International Journal of Management and Data Analytics
IJMADA

Vol 6 (1), 297-316
ISSN: 2816-9395
DOI: https://doi.org/10.5281/zenodo.22102917
Journal website: http://ijmada.com

A Framework for Optimising Phishing Websites Detection via Ensemble Learning and Synthetic Minority Oversampling Techniques

Author(s): Daniel Asuquo[1,2,3], Fredrick Umoh[3], Kingsley Attai[2,4], Kingsley Akputu[2], Avwokuruaye Oghenetega[2], Kitoye Ebire Okonny[5], Udoinyang G. Inyang[3], Isah R. Saidu[2]

1 Department of Information Systems, Faculty of Computing, University of Uyo, Uyo, Nigeria.
2 Department of Computing Sciences, Faculty of Science, Admiralty University of Nigeria, Ibusa, Nigeria.
3 Department of Cybersecurity, TETFund Centre of Excellence in Computational Intelligence Research, University of Uyo, Nigeria.
4 Department of Computer Science, Faculty of Computing, Ritman University, Ikot Ekpene, Nigeria.
5 ICT Centre, Ignatius Ajuru University of Education, Port Harcourt, Nigeria.

Corresponding Author: danielasuquo@uniuyo.edu.ng

Received: July, 2026 Published: Aug, 2026

ARTICLE INFO
Keywords:
Class imbalance; Cyber-attacks; Phishing websites; Supervised learning models; XAI

ABSTRACT
Phishing websites remain one of the most prevalent forms of cyber-attacks. They often exploit users, through deceptive web interfaces, to steal sensitive information such as login credentials, financial data, and personal records. The increasing sophistication of phishing strategies has reduced the effectiveness of traditional rule-based detection systems. This necessitates the adoption of intelligent and adaptive machine learning (ML) approaches. This study presents a robust ML-based framework for phishing websites detection using ensemble learning and Synthetic Minority Oversampling Technique (SMOTE). The proposed framework integrates comprehensive data preprocessing techniques, including constant and duplicate feature removal, Standard Scaler (Z-score normalization), and class balancing through SMOTE to improve model robustness and predictive capability. Three supervised learning algorithms, namely Decision Tree (DT), Random Forest (RF), and Extreme Gradient Boosting (XGBoost), were implemented and evaluated using an 8:2 train-test split, 5-fold cross-validation, and GridSearchCV-based hyperparameter optimization. Experimental results demonstrate that ensemble models outperform the standalone DT model across all evaluation metrics. Among the evaluated models, RF achieved the best overall performance with an accuracy of 97.13%, recall of 96.07%, F1-score of 95.86%, and area under the curve-receiver operating characteristics (AUC-ROC) of 0.9951 on the baseline dataset. Following the application of SMOTE, recall improved further to 96.46%, indicating enhanced capability in identifying phishing websites. Additionally, SHapley Additive exPlanations (SHAP) was incorporated to improve model interpretability and identify the most influential phishing indicators. The findings reveal that domain age and URL structural characteristics significantly influence phishing detection. Overall, the findings demonstrate the effectiveness and interpretability of the evaluated tree-based framework within the structured-feature dataset considered in this study. However, the age of the dataset limits the extent to which the findings can be generalized to contemporary phishing environments. Future work should therefore evaluate the framework on recent and continuously evolving datasets and investigate its robustness against emerging phishing strategies and changes in website characteristics.

1.0 Introduction

Phishing is one of the most common forms of social engineering attacks. It is widely used to gain unauthorized access to systems, financial resources, and confidential information

(Jabir et al., 2025; Alabdan, 2020; Heartfield & Loukas, 2016). Adversaries reportedly bypass cryptographic mechanisms by replicating website Cascading Style Sheets (CSS) and Document Object Model (DOM) structures with high fidelity (Kathrine et al., 2019). Subsequently, these deceptive assets are then distributed through multi-channel delivery networks using what could be considered as fast psychological triggers. It is by these triggers that target users are manipulated (Jabir et al., 2025). The aftermath of such automated lifecycle could even lead to catastrophic data breaches and systemic operational collapse (Jha & Jha, 2025; Safitra et al., 2023).

Attempt to mitigate phishing by legacy signature-based engines have failed, notably due to rigid reliance on static blacklists, which cannot detect rapid, zero-day polymorphic variations (Jain & Gupta, 2021). This technical gap has reportedly forced an architectural pivot toward predictive machine learning (ML) frameworks (Tang & Mahmoud, 2021). However, some of the contemporary predictive models are optimized under idealized, balanced datasets. In practice, therefore, they have suffered severe performance degradation in real-time environments (Asuquo et al., 2020; Manzano et al., 2022). In practical environments, network telemetry is characterized by extreme class imbalances, feature redundancy, and high multi-collinearity.

As a result, current research concentrates heavily on achieving high classification accuracy (Balogun et al., 2021). Although this approach improves performance on benchmark datasets, it introduces significant limitations when models are deployed in real-world operational environments. According to Innab et al. (2024), previous studies have often failed to simultaneously address the challenges of feature redundancy and severe class imbalance, despite their substantial influence on model performance. Neglecting these interconnected data issues increases the likelihood of overfitting to the training data (Manzano et al., 2022), reduces the model's ability to generalize to unseen data, and results in unnecessary computational complexity. Furthermore, current intrusion detection frameworks lack standardized preprocessing procedures and comprehensive evaluation methodologies. Most of them depend predominantly on threshold-based performance metrics, such as accuracy and F1-score, while overlooking other essential evaluation criteria like discriminative capability, probability calibration, and feature-level interpretability (Elshewey et al., 2026). This limited evaluation framework diminishes the transparency, reliability, and practical application of intelligent intrusion detection systems, thereby constraining their adoption in real-world cybersecurity environments.

To address these challenges, this study proposes a robust ML-based framework for phishing websites detection. The framework integrates both predictive performance and model interpretability. It implements three supervised learning algorithms namely, Decision Tree (DT), Random Forest (RF), and Extreme Gradient Boosting (XGBoost), evaluating their performance based on appropriate metrics. A comprehensive preprocessing pipeline is designed to enhance data quality and model efficiency, incorporating feature selection, Standard Scaler (Z-score normalization), and class balancing through the Synthetic Minority Oversampling Technique (SMOTE). To ensure fair and reliable comparison, all models are evaluated within a consistent experimental framework, including an 8:2 train-test split, 5-fold cross-validation, and systematic hyperparameter tuning using GridSearchCV. Model performance is assessed using multiple evaluation metrics, including accuracy, precision, recall, F1-score, log loss, and Area Under the Curve – Receiver Operating Characteristics (AUC-ROC).

In addition, this study incorporates Explainable Artificial Intelligence (XAI) through SHapley Additive exPlanations (SHAP) to enhance transparency and interpretability of the best-performing model. SHAP analysis is used to quantify the contribution of each feature to model predictions, enabling a deeper understanding of the factors driving phishing detection. By integrating SHAP into the framework, the proposed approach does not only achieve high predictive performance but also provides meaningful insights into feature

importance. This has the potential to improve model's trustworthiness and support more informed decision-making in real-world cybersecurity applications.

The contributions of this study are defined at the framework and empirical-analysis levels rather than as the introduction of new machine learning algorithms. Specifically, the study makes the following contributions:

A leakage-controlled phishing detection framework: A unified machine learning framework is developed in which feature preprocessing, class balancing, model selection, and evaluation are organized to prevent information from the independent test set entering the model-development process. In particular, SMOTE is applied exclusively within the training folds during cross-validation, while the independent test set remains untouched for final evaluation.

A controlled investigation of class-imbalance handling: The study systematically evaluates phishing detection under both baseline and SMOTE-assisted training conditions using a common experimental protocol. This enables the effect of synthetic minority oversampling on phishing identification, particularly recall and F1-score, to be assessed rather than assuming that oversampling necessarily improves performance.

Comparative evaluation of complementary tree-based learning approaches: DT, RF, and XGBoost are evaluated under the same preprocessing, cross-validation, and hyperparameter optimization framework. This provides a controlled comparison between a standalone tree learner and two ensemble approaches based on bagging and boosting.

Performance–interpretability analysis: The framework combines predictive evaluation with SHAP-based model interpretation to identify the website characteristics that most strongly influence the predictions of the selected model. The analysis provides an interpretable link between model performance and phishing indicators, including domain-related and URL-structural characteristics.

The study therefore does not claim novelty in the individual algorithms or techniques employed. Rather, its contribution lies in the systematic integration and leakage-controlled evaluation of established techniques, the empirical examination of the effect of class balancing on phishing detection, and the interpretation of the resulting model decisions.

The remainder of this manuscript is structured into four sequential sections. Section 2 reviews existing literature on approaches to phishing websites detection, highlighting their objectives, methods/metrics, results, strengths and weaknesses. Section 3 presents the proposed framework for optimising phishing websites detection using ensemble learning and SMOTE while Section 4 discusses the results and its implication, presenting the hyperparameters, calibration charts, and SHAP interpretability profiles. Section 5 concludes the paper, outlining directions for future research.

2.0 Literature Review

2.1 Traditional Heuristics and Classical Machine Learning Frameworks

Early cybersecurity defense architectures relied predominantly on deterministic techniques, including static blacklists, whitelists, and signature-based heuristics (Alabdan, 2020; Jain & Gupta, 2021). While blacklists enable near-instantaneous detection of known threats, they are fundamentally ineffective against zero-day attacks and emerging domain generation algorithms (DGAs) that rapidly rotate hosting locations (Heartfield & Loukas, 2016). To overcome these constraints, rule-based heuristics were introduced to examine web pages for specific malicious indicators, such as absent SSL certificates or suspicious DOM patterns (Babagoli et al., 2019). However, these rules rely heavily on manual feature engineering, high maintenance overhead, and are easily evaded through code obfuscation (Kathrine et al., 2019).

To address the rigidity of heuristic rules, researchers transitioned toward supervised classical machine learning (ML) paradigms capable of learning discriminative patterns directly from URL and webpage features (Alnemari & Alshammari, 2023; Tang & Mahmoud, 2021). Early efforts applied standalone classifiers, such as Support Vector Machines (SVM), Naïve Bayes, Decision Trees (DT), and Artificial Neural Networks (ANN), to lexical and structural web features (Omari, 2023; Sahingoz et al., 2019; Ul Hassan et al., 2022). Further improvements incorporated meta-heuristic optimization; for example, Alsenani (2023) and Saheed et al. (2020) demonstrated that coupling feature selection techniques (e.g., Genetic Algorithms) with baseline classifiers enhances predictive accuracy and computational efficiency across detection tasks.

Despite these advancements, both heuristic frameworks and classical ML models exhibit three persistent systemic limitations when deployed in real-world environments:

Susceptibility to Obfuscation and Infrastructure Shifts: Deterministic rules and single-model classifiers struggle when adversaries employ advanced visual rendering or dynamic hosting tactics that mask underlying source code (Ul Hassan et al., 2022).

Vulnerability to Feature Manipulation: Single-tree models and frequency-based classifiers assign high weights to specific feature occurrence thresholds (Omari, 2023). Attackers exploit this by injecting benign keywords or artificial DOM attributes to bypass decision boundaries.

Poor Out-of-Distribution Generalization: Models trained on isolated, single-source datasets fail to generalize across heterogeneous production environments characterized by evolving data distributions and unseen attack campaigns (Alnemari & Alshammari, 2023).

2.2 Critical Analysis of Data Resampling and Class Balancing Strategies

Since real-world network traffic is predominantly composed of benign interactions, class imbalance has become a major challenge in developing effective intrusion detection models. To address this issue, researchers commonly employ data resampling techniques to balance class distributions and reduce the tendency of classifiers to favor the majority class (Chawla et al., 2004; García et al., 2012). Among these techniques, the Synthetic Minority Over-sampling Technique (SMOTE) is one of the most widely adopted methods. The technique increases the representation of minority classes by generating synthetic samples along the line segments connecting neighboring minority-class instances (Chawla et al., 2002). Batista et al. (2004) and Haixiang et al. (2017) observed that though SMOTE improves class balance and mitigates the bias of empirical risk minimization toward the majority class, its independent application introduces several structural limitations into the training data. Because SMOTE generates synthetic instances without considering the global distribution of the data, it may produce samples within regions densely populated by the majority class. Consequently, this can introduce high-dimensional noise, increase the overlap between classes, and create ambiguous decision boundaries. The consequence is in reducing the model's generalization capability and degrading classification performance on unseen data (Khorshidi & Aickelin, 2025; Rout et al., 2018).

In addition to these geometric limitations, Haixiang et al. (2017) observed that the practice of performing data balancing before partitioning datasets into training and testing subsets is a serious methodological issue. When oversampling techniques are applied to the entire dataset prior to train–test splitting, synthetic samples generated from minority-class instances may be distributed across both the training and testing sets. This practice introduces data leakage because the model is evaluated on test samples that are derived from or closely related to instances encountered during training. As a result, the reported performance metrics become overly optimistic and fail to provide an accurate assessment of the model's ability to generalize to unseen data in real-world deployment scenarios.

2.3 Ensemble Learning and Class-Imbalance Handling in Phishing Detection

Recent research has increasingly adopted ensemble learning and gradient boosting techniques to overcome the high variance and stability limitations associated with standalone ML models (Ubing et al., 2019). Among these, XGBoost has gained prominence because of its scalability and regularized boosting mechanism, which reduces model complexity and mitigates overfitting. Studies have shown that integrating tree-based ensembles, such as RF, XGBoost, and LightGBM, with feature selection and data-balancing techniques enhances classification performance and robustness (Wilk-Jakubowski et al.,2025; Zheng et al., 2024; Ubing et al., 2019). Likewise, the Forest by Penalizing Attributes (FPA) algorithm further improves ensemble diversity and predictive accuracy (Alsariera et al., 2020).

Zamir et al. (2020) evaluated diverse ML algorithms for phishing detection and concluded that ensemble methods achieved better robustness and predictive capability than individual classifiers. Similarly, Adebowale et al. (2019) proposed an intelligent phishing detection framework integrating textual, image-based, and structural website features using ensemble learning techniques. Their study reported significant improvements in classification accuracy and stability across different datasets.

Despite the extensive application of ensemble learning and class-imbalance techniques in phishing detection, previous studies differ substantially in their data representations, preprocessing procedures, validation strategies, and treatment of class imbalance. In particular, performance comparisons are difficult to interpret when preprocessing or resampling is performed without a clearly separated model-development and independent evaluation process. The present study therefore focuses on establishing a common experimental protocol in which preprocessing, class balancing, cross-validation, and model optimization are incorporated into the training procedure, allowing the effect of SMOTE to be examined consistently across different tree-based learning approaches.

2.4 Contemporary Baselines: Deep Learning, NLP, and Graph-Based Extensions

Alongside tree-based ensemble methods, recent studies have increasingly explored Deep Neural Networks (DNNs), Natural Language Processing (NLP), and graph-based learning techniques for detecting complex and evolving cyber threats (Shaukat et al., 2023; Xiao et al., 2021). Deep learning (DL) models trained using optimization algorithms such as Adaptive Moment Estimation (ADAM) can automatically learn hierarchical feature representations and reduce reliance on manually engineered heuristics (Lakshmi et al., 2021). Yerima and Alzaylaee (2020) proposed a Convolutional Neural Network (CNN)-based phishing detection model that achieved high classification accuracy by automatically extracting patterns from URLs and webpage characteristics. Similarly, Alshingiti et al. (2023) combined CNN and Long Short-Term Memory (LSTM) architectures for phishing detection and reported improved detection capability against obfuscated phishing attacks. Elshewey et al. (2026) further investigated phishing detection using a hybrid CNN-VGG19 architecture optimized through Bayesian techniques, reporting high classification accuracy and reduced false detection rates.

NLP-based approaches have also been applied to analyse lexical and semantic characteristics of URLs and webpage content. Techniques such as Term Frequency–Inverse Document Frequency (TF-IDF) can be used to represent textual characteristics for malicious activity detection. For example, Shaukat et al. (2023) proposed a hybrid detection framework combining linguistic and hosting features to identify deceptive advertisements and malicious redirection chains. Benavides-Astudillo et al. (2023) similarly integrated NLP and DL techniques to analyse webpage content and URL semantics, reporting that semantic features can contribute to the detection of sophisticated phishing websites that mimic legitimate websites. Graph-based approaches provide another direction for modelling relationships among websites, domains, and other entities. Zhang et al. (2024), for example, introduced GrabPhisher, a Graph Neural Network (GNN)-based framework that exploits

dynamically evolving graph structures for phishing detection in decentralized environments. Xiao et al. (2021) further demonstrated the use of CNNs with multi-head self-attention mechanisms for identifying complex patterns in highly imbalanced intrusion-detection data.

These studies demonstrate the increasing diversity of modelling approaches used in phishing and related cybersecurity detection tasks. DL and graph-based methods can provide powerful representation-learning capabilities, particularly when large volumes of sequential, textual, visual, or relational data are available. However, their computational requirements, data representation requirements, and interpretability characteristics may differ substantially from those of conventional tree-based models. These differences are particularly relevant when phishing websites are represented using structured numerical features rather than raw webpage content, images, or graph relationships.

The reviewed studies are considered in this work to establish the broader research context rather than as direct experimental baselines. The experimental investigation is deliberately restricted to tree-based classifiers operating on structured numerical website features. Accordingly, the present study does not claim to establish superiority over CNN, LSTM, hybrid deep learning, NLP, or graph-based approaches. Instead, it focuses on the comparative behaviour of DT, RF, and XGBoost under a common experimental protocol and examines the effect of SMOTE on phishing-class detection.

This scope provides a focused basis for evaluating established tree-based methods within the selected structured-feature setting. It also allows the study to examine predictive performance alongside model interpretability through SHAP without introducing substantially different data representations or modelling paradigms. Nevertheless, the absence of direct experimental benchmarking against contemporary deep learning and graph-based approaches limits broader conclusions regarding the relative performance of the proposed framework. Such cross-paradigm evaluation, preferably using common datasets and consistent evaluation protocols, represents an important direction for future research.

The reviewed literature demonstrates that phishing website detection has evolved from conventional machine learning approaches to deep learning, natural language processing, and graph-based methods. These approaches, however, operate on different representations of website information. Tree-based classifiers such as Decision Tree, Random Forest, and XGBoost are particularly suited to structured tabular representations, whereas CNN-, LSTM-, NLP-, and graph-based approaches generally exploit spatial, sequential, textual, or relational representations, respectively. The present study focuses specifically on structured numerical representations of website characteristics. The dataset comprises engineered lexical, structural, and network-related attributes, and the experimental objective is to investigate the performance of tree-based classifiers within this tabular learning setting. Consequently, Decision Tree, Random Forest, and XGBoost were selected as representative models for the experimental evaluation. This scope does not imply that tree-based methods are universally superior to deep learning, NLP-based, or graph-based approaches. Rather, the study aims to provide a controlled comparison within the structured-feature setting, with particular emphasis on the influence of class imbalance and Synthetic Minority Oversampling Technique (SMOTE). Baseline and SMOTE-assisted models are therefore evaluated under a common data partitioning, cross-validation, hyperparameter optimization, and independent test-set evaluation protocol. The deep learning, NLP, and graph-based studies reviewed above are consequently used to position the proposed framework within the broader phishing detection landscape rather than as direct experimental baselines. Direct cross-paradigm comparison would require alternative data representations and model-specific experimental pipelines, which are outside the defined scope of the present study. Such comparisons are identified as an important direction for future research.

3.0 Methodology

In this section, the methodological framework followed in the detection of the phishing website through ML techniques is presented. As shown in Figure 1, the framework is composed of five stages: data acquisition, data preprocessing, model development and training, model evaluation, and results analysis. It is planned in each stage to obtain high performance of the model, an accurate classification and a reliable evaluation of the model.

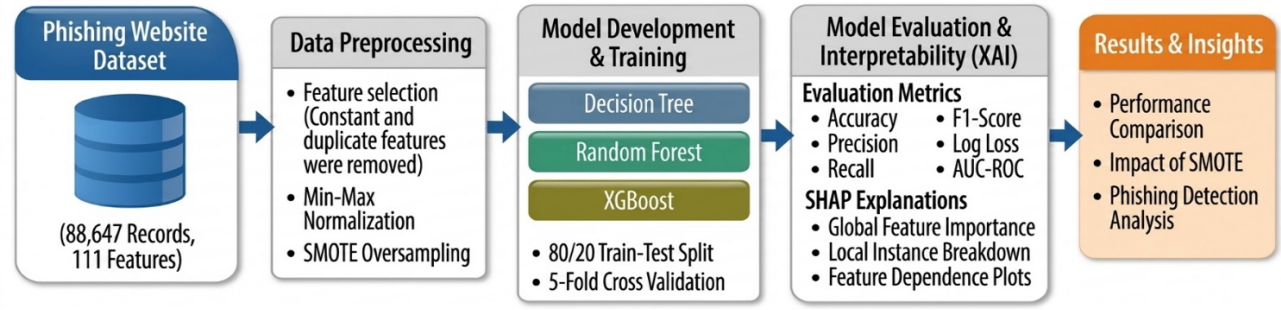

Figure 1: Workflow of the Machine Learning-Based Phishing Detection Framework

3.1 Dataset Description

The data used in this study was publicly available data collected by Vrbančič et al. (2020). This dataset, "Datasets for Phishing Websites Detection", is broadly utilized for phishing detection research and open data repositories, for example, GitHub. The data set comprises 88,647 examples, where each one is a website classified as phishing (1) or as legitimate (0). It originally has 111 features that represent different features of websites, summarised in Table 1 as three main categories: Lexical features (99 features), Host-based features (4 features), and Network-based features (8 features). These features come from various sources and may be classified into lexical features, host-based features and network-based features. These features are diverse and rich enough to serve as a basis for training and evaluating machine learning models for phishing detection, containing both structural and behavioural patterns of malicious and legitimate websites.

Table 1: Categorization and Description of Dataset Features

Feature Category	Structural Component / Type	Description & Example Features	Count
Lexical Features	URL Structure (Full URL, Domain, Directory, File, Parameters)	Character and symbol frequencies (qty_dot_*, qty_hyphen_*, qty_underline_*, qty_slash_*, qty_at_*, qty_percent_*), length metrics (length_url, domain_length, directory_length, file_length, params_length), parameter count (qty_params), and TLD presence (tld_present_params).	95
	Patterns & Heuristics	IP address embedded in domain (domain_in_ip), obfuscated terms (server_client_domain), email presence (email_in_url), and shortener usage (url_shortened).	4
Host-based Features	Domain & Server Info	Domain registration age (time_domain_activation), expiration window (time_domain_expiration), Autonomous System Number (asn_ip), and SSL/TLS certificate validity (tls_ssl_certificate).	4
Network-based Features	DNS & Protocol Metrics	Server response latency (time_response), SPF record presence (domain_spf), resolved IP count (qty_ip_resolved), nameserver count (qty_nameservers), MX record count (qty_mx_servers), Time-To-Live (ttl_hostname), redirect counts (qty_redirects), and search engine indexing status (url_google_index, domain_google_index).	8
Total			111

3.2 Data Preprocessing

Data preprocessing was performed to improve the quality, consistency, and suitability of the dataset for machine learning model development. The original dataset contained 111 numerical attributes representing different lexical, structural, and network characteristics of websites. A two-stage feature redundancy reduction process was performed to optimize the feature space and computational efficiency. First, columns with zero variance (cardinality ≤ 1) were removed because they provide no discriminatory information for the classification task. Consequently, 13 constant features, namely qty_slash_domain, qty_questionmark_domain, qty_equal_domain, qty_and_domain, qty_exclamation_domain, qty_space_domain, qty_tilde_domain, qty_comma_domain, qty_plus_domain, qty_asterisk_domain, qty_hashtag_domain, qty_dollar_domain, and qty_percent_domain, were eliminated. Subsequently, five duplicate features (qty_hashtag_directory, qty_slash_file, qty_questionmark_file, qty_hashtag_file, and qty_dollar_file) were removed because their values were identical to those of other features. These procedures reduced the feature space to 93 features while retaining all 88,647 observations.

To prevent information leakage, the resulting dataset was subsequently partitioned into training and independent test sets using an 80:20 stratified train-test split. The training set was used exclusively for model development, including feature scaling, class balancing, cross-validation, and hyperparameter optimization. In contrast, the test set was retained as an unseen dataset and was not used during model training or model selection.

Feature normalization was performed using the Standard Scaler technique. Importantly, the scaling parameters were estimated exclusively from the training data and subsequently applied to both the training and test data. This procedure transformed the features to a common range while preventing information from the independent test set from influencing the preprocessing stage.

The training data contained an imbalance between legitimate and phishing websites, with 58,000 legitimate and 30,647 phishing instances in the complete dataset. To address this imbalance, Synthetic Minority Oversampling Technique (SMOTE) was applied only to the training data. The independent test set was not oversampled and retained its original class distribution. SMOTE generated synthetic minority-class observations by interpolating between existing minority-class samples. In the training data, the minority phishing class was consequently increased to match the number of legitimate training instances, resulting in a balanced training distribution of approximately 46,400 instances per class.

For the cross-validation-based model selection, SMOTE was incorporated into the model-development pipeline such that oversampling was performed independently within each training fold. Thus, synthetic observations generated from one training fold could not influence its corresponding validation fold. This procedure prevents information leakage and provides a more reliable estimate of model performance on unseen data. The SMOTE procedure can be summarized as follows:

Step 1: Identify the Minority Class

Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ represent the training dataset, where $x_i \in \mathbb{R}^d$ denotes a d-dimensional feature vector and $y_i \in \{0,1\}$ represents the target class label. SMOTE isolates the minority class instances $S_{min} \subset D$, where $y_i = 1$ (phishing samples), from the majority class S_{maj}, where $y_i = 0$ (benign samples).

Step 2: Select *k*-Nearest Neighbours

For each minority class sample $x_i \in S_{min}$:

Compute the distance to all other minority class instances in S_{min} using Euclidean distance:

$$d(x_i, x_j) = \sqrt{\sum_{m=1}^{d} (x_{i,m} - x_{j,m})^2}$$

Identify the k-nearest neighbours within S_{min} (where $k = 5$).

Step 3: Generate Synthetic Samples

For each targeted minority instance x_i:

Randomly select one neighbour x_{nn} from its k-nearest minority neighbours.

Compute the feature difference vector δ:

$$\delta = x_{nn} - x_i$$

Draw a random scaling parameter λ from a uniform distribution:

$$\lambda \sim U(0,1)$$

Construct the synthetic sample x_{syn} along the line segment joining x_i and x_{nn}:

$$x_{syn} = x_i + \lambda \cdot (x_{nn} - x_i)$$

Step 4: Repeat Until Balance Is Achieved

This procedure is iterated until:

The total number of minority instances equals the majority class size ($|S_{min}| = |S_{maj}|$), achieving complete balance, or

The pre-specified sampling ratio $\beta = \frac{|S_{min}|}{|S_{maj}|}$ is attained.

3.3 Model Development

Three supervised machine learning algorithms, DT, RF, and XGBoost, were implemented and evaluated. These models were selected because they are suitable for structured, high-dimensional classification problems such as phishing website detection. DT was used as an interpretable baseline, whereas RF and XGBoost were used as ensemble learning approaches based on bagging and boosting, respectively.

The DT classifier recursively partitions the feature space according to feature values to maximize class separation. Although DT provides a relatively interpretable classification mechanism, excessive tree depth can result in overfitting. Therefore, parameters controlling tree complexity, including maximum depth, minimum samples required for splitting, and minimum samples required at leaf nodes, were considered during hyperparameter optimization.

RF extends the decision-tree approach by constructing multiple trees using bootstrap samples and randomly selected feature subsets. The predictions of the individual trees are aggregated to obtain the final classification. This ensemble mechanism reduces variance and generally provides greater robustness than a single decision tree. The principal RF hyperparameters, including the number of trees, maximum tree depth, minimum samples required for splitting and leaf formation, and the number of features considered at each split, were optimized using GridSearchCV.

XGBoost is a gradient-boosting ensemble method in which decision trees are constructed sequentially, with subsequent trees attempting to correct errors made by preceding trees. Its regularization and subsampling mechanisms can improve generalization and reduce overfitting. Accordingly, parameters including the number of estimators, maximum tree depth, learning rate, subsampling ratio, and column-sampling ratio were optimized.

For each model, two experimental conditions were considered: (i) baseline learning without SMOTE and (ii) SMOTE-assisted learning. In the baseline condition, the model was trained using the original training distribution. In the SMOTE condition, SMOTE was applied only within the training portion of each cross-validation fold. In both conditions, the independent test set remained completely untouched until final evaluation.

3.4 Training Strategy, Hyperparameter Tuning, and Evaluation Metrics

A leakage-controlled training strategy was adopted to ensure that the reported performance reflects the ability of the models to generalize to previously unseen websites. The complete dataset was first divided into training and test sets using an 80:20 stratified split, with 80% allocated to model development and 20% reserved exclusively for final evaluation.

A stratified 5-fold cross-validation procedure was subsequently applied to the training data. Within each fold, the training portion was used for feature scaling, SMOTE where applicable, and model fitting, while the corresponding validation portion remained independent of these operations. In the SMOTE condition, oversampling was therefore performed separately within each cross-validation training fold rather than before cross-validation. This arrangement prevents synthetic observations from being generated using information from validation data.

Hyperparameter optimization was performed using GridSearchCV with stratified 5-fold cross-validation strictly for hyperparameter tuning and model selection on the training data. The optimal configuration for each model was selected according to the mean validation F1-score obtained across the 5 cross-validation folds. The best-performing hyperparameter combination was subsequently used to refit the model on the full 80% training set under the corresponding pipeline before final evaluation.

The hyperparameter search spaces were defined to provide a balance between model flexibility, generalization, and computational efficiency. For DT, the principal parameters included criterion, maximum depth, minimum samples for splitting, and minimum samples per leaf. For RF, the search included the number of estimators, maximum depth, minimum samples for splitting and leaf nodes, and maximum features. For XGBoost, the search included the number of estimators, maximum depth, learning rate, subsampling ratio, and column-sampling ratio. The principal hyperparameter search configurations are presented in Table 2.

Table 2. Hyperparameter search spaces used for model optimization

Model	Hyperparameter	Values
Decision Tree	Criterion	Gini, Entropy
	max_depth	None, 10, 20, 30
	min_samples_split	2, 5, 10
	min_samples_leaf	1, 2, 4
Random Forest	n_estimators	100, 200, 300
	max_depth	None, 10, 20, 30
	min_samples_split	2, 5, 10
	min_samples_leaf	1, 2, 4
	max_features	sqrt, log2
XGBoost	n_estimators	100, 200, 300
	max_depth	3, 5, 7

	learning_rate	0.01, 0.05, 0.10
	Subsample	0.8, 1.0
	colsample_bytree	0.8, 1.0

Model performance was evaluated on the untouched test set using accuracy, precision, recall, F1-score, log loss, and AUC-ROC. Accuracy measured the proportion of correctly classified websites. Precision measured the proportion of websites predicted as phishing that were actually phishing, thereby reflecting the control of false-positive classifications. Recall measured the proportion of actual phishing websites correctly identified by the model and was therefore particularly important for assessing the effectiveness of the phishing detection system. F1-score provided a harmonic mean of precision and recall. Log loss was used to assess the quality of the predicted probabilities, with lower values indicating better probabilistic performance. Finally, AUC-ROC measured the ability of the classifier to distinguish between phishing and legitimate websites across different classification thresholds.

All final performance metrics reported were calculated using predictions generated from the independent 20% test set, which was held out throughout feature scaling, SMOTE oversampling, cross-validation, and hyperparameter optimization. Cross-validation was utilized solely as an internal tuning mechanism to prevent overfitting during parameter selection, whereas the holdout test set metrics represent the final evaluation of generalization capability.

3.5 Explainable Artificial Intelligence using SHAP

To make the suggested phishing detection framework more interpretable and transparent, this study includes XAI using SHAP in the framework. Ensemble models like RF and XGBoost are often not interpretable and produce very strong predictive results. To solve this, SHAP measures the importance of each feature according to its contribution to the model's predictions, which is based on cooperative game theory. In this study, the best-performing model is used for the SHAP analysis, which highlights and prioritizes the most important features for the phishing website detection task. It quantifies the overall effect of each feature on the dataset, providing a global interpretation of the model's behaviour, via the mean absolute SHAP values. This not only gives indications of the important features of phishing sites but also confirms that the model is learning patterns rather than random correlations. The inclusion of SHAP within this framework not only enhances predictive accuracy but also boosts the explainability of the models, fostering trust and adoptions of dependable cybersecurity solutions.

3.6 Experimental Setup and Computing Infrastructure

To ensure complete experimental reproducibility, all data preprocessing, model training, cross-validation, and SHAP interpretability workflows were executed within the Google Colab cloud computing environment running Ubuntu 22.04.3 LTS (Linux 6.1 x86_64). The underlying hardware infrastructure featured an Intel(R) Xeon(R) CPU @ 2.20GHz with two virtual cores, 12.7 GB of system RAM, and an NVIDIA T4 GPU equipped with 16 GB GDDR6 VRAM running CUDA 12.1. The core computational pipeline was built on Python 3.10.12, utilizing numpy (v1.25.2) and pandas (v2.0.3) for data manipulation and structure. Model development, hyperparameter tuning, and class imbalance handling were implemented using scikit-learn (v1.3.2), xgboost (v2.0.3), and imbalanced-learn (v0.11.0 for SMOTE oversampling), respectively. Model explainability and visual performance metrics were produced using shap (v0.44.1), matplotlib (v3.7.1), and seaborn (v0.13.1).

4.0 Results and Discussion

Three ML models namely, DT, RF and XGBoost were evaluated for their performance using the following metrics: accuracy, precision, recall, F1 score, log loss and AUC-ROC. The evaluation was done on the original imbalanced dataset (baseline) as well as the SMOTE balanced dataset. The results for the three models trained on the original imbalanced dataset are shown in Table 3 and Fig. 2. In general, the results demonstrate that ensemble models deliver much better performance than the single DT model. The best overall performance is obtained by the RF method, where the highest accuracy (97.1292 %), recall (96.0685 %), F1-score (95.8574 %), AUC-ROC (0.9951) are obtained, showing good classification ability and discrimination between phishing and legitimate websites. XGBoost is right next behind in terms of accuracy (96.9092%) and F1-score (95.5229%) as well as the lowest log loss (0.08471) which means that it gives better calibrated probability estimates. The DT, on the other hand, has the poorest performance on most of the metrics, especially in terms of log loss (1.7300), which indicates instability and a higher chance of overfitting. However, its accuracy (0.9520) and F1-score (0.9305) are still reasonable for its classification capabilities. In total, these results prove that ensemble learning methods work better than other single classifiers in detecting phishing problems, particularly when dealing with complex patterns in imbalanced data.

Table 3: Overall Model Performance (Baseline)

Model	Accuracy	Precision	Recall	F1-Score	Log Loss	AUC-ROC
DT	0.952002	0.931502	0.929527	0.930514	1.730014	0.946703
RF	0.971292	0.956472	0.960685	0.958574	0.096829	0.995125
XGBoost	0.969092	0.956792	0.953670	0.955229	0.084708	0.994741

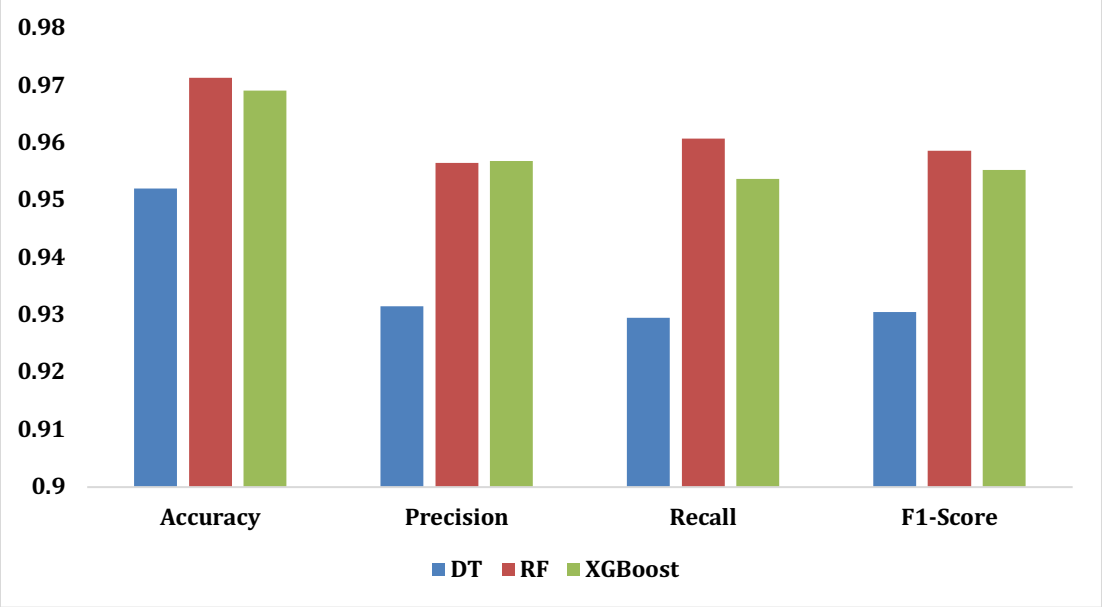

Figure 2: Graphical Performance of the Models (Baseline)

The results of the same models with SMOTE to overcome the class imbalance problem are shown in Table 4 and Figure 3. The increased recall for all models is the most significant effect of SMOTE, with DT's from 92.9527% to 93.3931%, RF's from 96.0685% to 96.4600%, and XGBoost from 95.3670% to 95.8075%. This means the models are more effective at correctly recognising phishing sites, the minority class. This improvement in recall comes with a slight loss of precision in RF and XGBoost, indicating a slight rise in false positives.

However, the F1-scores are reasonably consistent, suggesting the overall trade-off of precision and recall is maintained. Moreover, log loss seems to improve slightly with DT and RF, suggesting that their respective probabilities are estimated better, and AUC-ROC values are consistently high showing that they are not significantly affected by the use of SMOTE for the separation of classes.

Table 4: Overall Model Performance with SMOTE

Model	Accuracy	Precision	Recall	F1-Score	Log Loss	AUC-ROC
DT	0.952735	0.929685	0.933931	0.931803	1.703586	0.948302
RF	0.969882	0.949117	0.964600	0.956796	0.093548	0.995250
XGBoost	0.967738	0.949095	0.958075	0.953564	0.085880	0.994695

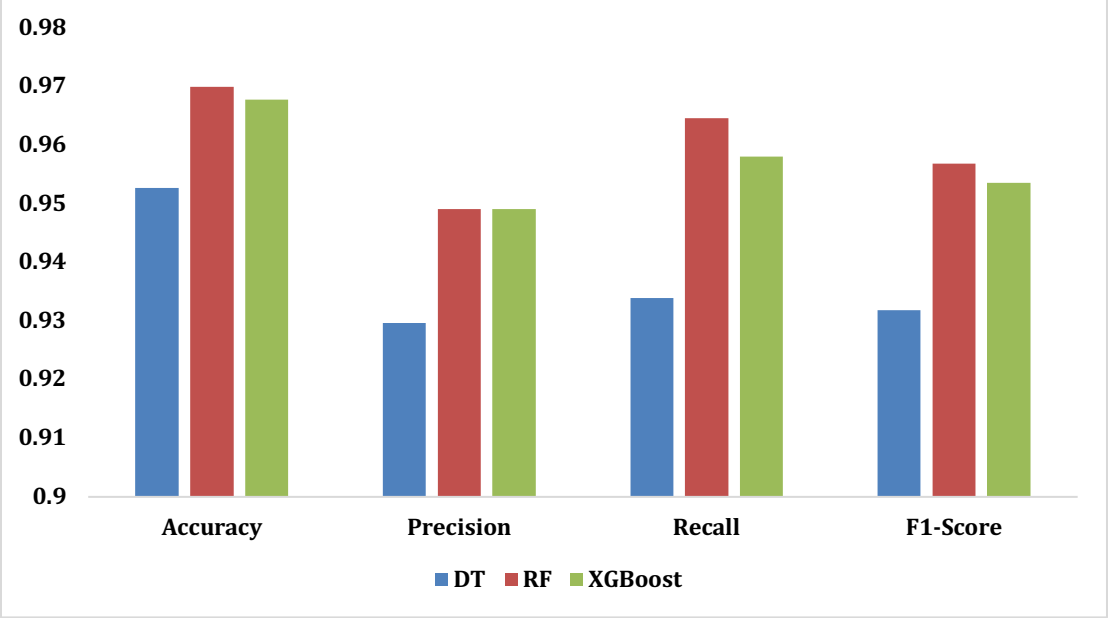

Figure 3: Graphical Performance of the Models with SMOTE

By comparing the results in both tables, the utilization of SMOTE leads to a better model performance, especially in recall. In both baseline and SMOTE cases, the ensemble models (particularly RF) still outperform the DT, and are robust to class imbalance and resampling effects. The trade-off between precision and recall is seen, especially for RF and XGBoost, but that is to be expected since we are seeking to improve recall, and so more false positives will be seen. This is fine in phishing detection because the chances of getting a phishing site false classified are much higher than getting a legitimate site wrong. Moreover, Figure 4 shows the confusion matrices of the DT, RF and XGBoost models for both baseline and SMOTE experiments, which could also be used as a visual confirmation of these results, showing how the true positive rates were improved and how the misclassification patterns were changed. The results confirm the use of ensemble learning combined with SMOTE can improve the performance of detecting phishing websites.

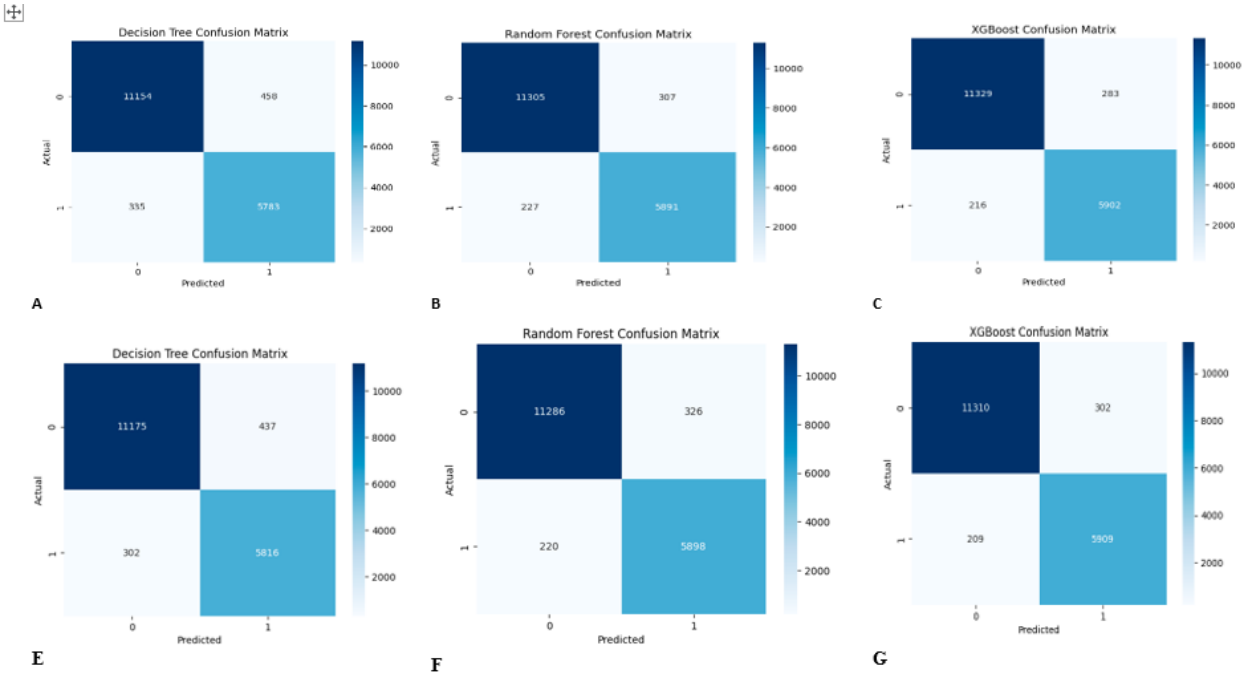

Fig 4: Confusion Matrices: (A-C) Baseline results for DT, RF, XGBoost; (D-F) Results with SMOTE for DT, RF, XGBoost

As shown in Figure 5, the AUC-ROC results of all three models were good for both baseline and SMOTE-balanced datasets. For all the models, RF has the best performance with AUC-ROC scores of 0.995125 and 0.99525 for the baseline and SMOTE datasets, respectively. XGBoost also achieved very high prediction accuracy with scores greater than 0.994 for both baseline and SMOTE experiments, and the comparatively lower but still good performance was achieved by DT. The use of SMOTE generated very little improvement, indicating that the models were already very good for dealing with the class distribution.

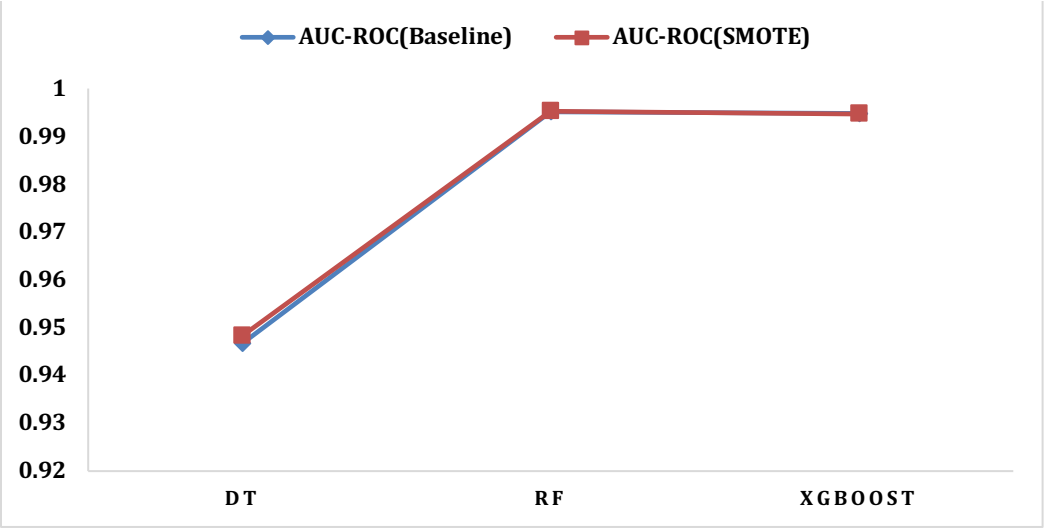

Fig 5: Comparison of AUC-ROC Performance Across Baseline and SMOTE Datasets

To explain the contribution of features in the best-performing model, the SHAP analysis was used with the model that was RF with SMOTE, as depicted in Figure 6. The results are shown as a SHAP summary bar plot, which ranks features by their mean absolute SHAP value (i.e.,

their average effect on model predictions). In comparison with all other features, time_domain_activation is the most influential feature - it contributes significantly more to the model than the rest of the features. This indicates that the length of time a domain has been activated is an important factor in identifying phishing websites, since it is likely that a malicious domain will be new and temporary. The second most important feature is directory_length meaning that longer or more complex URL directory structures are associated with phishing behaviour. In addition, several features of the URL are important such as qty_slash_url, length_url, and qty_slash_directory, indicating that phishing URLs are on average deeper and more structurally complex than non-phishing ones. Further, patterns of using special characters like qty_dot_domain, qty_underline_directory, and qty_dot_directory further underscore the significance of patterns of abnormal character usage in identifying malicious URLs. Other features such as ttl_hostname, file_length and time_domain_expiration are still included in the model but in a relatively lower contribution. These features are supplementary features associated with domain lifecycle and hosting features that can help improve classification. In general, the results of the SHAP analysis validate that the features related to the domain and the URL structure are also important in determining the model's predictive power.

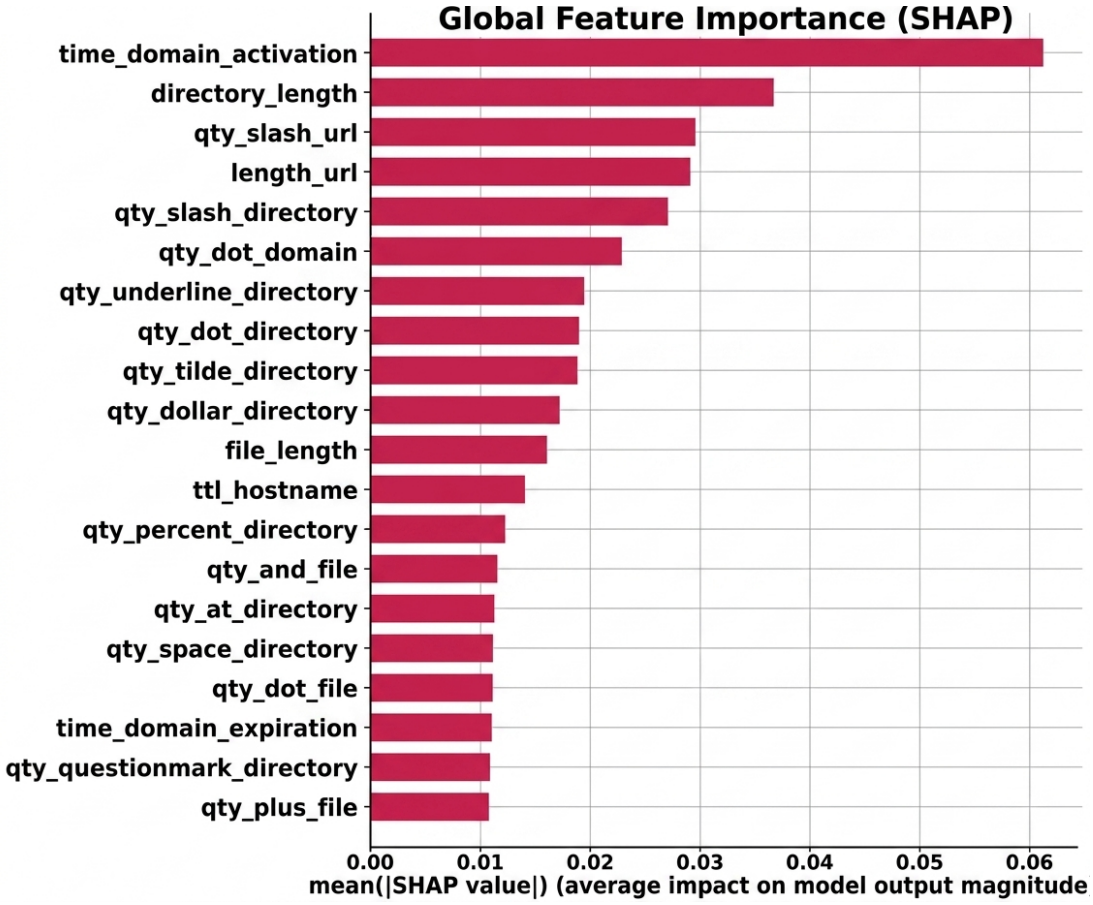

Fig. 6: SHAP Feature Importance for the Random Forest Model with SMOTE

The classification performance of the models using confusion matrices shows that ensemble methods, especially RF and XGBoost, always have higher true positive rates in identifying phishing websites. This ability is further enhanced by the application of SMOTE as illustrated in Table 3 with the number of correctly classified phishing instances and better recall values. There is a small increase in false positives, but the increase is relatively small, from 268 to 317 cases for RF (an increase of 49 false positives) and from 264 to 315 cases

for XGBoost (an increase of 51 false positives) after the application of SMOTE. Given the higher cost of undetected phishing attacks, this trade-off remains acceptable.

The quantitative results in Tables 2 and 3 show that ensemble models and RF are superior and exhibit the highest and most stable overall performance in both the baseline and SMOTE conditions. XGBoost also shows a good predictive ability especially in predicting the probabilities as it has a low log loss. Though the DT is effective to some extent, the robustness and generalisation performance are comparatively lower.

The analysis of SHAP provides a better understanding of why the RF model performs so well. The significance of domain age as a key indicator of phishing activity is highlighted by the prominence of time_domain_activation, consistent with known attacker behaviours involving short-lived domains. Likewise, the prominence of features associated with URL structure, such as directory length, the number of slashes, and counts of special characters, suggests that phishing websites frequently employ complex and irregular URL patterns to deceive users. This indicates that the proposed model not only achieves high predictive accuracy but also learns meaningful and interpretable patterns that reflect the characteristics of real-world phishing attacks.

The application of SMOTE resulted in a notable improvement in recall, demonstrating enhanced sensitivity to phishing instances. Furthermore, the SHAP analysis confirmed that these performance gains were driven by the meaningful contribution of relevant features rather than synthetic noise introduced during oversampling. Although a slight reduction in precision was observed, the balance between precision and recall remained stable, as reflected by the consistent F1-scores.

These findings are consistent with those reported by Safi and Singh (2023), who highlighted the effectiveness of ensemble learning in modelling complex, non-linear relationships within cybersecurity datasets. The integration of SMOTE further enhanced detection performance by mitigating class imbalance, while SHAP improved model transparency and interpretability. Collectively, these results demonstrate that combining ensemble learning, data resampling techniques, and Explainable Artificial Intelligence (XAI) provides an effective and interpretable framework for phishing website detection within the evaluated experimental setting. However, the use of a single dataset limits conclusions regarding generalizability across different phishing environments and attack patterns. Future research should therefore evaluate the framework on multiple independent and recent datasets, together with cross-dataset and temporal validation, to assess its performance under changing data distributions.

In addition, the present study does not experimentally evaluate the robustness of the proposed models against deliberate feature manipulation or adversarial evasion attacks. Although SHAP identifies influential features contributing to the model's predictions, this does not establish that these features remain reliable under adversarial manipulation. An attacker may potentially modify observable website characteristics while preserving the malicious functionality of a phishing website. Consequently, the reported results should not be interpreted as demonstrating resistance to adversarial attacks. Future research should systematically investigate plausible perturbations of influential features, evaluate prediction stability and attack success rates, and examine defensive mechanisms such as adversarial training, robust feature selection, and ensemble diversity.

Despite the promising classification performance obtained in this study, several limitations should be considered when interpreting the findings. First, the evaluation was conducted using a single publicly available phishing website dataset. Although this provides a consistent basis for comparing the evaluated models, a single dataset may not adequately represent the diversity of phishing websites, attack strategies, feature distributions, and operational environments encountered in practice. Consequently, the generalizability of the

findings to independent datasets and contemporary phishing environments remains to be established.

Second, the study does not include temporal validation using recently collected phishing websites. The dataset represents an earlier phishing environment, whereas phishing techniques, URL structures, domain practices, and website characteristics continue to evolve. The reported results should therefore not be interpreted as evidence that the models will maintain the same level of performance against emerging phishing campaigns. Future research should incorporate temporal validation and evaluation on recent datasets to assess model performance under changing data distributions.

Third, adversarial robustness was not experimentally evaluated. Although SHAP identifies influential features associated with model predictions, this does not establish that the models are resistant to deliberate manipulation of those features. Attackers may potentially modify observable website characteristics to evade detection while maintaining malicious functionality. Future studies should therefore investigate plausible adversarial perturbations, prediction stability, attack success rates, and defensive strategies such as adversarial training and robust feature selection.

Fourth, the methodological contribution of the study should be interpreted within the context of the use of established machine learning techniques. DT, RF, XGBoost, SMOTE, and SHAP are well-established methods, and the study does not introduce a fundamentally new learning algorithm or architecture. The contribution is instead centred on their controlled integration and comparative evaluation within a structured-feature phishing detection setting, including class-imbalance handling and model interpretability. Accordingly, the findings should not be interpreted as establishing algorithmic or state-of-the-art superiority over contemporary deep learning, NLP, or graph-based approaches.

Fifth, the study does not include dedicated deployment-oriented experiments measuring inference latency, throughput, memory consumption, or computational scalability under realistic operational conditions. Therefore, although the evaluated models demonstrate high predictive performance on the selected dataset, their suitability for real-time or large-scale operational deployment has not been experimentally established.

Finally, particular attention was given to preventing data leakage associated with SMOTE. Applying SMOTE before train-test splitting can allow information from observations subsequently used for testing to influence model development and can result in overly optimistic performance estimates. The revised experimental procedure therefore performs the data partition before oversampling and restricts SMOTE to the training data within the model-development process. This limitation highlights the importance of maintaining strict separation between training and evaluation data when applying synthetic oversampling techniques.

5.0 Conclusion

This study presented an integrated machine learning framework for phishing website detection based on structured website features, tree-based learning, class-imbalance handling, and explainable artificial intelligence. The framework incorporated feature preprocessing, including the removal of constant and duplicate features, normalization, and SMOTE-based class balancing. Decision Tree, Random Forest, and Extreme Gradient Boosting were evaluated using cross-validation and hyperparameter optimization within a leakage-controlled experimental procedure. The experimental results provide evidence that the evaluated ensemble models can achieve high classification performance on the selected benchmark phishing website dataset. RF achieved the highest observed overall performance among the evaluated models, while XGBoost also demonstrated competitive performance. The observed differences between the models and the changes associated with SMOTE were interpreted in conjunction with the statistical analysis rather than being treated solely as evidence of model superiority. In particular, the improvement in RF recall

following SMOTE should be understood within the context of the observed variability and statistical significance of the results. SHAP analysis provided global feature-level insight into the selected model's predictions and identified characteristics such as domain age, URL length, directory structure, and abnormal character usage as influential features. These findings provide an interpretable perspective on the characteristics associated with phishing classification and may inform future feature engineering and detection-rule refinement. Within the constraints of single-dataset evaluation and the structured-feature experimental setting, the findings provide evidence that tree-based ensemble learning combined with appropriate class-imbalance handling and SHAP-based interpretation can achieve high phishing website classification performance. However, the study does not establish universal superiority over deep learning or graph-based approaches, temporal robustness against emerging phishing attacks, resistance to adversarial manipulation, or scalability under real-world deployment conditions. Future research should therefore extend the evaluation to multiple independent and recent phishing datasets, conduct temporal and cross-dataset validation, investigate adversarial robustness, and compare the framework with contemporary deep learning and graph-based approaches under consistent evaluation protocols. Deployment-oriented studies examining computational efficiency, inference latency, memory requirements, and real-time detection capabilities would further establish the practical applicability of the framework.

Acknowledgment

The authors are grateful to Tertiary Education Trust Fund (TETFund), Nigeria for funding this research and the Management of the University of Uyo for creating a conducive environment for research.

References

[1] Adebowale, M. A., Lwin, K. T., Sanchez, E., & Hossain, M. A. (2019). Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text. Expert Systems with Applications, 115, 300-313. https://doi.org/10.1016/j.eswa.2018.07.067.

[2] Alabdan, R. (2020). Phishing Attacks Survey: Types, Vectors, and Technical Approaches. Future Internet, 12(10), 168. https://doi.org/10.3390/fi12100168

[3] Alnemari, S., & Alshammari, M. (2023). Detecting phishing domains using machine learning. Applied Sciences, 13(8), 4649. https://doi.org/10.3390/app13084649

[4] Alsariera, Y. A., Elijah, A. V., & Balogun, A. O. (2020). Phishing website detection: Forest by penalizing attributes algorithm and its enhanced variations. Arabian Journal for Science and Engineering, 45, 10459–10470. https://doi.org/10.1007/s13369-020-04812-7

[5] Alsenani, T. R., Ayon, S. I., Yousuf, S. M., Anik, F. B. K., & Chowdhury, M. E. S. (2023). Intelligent feature selection model based on particle swarm optimization to detect phishing websites. Multimedia Tools and Applications, 82(29), 44943-44975.

[6] Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., & Faheem, M. H. (2023). A Deep Learning-Based Phishing Detection System Using CNN, LSTM, and LSTM-CNN. Electronics, 12(1), 232. https://doi.org/10.3390/electronics12010232

[7] Asuquo, D. E., Umoh, U. A., Osang, F. B. and Okokon, E. W. (2020). Performance Evaluation of C4.5, Random Forest and Naïve Bayes Classifiers in Employee Performance and Promotion Prediction, African Journal of Management Information System, 2(4), 41-55.

[8] Babagoli, M., Aghababa, M. P., & Solouk, V. (2019). Heuristic nonlinear regression strategy for detecting phishing websites. Soft Computing, 23(12), 4315-4327.

[9] Balogun, A. O., Adewole, K. S., Raheem, M., Oluwatobi, A. N., Hamza-Usman,F. E., Mabayoje, M. A., Akintola, A., Asaju-Gbolagade, A. W., Muhammed, K. J., Gbenga, J. R., & Adeyemo, V. E. (2021). Improving the phishing website detection using empirical analysis of Function Tree and its variants. Heliyon, 7(7). https://doi.org/10.1016/j.heliyon.2021.e07437.

[10] Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD explorations newsletter, 6(1), 20-29.

[11] Benavides-Astudillo, E., Fuertes, W., Sanchez-Gordon, S., Nuñez-Agurto, D., & Rodríguez-Galán, G. (2023). A Phishing-Attack-Detection Model Using Natural Language Processing and Deep Learning. Applied Sciences, 13(9), 5275. https://doi.org/10.3390/app13095275.

[12] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357. https://doi.org/10.1613/jair.953

[13] Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. ACM SIGKDD explorations newsletter, 6(1), 1-6.

[14] Elshewey, A. M., Osman, A. M., El-Bakry, H. M., & Youssef, R. Y. (2026). An enhanced deep learning model for phishing detection based on bayesian optimization and hybrid CNN VGG19 model. Cluster Computing, 29(4), 231. https://doi.org/10.1007/s10586-025-05896-8

[15] García, V., Sánchez, J. S., & Mollineda, R. A. (2012). On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. Knowledge-Based Systems, 25(1), 13–21. https://doi.org/10.1016/j.knosys.2011.06.013

[16] Gupta, B.B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A. and Chang, X. (2021). A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. Comput. Commun., 175, 47-57. https://doi.org/10.1016/j.comcom.2021.04.023.

[17] Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. Expert systems with applications, 73, 220-239.

[18] Heartfield, R., & Loukas, G. (2016). A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks. ACM Computing Surveys, 48(3), 1–37. https://doi.org/10.1145/2835375

[19] Innab, N., Abdelgader, A., Awad, M., Abu-Zanona, M., Elzaghmouri, B., Zawaideh, F., & Alawneh, M. (2024). Phishing attacks detection using ensemble machine learning algorithms. Computers, Materials, & Continua, 80(1), 1325. https://doi.org/10.32604/cmc.2024.051778

[20] Jabir, R., Le, J., & Nguyen, C. (2025). Phishing Attacks in the Age of Generative Artificial Intelligence: A Systematic Review of Human Factors. AI, 6(8), 174. https://doi.org/10.3390/ai6080174

[21] Jain, A. K., & Gupta, B. B. (2021). A survey of phishing attack techniques, defence mechanisms and open research challenges. Enterprise Information Systems, 16(4), 527–565. https://doi.org/10.1080/17517575.2021.1896786

[22] Jha, A., & Jha, A. (2025). Phishing Forensics: A Systematic Approach to Analyzing Mobile and Social Media Fraud. https://doi.org/10.32604/jcs.2025.064429

[23] Kathrine, G.J.W., Praise, P.M., Rose, A.A., Kalaivani, E.C. (2019). Variants of phishing attacks and their detection techniques. Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019, 255-259. https://doi.org/10.1109/ICOEI.2019.8862697.

[24] Khorshidi, H. A., & Aickelin, U. (2025). A synthetic over-sampling method with minority and majority classes for imbalance problems. Knowledge and Information Systems, 67(7), 5965-5998.

[25] Lakshmi, L., Reddy, M. P., Santhaiah, C., & Reddy, U. J. (2021). Smart phishing detection in web pages using supervised deep learning classification and optimization technique ADAM. Wireless Personal Communications, 118, 3549–3564. https://doi.org/10.1007/s11277-021-08188-x

[26] Manzano, C., Meneses, C., Leger, P., & Fukuda, H. (2022). An empirical evaluation of supervised learning methods for network malware identification based on feature selection. Complexity, 2022(1), 6760920.

[27] Omari, K. (2023). Comparative study of machine learning algorithms for phishing website detection. International Journal of Advanced Computer Science and Applications, 14(9), 415–424. https://doi.org/10.14569/IJACSA.2023.0140945

[28] Rout, N., Mishra, D., & Mallick, M. K. (2018). Handling imbalanced data: A survey. International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications, 431–443. Springer.

[29] Saheed, Y. K., Hambali, M. A., Arowolo, M. O., & Olasupo, Y. A. (2020, November). Application of GA feature selection on Naive Bayes, random forest and SVM for credit card fraud detection. In 2020 international conference on decision aid sciences and application (DASA) (pp. 1091-1097). IEEE.

[30] Safi, A. & Singh, S. (2023). A systematic literature review on phishing website detection techniques. Journal of King Saud University - Computer and Information Sciences, 35(2), 590-611. https://doi.org/10.1016/j.jksuci.2023.01.004.

[31] Safitra, M. F., Lubis, M., & Fakhrurroja, H. (2023). Counterattacking Cyber Threats: A Framework for the Future of Cybersecurity. Sustainability, 15(18), 13369. https://doi.org/10.3390/su151813369

[32] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345-357. https://doi.org/10.1016/j.eswa.2018.09.029

[33] Shaukat, M. W., Amin, R., Muslam, M. M. A., Alshehri, A. H., & Xie, J. (2023). A hybrid approach for alluring ads phishing attack detection using machine learning. Sensors, 23(19), 8070. https://doi.org/10.3390/s23198070

[34] Tang, L., & Mahmoud, Q. (2021). A survey of machine learning-based solutions for phishing website detection. Machine Learning and Knowledge Extraction, 3(3), 672–694. https://doi.org/10.3390/make3030034

[35] Ubing, A., Kamilia, S., Abdullah, A., Zaman, N., & Supramaniam, M. (2019). Phishing website detection: An improved accuracy through feature selection and ensemble learning. International Journal of Advanced Computer Science and Applications, 10(3), 252–257Vrbančič, G., Fister, I., Jr, & Podgorelec, V. (2020). Datasets for phishing websites detection. Data in brief, 33, 106438. https://doi.org/10.1016/j.dib.2020.106438

[36] Ul Hassan, I., Ali, R. H., Ul Abideen, Z., Khan, T. A., & Kouatly, R. (2022). Significance of machine learning for detection of malicious websites on an automated framework under an unbalanced dataset. Digital, 2(4), 501–519. https://doi.org/10.3390/digital2040027

[37] Vrbančič, G., Fister, I., Jr, & Podgorelec, V. (2020). Datasets for phishing websites detection. Data in brief, 33, 106438. https://doi.org/10.1016/j.dib.2020.106438

[38] Wilk-Jakubowski, J. L., Pawlik, L., Wilk-Jakubowski, G., & Sikora, A. (2025). Machine Learning and Neural Networks for Phishing Detection: A Systematic Review (2017–2024). Electronics, 14(18), 3744. https://doi.org/10.3390/electronics14183744

[39] Xiao, X., Xiao, W., Zhang, D., Zhang, B., Hu, G., Li, Q., & Xia, S. (2021). Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets. Computers & Security, 108, 102372. https://doi.org/10.1016/j.cose.2021.102372

[40] Yerima, S. Y., & Alzaylaee, M. K. (2020). High accuracy phishing detection based on convolutional neural networks. In 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS) (pp. 1-6). IEEE. https://doi.org/10.1109/ICCAIS48893.2020.9096869

[41] Zamir, A., Khan, H. U., Iqbal, T., Yousaf, N., Aslam, F., Anjum, A., & Hamdani, M. (2020). Phishing web site detection using diverse machine learning algorithms. The Electronic Library, 38(1), 65-80. https://doi.org/10.1108/EL-05-2019-0118

[42] Zhang, J., et al. (2024). GrabPhisher: Phishing scams detection in ethereum via temporally evolving GNNs. IEEE Transactions on Services Computing, 17(2), 512–526.

[43] Zheng, Q., Yu, C., Cao, J., Xu, Y., Xing, Q., & Jin, Y. (2024). Advanced payment security system: XGBoost, LightGBM, and SMOTE integrated. arXiv preprint arXiv:2406.04658.