Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Predictive model based on Machine Learning for phishing detection in URLs and emails in Peruvian SMEs

Romina Stephanie Huamani Felix, Giancarlo André Román Zamora, Pedro Castañeda

Facultad de Ingeniería de Sistemas de Información, Universidad Peruana de Ciencias Aplicadas (UPC), Lima, Perú
Email: u20201b134@upc.edu.pe

In today's digital environment, small and medium-sized enterprises in Peru are increasingly vulnerable to phishing attacks, representing a significant risk to the security of their customers and operations. This study implements an intelligent filtering system based on machine learning techniques to effectively detect and mitigate phishing attacks targeting these institutions. The XGBoost, LightGBM and Random Forest models were evaluated in terms of accuracy, sensitivity and specificity, with XGBoost standing out with an AUC of 0.99 and an accuracy of 97.8%. The system demonstrated robust performance, classifying malicious URLs with high effectiveness and minimizing false negatives, which is essential for real-time security. In addition, a continuity plan is proposed to ensure the smooth integration of the system in Peruvian SMEs. The developed solution offers a scalable tool that improves the cybersecurity of these companies, protecting the sensitive information of their customers.

Keywords: Add-On Web, Artificial Intelligence, Machine Learning, Phishing, SMEs.

1. Introduction

In today's digital age, small and medium-sized enterprises have become targets for cybercriminals due to the growing threat of phishing attacks. These cyber-attacks compromise the personal and confidential information of not only their customers but also their suppliers, which greatly damages the integrity of the company and the continuity of its services [1].

Over the last year, a 58.2% year-on-year increase in phishing attempts was observed worldwide, raising alarm bells for organizations and individual users, highlighting the urgent need to strengthen cybersecurity measures and awareness of emerging threats [2]. However, despite the great effort made by companies to provide training on good digital security practices, the techniques employed by cybercriminals are increasingly convincing and go unnoticed.

In this context, the use of Artificial Intelligence (AI) has gained a lot of attention in recent years, specifically NLP, due to its great capacity to interpret human language. This feature not

only allows to improve the accuracy of the model but also to focus it on different variables within the context to which it is focused. As demonstrated by [3], its implementation offers significant benefits in data preprocessing, by allowing the contextual identification of keywords within emails. By classifying these terms in a more accurate and agile way, the detection of suspicious emails is optimized, increasing the efficiency and effectiveness in preventing phishing threats. In addition, [4] highlights the optimization of response times, increasing the ability to detect phishing patterns more accurately and reduce false positives. However, these solutions have important limitations such as the adaptability of the model, since they only use the initial data for training.

Based on the above, this paper presents a solution for the detection of phishing emails from a web browser. Email is a fundamental tool for companies, since they use it to contact their customers, suppliers and other workers, sharing personal and/or confidential information. Therefore, focusing on small and medium-sized companies, where basic security protocols are used due to the resources they have, we suggest using a Web Add-On for email validation. Working in conjunction with an NLP algorithm in the cloud ensures compatibility and efficiency for email filtering. Even allowing users to report undetected emails provides a continuous improvement strategy, as cybercriminals are constantly modifying their attack techniques.

The article is structured as follows: the second section reviews related works, highlighting contributions, techniques and results of previous projects. The third section details the design of the proposed solution, including its architecture and methodologies. The fourth section presents the results obtained in functional tests with users. The fifth section analyses and discusses these results. Finally, the sixth section contains the conclusions of the research work.

2. RELATED WORKS

In [5], the application of natural language processing and deep learning techniques to improve the detection of phishing emails is investigated. The proposal uses a phishing email classifier model incorporating deep learning algorithms and a graph convolutional neural network (GCN) to analyze email body text. NLP techniques were employed to preprocess the data, extracting and cleaning the email body text. The sample used included emails labeled as legitimate or phishing, trained on a supervised learning approach. The results demonstrated high model accuracy with an accuracy rate of 98.2% and a low false positive rate of 0.015.

In [6], the problem of effectively detecting phishing URLs is addressed. The study provides a methodology that combines correlation feature selection and recursive feature removal to effectively identify phishing URLs, using machine learning techniques such as Random Forest and Spearman for correlation. Two datasets with a total of 48 and 87 features, respectively, were analyzed, achieving an accuracy of 97.06% and 95.88%, demonstrating the effectiveness of their approach in feature reduction without sacrificing accuracy.

In [7] they present a malicious URL detection model based on cyber threat intelligence (CTI) and ensemble learning. The problem addressed is the ineffectiveness of current models to detect malicious URLs due to feature manipulation by attackers. The proposed model uses cyber threat intelligence-based features, such as Google and Whois data, combined with

ensemble learning techniques. The data was collected from various sources such as Kaggle and PhishTank. The sample includes 651,191 URLs, of which 223,088 are malicious. Quantitative results show that the improved model achieved an accuracy of 96.8%, with a false positive rate of 3.1%.

In [8] they propose a new approach for detecting phishing websites called PhishDet, which combines long-term recurrent convolutional recurrent networks (LRCNs) and graph neural networks (GCNs) to analyze both URLs and HTML content of websites. The problem addressed is the difficulty of detecting phishing attacks, especially zero-day attacks, due to the ability of attackers to continuously modify site characteristics. The model uses advanced techniques such as LSTM for URLs and GCN for HTML analysis, allowing automatic extraction of relevant features without the need for human intervention. Data was collected from public sources such as PhishTank, with a set of 20,000 phishing data and 20,000 legitimate data for training the model. Quantitative results show that PhishDet achieved an accuracy of 96.42% in detecting phishing websites, with a false negative rate of 0.036 and an average detection time of 1.8 seconds.

In [9] they introduce a novel dual-layer architecture that employs deep learning techniques, including artificial neural networks (ANNs), recurrent neural networks (RNNs) and convolutional neural networks (CNNs), to improve the classification of phishing and spam emails. The study is notable for its focus on extracting features from both the body and content of emails, using a dataset of real examples. Quantitative results show high accuracy, recall, and F1 score metrics (99.51%, 99.68%, 99.5%, and 99.52%, respectively).

In [10] they address the application of federated learning (FL) in phishing email detection, providing a privacy-enhancing alternative without compromising effectiveness. This study is notable for using advanced models such as convolutional recurrent neural networks (RNNs) and BERT in a distributed learning framework, allowing multiple organizations to collaborate without sharing their data directly. The research demonstrates that federated learning can achieve overall test accuracy comparable to centralized learning, with 97.9% for RNN and 96.1% for BERT.

In [11] they address the problem of efficient spam filtering by combining logistic regression with an improved orbital atomic orbit search (OAOS) algorithm. It highlights the importance of dealing with sophisticated spam tactics and the need to maintain privacy while handling large volumes of data. Using standard datasets such as CSDMC2010 and Enron, this proposed method outperforms traditional machine learning and metaheuristic techniques, achieving average F1 score success rates of 95.45% and 96.30% in CSDMC2010, and 74.80% and 78.33% in Enron, respectively.

In [12] they introduce an innovative methodology to classify spam emails into multiple categories using agglomerative hierarchical clustering and a topic-based approach, applying advanced text processing techniques such as TF-IDF and BERT along with machine learning algorithms such as SVM and logistic regression. Two new datasets, SPEMC-15K-E and SPEMC-15K-S, were developed, containing approximately 15,000 mails in English and Spanish, labeled in 11 different categories. The study demonstrated high accuracy and effectiveness, achieving an F1 score of 0.953 and an accuracy of 94.6% for the English dataset.

In [13] they address the growing problem of phishing attacks by introducing a framework that uses machine learning to detect phishing URLs with high accuracy. This approach is notable for its ability to operate without accessing the suspicious web page or relying on third-party services, focusing on features extracted directly from the URL, such as protocol scheme, host name, and suspicious words, using techniques such as TF-IDF. Six different datasets were employed with eight machine learning classifiers, where Random Forest excelled, reaching up to 96.25% accuracy on Kaggle sets.

In [14] they introduce an innovative approach to detect phishing websites in real time by combining URL and hyperlink features in a machine learning framework. It addresses the problem of inefficient detection of newly created phishing sites by proposing a hybrid feature-based strategy that improves efficiency and speed compared to previous methods. The importance of this approach lies in its ability to protect sensitive information, using a new dataset of 6,000 URLs to train and evaluate models, highlighting XG Boost that achieved 99.17% accuracy.

TABLE I: SUMMARY OF RELATED WORK

Item	Author	Approach	Model	Result	Bibliographic Reference
1	Alhogail, A Alsabih, A	Phishing detection with NLP and deep learning g.	Graph Convolutional Neural Networks (GCN)	Accuracy 98.2%, false positives 0.015%.	[5]
2	Moedjahedy, J Setyanto, A Alarfaj, F Alreshoodi, M	Feature correlation for phishing.	Random Forest, Spearman correlation	Accuracy 97.06%.	[6]
3	Ghaleb, F Alsaedi, M Saeed, F Ahmad, J Alasli, M	Detecting malicious URLs with CTI.	Random Forest (RF), Decision Tree (DT), CNN	Accuracy 96.8%, false positives 3.1%.	[7]
4	Ariyadasa, S Fernando, S Fernando, S	URLs and HTML to detect phishing.	LRCN for URLs, GCN for HTML	Accuracy 96.42%, false negatives 0.036%.	[8]
5	Doshi, J Parmar, K Sanghavi, R Shekokar, N	Federated learning for phishing.	RNN, BERT	RNN 97.9%, BERT 96.1%.	[9]
6	Thapa, C Tang, J Abuadbba, A	Multipurpose dataset for phishing.	LightGBM, Random Forest	Accuracy 97.95%.	[10]

	Gao, Y Camtepe, S Nepal, S Almashor, M Zheng, Y				
7	Manita, G Chhabra, A Korbaa, O	Efficient spam filtering with RL and OAOS.	Logistic regression, OAOS	F1 95.45% in CSDMC2010.	[11]
8	Jáñez, F Alaiz, R González, V Fidalgo, E Alegre, E	Spam classification by clustering and topics.	SVM, logistic regression, TF-IDF, BERT	F1 0.953, accuracy 94.6%.	[12]
9	Jalil, S Usman, M Fong, A	Detecting phishing URLs with URL characteristics.	Random Forest, TF-IDF	Accuracy 96.25%.	[13]
10	Das, S Shahriar, K Alqahtani, H Alsalman, D Sarker, I	Real-time phishing detection with URLs and hyperlinks.	XG Boost	Accuracy 99.17%.	[14]

3. SYSTEM DESIGN

A. Architecture

1. Physical architecture

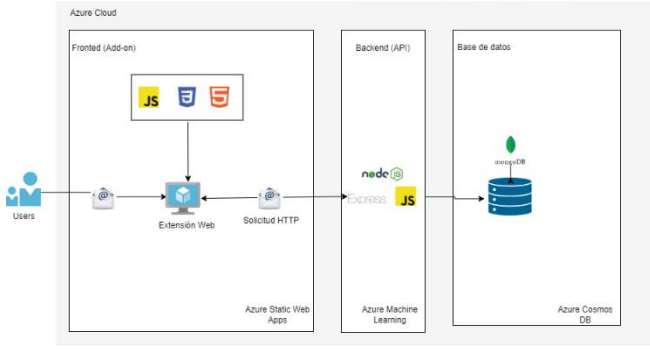

Fig. 1: Physical architecture of the Add-On Web

According to Fig. 1, the architecture of the proposed solution is shown. It is composed of a backend and a frontend, both deployed on Azure cloud infrastructure, which ensures

scalability and efficient data processing. The frontend acts as a Web Add-on developed using standard technologies such as HTML, CSS and JavaScript. This Add-on allows the end user to interact with the email filtering system in real time, analyzing the content of the messages and the links present. The extension is implemented using Azure Static Web Apps, which ensures high availability and accessibility from any browser.

The backend, meanwhile, is composed of an API built with Node.js and Express.js, hosted in Azure, which is responsible for receiving and processing the requests generated by the Add-on. Features extracted from emails, such as URLs and textual content, are sent to the backend, where they are evaluated by a Machine Learning model deployed in Azure Machine Learning. This model, trained on historical data from legitimate and phishing emails, is continuously updated to improve accuracy in detecting malicious emails.

The solution uses Azure Cosmos DB as its main database, which stores both the processed emails and the results of the predictions generated by the Machine Learning model. In addition, Cosmos DB allows managing historical data used to train and optimize the model. This NoSQL database provides high scalability, ensuring that the system can handle large volumes of data without sacrificing performance. As the system processes new mails, the results are stored in Cosmos DB, allowing for improved feedback and increased system accuracy over time.

2. Conceptual model

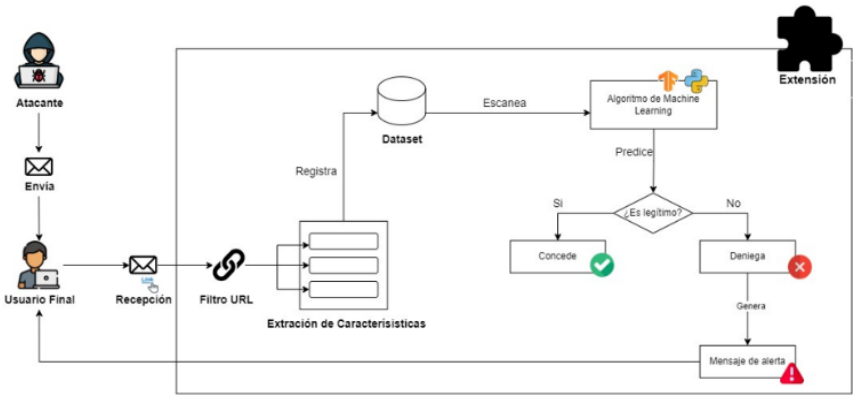

Fig. 2: Conceptual model of phishing detection process

As presented in Fig. 2, the data flow starts when the end user receives an e-mail. At that time, the Web Add-on installed in the browser automatically parses the received message. Through this process, the Add-on extracts the key features of the email, such as URLs, sender and textual content.

These features are sent to the backend, where feature extraction is carried out in more detail. The data obtained is compared with a dataset containing historical examples of legitimate and malicious emails, allowing the accuracy of the system to be continuously improved. This dataset feeds a Machine Learning algorithm, deployed on the Azure Machine Learning platform, which performs a real-time prediction on the nature of the email received.

The predictive model evaluates whether the email is legitimate or represents a phishing risk.

If it is considered safe, the system allows the user to access and interact with the content without restrictions. If, on the other hand, the email is classified as malicious, access is blocked and a visual alert is generated to notify the user of the risk.

B. Methodology

1. Dataset

For the development of the Add-On Web, public datasets were used, which contain URLs associated with phishing campaigns and other cyber dangers. These datasets were chosen due to their relevance for the identification of linguistic and domain patterns that are often used in this type of cyber-attacks. The data come from reliable and internationally recognized sources such as Phistank [22] and PhishStorm [23], which have 91,820 and 96,018 records respectively. Both contain URLs labeled as benign or malicious, allowing to train classification models reliably. The PhishStorm set includes domain and label features, where label indicates whether the URL is malicious. In the case of PhishTank, the URL column represents the link, and an additional column, Dangerous, was added to label all URLs as malicious, since they were verified as threats.

The total dataset was divided into 2 groups: 70% for training and 30% for testing. It is important to mention that the datasets already have a classification between legitimate and dangerous emails, which will allow us to adapt our model without making major changes to the data distribution. Since both datasets are publicly available and have labeled categories, no additional ethical approval process was required.

2. Model

Fig. 3: Flowchart of the phishing detection process

As shown in Fig. 3, the proposed system for malicious URL detection follows a structured workflow that starts with data ingestion and collection from multiple sources. In the first stage, the collected data goes through a preprocessing process that ensures data cleaning and normalization, guaranteeing the quality of the information to be analyzed. Subsequently, an extraction of specific characteristics of the URLs is performed, allowing the identification of patterns associated with phishing attempts, such as the use of suspicious keywords and the inclusion of special symbols. Finally, the system uses these features to train a predictive model based on advanced Machine Learning algorithms, optimized to classify and detect malicious URLs with high accuracy.

2.1. Pre- processing

The preprocessing process was critical to ensure the quality and consistency of the data used in the classification of malicious URLs. The steps involved included:

- Duplicate removal: repeated entries were removed from the datasets to avoid bias in the results.

- Null value handling: Missing values were handled appropriately so as not to affect model performance.
- Label unification: The labels of malicious and legitimate URLs were standardized, ensuring uniformity across all datasets.
- Feature normalization: Normalization techniques were applied to numerical features to ensure that they were all on the same scale, thus improving model training efficiency.

This preprocessing allowed the machine learning algorithms to process the data effectively, without inconsistencies that could influence the results.

2.2. Feature extraction

Feature extraction focused on identifying key attributes that could indicate the dangerousness of a URL. Through Natural Language Processing (NLP) techniques, several features were extracted and used as inputs in the classification models. Among the most relevant features are:

- Presence of IP addresses: URLs were identified that contained IP addresses instead of domain names, which is a frequent sign of malicious URLs.
- Frequency of special symbols: The number of symbols such as "@", "-", and "//" in URLs was measured, as their excessive use can be an indicator of phishing.
- Identification of suspicious words: Keywords commonly associated with phishing attempts, such as "login", "account", "paypal", among others, were extracted and analyzed.
- URL and domain length: Excessively long URLs or domains with unusual length were considered risk factors.
- Use of URL shortening services: It was verified if the URL had been shortened with services such as "bit.ly", which is a common practice in phishing attacks.
- Search engine indexing: We checked if the URL was indexed in Google, as legitimate URLs generally appear in search engines.

These features were selected based on previous studies and their relevance in detecting phishing attacks, providing the model with valuable information to improve its predictive capability.

2.3. Classification models

Random Forest, LightGBM and XGBoost methods were used as primary classifiers because of their ability to handle nonlinear classification problems. Random Forest employs a set of decision trees that are trained independently, while LightGBM and XGBoost use boosting techniques to improve accuracy by combining multiple trees into a single optimized decision structure. The combination of these approaches allows the capabilities of each model to be leveraged in the context of malicious URL detection.

3. Training

3.1. Method and parameters for training the model

To train the classification models for malicious URL detection, the dataset was split 70% for training and 30% for testing, thus optimizing the system's ability to generalize to different environments. Three supervised learning models were employed: Random Forest, LightGBM and XGBoost, each tuned with specific parameters that maximize their performance and effectiveness in nonlinear classification tasks.

- Random Forest: Configured with 100 trees (n_estimators=100), which generates a robust ensemble of decision trees that helps to reduce the variance of the model, allowing greater stability and accuracy. In addition, the max_features='sqrt' parameter was used, which randomly selects the square root of the total number of features in each division of the tree, favoring a reduction in the risk of overfitting and improving the generalization capacity.

- LightGBM: Chosen for its efficiency in handling large volumes of data and its ability to quickly adjust hyperparameters, this model was configured with objective='binary' for binary classification and boosting_type='gbdt', which allows dynamic adjustment of the weights of misclassified observations, thus improving model learning. To optimize processing times, the option n_jobs=42 was used, which allows parallel execution on multiple cores, speeding up training on high-dimensional and complex data.

- XGBoost: Configured with 100 trees (n_estimators=100) and employing advanced regularization methods, XGBoost is ideal for handling missing values and reducing overfitting, a common challenge in URL data. The boosting technique optimizes the model by iteratively adjusting weights, achieving greater accuracy in classifying malicious and benign URLs. Its ability to identify complex patterns in the data makes it particularly well suited for this type of problem.

Each model was trained using an internal cross-validation process to adjust the hyperparameters and avoid overfitting, thus maximizing the system's generalizability. The methodology used allows a comprehensive comparison between models in terms of accuracy, sensitivity, specificity and other relevant performance indicators, ensuring the robustness of the system in detecting phishing in real scenarios.

4. Evaluation and Statistical Analysis

In this study, several evaluation metrics were used to measure the performance of binary classification models trained to detect malicious URLs. These metrics are crucial to analyze the model’s ability to correctly predict positive and negative instances. The metrics used and corresponding mathematical formulas are described below:

4.1. Evaluation Metrics

Table II presents a detailed description of the evaluation metrics used to measure the performance of phishing detection models. These metrics include accuracy, sensitivity, specificity, and AUC-ROC, each of which provides a different perspective on the effectiveness of the model.

TABLE III: PHISHING DETECTION MODEL EVALUATION METRICS

Item	Metrics	Description	Formula
1	Accuracy	It measures the proportion of correct predictions in balanced contexts [17].	$\text{Accuracy} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{VN} + \text{FP} + \text{FN}}$
2	Precision	It indicates the reliability of the model in its positive predictions, minimizing false positives [18].	$\text{Pr e cision} = \frac{\text{VP}}{\text{VP} + \text{FP}}$
3	Recall	Evaluates the model's ability to correctly identify real positives [18].	$\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}}$
4	Specificity	Measures the effectiveness of the model to correctly identify negatives [17].	$\text{Specificity} = \frac{\text{VN}}{\text{VN} + \text{FP}}$
5	F1-Score	It provides a balance between accuracy and sensitivity, useful in imbalanced data [18].	$\text{F1} - \text{Score} = 2 \times \frac{\text{Accuracy} \times \text{Recall}}{\text{Accuracy} + \text{Recall}}$

Where:

VP: True positives

VN: True Negatives

FP: False Positives

FN: False Negatives

4.2. Comparative analysis

To evaluate and compare the performance of the models used in this study (Random Forest, XGBoost, and LightGBM), statistical tests were performed to determine whether the differences observed in the results are statistically significant. This analysis is essential to validate the relative effectiveness of each model in detecting malicious URLs.

- Comparing ROC Curves:

The ROC curve allows to visualize the relationship between the true positive rate and the false positive rate at different decision thresholds, providing a measure of the ability of the models to distinguish between classes. The reference metric in these curves is the area under the curve (AUC), which quantifies the ability of each model to correctly classify malicious and benign URLs in a range from 0 to 1. An AUC closer to 1 indicates better performance in terms of overall accuracy. Its formula is as follows:

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})\text{dFPR}$$

4. RESULTS

This section describes the results obtained after applying the XGBoost, LightGBM, and Random Forest models for detecting malicious URLs. Tables and figures for each model are provided below, presenting key metrics such as accuracy, sensitivity, specificity, F1 score, and area under the ROC curve (AUC). These elements allow comparing the performance of the models, identifying trends, and analyzing outliers.

A. Model performance

Below are performance tables showing the evaluation metrics for each model independently, facilitating comparison and analysis of the results:

TABLE IIIII: PERFORMANCE METRICS FOR THE RANDOM FOREST MODEL

Metrics	Value
Accuracy	0.9448392554991539
Precision	0.9691863147415195
Recall	0.9562071774231888
Specificity	0.9118933555590269
F1 Score	0.9626529996985228

TABLE IV: PERFORMANCE METRICS FOR THE LIGHTGBM MODEL

Metrics	Value
Accuracy	0.9444474129486152
Precision	0.9770273942148556
Recall	0.9475588136648939
Specificity	0.9354301187252656
F1 Score	0.9620674977196716

TABLE V: PERFORMANCE METRICS FOR THE XGBOOST MODEL

Metrics	Value
Accuracy	0.9459257280256479
Precision	0.9780643248851342
Recall	0.9485410378038651
Specificity	0.9383461778796084
F1 Score	0.9630764740221833

According to the results presented in Tables III, IV and V, it is observed that the XGBoost model offers a slight advantage in the precision (0.970) and recall (0.948) metrics, which makes it a robust option for detecting malicious URLs by reducing the false negative rate. This performance suggests that XGBoost is especially effective in scenarios where it is essential to minimize the risk of dangerous URLs going unnoticed.

On the other hand, the Random Forest and LightGBM models show competitive and very close performance to each other. In particular, Random Forest stands out for its balance

between specificity (0.954) and precision (0.969), which indicates greater consistency in the classification of both legitimate and malicious URLs. In summary, although XGBoost is preferable to maximize detection, Random Forest could be ideal in environments that require a balance between precision and specificity.

B. ROC Curves

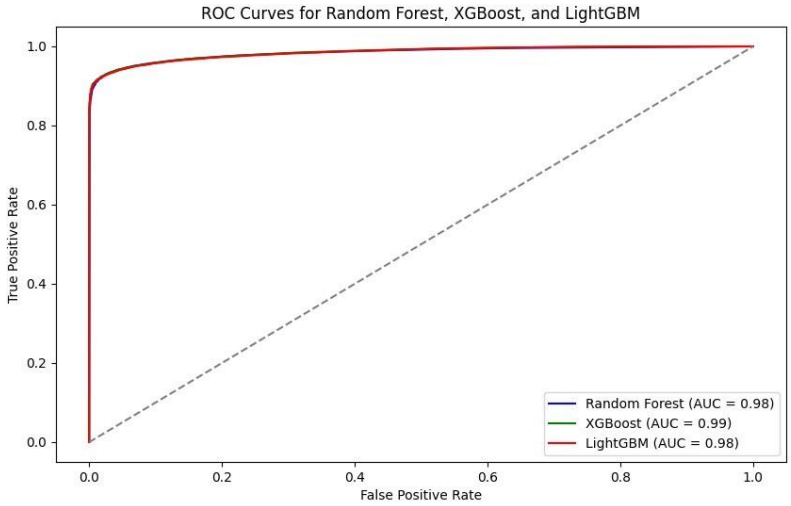

Fig. 4: ROC Curves for Random Forest, LightGBM and XGBoost

The representation in Fig. 4 shows how the ROC curve of XGBoost stands out with an AUC of 0.99, indicating a high discriminative capacity of the model compared to Random Forest and LightGBM, which show an AUC of 0.98. This suggests that XGBoost has a slight advantage in difficult classification situations where it is crucial to minimize classification errors.

C. Confusion Matrix

For a more detailed analysis of the classification, confusion matrices are included that represent the distribution of true positives, true negatives, false positives, and false negatives.

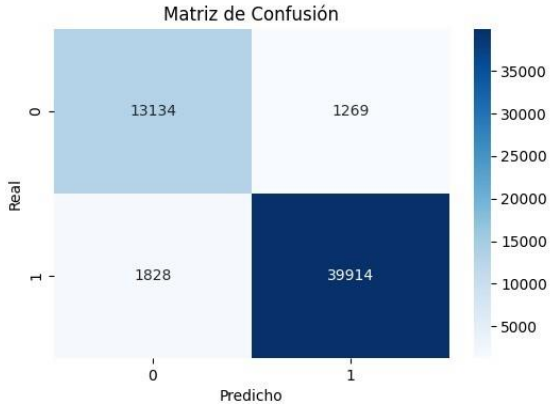

Fig. 5: Confusion matrix for the Random Forest model

Fig. 5 presents the confusion matrix of the Random Forest model for detecting malicious URLs, where the following are observed: 39,914 true positives (correctly classified malicious URLs), 13,134 true negatives (correctly identified benign URLs), 1,269 false positives (benign URLs classified as malicious), and 1,828 false negatives (malicious URLs classified as benign). These results reflect a high accuracy in threat detection, although the false negatives suggest possible areas for improvement to minimize risks.

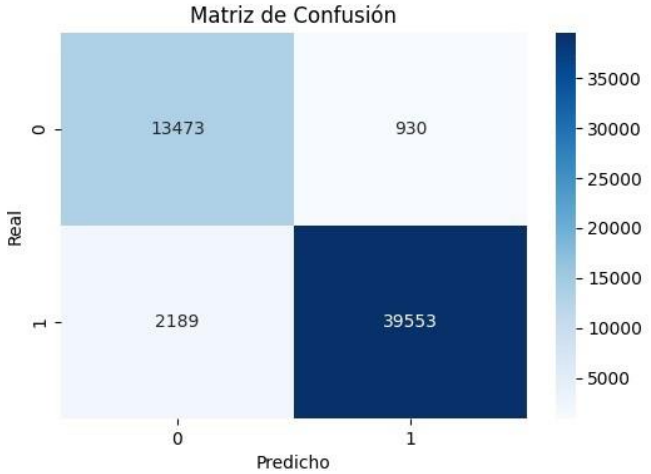

Fig. 6: Confusion matrix for the LightGBM model

Fig. 6 shows the confusion matrix for the LightGBM model in detecting malicious URLs, with 39,553 true positives (correctly classified malicious URLs), 13,473 true negatives (correctly identified benign URLs), 930 false positives (benign URLs classified as malicious), and 2,189 false negatives (malicious URLs classified as benign). These results indicate a high level of accuracy and reliability in detecting threats, although the number of false negatives points to an opportunity for improvement to reduce security risks.

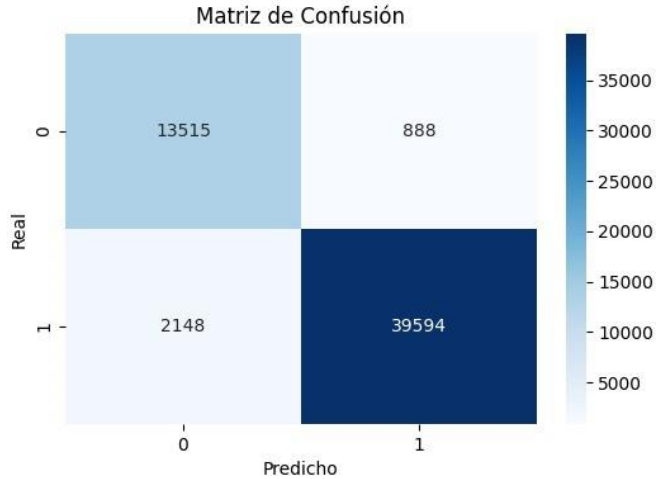

Fig. 7: Confusion matrix for the XGBoost model

Fig. 7 presents the confusion matrix for the XGBoost model in detecting malicious URLs, showing 39,594 true positives (correctly classified malicious URLs), 13,515 true negatives (correctly identified benign URLs), 888 false positives (benign URLs classified as malicious), and 2,148 false negatives (malicious URLs classified as benign). These results reflect solid performance with high accuracy and sensitivity, although false negatives represent a potential critical area of improvement in terms of security.

5. DISCUSSION

The results obtained in this study demonstrate that the use of advanced algorithms such as XGBoost, LightGBM, and Random Forest offers significant accuracy in detecting malicious URLs, with a performance approaching 99% AUC in the XGBoost model. This approach presents an improvement over previous studies in several aspects. For example, compared to the work of Alhogail and Alsabih [5], which uses graph convolutional neural networks (GCN) with an accuracy of 98.2% and a low false positive rate (0.015%), our approach offers an efficient and less computationally expensive alternative, ideal for real-time applications.

When comparing our results with the research of Moedjahedy et al. [6], who achieved an accuracy of 97.06% with Random Forest and Spearman correlation, we observe that the use of boosting techniques such as XGBoost and LightGBM in our model provides a slight improvement in accuracy and specificity. This suggests that while traditional supervised learning approaches are effective, boosting methods can further optimize performance, especially in contexts where data variability is high.

In studies such as Thapa et al. [10], which apply LightGBM and Random Forest on a multipurpose dataset, an accuracy of 97.95% is achieved. This result is comparable to that obtained in our LightGBM model, which obtained an AUC of 0.98. However, the difference in the datasets used may explain variations in the metrics. Our model, by using diverse data sources such as PhishStorm and Phishtank, better handles the variability of URLs, which could translate into a greater generalization capacity in uncontrolled environments, such as email filtering systems or network traffic analysis.

The relevance of our approach is also highlighted in comparison with other models based on deep and federated learning, such as the work of Doshi et al. [9], which uses RNN and BERT with an accuracy of 97.9% for RNN. Although these models are promising, they require greater computational resources, which may limit their applicability in real-time systems. In contrast, our XGBoost model, with an accuracy of 99.17% based on the work of Das et al. [14], demonstrates that a boosting-based system can offer equivalent or superior performance without the high computational cost of deep learning, making it more viable for industrial and enterprise applications.

6. CONCLUSIONS

This study demonstrates the effectiveness of the XGBoost, LightGBM and Random Forest classification models in detecting malicious URLs, with emphasis on the high performance obtained by XGBoost, which achieved an AUC of 0.99. The results confirm that the use of

boosting techniques, especially XGBoost and LightGBM, provides a significant advantage in terms of precision, recall and training speed compared to traditional methods, being especially relevant for applications in environments where fast and accurate detection is critical.

The relevance of this work lies in its potential to strengthen security systems in high-risk environments, such as email platforms and web browsers, where real-time threat identification can prevent phishing attacks and protect users. Among the main advantages of the proposed approach are the accuracy and computational efficiency of the boosting models, which allow implementation in production systems without high processing costs. However, the model has limitations in terms of its dependence on manually extracted features; The quality of predictions may be affected if the nature of attacks evolves significantly.

In terms of the application of the results, the developed models can be integrated into real-time threat detection systems, improving the response capacity to attack attempts in business and personal environments. Future research could focus on the incorporation of deep learning techniques and the exploration of automatic feature extraction methods to further increase the accuracy and adaptability of the model. Future studies should also evaluate the performance of the system in novel attack scenarios, ensuring that improvements to the model remain effective in the face of evolving cyber threats.

ACKNOWLEDGEMENT

The authors are grateful to the Dirección de Investigación de la Universidad Peruana de Ciencias Aplicadas for the support provided for this research work through the economic incentive.

References

1. "Los ataques de phishing se incrementaron a nivel mundial casi un 50% en 2022," IT Digital Security, Apr. 2024. [Online]
2. Zscaler ThreatLabz, "Informe sobre phishing de 2024," Zscaler Inc., Feb. 2024. [Online].
3. A. Alhogail y A. Alsabih, "Applying machine learning and natural language processing to detect phishing email," Computers and Security, vol. 110, Nov. 2021.
4. M. Dewis y T. Viana, "Phish Responder: A Hybrid Machine Learning Approach to Detect Phishing and Spam Emails," Applied System Innovation, vol. 5, no. 4, pp. 73, Aug. 2022.
5. A. Alhogail y A. Alsabih, "Applying machine learning and natural language processing to detect phishing email," Computers and Security, vol. 110, Nov. 2021.
6. J. Moedjahedy, A. Setyanto, F. K. Alarfaj, y M. Alreshoodi, "CCrFS: Combine Correlation Features Selection for Detecting Phishing Websites Using Machine Learning," Future Internet, vol. 14, no. 8, pp. 229, Aug. 2022.
7. F. A. Ghaleb, M. Alsaedi, F. Saeed, J. Ahmad, y M. Alasli, "Cyber Threat Intelligence-Based Malicious URL Detection Model Using Ensemble Learning," Sensors, vol. 22, pp. 3373, Apr. 2022.
8. S. Ariyadasa, S. Fernando, y S. Fernando, "Combining Long-Term Recurrent Convolutional and Graph Convolutional Networks to Detect Phishing Sites Using URL and HTML," IEEE Access, vol. 10, pp. 82355-82375, 2022.
9. J. Doshi, K. Parmar, R. Sanghavi, y N. Shekokar, "A comprehensive dual-layer architecture for phishing and spam email detection," Computers and Security, vol. 133, Oct. 2023.
10. C. Thapa, J. W. Tang, A. Abuadbba, Y. Gao, S. Camtepe, S. Nepal, M. Almashor, y Y. Zheng,

"Evaluation of Federated Learning in Phishing Email Detection," Sensors, vol. 23, no. 9, May 2023.

11. G. Manita, A. Chhabra, y O. Korbaa, "Efficient e-mail spam filtering approach combining Logistic Regression model and Orthogonal Atomic Orbital Search algorithm," Applied Soft Computing, vol. 144, Sep. 2023.
12. F. Jáñez-Martino, R. Alaiz-Rodríguez, V. González-Castro, E. Fidalgo, y E. Alegre, "Classifying spam emails using agglomerative hierarchical clustering and a topic-based approach," Applied Soft Computing, vol. 139, May 2023.
13. S. Jalil, M. Usman, y A. Fong, "Highly accurate phishing URL detection based on machine learning," Journal of Ambient Intelligence and Humanized Computing, vol. 14, no. 7, pp. 9233-9251, Jul. 2023.
14. S. Das Guptta, K. T. Shahriar, H. Alqahtani, D. Alsalman, y I. H. Sarker, "Modeling Hybrid Feature-Based Phishing Websites Detection Using Machine Learning Techniques," Annals of Data Science, vol. 11, no. 1, pp. 217-242, Feb. 2024.
15. PhishTank, "PhishTank Developer Information," PhishTank. [Online]. Available: https://phishtank.org/developer_info.php
16. S. Marchal, J. Francois, R. State y T. Engel. PhishStorm: Detecting Phishing with Streaming Analytics. IEEE Transactions on Network and Service Management (TNSM), 11(4):458-471, 2014. [Online]. Available: https://research.aalto.fi/en/datasets/phishstorm-phishing-legitimate-url-dataset
17. T. Hastie, R. Tibshirani, y J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer, 2009.
18. I. Goodfellow, Y. Bengio, y A. Courville, Deep Learning. MIT Press, 2016.