our approach improves detection of malicious URLs for more robust measures of cybersecurity.

The working dataset contains 651,191 records, which include the two most significant features: 'url' and 'type'. Now, for each URL, there exists a column where each will be classified into any of these four categories: 'benign', 'defacement', 'phishing', and 'malware'. Such distribution is represented by 'benign' URLs, which are 428,103; 'defacement' URLs amount to 96,457; 'phishing' URLs come to a total of 94,111; and lastly, 'malware' URLs add up to 32,520 entries. It provides a fair view of the kinds of URLs present in the dataset, hence essential for model training and evaluation to ensure accurate classification of URL types.



	url	type
0	br-icloud.com.br	phishing
1	mp3raid.com/music/krizz_kaliko.html	benign
2	bopsecrets.org/rexroth/cr/1.htm	benign
3	http://www.garage-pirenne.be/index.php?option=...	defacement
4	http://adventure-nicaragua.net/index.php?optio...	defacement

Fig 1 Dataset First 5 Rows



```
Dataset shape: (651191, 2)

    1   dataset.keys()
Index(['url', 'type'], dtype='object')

[11]  1   dataset['type'].value_counts()

                 count
         type
       benign   428103
   defacement    96457
     phishing    94111
      malware    32520

dtype: int64
```

Fig 2 Analysis of Results

In the distribution of URL types in the dataset, what is immediately recognizable is that most entries are for Type 0; in fact, it tops the list with more than 400,000 entries. That dominant presence is contrasted by URL Type 1 and URL Type 3, both with a count of about 100,000 each, therefore showing some reasonable level of representation. When compared to these, URL Type 2 represents the smallest count among all types and hence is grossly underrepresented. This clearly shows a strongly one-tailed distribution with URL Type 0 dominating the dataset. This kind of distribution would suggest that although it contains a variety of URL types, most of the entries in a dataset pertain to URL Type 0; if proper care is not taken, this will likely impact the performance and generalization of machine learning models.

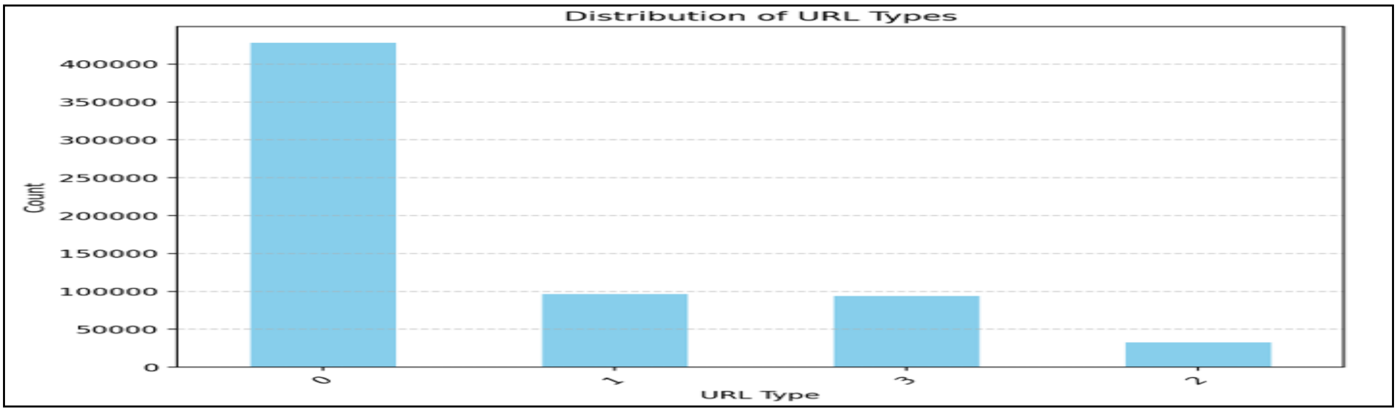
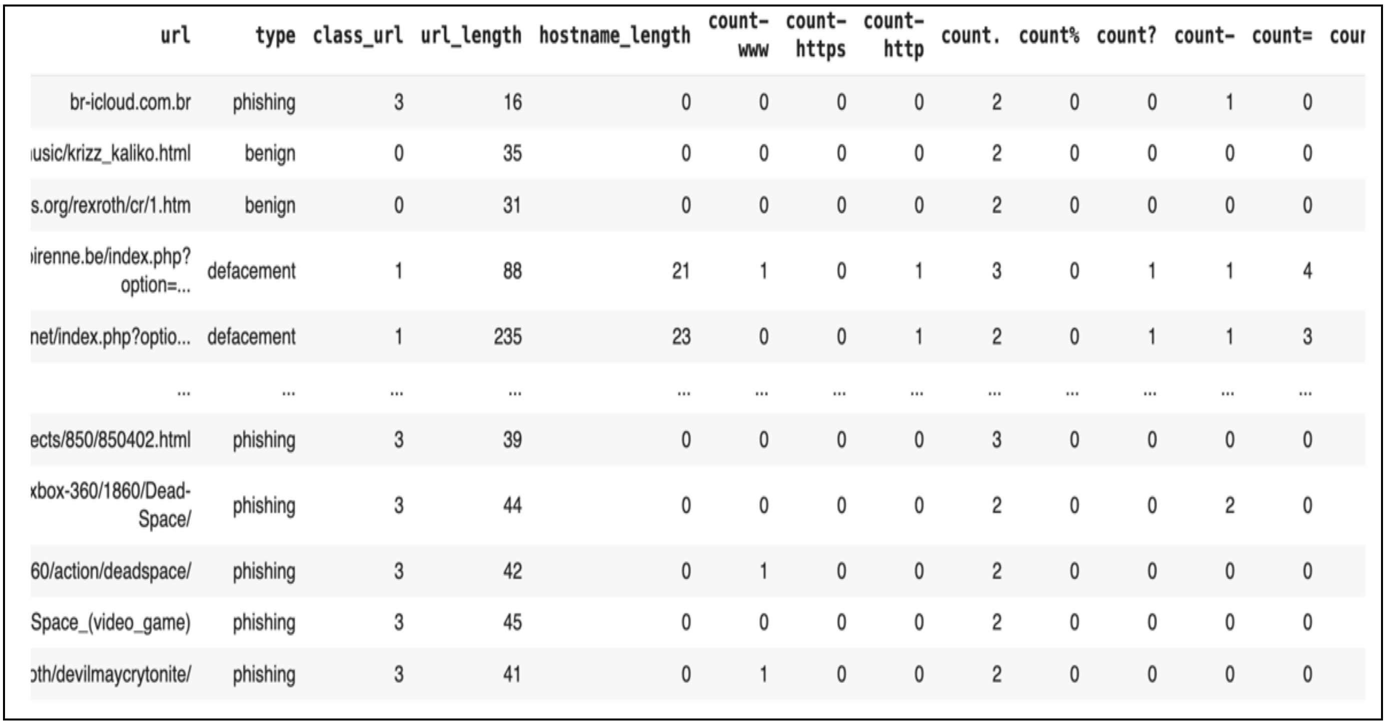


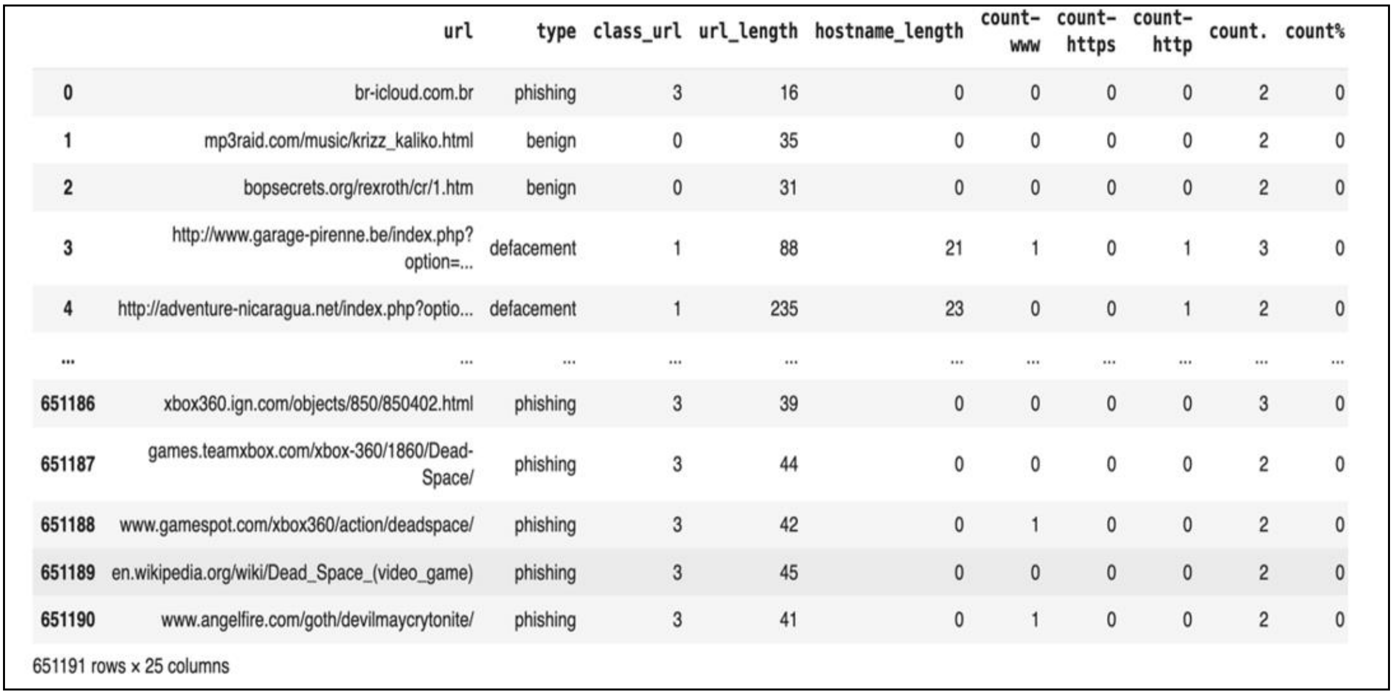
Fig 3 Distribution of URL Types

ISSN No: 2456-2165 https://doi.org/10.38124/ijisrt/25may1412



url	type	class_url	url_length	hostname_length	count-www	count-https	count-http	count.	count%	count?	count-	count=	cour
br-icloud.com.br	phishing	3	16	0	0	0	0	2	0	0	1	0	
usic/krizz_kaliko.html	benign	0	35	0	0	0	0	2	0	0	0	0	
s.org/rexroth/cr/1.htm	benign	0	31	0	0	0	0	2	0	0	0	0	
irenne.be/index.php? option=...	defacement	1	88	21	1	0	1	3	0	1	1	4	
net/index.php?optio...	defacement	1	235	23	0	0	1	2	0	1	1	3	
...	...	...	...	...	...	...	...	...	...	...	...	...	
ects/850/850402.html	phishing	3	39	0	0	0	0	3	0	0	0	0	
xbox-360/1860/Dead-Space/	phishing	3	44	0	0	0	0	2	0	0	2	0	
60/action/deadspace/	phishing	3	42	0	1	0	0	2	0	0	0	0	
Space_(video_game)	phishing	3	45	0	0	0	0	2	0	0	0	0	
oth/devilmaycrytonite/	phishing	3	41	0	1	0	0	2	0	0	0	0	

Fig 4 Results of Lexical Analysis: Faction for Creating features from URL



	url	type	class_url	url_length	hostname_length	count-www	count-https	count-http	count.	count%
0	br-icloud.com.br	phishing	3	16	0	0	0	0	2	0
1	mp3raid.com/music/krizz_kaliko.html	benign	0	35	0	0	0	0	2	0
2	bopsecrets.org/rexroth/cr/1.htm	benign	0	31	0	0	0	0	2	0
3	http://www.garage-pirenne.be/index.php? option=...	defacement	1	88	21	1	0	1	3	0
4	http://adventure-nicaragua.net/index.php?optio...	defacement	1	235	23	0	0	1	2	0
...	...	...	...	...	...	...	...	...	...	...
651186	xbox360.ign.com/objects/850/850402.html	phishing	3	39	0	0	0	0	3	0
651187	games.teamxbox.com/xbox-360/1860/Dead-Space/	phishing	3	44	0	0	0	0	2	0
651188	www.gamespot.com/xbox360/action/deadspace/	phishing	3	42	0	1	0	0	2	0
651189	en.wikipedia.org/wiki/Dead_Space_(video_game)	phishing	3	45	0	0	0	0	2	0
651190	www.angelfire.com/goth/devilmaycrytonite/	phishing	3	41	0	1	0	0	2	0

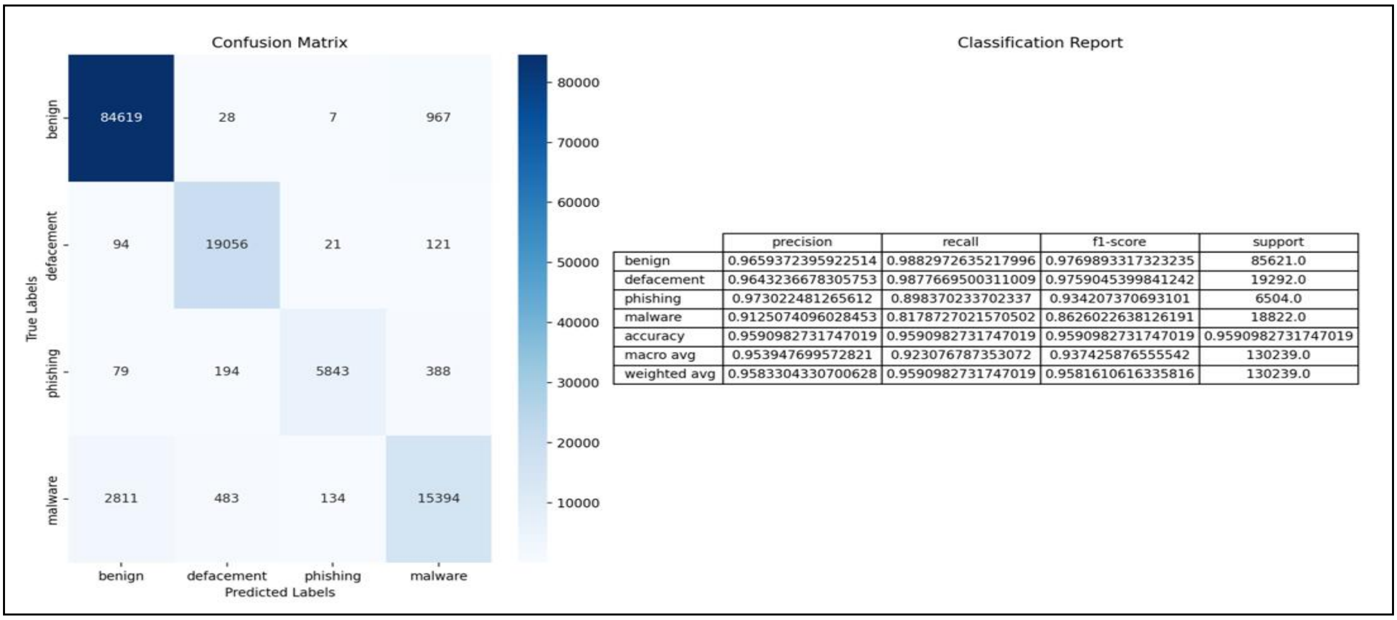
651191 rows × 25 columns

Fig 5 Result of Heuristic Feature Engineering

Because IEEE will do the final formatting of your paper, you do not need to position figures and tables at the top and bottom of each column. Large figures and tables may span.

➤ *Result of Ensemble Machine Learning Techniques*

Results from the CatBoost Classifier turn out very outstanding with respect to most of the evaluation metrics. The confusion matrix indicates that the model efficiently maps URLs into their corresponding classes; most of the misclassifications are between classes 'benign' (0) and 'malware' (3). The classification report contains an overall accuracy of about 96%, with precisions for the different classes from 91% to 97%, and recalls from 82% to 99%. The more balanced F1 score ranged from 0.86 to 0.98 for various classes, thereby showing its robustness in the correct identification of URLs of different types. The macro and weighted averages of precision, recall, and F1 score reinforce the classifier's effectiveness in establishing a well. rounded performance that will sensibly deal with class imbalances and achieve reliable results in URL classification.



	precision	recall	f1-score	support
benign	0.9659372395922514	0.9882972635217996	0.9769893317323235	85621.0
defacement	0.9643236678305753	0.9877669500311009	0.9759045399841242	19292.0
phishing	0.973022481265612	0.898370233702337	0.934207370693101	6504.0
malware	0.9125074096028453	0.8178727021570502	0.8626022638126191	18822.0
accuracy	0.9590982731747019	0.9590982731747019	0.9590982731747019	0.9590982731747019
macro avg	0.953947699572821	0.923076787353072	0.937425876555542	130239.0
weighted avg	0.9583304330700628	0.9590982731747019	0.9581610616335816	130239.0

Fig 6 Results of Cat Boost Classifier

The above-mentioned code trains several instances of the CatBoostClassifier to build an ensemble as snapshots. Concretely, 5 models are trained with 200 iterations each. In the initialization, the CatBoostClassifier is run with a different random seed for each snapshot to 33add some variability between the models. After training, it makes a prediction with each model on the test set and averages the predicted probabilities to obtain the result. These are further converted to get the final class labels for the ensemble prediction. Model accuracy follows a wave-like pattern: first, it improves with the number of snapshots, then it deteriorates, and later improves again. One may therefore believe that this very good performance by the Snapshot Ensemble is sensitive to the number of models involved. Under these circumstances, peak accuracy with 2 snapshots would then imply that this configuration keeps model diversity versus complexity in a balance that gives good performance without overfitting. The probably initial rise in accuracy could result from improved generalization because of increased ensemble diversity. However, accuracy tumbles down with increasing snapshots, probably due to the development of too complex a model that overfits. It finally rises again, indicating that some level of complexity in the ensemble helps in the capturing of the underlying patterns in the data effectively. The snapshot ensemble is effective overall, with the number of snapshots tuned to avoid overfitting into optimum performance.

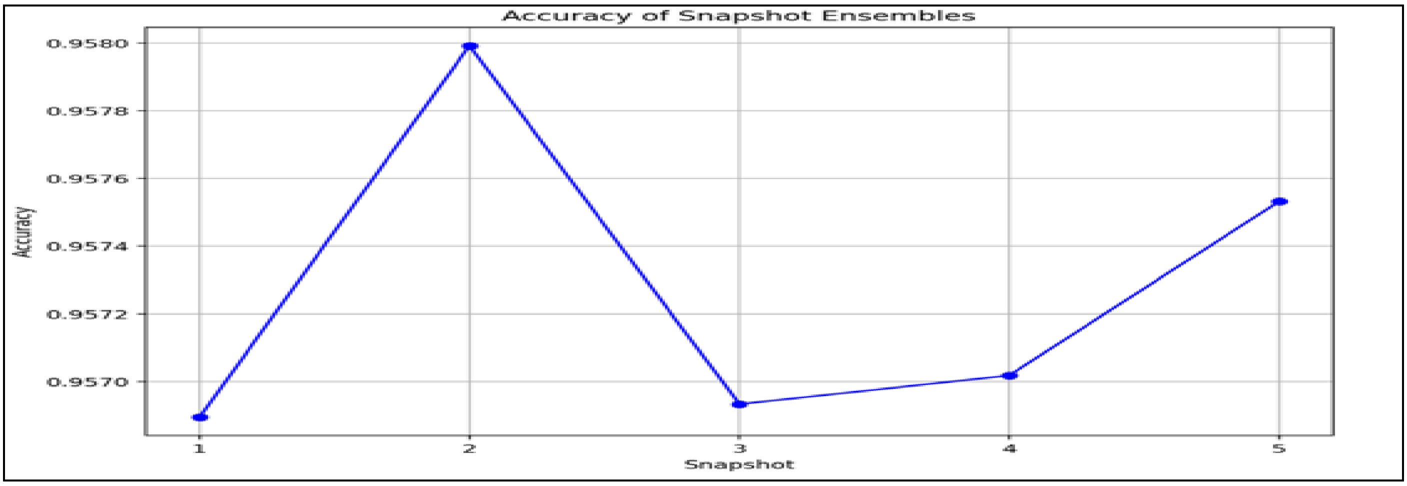
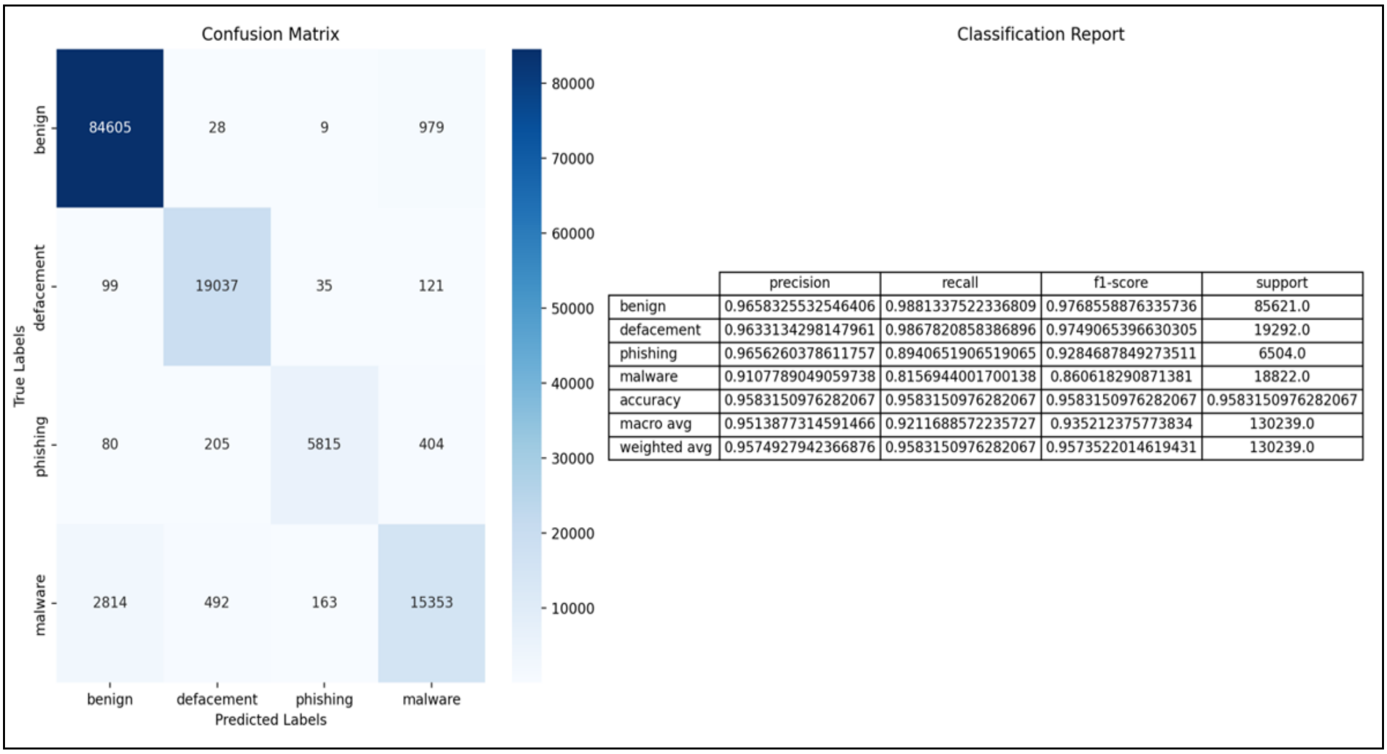


Fig 7 Accuracy of Snapshots Ensembles

The confusion matrix reveals that, all in all, the performance of the model is good, with accuracy as high as 95.83%. The diagonals in the matrix relate to the correctly classified instances, such as 84,405 of the benign instances identified correctly as true positives. Off diagonal elements of the matrix correspond to misclassifications: for example, 28 benign classifieds as defacement. It contains the following performance metrics: precision, which estimates a model's ability to correctly identify positive instances; recall, which is a model's ability to find all positive instances; and F1-score, which balances precision and recall into one number. The macro average will return an unweighted average of metrics per class, while the weighted average will balance these metrics by class frequencies.



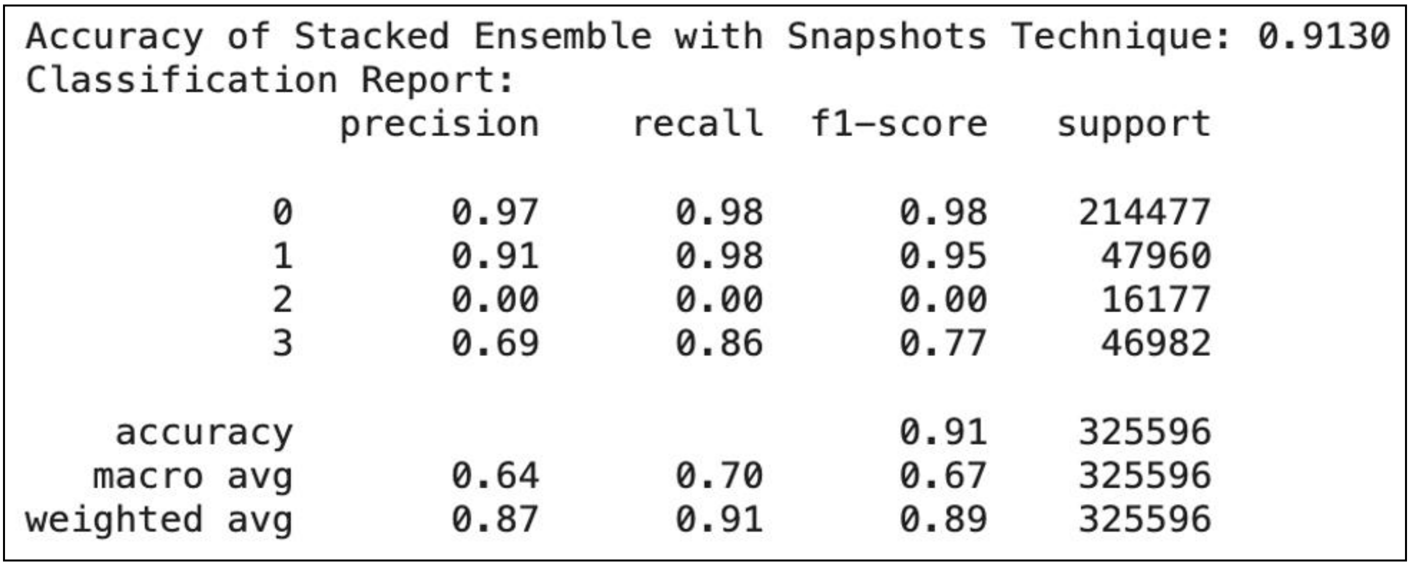
	benign	defacement	phishing	malware
benign	84605	28	9	979
defacement	99	19037	35	121
phishing	80	205	5815	404
malware	2814	492	163	15353

	precision	recall	f1-score	support
benign	0.9658325532546406	0.9881337522336809	0.9768558876335736	85621.0
defacement	0.9633134298147961	0.9867820858386896	0.9749065396630305	19292.0
phishing	0.9656260378611757	0.8940651906519065	0.9284687849273511	6504.0
malware	0.9107789049059738	0.8156944001700138	0.860618290871381	18822.0
accuracy	0.9583150976282067	0.9583150976282067	0.9583150976282067	0.9583150976282067
macro avg	0.9513877314591466	0.9211688572235727	0.935212375773834	130239.0
weighted avg	0.9574927942366876	0.9583150976282067	0.9573522014619431	130239.0

Fig 8 Confusion Matrix and Classification Report of Snapshots Approach

The high accuracy indicates excellent performance in general, but class balance issues the benign instances are most prevalent—may impact model performance appraisal. In other words, the model is very good for the classification of the benign class, leaving room for improvement in the defacement, phishing, and malware classes. For these latter classes, it was significantly low in recall. A confusion matrix and classification report can help highlight areas that might be particularly appropriate to focus on in enhancing the model to cut down on false positives and negatives.

Results using Stacked Ensemble with Snapshots Technique: The performance for the different classes is mixed. The model performs to an accuracy of 91.30%, hence performing very well in general. It does very well on class 0 with high precision and recall, hence high F1-score, showing that it classifies that class effectively.


```
Accuracy of Stacked Ensemble with Snapshots Technique: 0.9130
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.98      0.98    214477
           1       0.91      0.98      0.95     47960
           2       0.00      0.00      0.00     16177
           3       0.69      0.86      0.77     46982

    accuracy                           0.91    325596
   macro avg       0.64      0.70      0.67    325596
weighted avg       0.87      0.91      0.89    325596
```

Fig 9 Classification. Report of Stacking Techniques

Class 1 also turns out to be a little less effective than class 0, evidenced by its high precision and recall but lower F1-score. In contrast, the model performs very poorly on class 2 with a precision, recall, and F1-score of 0.00, thus class 2 is poorly predicted. Class 3 demonstrates medium performance with quite a high recall but lower precision and

F1-score. These macro average metrics reflect this variability, with lower values indicating the challenges the model had with less frequent or more complex classes.

The table clearly contrasts the accuracy obtained by three different machine learning techniques run for URL classification: Cat Boost Classifier, Snapshot Ensemble, and Stacked Ensemble with Snapshots. The Cat Boost Classifier has the highest accuracy at about 96%, while the Snapshot Ensemble was very close at about 95.83%. On the other end, the Stacked Ensemble with Snapshots trailed at 91.30%. High accuracy from the Cat Boost Classifier shows this classifier's effectiveness in correctly classifying URLs into their respective categories. This can be because Cat Boost works efficiently on categorical features and does not result in overfitting. It also provides quite strong results for imbalanced datasets. The way the classifier works to avoid overfitting with ordered boosting, coupled with random permutations of the dataset, helps in gaining better generalization. Consequently, this model has very high precision and recall for most classes; hence, reliable URL classification. On the other side, the snapshot ensemble, as high as about 95.83% in accuracy, turns out to be quite considerable in its effect. Snapshot Ensembles use multiple models that capture different aspects of data through diverse snapshots, all helping to improve prediction accuracy. One of the ensemble methods reduces overfitting by predicting with models trained on different initial conditions or on different parts of the training data. This introduces diversity that enables the model to generalize well to new, unseen data. However, performance is slightly lower compared with Cat Boost Classifier, maybe due to all the balancing involved in achieving this diversity and having model accuracy. Moreover, this effectiveness depends on the number of snapshots, which has to be cautiously tuned since increased complexity deteriorates performance. The Stacked Ensemble with Snapshots is more modest, having an accuracy of 91.30%. Stacking ensembles try to improve predictive accuracy by combining multiple models and using a meta-model for final prediction. The Stacked Ensemble with Snapshots method further ensembles all snapshot models. Although this technique is still effective in general, it does show quite a large drop in accuracy compared to the Cat Boost Classifier and the Snapshot Ensemble. This could be a result of a variety of factors, including increased model complexity and potential optimization challenges for combining the base models. Poor performance on categories like Phishing suggests that the stacked model has a very high chance of performing poorly on less represented classes, which is exactly what the challenge is in obtaining balanced performance across all categories.

Table 1 Accuracy

Technique	Accuracy
Catboost Classifier	96%
Snapshot Ensemble	95.83%
Stack Ensemble	91.30%

Accuracy results show that the choice of the right technique of machine learning should be based on dataset characteristics and classification goals. The Cat Boost Classifier remains a strong choice for high accuracy and very balanced performance across different URL types. Advanced handling of categorical data and the capacity for reducing overfitting make the Cat Boost Classifier quite suitable for complex classification tasks with imbalanced datasets. The snapshot ensemble itself is only slightly less accurate than cross-validation and hence makes for a powerful alternative, especially in a situation when one is interested in harnessing the power of multiple diverse models for better generalization. Its performance is an indication of its strong capacity to yield accurate predictions, provided the complexity of the ensemble is effectively managed. Lastly, while providing no comparative accuracy to other solutions, the Stacked Ensemble with Snapshots retains the advantage of using a multi-model approach to improve the quality of the classification. On the other hand, it requires very careful tuning and optimization so as not to lose in performance because of the added complexity in handling very diverse classes. These benchmarked techniques therefore connote the fact that, for model selection, careful consideration is required on a dataset and classification requirements. The Cat Boost Classifier having the highest accuracy shows its worth as it is definitely a very handy tool in URL classification. The snapshot ensemble and the stacked ensemble with snapshots give insight into the benefits and challenges of ensemble learning approaches.

IV. CONCLUSION

In particular, URL classification that is, identifying and classifying a URL into classes such as benign, defacement, phishing, and malware has made immense progress in the recent past years. A number of studies have contributed a variety of algorithms and methodologies that used unique datasets and lexical features to realize very high accuracies. Comparing these already existent studies with the results from CatBoost Classifier, Snapshot Ensemble, and Stacked Ensemble with Snapshots, several insights and comparisons can be derived that help put into perspective the efficacy and limitations of these techniques more elaborately within the context of URL classification research. The CatBoost Classifier also turned in quite a decent performance, with its accuracy at about 96%. This classifier's high efficacy is due mostly to its advanced handling of categorical data, the ability to mitigate overfitting, and ordered boosting and random permutations that prop up generalization. This performance compares favorably with some of the highest accuracies ever reported in the literature. For instance, Cui et al. (2018) achieved an accuracy of 99.89% using Support Vector Machine with 22 lexical features, thus proving that detailed feature engineering could lead to heightened performance. Afzal et al. (2021) created k-means clustering on Phish Tank and Kaggle datasets to arrive at an accuracy as high as 99.70%. These pieces of research show a strong role of feature selection and its engineering in improving classifier performance this shared strength in Cat Boost's approach. Snapshot assembling retains an accuracy of about 95.83%. This very high strength is a hallmark of how ensemble learning generally imparts to models with respect to both generalization and performance. The ensemble methods, like snapshot enfeebling, leverage diversity across a number of models to capture the different aspects of data, hence improving predictive accuracy. It is almost as good as the Cat Boost Classifier, probably due to the balancing act between ensemble diversity and model accuracy. In fact, different ensemble methods have been accurate in other studies. For example, Aljabri et al. applied the Majority Voting-based Classifier on UNB and Kaggle datasets and reported an excellent accuracy of 99.72% for the 2022b study, having 47 lexical features. Different ensemble methods succeeded in different studies, proving their capability to enhance robustness and accuracy for the classification of URLs.

The Stacked Ensemble with Snapshots realizes more moderate performance, achieving an accuracy of 91.30%. Stacking ensembles are designed to improve predictive accuracy by combining multiple base models and then making final predictions using a meta-model. The integration of the snapshot models in this ensemble is for further improving performance. While this model was more accurate than Gradient Boosting, Cat Boost, and Snapshot Ensemble, its results were less accurate than those of Cat Boost and Snapshot Ensemble alone. 39These results suggest it had problems optimizing the combination of base models and dealing with less well-represented classes. Large performance differences across different URL categories, but mainly poor performance on phishing URLs, suggest likely problems of the stacked model with imbalanced datasets or complex class distributions. This challenge is echoed in studies like Joshi et al. (2019), wherein the use of a Random Forest classifier reached an accuracy of 92% on Open Phish, Alexa whitelists, and FireEye datasets. Although the accuracies reported are high, they reduce drastically over all URL categories, thereby posing the challenge in maintaining high accuracy. Accuracies that are reported in URL classification research differ to a great extent due to the varying algorithms, datasets, and features used in the studies. For instance, Yuan et al. applied in 2018 XG Boost and reached an accuracy of 99.69% on the Alexa, Phish Tank, and Reasonable Anti-phishing datasets. It is also shown that high performance with gradient boosting algorithms like XG Boost exists in Cat Boost, corresponding to the successes in stating the efficacy of boosting techniques on complex classification tasks. On the contrary, simpler models like Logistic Regression and Decision Trees, used by Vanitha and Vinodhini, 2019a, and Aalla et al., 2021, have recorded accuracies of 98% and 97.50%, respectively. This means that even less complex models can be pretty competitive if equipped with appropriate feature engineering and dataset selection. Feature engineering at the lexical feature level is very important for the success of any URL classification model. Studies, such as the one done by Raja, Vinodini, and Kavitha, which used 20 lexical features on the UNB dataset and achieved an accuracy of 99%, further underline the importance of extracting meaningful features from URLs. On the other hand, Johnson et al. used 78 lexical features with a Random Forest classifier on the ISCX-URL-2016 dataset and obtained an accuracy of 99%. The high accuracies that most of the studies reported underline the key role of detailed and relevant feature extraction in improving the model's performance. This also goes hand in hand with the robust handling of features by Cat Boost Classifier, which stipulates that complex feature engineering combined with advanced algorithms is enough to drive high classification accuracies. However, the accuracy reported in most of the studies could, to a great extent, be explained by diversity in the used datasets. We have hugely used datasets such as Phish Tank and Common Crawl, along with a number of private datasets, all of which have different challenges and distributions. For example, Shivangi et al. combined Phish Tank and Common Crawl datasets with LSTM models to obtain an accuracy of 96.89%. Zhao et al. applied, in 2019, deep learning models such as LSTM and GRU on datasets derived from a Chinese Internet security company and achieved an accuracy of 98.50%. Alone, these results imply that deep learning has the potential to effectively tap into complex patterns in URL data. In contrast, this is in variances with the lower performance obtained for the Stacked Ensemble with Snapshots in our results. That means, although ensemble methods are very strong, their success really depends on the underlying models and characteristics of the data. To sum up, Cat Boost Classifier, Snapshot Ensemble, and Stacked Ensemble with Snapshots have been compared to related studies with the following main insights. The leading accuracy 40of the Cat Boost Classifier underlines the efficiency of the boosting technique and advanced feature handling in URL classification. Competitive performance by the Snapshot Ensemble speaks to the strength of ensemble learning in generalization and accuracy. The Stacked Ensemble with Snapshots returned moderate performance and experienced challenges handling class-imbalance thus complexity of optimizing ensemble methods. The role of feature engineering and the choice of dataset have been very important in attaining high accuracies, as shown by existing studies. In these studies, there were reported accuracies ranging from logistic regression to a decision tree, deep learning models, and ensemble techniques, which are all successful options for trying to perform the complex task of URL classification.

REFERENCES

[1]. Y. Zeng, “Malicious URLs and Attachments Detection on Lexical-based Features using Machine Learning Techniques,” 2018.

[2]. B. Banik and A. Sarma, “Lexical Feature Based Feature Selection and Phishing URL Classification Using Machine Learning Techniques,” Commun. Comput. Inf. Sci., vol. 1241 CCIS, pp. 93–105, Jul. 2020, doi: 10.1007/978-981-15-6318-8_9.

[3]. K. L. Chiew et al., “Building Standard Offline Anti-Phishing Dataset for Benchmarking,” Int. J. Eng. Technol., vol. 7, no. 4.31, pp. 7–14, Dec. 2018, doi: 10.14419/ijet. v7i4.31.23333.

[4]. B. Banik and A. Sarma, “Phishing URL detection system based on URL features using SVM,” International Journal of Electronics and Applied Research, 2018.

[5]. O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, “Machine learning based phishing detection from URLs,” Expert Syst. Appl., vol. 117, pp. 345–357, Mar. 2019, doi: 10.1016/J.ESWA.2018.09.029.

[6]. “Yandex.XML — Yandex Teknolojileri.” https://yandex.com.tr/dev/xml/. A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. Gonzalez, “Classifying phishing URLs using recurrent neural networks,” in eCrime Researchers Summit, eCrime, Jun. 2017, pp. 1–8, doi: 10.1109/ECRIME.2017.7945048.

[7]. W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Scherer, and M. Woźniak, “Accurate and fast URL phishing detector: A convolutional neural network approach,” Comput. Networks, vol. 178, no. January, 2020, doi: 10.1016/j.comnet.2020.107275. “Technical challenge of network security.”

[8]. https://www.kesci.com/apps/home/dataset/58f32a96a 686fb29e42 5a567. “Reasonable Antiphishing,” [Online]. Available: http://antiphishing.reasonables.com/BlackList.aspx.

[9]. R. Yang, K. Zheng, B. Wu, C. Wu, and X. Wang, “Phishing Website Detection Based on Deep Convolutional Neural Network and Random Forest Ensemble Learning,” Sensors, vol. 21, no. 24, p. 8281, Dec. 2021, doi: 10.3390/S21248281.

[10]. “Yandex.Toloka Open Datasets.” https://research.yandex.com/datasets/toloka (accessed Jan. 16, 2022).

[11]. J. Yuan, Y. Liu, and L. Yu, “A Novel Approach for Malicious URL Detection Based on the Joint Model,” Secur. Commun. Networks, vol. 2021, pp. 1–12, Dec. 2021, doi: 10.1155/2021/4917016. “Hphosts.” https://www.hosts-file.net/.

[12]. B. Altay, T. Dokeroglu, and A. Cosar, “Context-sensitive and keyword density-based supervised machine learning techniques for malicious webpage detection,” Soft Comput., vol. 23, no. 12, pp. 4177–4191, Jun. 2019, doi: 10.1007/s00500-018-3066-4.

[13]. “Cybersecurity to Prevent Breaches. | Comodo Cybersecurity.” = https://www.comodo.com.

[14]. J. McGahagan, D. Bhansali, C. Pinto-Coelho, and M. Cukier, “A Comprehensive Evaluation of Webpage Content Features for Detecting Malicious Websites,” Nov. 2019, doi: 10.1109/LADC48089.2019.8995713.

[15]. talosintelligence.com, “Snort || Cisco Talos Intelligence Group - Comprehensive Threat Intelligence,” Cisco, 2020. https://talosintelligence.com/snort. A. K. Jain and B. B. Gupta, “A machine learning based approach for phishing detection using hyperlinks information,” J. Ambient Intell. Humaniz. Comput., vol. 10, no. 5, pp. 2015– 2028, May 2019, doi: 10.1007/S12652-018-0798-Z. “Welcome to CentOS.” http://www.stuffgate.com/ (accessed Jan. 19, 2022).

[16]. M. Al-Kabi, H. Wahsheh, I. Alsmadi, E. Al-Shawakfa, A. Wahbeh, and A. Al-Hmoud, “Content-based analysis to detect Arabic web spam,” J. Inf. Sci., vol. 38, no. 3, pp. 284–296, Jun. 2012, doi: 10.1177/0165551512439173.

[17]. Alsmadi, “The automatic evaluation of website metrics and state,” Int. J. Web-Based Learn. Teach. Technol., vol. 5, no. 4, pp. 1–17, 2010, doi: 10.4018/jwltt.2010100101.

[18]. M. N. Al-Kabi, H. A. Wahsheh, and I. M. Alsmadi, “OLAWSDS: An Online Arabic Web Spam Detection System,” 2014.

[19]. E. M., A. F., and H. E., “Web Mining Techniques to Block Spam Web Sites,” Int. J. Comput. Appl., vol. 181, no. 8, pp. 36–42, Aug. 2018, doi: 10.5120/ijca2018917622.