International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 13 Issue 9, September-2024, Impact Factor: 8.375

Artificial Intelligence Approaches to Explainability of Cybersecurity Threats Data Using SHAP

Nafea Alayyat Alanazi[1], Naif Ghazi Alotaibi[2]

[1]Department of Computer and Information Technology, Technical and Vocational Training Corporation, TVTC, Hafer Albatin, Technical College, TVTC KSA
[2]Department of Business Technology, Technical and Vocational Training Corporation, TVTC, Hafer Albatin Technical College, TVTC, Hafer Albatin, KSA

ABSTRACT

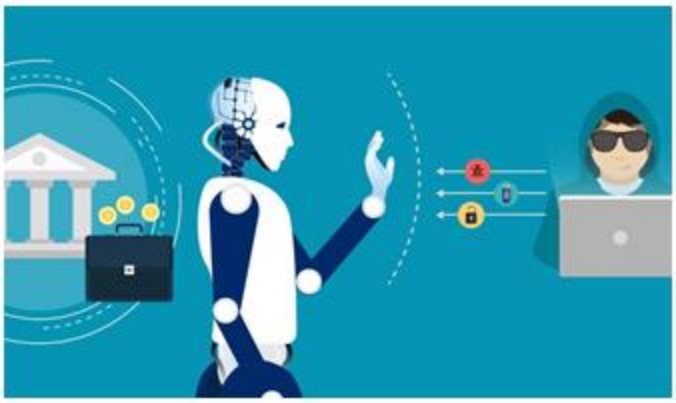
In today's digital era, cybersecurity threats represent a critical global challenge that must be addressed with urgency. With the widespread use of smart devices and the constant sharing of personal information through social media platforms, individuals are increasingly vulnerable to cyberattacks. Malicious URLs and malware are among the most common threats that can significantly harm everyday users. For the research community, the emergence of new attack types creates continuous difficulties in detection. Traditional approaches have largely focused on surveying malicious activity, while more recent advancements highlight the role of Artificial Intelligence (AI) in strengthening cybersecurity. AI-powered classification models, particularly those based on machine learning, are widely applied to identify and categorize cyberattacks with greater precision. Yet, beyond detection, one of the pressing challenges is understanding how these AI models arrive at their decisions. Model interpretability is crucial for ensuring trust, transparency, and actionable insights. In this regard, SHAP (Shapley Additive Explanations) methods—Tree SHAP, Deep SHAP, and Kernel SHAP—are increasingly recognized for their effectiveness in explaining AI-driven outcomes. In this study, two five-class cybersecurity datasets are analyzed using advanced AI algorithms, including Random Forest, XGBoost, and Keras Sequential models. To enhance explainability, three SHAP techniques are employed to interpret and validate the outputs of these AI-based models, ultimately providing a deeper understanding of cybersecurity threats and supporting more reliable defense mechanisms.

Index Terms—SHAP, XGBoost, RFC, Sequential, Explain, TreeShap, KernelShap, DeepShap

INTRODUCTION

The rapid evolution of the web has introduced remarkable advancements in digital communication. However, despite this exponential growth, the online environment continues to encounter persistent challenges, most notably cyber threats. Reports of financial fraud are increasingly common, with hackers exploiting personal data often extracted from social media platforms to facilitate criminal activities [1]. As technology advances, cybercriminals develop increasingly sophisticated penetration strategies, constantly experimenting with innovative attack techniques. This dynamic landscape compels information technology specialists and digital users to remain vigilant, yet such vigilance demands substantial time and resources. Consequently, there is a pressing need for artificial intelligence–driven protective mechanisms capable of proactively safeguarding online users from malicious attacks [2]. Given the fluidity of technological change, organizations and individuals must adopt continuous, AI-enhanced defensive measures to secure their digital assets.

One of the major challenges for cybersecurity researchers is predicting attack risks with precision. Developing AI-powered, reliable, and secure analysis systems is critical to fortifying digital infrastructures. Proactive prevention strategies, such as automated scans, have already demonstrated their value in minimizing the impact of cybercrime [3]. For instance, Mamun et al. [4] proposed AI-based approaches that detect and classify malicious URLs according to attack types using lexical analysis. Similarly, researchers have concentrated on leveraging AI in malware detection, such as identifying Android malware variants [5]. In the highly dynamic Android ecosystem, AIdriven safeguards are increasingly essential to block malicious code infiltration. Nevertheless, without a comprehensive understanding of malware infection patterns, building effective AI-based mitigation solutions remains challenging. A lack of structured defenses inevitably compromises the security of online databases and sensitive information.



This study focuses on two widely recognized datasets of cyber-attacks: malicious URLs and Android malware. To enhance transparency in AI-driven cybersecurity, explainable artificial intelligence (XAI) is integrated using Shapley Additive Explanations (SHAP). Three Machine Learning (ML) models—Random Forest Classifier (RFC), XGBoost Classification, and a Keras Sequential algorithm—are implemented to predict, analyze, and assess performance against these datasets. Through this AI-centric systemic analysis, researchers can identify critical infiltration patterns and gain insight into the underlying causes of attacks.

Equally important is the interpretability of AI predictions, which helps distinguish the strengths and weaknesses of different models. SHAP facilitates this by quantifying the contribution of each feature to the model's decision, thereby offering deeper clarity into AI-driven outcomes. Furthermore, SHAP enables the removal of irrelevant information from the vast and complex intelligence processed by ML systems. Using TreeShap, KernelShap, and DeepShap, this study provides detailed explanations of model outputs across five attack categories, with features representing the causal factors behind these threats.

The remainder of this paper is structured as follows. Section II reviews related studies, particularly AI-driven approaches to cybersecurity. Section III describes the proposed methodology, highlighting the ML algorithms applied in conjunction with SHAP. Section IV presents the experimental setup, results, and SHAP-based interpretability analysis. Finally, Section V summarizes the study's contributions, emphasizing the role of AI and explainability in strengthening cybersecurity defenses.

RELATED WORK

Most studies in the cybersecurity domain have traditionally emphasized identifying risks, reporting cybercrime incidents, and assessing their broader impacts on individuals and organizations. Nurse [2] introduced a conceptual framework to understand cybercrime and the associated societal challenges, highlighting the recurring risks across digital platforms. Different attack strategies used by cybercriminals, such as malware deployment, account hijacking, and malicious URL injections, were discussed in detail. In [6], Artificial Intelligence (AI) was integrated into cybersecurity through machine learning (ML), where models based on Random Forest and Decision Tree algorithms were applied to detect attacks in real-time, achieving superior performance.

In [7], the authors explored Explainable AI (XAI) by applying Ontology Graphs (OG) and transfer learning to simulate human cognitive processes and thereby enhance the interpretability of model outputs. Their findings suggested that the primary challenge lies in ensuring explainability within AI systems, which is fundamental for future applications. This work demonstrated that reliable interpretability enables more accurate decision-making while strengthening both validity and reliability of outcomes. Similarly, Fernando et al. [8] implemented two XAI techniques, DeepSHAP and LIME, to explain Neural Retrieval Models (NRMs) in text-based ad hoc search tasks. Their results indicated that DeepSHAP offered more precise term-level insights, whereas LIME concentrated on highlighting only the most prominent features.

Further advancement was reported by Ibor et al. [9], who applied unsupervised and supervised AI-driven learning techniques to detect threats through hyper-alert mapping into class based risk categories. According to [10], Lakshmanaprabu et al. used Random Forest Classifiers (RFC) for e-health data classification, reaching a precision of 94.2
Chen et al. [12] introduced the use of XGBoost, an AI model that incrementally improves classification by focusing on prior errors, thereby addressing large-scale real-world problems with minimal resource consumption. Complementary work by Ho [13] highlighted KernelSHAP as a method to interpret RNN predictions, where the computational efficiency allowed

explanations to be generated in under ten minutes. Importantly, these explanations aligned with clinical expectations across multiple analysis scales (individuals, cohorts, and populations).

Building on these contributions, this paper applies three SHAP-based AI methods (TreeSHAP, KernelSHAP, and DeepSHAP) to interpret the predictions of RFC, XGBoost, and Sequential models. These explainability strategies not only enhance the transparency of AI-based cybersecurity models but also enable the mapping of threat alerts across web and social media platforms. Additionally, the comparative performance and accuracy of these models are systematically evaluated.

PROPOSED APPROACH

In this section, Artificial Intelligence (AI) techniques are emphasized through the use of Machine Learning (ML) models and the application of the SHAP explainability method. Specifically, RFC, XGBOOST, and SEQUENTIAL AI-driven models are introduced and described. Following this, the SHAP approach is detailed, highlighting how it is applied to interpret the outcomes of these AI models when executed on two cybersecurity data sets.

A. Random Forest Classifier

In the context of Artificial Intelligence–based cybersecurity analytics, the Random Forest Classifier (RFC) is applied together with the TreeShap explanation method, which is one of the ensemble approaches widely recognized for enhancing the interpretability of AI-driven predictions [14]. Within RFC, three key parameters guide the decision-making process. The first is the splitting criterion, where gini evaluates the quality of a split using the Gini index as expressed in Equation (1). This index leverages class probabilities to calculate the Gini value for each node branch, thereby indicating which outcome is more likely to occur. Here, P_i denotes the relative frequency of a class in the dataset, and C refers to the total number of classes. Alternatively, the entropy criterion uses the probability distribution to determine branching in decision trees, as described in Equation (2). For the AI model optimization, Gini was selected due to its bounded range (0 to 0.5), which offers greater interpretability compared to Entropy (0 to 1) [11]. The other important hyperparameters include “max depth”, which regulates the depth of the decision trees, and “n estimators”, which defines the number of trees used in the forest, directly influencing the strength and stability of the AI ensemble model.

$$Gini = \mathbf{1} - \sum_{i=\mathbf{1}}^{C} (P_i)^{\mathbf{2}} \quad (1)$$

$$Entropy = \sum_{i=\mathbf{1}}^{C} -P_{\mathbf{i}} * \mathbf{log_2}(P_{\mathbf{i}}) \quad (2)$$

In Artificial Intelligence, the Random Forest Classifier (RFC) represents a learning approach that builds predictive models by drawing random subsets from a dataset. For each subset, an individual decision tree is constructed, and AI algorithms then generate prediction outcomes from these trees. The final classification is determined through an ensemble mechanism, where the prediction with the highest number of votes across all decision trees is selected as the output result [15].

B. XGBoost Classification

Artificial Intelligence methods are applied by utilizing XGBoost classification in combination with KernelSHAP for interpretability. XGBoost, an advanced ensemble learning algorithm, operates within the gradient boosting framework and is widely recognized in AI applications for its high efficiency and scalability [16]. As an evolution of extreme gradient boosting, XGBoost leverages second-order gradients along with advanced regularization techniques to enhance predictive accuracy. This makes it a powerful AI-driven model capable of handling complex datasets and generating precise approximations. In the boosting process, initial predictions and corresponding errors are iteratively refined, where AI models are trained using independent variables and residual errors to produce improved predictions [12]. The approach involves optimizing three key parameters:

1) binary logistic: logistic regression is used for binary classification
2) output probability is max depth =10 is the maximum depth of a tree
3) n estimators=800 is the number of boosting rounds [16]

The model undergoes training and evaluation, producing predictions.

C. Sequential Model

In the Artificial Intelligence framework, a sequential deep learning model is implemented as a classifier, structured as a linear stack of interconnected layers. Each layer processes an input tensor and generates an output tensor, enabling hierarchical feature learning.

The initial layer employs the ReLU (Rectified Linear Unit) activation function, which enhances non-linearity and accelerates training performance [17]. ReLU is formally expressed in Equation (3). At the final stage of the network, the softmax activation function is applied, allowing the model to perform probabilistic multi-class classification by extending the effect of the initial transformation [17]. Softmax is represented in Equation (4).

Once the AI-driven architecture is established, the system specifies the loss function, optimizer, and evaluation metrics that guide the training process and improve prediction accuracy.

$$F(x) = (0,x) \quad (3)$$

$$Softmax(x_j) = \frac{e^{(x_i)}}{\sum_i e^{(x_i)}} \quad (4)$$

D. Introduction to SHAP

Shapley values were utilized as the foundation of SHapley Additive exPlanations (SHAP) to quantify the influence of individual features in AI-driven predictions. By design, Shapley values evaluate all possible prediction outcomes for a given instance across every possible subset of input features. This exhaustive computation ensures that SHAP inherits desirable theoretical properties such as consistency and local accuracy [18]. The formal definition of the Shapley value is expressed in Equation (5) where $s \subseteq F$ represents all feature subsets, F is the set of all features, $f_{S \cup i}$ is trained with that feature present, f_S is trained with the feature withheld. Then, predictions from the two models are compared based on the current input

$[f_{S \cup i}(x_{S \cup i}) - f_S(x_S)]$, where (x_S) represents the values of the input features in set S [19].
(5)

$$\emptyset_i = \sum_{S \subseteq \mathbf{F}\{\mathbf{i}\}} \frac{|\mathbf{S}|!(|\mathbf{F}| - |\mathbf{S}| - \mathbf{1})!}{|\mathbf{F}|!} [f_{S \cup i}(x_{S \cup i}) - f_S(x_S)]$$

SHAP takes three steps [20] which are: (1) computation of Shapley value explanations, (2) capture feature interactions by extending local explanations, and (3) interpreting global model structure based on many local explanations by defining desirable properties.

E. TreeExplainer of SHAP

In this study, Artificial Intelligence techniques are integrated through the use of the TreeExplainer in conjunction with the Machine Learning Random Forest Classifier (RFC). This combination enables efficient and interpretable computation of optimal local explanations by leveraging the explainability framework defined by SHAP. Within this AI-driven process, the decision path is decomposed into feature-level components, allowing each attribute's influence on the prediction to be clarified. Consequently, the model's prediction y is explained as the cumulative additive contributions of features along the decision path, with the explanation formally represented in Equation (6).

$$y = bias + \sum_{m=1}^{M} feature\ contribution\{m,x\} \quad (6)$$

m=**1** where *bias* is the contribution of root node and feature contribution, $\{m,x\}$ is the contribution of feature m in predicting the outcome corresponding to an input x [21].

F. KernelExplainer of SHAP

The AI-powered KernelExplainer is applied to the machine learning XGBOOST classifier model. It leverages artificial intelligence techniques to perform a local regression, utilizing the model's prediction method and the underlying data in order to generate SHAP values, which represent the contribution and importance of each feature as Shapley values. Within this AI-driven process, the KernelExplainer relies on two key parameters: (1) *predict proba*() is used to retrieve the probabilities of each target class; (2) *link = Logit* is a function to make the feature importance values have logodds units [22]. The approximation in Equation (7) is used to evaluate the conditional expectation [23].

$$f_{T,KernelSHAP}(\mathbf{x}) \approx \frac{1}{K} \sum_k f(x_T, x_T^k) \quad (7)$$

where x^k_T, k=1,...,K are samples from x_T.

G. DeepExplainer of SHAP

In the context of Artificial Intelligence, the DeepExplainer is employed alongside the ML Sequential model as an advanced interpretability technique. It builds upon the DeepLIFT algorithm (Deep SHAP), enhancing explainability by approximating the conditional expectations of SHAP values through the use of diverse background samples [19]. This AI-driven method integrates across numerous background datasets, enabling more reliable estimation. Specifically, DeepExplainer computes approximate SHAP values by measuring the difference between the expected model outputs on the selected background samples and the actual outputs of the current AI model, as expressed in Equation (8).

$$Difference = f(x) - Exp(f(x)) \quad (8)$$

where $f(x)$ is current outputs, and $Exp(f(x))$ is the expected output.

EXPERIMENTS AND RESULTS

In this section, the two data sets are used and described followed by the explanation of the outputs of the different SHAP explainer algorithm that are used based on the ML models. Furthermore, the accuracy of the resulting models are analyzed.

A. Data Set Description

Two cybersecurity data sets were obtained from the Canadian Institute for Cybersecurity (UNB) [24], and both were employed to evaluate the role of Artificial Intelligence in enhancing explainability and detection of threats.

The first dataset, "URL data set (ISCX-URL2016)", is labeled as malicious URLs. It consists of 36,707 records with 80 attributes (79 input features and one target column). The target column represents five URL attack categories: Benign, Spam, Phishing, Malware, and Defacement. AI-driven analysis and classification of these features enabled the identification and explanation of the behavioral patterns behind malicious URLs across the five classes.

The second dataset, "CICMalDroid 2020", referred to as the Android Malware dataset, contains 11,598 APK samples with 470 attributes (469 extracted features plus one target column). The target column categorizes the data into Adware, Banking malware, SMS malware, Riskware, and Benign classes. The features encompass system calls, binders, and composite behaviors, which are critical in modeling Android malware.

AI-based experiments were conducted to interpret and detect obfuscation techniques impacting each URL attack type and to recognize Android malware by analyzing system calls, binders, and behavioral patterns. Leveraging AI explainability techniques allowed the results to be presented in a transparent and user-friendly manner, ensuring that end users could clearly understand the underlying factors driving each threat.

B. Results - Tree-Model Explanation

A SHAP bar chart is utilized to illustrate the features that affect the classification within the dataset. The importance of features is arranged in descending order according to their influence on the prediction of each class. Consequently, the primary function of the Tree-Explainer is to provide interpretation by clarifying how each feature contributes to the model's output for the target classes, based on the test sets of both datasets.

1) Malicious URLs Data Set: Figure 1 illustrates the key SHAP features that influence different classes of attack-type URLs. As depicted, the colors indicate the obfuscation technique features represented by the SHAP values. For instance, the obfuscation feature named domain token count," which appears at the upper portion of the plot, has a stronger association with the Benign" and Spam" URL classes compared to the other categories. In contrast, features shown at the lower portion of the plot, such as fileNameLen," predominantly impact the Malware URLs" class more than the remaining URL categories. Similarly, the urlLen" attribute also exerts greater influence on the Malware" class compared to the others. Overall, based on the relative impact of the features, the Spam" class, represented by the blue color, emerged as the most frequently identified.

2) Android Malware Data Set: Figure 2 illustrates the SHAP values used to determine which features contributed to malware classification across the different classes, each represented by distinct colors. For instance, the feature pread64," positioned at the top of the plot, had a stronger influence on the SMS malware" class compared to the others. Conversely, the getSubscriberId" feature, found toward the lower portion of the plot, was more influential in classifying Riskware" than in other malware categories. Overall, the plot highlights that the class "SMS malware," depicted in blue, appeared as the most frequent outcome.

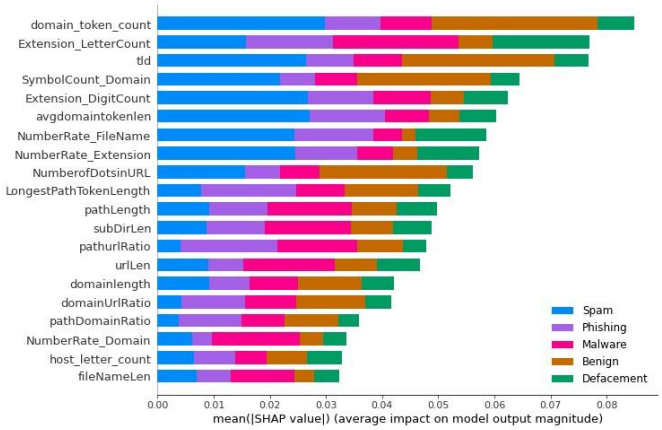


Figure 1. Five-Class Tree-SHAP of Malicious URLs

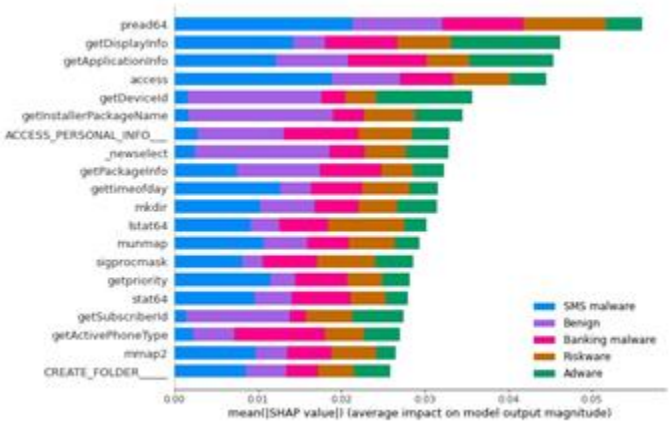


Figure 2. Five-Class Tree-SHAP of Android Malware

C. Results - Kernel-Model Explainer

In the subsequent experiments, SHAP is applied to interpret a single class using the Dot-SHAP approach. This technique

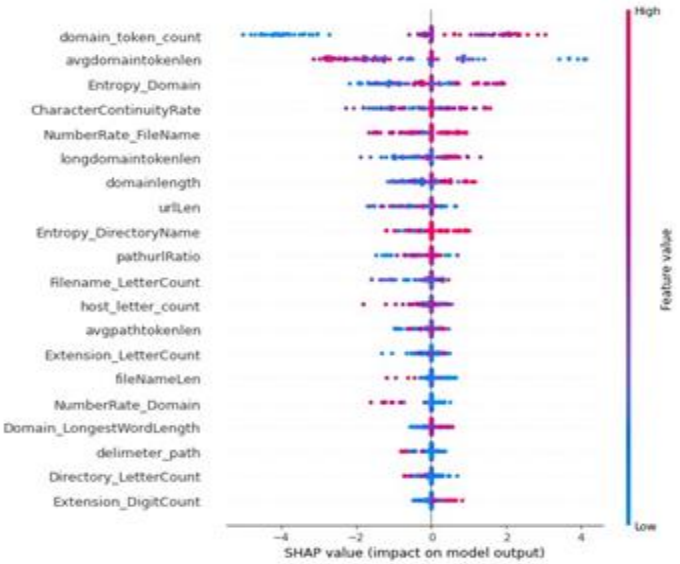


Figure 3. Five-Class Kernel SHAP of Malicious URLs

illustrates the outcomes through a visualization that blends scatter plotting with density estimation, where dots accumulate when they overlap or fail to fit. The color of the dots indicates the mean feature value at that location: red dots generally denote higher feature values, whereas blue dots indicate lower values, depending on whether the dots fall to the left or right of the vertical axis. Through this visualization, Dot-SHAP conveys both the positive and

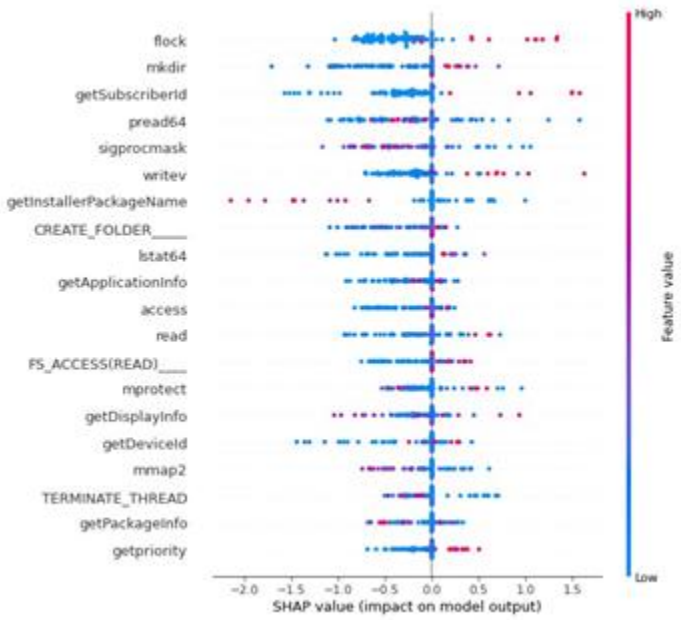


Figure 4. Five-Class Kernel-SHAP of Android Malware

are represented in red, while the "positive" effect is indicated along the horizontal axis. The horizontal placement demonstrates how this impact corresponds to either an increase or decrease in prediction, depending on the scale. In contrast, the feature "avgdomaintonkelen" exhibits a negative correlation with the "Spam" class. Likewise, larger values of "EntropyDirectoryName" markedly elevate the probability of a "Spam" prediction, whereas greater values of "fileNamemLen" contribute to a lower likelihood of predicting "Spam".

Android Malware Data: Figure 4 illustrates that the feature "clock" exerts a strong positive influence on the class SMS malware," as indicated by the red color along the horizontal axis, thereby increasing the probability of predicting SMS malware." In contrast, higher values of the feature "pread64" reduce the likelihood of classifying an instance as SMS malware." Similarly, the features "getSubscribed" and "writev" contribute to a greater probability of predicting the class SMS malware." On the other hand, the features "sigprocmask" and "getinstallerPackageName" have the opposite effect, lowering the chances of predicting the class "SMS malware." negative associations between predictors and the target class, grounded in the training data. Furthermore, features are ordered in descending importance according to the magnitude of their influence on predicting the class.

1) Malicious URLs Data Set: Figure 3 illustrates that the feature “domain token count” exerts a strong positive influence on predicting the class “Spam”. Higher values of this feature significantly raise the likelihood of classifying an instance as “Spam”. In the visualization, “high” values

D. Results - Deep-Model Explainer

In the next section, SHAP is applied to interpret a single class through the Force Plots-SHAP technique. This visualization illustrates how each feature contributes to shifting the model output from the base value prediction—defined as the average predicted result across the full training dataset—towards the actual output of the target class. The x-axis is centered on the expected feature’s contribution, which directly impacts the target class. Features that push the prediction upward are highlighted in red, whereas those that decrease the prediction are displayed in blue. The direction of influence, whether towards higher or lower values, depends on the comparison between the output value (f(x)) and the base value. When the output value exceeds the base value, red features move the prediction further to the right (higher), and the opposite occurs if it is less. Additionally, the most significant feature effects are displayed at the bottom of the plot.

1) Malicious URLs Data Set: Figure 5 illustrates how various features contribute to shifting the model's output toward the class Spam" relative to the base value. The output value (f(x)) is 0.73, which exceeds the base value of 0.5647. As a result, the red-colored features associated with the Spam" class push the prediction to the right (towards higher values). For instance, the feature Querylength" positively influences the Spam" prediction, as indicated by its placement in the red area, thereby driving the output further right. In contrast, the feature charcompvowels," represented in the blue area, has a negative effect on the Spam" class, shifting the prediction left (towards lower values). Nonetheless, the overall force pushing the output to the right is stronger, since the magnitude of the red-area feature contributions outweighs that of the blue-area features.

2) Android Malware Data Set: Figure 6 illustrates the contribution of each feature in shifting the model's prediction for the class SMS malware" away from the base value. The output value (f(x)) is 0.72, which exceeds the base value of 0.5881. Features highlighted in red, associated with the class SMS malware," push the prediction toward the right (indicating a higher value). For instance, the feature FS ACCESS(READ) " in the red zone exerts a force that drives the prediction of SMS malware" to the right (higher value). In contrast, the feature "FS ACCESS(CAREATE APPEND) " in the blue zone pulls the prediction leftward (toward a lower value). However, the stronger influence comes from the red region, as its feature values dominate those in the blue region, thereby driving the prediction more strongly toward the right (higher value).

E. ML Model Performance

In the following section, the confusion matrix of each model is discussed, along with the classification result tables of the ML models and their corresponding ROC curves. This is then followed by a consolidated summary of the outcomes of all applied ML models.

1) Confusion Matrix: The confusion matrices corresponding to the five classes across all ML models for both datasets are presented in Figures 7 through 12.

• Confusion Matrices of Malicious URLs Data Set:

The confusion matrix in Figure 7 illustrates that a total of 3,677 instances were accurately classified across all five categories by the RFC model applied to the Malicious URLs data set. Among these, the Spam category recorded the highest correct classifications with 1,091 samples. This was followed by the Malware category with 854 correctly identified samples, the Phishing category with 781, the Benign category with 508, and the Defacement category with 443. The matrix also indicates that each class contained a small number of misclassified instances.

Figure 8 presents the Confusion Matrix of the XGBOOST model on the Malicious URLs dataset. The model successfully identified 3,722 instances across all five categories. Specifically, the Spam class recorded 1,108 correct predictions, while the Defacement class had the fewest with 449 accurate classifications. The remaining results were 868 correct samples for Malware, 787 for Phishing, and 510 for Benign.

Similarly, Figure 9 illustrates the Confusion Matrix of the Sequential model on the Malicious URLs dataset. This model correctly classified a total of 3,546 instances across the five categories. The distribution of correct predictions was as follows: Spam with 1,095 samples, Malware with 814, Phishing with 722, Benign with 482, and Defacement with 433 correctly classified cases.

Overall, the XGBOOST model demonstrated superior performance by achieving the highest number of correct classifications (3,722) among all tested ML models. Additionally, it is noteworthy that the Spam class consistently achieved the largest number of accurate classifications when compared across all models applied to the dataset.

• Confusion Matrices of Android Malware Data Set:

In Figure 10, the confusion matrix illustrates that 2,183 samples were accurately identified across all five categories of the RFC model applied to the Android Malware data set. The SMS malware class recorded the largest number of correct classifications with 807 samples, followed by the Riskware class with 467, the Bank malware class with 373, and the Benign class with 307. The Adware class had the smallest number of correct classifications, totaling 229 samples.

The confusion matrix in Figure 11 illustrates that the XGBOOST model correctly identified 2,197 samples across all five categories of the Android Malware dataset, as shown along the diagonal. Specifically, it classified 810 instances of the SMS malware class, 473 instances of the Riskware class, 381 instances of the banking malware class, 306 instances of the Benign class, and 227 instances of the Adware class.

Similarly, Figure 12 presents the confusion matrix for the Sequential model applied to the Android Malware dataset, showing that a total of 1,986 samples were correctly classified. These include 790 for SMS malware, 407 for Riskware, 351 for banking malware, 240 for Benign, and 198 for Adware.

From these results, it is evident that the XGBOOST model achieved the best performance, as it produced the highest number of correct classifications (2,197 samples) compared

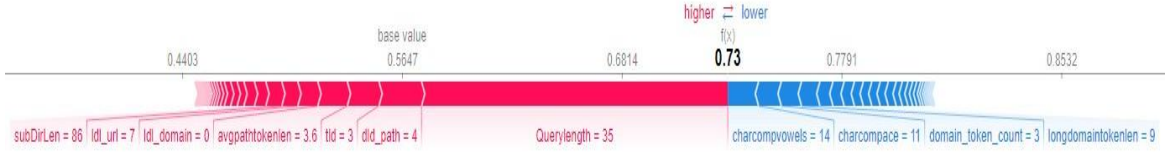


Figure 5. Five-Class Deep-Malicious URLs

Figure 6. Five-Class Deep-SHAP of Android Malware

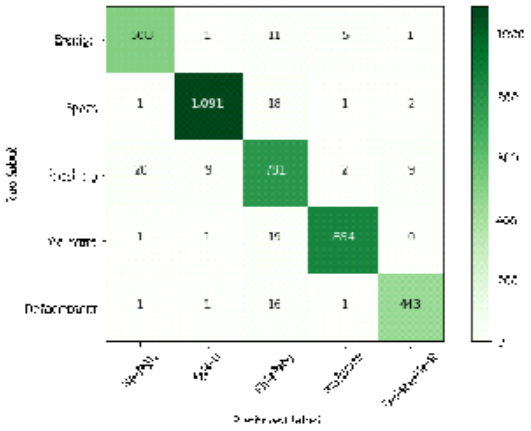


Figure 7. Confusion Matrix for RFC model of Malicious URLs

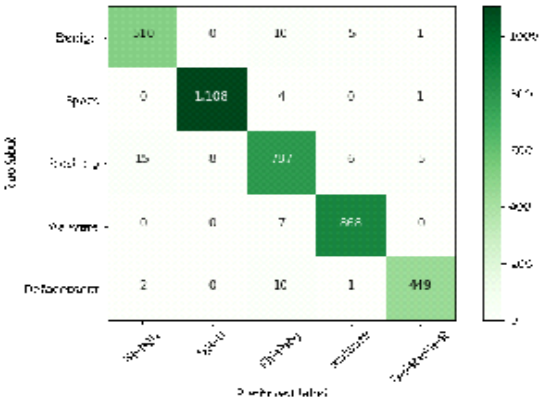


Figure 8. Confusion Matrix for XGBOOST model of Malicious URLs

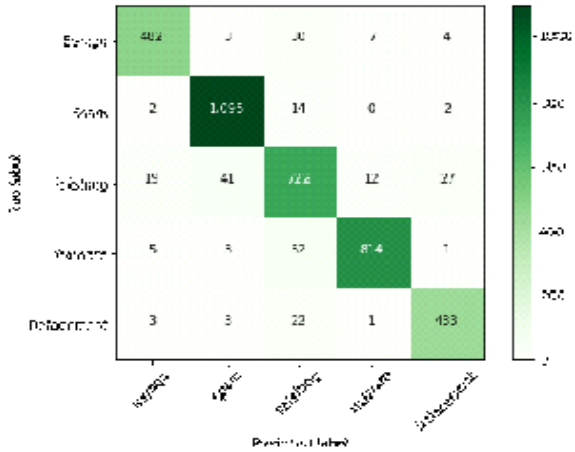


Figure 9. Confusion Matrix for Sequential model Malicious URLs

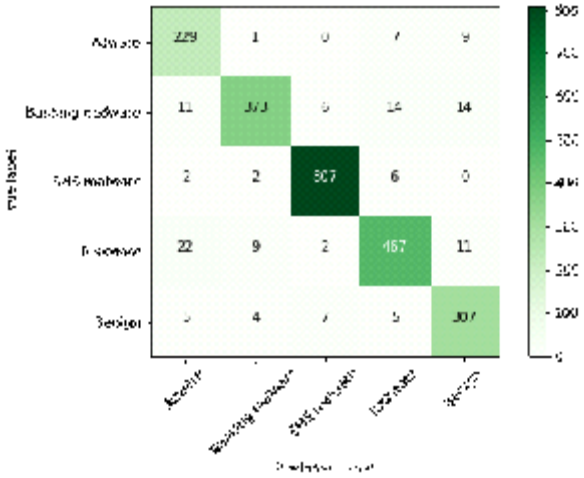


Figure 10. Confusion Matrix of RFC model of Android Malware

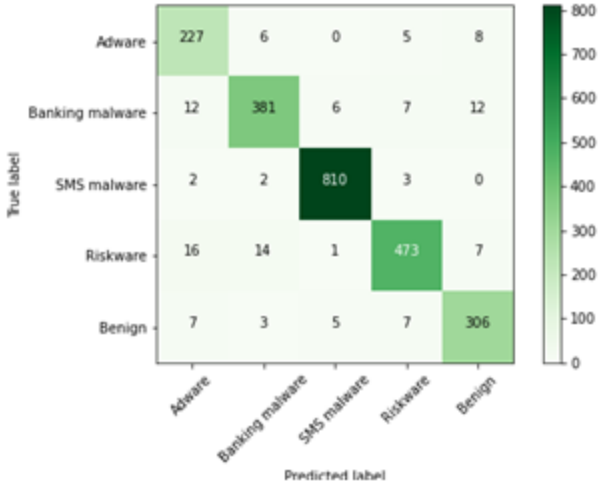


Figure 11. Confusion Matrix of XGBOOST model of Android Malware

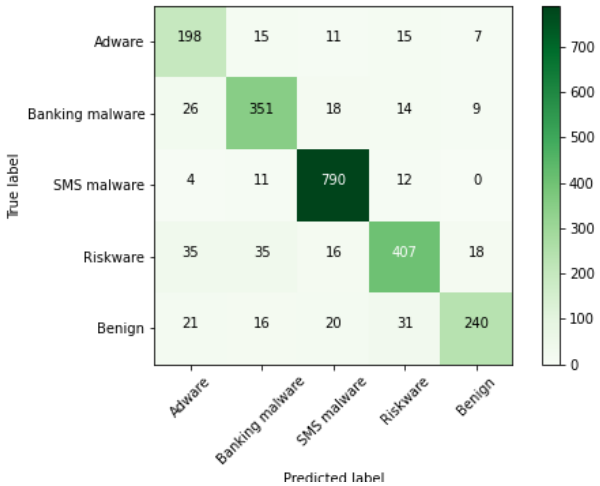


Figure 12. Confusion Matrix of Sequential model of Android Malware

(ML) model, both the ROC curve and the accuracy score for each class under every ML model were analyzed.

Figure 13 presents the ROC curve for the XGBOOST ML model applied to the Malicious URLs dataset, demonstrating superior performance compared to the other models. Table I provides a summary of the ROC results across all ML models tested on the Malicious URLs dataset.

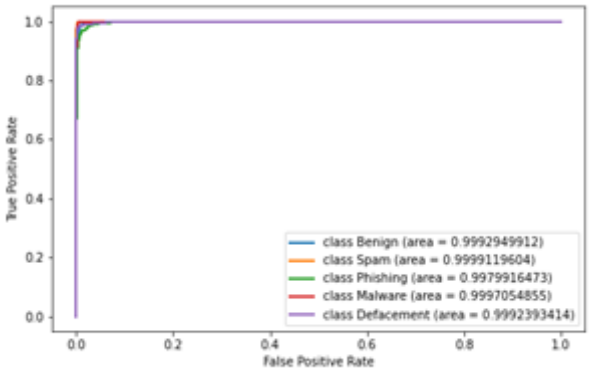


Figure 13. ROC Curve of 5-class to XGBOOST model of Malicious URLs

Furthermore, Tables II through IV report the precision, recall, f1-score, and support metrics for the five-class dataset, based on the performance of all ML models applied to the Malicious URLs dataset.

3) ROC and Classification Tables of Android Malware Data Set: Firstly,an analysis of the ROC values and the accuracy rates across all classes for each ML model revealed that XGBOOST outperformed the other models. Figure 14 illustrates the ROC curve of the XGBOOST model, while Table V presents the ROC results of all ML models applied to evaluate the five classes of the Android Malware dataset.

Table V presents the ROC results for all machine learning

Table I : ROC OF ML MODELS FOR MALICIOUS URLS

class	Sequential	XGBOOST	RFC
Benign	0.9873	0.9992	0.9988
Spam	0.9942	0.9999	0.9996
Phishing	0.9765	0.9979	0.9958
Malware	0.9928	0.9997	0.9996
Defacement	0.9896	0.9992	0.9988

Table II: RFC MODEL OF MALICIOUS URLS

	Precision	Recall	f1-Score	Support
Benign	0.96	0.97	0.96	526
Spam	0.99	0.98	0.98	1113
Phishing	0.92	0.95	0.94	821
Malware	0.99	0.98	0.98	875
Defacement	0.97	0.96	0.97	462
accuracy			0.97	3797
macro avg	0.97	0.97	0.97	3797
weighted avg	0.97	0.97	0.97	3797

models evaluated on the testing subset of the Android Malware dataset.
In addition, Tables VI to VIII provide the precision, recall, f1-score, and support metrics for the 5-Class data derived from the application of all machine learning models to the Malicious URLs dataset.

4) Summary of Results: Table IX presents a summary of the conducted experiments, outlining and comparing the two data sets employed. Both are considered relatively large

Table III XGBO OST MODEL OF MALI CIOUS URLS

	Precision	Recall	f1-Score	Support
Benign	0.97	0.97	0.97	526
Spam	0.99	1.00	0.99	1113
Phishing	0.96	0.96	0.96	821
Malware	0.99	0.99	0.99	875
Defacement	0.98	0.97	0.98	462
accuracy			0.98	3797
macro avg	0.98	0.98	0.98	3797
weighted avg	0.98	0.98	0.98	3797

Table IV: SEQUENTIAL MODEL OF MALICIOUS URLS

	Precision	Recall	f1-Score	Support
Benign	0.94	0.92	0.93	526
Spam	0.96	0.98	0.97	1113
Phishing	0.86	0.88	0.87	821
Malware	0.98	0.93	0.95	875
Defacement	0.93	0.94	0.93	462
accuracy			0.93	3797
macro avg	0.93	0.93	0.93	3797
weighted avg	0.93	0.93	0.93	3797

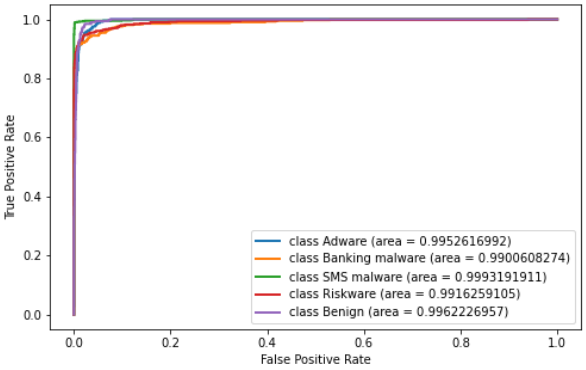


Figure 14. ROC Curve of 5-Class to XGBOOST model of Android Malware

Table V: ROC OF ML MODELS FOR ANDROID MALWARE

Class	Sequential	XGBOOST	RFC
Adware	0.8901	0.9952	0.9946
Banking malware	0.9068	0.9900	0.9890
SMS malware	0.9662	0.9993	0.9985
Riskware	0.8862	0.9916	0.9929
Benign	0.8661	0.9962	0.9943

Table VI: RFC MODEL OF ANDROID MALWARE

	Precision	Recall	f1-Score	Support
Adware	0.85	0.93	0.89	246

Banking malware	0.96	0.89	0.92	418
SMS malware	0.98	0.99	0.98	817
Riskware	0.94	0.91	0.92	511
Benign	0.90	0.94	0.92	328
Accuracy			0.94	2320
macro avg	0.93	0.93	0.93	2320
weighted avg	0.94	0.94	0.94	2320

in scale. The Malicious URLs data set consists of 36,707 samples with 79 features, whereas the Android Malware data set comprises 11,598 samples with 469 features. The features from both sets were examined and categorized into five classes using their respective test portions. Overall, the classification models demonstrated strong performance for the 5-Class tasks. For instance, the RFC model achieved 97% accuracy on the Malicious URLs data set and 94% on the Android Malware data set. Similarly, the XGBOOST model attained 98% accu-

Table VII: XGBOOST MODEL OF ANDROID MALWARE

	Precision	Recall	f1-Score	Support
Adware	0.86	0.92	0.89	246
Banking malware	0.94	0.91	0.92	418
SMS malware	0.99	0.99	0.99	817
Riskware	0.96	0.93	0.94	511
Benign	0.92	0.93	0.93	328
Accuracy			0.95	2320
macro avg	0.93	0.94	0.93	2320
weighted avg	0.95	0.95	0.95	2320

Table VIII: SEQUENTIAL MODEL OF ANDROID MALWARE

	Precision	Recall	f1-Score	Support
Adware	0.70	0.80	0.75	246
Banking malware	0.82	0.84	0.83	418
SMS malware	0.92	0.97	0.94	817
Riskware	0.85	0.80	0.82	511
Benign	0.88	0.73	0.80	328
Accuracy			0.86	2320
macro avg	0.83	0.83	0.83	2320
weighted avg	0.86	0.86	0.86	2320

Table IX SUMMARY OF COMPARISON OF BOTH DATA SETS

Comparison	URLs Data Set	Android Malware Data Set
Samples	36707	11598
Support	3797	2320
Features	79	469
Classes	5	5
RFC accuracy	97%	94%
XGBOOST accuracy	98%	95%
Sequential accuracy	92%	86%
High Correct Classified	XGBOOST model	XGBOOST model

racy on the Malicious URLs data set and 95% on the Android Malware data set. In contrast, the Sequential model obtained 92% accuracy for the Malicious URLs data set and 86% for the Android Malware data set. In conclusion, among the tested ML approaches, the XGBOOST model delivered the best results, as it correctly identified the largest proportion of samples in both data sets.

CONCLUSION

The study utilized Artificial Intelligence to analyze two large-scale cybersecurity datasets, namely Malicious URLs and Android Malware, both obtained from the UNB repository. Each dataset contained five distinct classes. Several AIdriven machine learning algorithms were applied, including Random Forest Classifier (RFC), XGBOOST, and a Sequential model. To enhance explainability, three SHAP-based methods were employed: Bar-SHAP, Dot-SHAP, and Force Plot-SHAP. Model performance was assessed using key evaluation metrics such as accuracy, ROC curves, classification tables, and confusion matrices.

AI-powered explainability through SHAP enabled the identification of features with the greatest influence on model predictions. The SHAP visualizations highlighted the relative contribution of individual features across different models and classes. Specifically, Bar-SHAP effectively illustrated the feature importance across all classes in both datasets, assigning them distinct colors for clarity. The analysis revealed that the feature-driven predictions were most successful in identifying the “Spam” class in the Malicious URLs dataset and the “SMS malware” class in the Android Malware dataset. These findings were further validated by classification tables and confusion matrices, while the other SHAP approaches provided detailed class-specific feature insights.

In terms of AI model performance, the RFC achieved an accuracy of 94.0% on the Android Malware dataset and 96.7% on the Malicious URLs dataset. XGBOOST performed even better, with 98.0% accuracy on the Malicious URLs dataset and 94.7% on the Android Malware dataset. The Sequential model attained 93.3% accuracy for the Malicious URLs dataset and 85.6% for the Android Malware dataset. Overall, the integration of SHAP with AI-based algorithms delivered highly efficient explanations, aligning with the strong accuracy results achieved across the models.

REFERENCES

[1] R. P. Khandpur, T. Ji, S. Jan, G. Wang, C.-T. Lu, and N. Ramakrishnan, Crowdsourcing Cybersecurity: Cyber Attack Detection using Social Media. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, doi: 10.1145/3132847.3132866.
[2] J. R. C. Nurse, Cyber crime and You: How Criminals Attack and the Human Factors That They Seek to Exploit. The Oxford Handbook of Cyber psychology, pp. 662-690, Nov. 2018, doi: 10.1093/oxfordhb/9780198812746.013.35.
[3] B. Seemma, S. Nandhini, and M. Sowmiya. Overview of Cyber Security. Nov. 11, 2018. https://www.researchgate.net/publication/329678338 (accessed: Dec. 24, 2024).
[4] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, Detecting Malicious URLs Using Lexical Analysis. Network and System Security, pp. 467-482, 2016, doi: 10.1007/978-3319-46298-1 30.
[5] F. Alswaina and K. Elleithy, Android Malware Family Classification and Analysis: Current Status and Future Directions. Electronics, vol. 9, no. 6, p. 942, 2020, doi: 10.3390/electronics9060942.
[6] M. Teixeira, T. Salman, M. Zolanvari, R. Jain, N. Meskin, and M.
[7] Samaka, SCADA System Testbed for Cybersecurity Research Using Machine Learning Approach. Future Internet, vol. 10, no. 8, p. 76, 2018, doi: 10.3390/fi10080076.
[8] N. SHI, Q. Zeng, and R. Lee, The Design and Implementation of
[9] Language Learning Chatbot with XAI using Ontology and Transfer Learning. Computer Science Information Technology (CS IT), 2020, doi: 10.5121/csit.2020.101124.
[10] Z. T. Fernando, J. Singh, and A. Anand, A study on the Interpretability of Neural Retrieval Models using DeepSHAP. Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, doi: 10.1145/3331184.3331312.
[11] A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and C. O. Uwadia, Deep learning model for preicting multistage cyber attacks. Journal of Computer Science and Its Application, vol. 26, no. 1, 2020, doi: 10.4314/jcsia.v26i1.6.
[12] S. K. Lakshmanaprabu, K. Shankar, M. Ilayaraja, A. W. Nasir, V. Vijayakumar, and N. Chilamkurti, Random forest for big data classification in the internet of things using optimal features. International Journal of Machine Learning and Cybernetics, vol. 10, no. 10, pp. 2609-2618, 2019, doi: 10.1007/s13042-018-00916-z.
[13] L. Khaidem, S. Saha, and S. Roy Dey. Predicting the direction of stock market prices using random forest. arXiv.org. Apr. 29, 2016. https:// arxiv.org/abs/1605.00003 (accessed: Dec. 24, 2023).

[14] T. Chen and C. Guestrin. XGBoost: A Scalable Tree Boosting System. arXiv.org. 2016. https://arxiv.org/abs/1603.02754 (accessed: Dec. 24, 2024).
[15] L. V. Ho, M. D. Aczon, D. Ledbetter, and R. Wetzel. Interpreting a Recurrent Neural Network. arXiv.org. Apr. 06, 2020. https://arxiv.org/pdf/1905.09865 (accessed: Dec. 24, 2023).
[16] S. M. Lundberg, G. G. Erion, and S.-I. Lee. Consistent Individualized Feature Attribution for Tree Ensembles. Feb. 12, 2018.
[17] S. K. Lakshmanaprabu, K. Shankar, M. Ilayaraja, A. W. Nasir, V. Vijayakumar, and N. Chilamkurti, Random forest for big data classification in the internet of things using optimal features. International Journal of Machine Learning and Cybernetics, vol. 10, no. 10, pp. 2609-2618, 2019, doi: 10.1007/s13042-018-00916-z.
[18] R. Kumar and G. S, Malware classification using XGboost-Gradient Boosted Decision Tree. Advances in Science, Technology and Engineering Systems Journal, vol. 5, no. 5, pp. 536-549, 2020, doi: 10.25046/aj050566.
[19] A. Gulli, A. Kapoor, and S. Pal, Deep Learning with TensorFlow 2 and Keras. Packt Publishing Ltd, 2019.
[20] M. Zaeri-Amirani, F. Afghah, and S. Mousavi, A feature selection method based on Shapley value to false alarm reduction in ICUs A genetic-algorithm approach, Annu Int Conf IEEE Eng Med Biol Soc, vol. 2018, pp. 319–323, 2018.
[21] S. Lundberg and S.-I. Lee, A unified approach to interpreting model predictions, arXiv [cs.AI], 2017.
[22] S. M. Lundberg et al., Explainable AI for trees: From local explanations to global understanding, arXiv [cs.LG], 2019.
[23] P. Sharma, Evaluating Tree Explanation Methods for Anomaly Reasoning: A Case Study of SHAP TreeExplainer and TreeInterpreter. Lecture Notes in Computer Science, pp. 35-45, 2020, doi: 10.1007/978-3-03065847-2 4.
[24] Welcome to the SHAP documentation — SHAP latest documentation. https://shap.readthedocs.io/en/latest/ (accessed: Dec. 24, 2024).
[25] D. Janzing, L. Minorics, and P. Blobaum, Feature relevance quantifica-¨ tion in explainable AI: A causal problem, arXiv [stat.ML], 2019.
[26] Canadian Institute for Cybersecurity — UNB. https://www.unb.ca/cic/ (accessed: Dec. 24, 2024).