Medicon Engineering Themes
Volume 5 Issue 3 September 2023
Article Type: Research Article
ISSN: 2834-7218

Explainable Prediction of features contributing to Intrusion Detection using ML algorithms and LIME

Satish Kumar Karna[1]*, Prakash Paudel[1], Ruby Saud[1] and Mohan Bhandari[2]

[1]*NCIT Lalitpur, Kathmandu, Nepal*

[2]*Samriddhi College Bhaktapur, Nepal*

***Corresponding Author:** Satish Kumar Karna, NCIT Lalitpur, Kathmandu, Nepal.

Received: August 16, 2023; **Published:** August 30, 2023

DOI: 10.55162/MCET.05.160

Abstract

Intrusion Detection System is software or hardware that checks a network for malicious activities. Each illegal activity is often recorded either centrally using a Security information and event management (SIEM) system or notified to an administration. This paper proposes an intrusion detection system using machine learning algorithms such as decision trees, random forests and explainable AI (XAI) using a real-world Software-Defined Networking (SDN) dataset. The evaluation includes various intrusion scenarios like network scanning, denial of service (DoS) attacks, and unauthorized access attempts. Random Forest exhibits the best performance, achieving an average training accuracy of 99.23%, while the decision tree achieves 98.78% accuracy. The result of this study contributes to the advancement of intrusion detection systems and fosters the development of resilient security solutions in the realm of SDN. The research highlights the importance of leveraging ML algorithms in effectively identifying and mitigating network intrusions, ultimately enhancing the security of SDN environments.

Keywords: Decision Tree; Intrusion Detection; Random Forest; XAI; LIME

Introduction

In today's developing world, the Internet has become an essential tool in daily life. Whether it's education, business, or even just paying a simple bill, the internet has push our limits so far [1]. Especially unforeseen attacks can cause catastrophic result. Hence the security of computer and web frameworks has become the major challenge and urgent point for many researchers [2]. As conventional intrusion detection systems struggle to keep pace with the advancing nature of these attacks, novel approaches are required to effectively protect network environments. The primary motivation behind this study stems from the limitations of traditional intrusion detection methods, which often rely on signature-based or rule-based approaches. Such systems struggle to detect novel and sophisticated attacks, as they heavily depend on predefined rules or patterns by leveraging the flexibility and visibility provided by Software-Defined Networking (SDN), we can upgrade the effectiveness of intrusion detection and reaction instruments, driving to moved forward arrange security [3].

In recent years, the proliferation of network attacks and security breaches has posed significant challenges to the integrity and confidentiality of data in computer networks. As traditional intrusion detection systems struggle to keep pace with the evolving nature of these attacks, novel approaches are required to effectively safeguard network environments. SDN has emerged as a promising paradigm for centralized network management and control, offering enhanced flexibility, scalability, and security capabilities [4]. The primary motivation behind this study stems from the limitations of traditional intrusion detection methods, which often rely on

signature-based or rule-based approaches. Such systems struggle to detect novel and sophisticated attacks, as they heavily depend on predefined rules or patterns. By contrast, Random Forest (RF) model offer the potential to learn complex patterns and detect anomalies by analyzing the sequential nature of network traffic data [5]. Moreover, SDN introduces a unique networking environment that facilitates centralized control, network programmability, and dynamic reconfiguration which make it an ideal platform for deploying intelligent intrusion detection systems.

By leveraging the capabilities of ML and LIME, the research sought to advance the cutting edge of intrusion detection and prediction, ultimately ensuring the integrity and privacy of data in computer network. Then, the extracted feature is classified using two ML methods, such as Decision Trees (DT) and RF. Model accuracy was reported and analyzed using Interpretable Artificial Intelligence (XAI), to demonstrate the reliability, capability, and reliability of AI-based solutions in IDS. XAI [6] is a method that allows humans to understand the results of a model, as these models are too difficult to understand and interpret due to their black box concept, and aims to improve the accuracy of the model. Exactly. Our approach is to provide the solution as a white box approach for better model understanding and reliable predictions, so that all stakeholders are in agreement [7, 8]. Differences between models and data can be detected early in the modeling process and corrected accordingly. From experiment, we have successfully shown the effect of XAI techniques such as LIME framework to improve the accuracy and reliability of the model [9]. “Black box” models such as aggregation methods or support vector machines are difficult to interpret and understand. XAI provides a solution to this problem by displaying a black box prediction and interpretation.

The objectives of the study are

1. To classify the intrusion samples using DT and RF algorithms in SDN environment and compare the result.
2. To implement LIME to find out the major contributing features in network environment.

Related Works

The study performed by M. Cavojsk y et al. [10] evaluated prediction model performance using the new UNSW-NB1 dataset. After careful reliability assessment, a subset of the dataset was chosen, preprocessed, and formatted for neural network training. Two accurate models with dense layers were created and adjusted, along with appropriately tuned LSTM models. A separate validation set from the dataset assessed prediction accuracy, post-training. Initial validation accuracy for the dense-layer model was 78.94%, and for LSTM, 76.84%. Weight balancing enhanced accuracy, resulting in 79.34% for dense-layer and 79.21% for LSTM model.

Meftah, Souhail, and their colleagues [11] conducted a study (NBID) where they implemented a two-step method for detecting network intrusions using the UNSW-NB15 dataset. They used a variety of methods, including Recursive Feature Elimination and Random Forests, to choose the dataset’s most pertinent features for machine learning. To differentiate between intrusive and legitimate network traffic, they initially performed binary classification. To do this, they used data mining methods like Logistic Regression, Gradient Boost Machine, and Support Vector Machine (SVM). The outcomes revealed that SVM had the highest accuracy, coming in at 82.11%.To increase the precision of predicting various attack types, the output of the SVM classifier was used as input for a number of multinomial classifiers in the second step. They tested Naive Bayes, Multinomial SVM, and Decision Trees (C5.0), with C5.0 getting the greatest F1 score (86%) and accuracy(74%) of the three. The overall accuracy was raised by up to 12 percent by using this two-stage hybrid classification approach, yielding a multi-classification accuracy of 86.04%.

Sydney Mambwe Kasongo [12] utilized an XGBoost-based approach to trim feature counts in datasets, decreasing attributes. From the NSL-KDD dataset, 22 attributes were retained; from UNSW-NB15, 17 were kept. Performance metrics included F1-Score, validation and test accuracy, and training time. Results indicated XGBoost-LSTM’s superiority for NSL-KDD, with 225.46s training, 88.13% test accuracy, and 99.49% validation accuracy. For UNSW-NB15, XGBoost-Simple-RNN was most effective, achieving 87.07% test accuracy. XGBoost-LSTM hit 86.93% TAC for multiclass NSL-KDD, while XGBoost-GRU scored 78.40% for UNSW-NB15.

The authors [13] introduced BLoCNet, a novel deep learning model fusing CNN and bidirectional LSTM layers. It swiftly detects network patterns via CNN, then processes results through dual BLSTM layers for malicious traffic identification. BLoCNet was compared with 5 DL models and 7 related studies across 4 datasets. It outperformed others in attack detection for CIC-IDS2017, IoT-23, and UNSW-NB15 datasets. Impressively, BLoCNet reached 98% and 99.8% accuracies for CIC- IDS2017 and IoT-23 respectively. Despite sampling differences, these levels were competitive. For UNSW-NB15, BLoCNet surpassed related work's 75.56% with 76.34% accuracy.

The article by Suhana, S et al. [14] used hybrid method for network intrusion detection, combining ensemble techniques. The process commences by employing Blended Linear Discriminant Analysis (BLDA) to extract essential features. Subsequently, a Random Forest Classifier is utilized for intrusion detection on a dimensionally reduced dataset. Evaluation of this approach is conducted on the NSL-KDD and UNSW-NB15 benchmark datasets. The method's efficacy is compared against conventional feature selection methods such as LDA, PCA, and PLS. Results indicate that the proposed method achieves an accuracy of 90.12% for the NSL-KDD dataset and 91.0% for the UNSW-NB15 dataset, respectively.

Methodology

The proposed model displays the abstract architecture of workflow consisting of data preprocessing, different algorithms implemented, statistical performance measures and explanation Extraction from the Model as shown in Fig.1.

Collection of Data set

The UNSW-NB 15 dataset raw network packets [15, 16] were generated in the Cyber Range Lab of UNSW Canberra using the IXIA PerfectStorm tool. This unbalanced dataset combines real modern normal activities with synthetic contemporary attack behavior. It includes nine types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. To generate the dataset, the Argus and BroIDS tools were utilized, and a total of twelve algorithms were developed. These algorithms produced 49 features along with their corresponding class label.

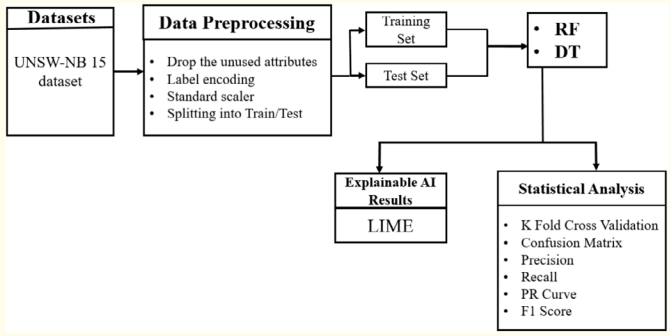


Figure 1: Methodology.

Data Pre-Processing

For training and testing purposes, the dataset was split into two sections that are specifically referred to as UNSW NB15 training set.csv and UNSW NB15 testing set.csv, respectively. Both the files downloaded and accessed from google colab [17] using python as programming language, were combined. Among the features from the dataset, 'id','dur', 'attack cat' were dropped.

Random Forest

Random Forest is a classification model that incorporates multiple decision trees trained on different subsets of the provided dataset. RF constructs an assortment of decision trees, with each tree being trained on arandomized portion of the training data. It combines the predictions from each tree by considering the majority votes, thus enhancing the overall predictive accuracy compared to relying on a single decision tree [18].

The steps followed in RF algorithm is summarised as follows.

1. Selecting random data points(same as the total number of training samples) from the training set.
2. Building decision trees associated with the selected data points.
3. Choosing the number N (default 100) for decision trees to build.
4. Repeating 1 and 2.

Decision Tree

The decision tree works by analyzing the dataset and making assumptions about how it should be classified. It begins at the tree's root node and contrasts the root attribute's value with the attribute of each record in the actual dataset [19]. Depending on the outcome of the comparison, it moves to the subsequent node by following the corresponding Up until the leaf node of the tree is reached, this process is repeated for each additional node by comparing their attribute values with those of the sub-nodes. The following steps can helps us to comprehend the entire procedure.

1. Begin with the root node, denoted as S, which contains the complete dataset.
2. Determine the best attribute in the dataset using an Attribute Selection Measure (ASM).
3. Divide S into subsets that contain possible values for the best attribute.
4. Create a decision tree node that represents the best attribute.
5. Recursively construct new decision trees using the subsets of the dataset.
6. Repeat this process until a point is reached where further classification is not possible, and designate the final node as a leaf node.

LIME

LIME is a technique used to provide local, interpretable explanations for the output of machine learning models. LIME generates these explanations by training a simpler, interpretable model on a local region around a specific data point [20]. LIME makes it possible to visualize every aspect for result analysis. LIME demands information about the model separately in order to assess a model's local faithfulness. How effectively a model represents the characteristics around a given prediction is measured by local fidelity. Local fidelity can help to explain how the prediction was made, even if it may not be relevant to the full model. On the other hand, a model's overall explanation could not explain a certain local prediction [21]. Global representation of an instance x can be represented as follows: as shown in equation 1.

$$x \in R^d \quad (1)$$

However, an interpretable representation of an instance is a binary vector as in equation 2.

$$x \in 0, 1^d \quad (2)$$

The local presence or absence of one or more characteristics is determined by the interpretable representation. In terms of LIME's model-agnostic attribute, g stands for a machine learning model. G represents a set of models containing g among other models, then.

$$g \in G \quad (3)$$

As such, LIME's algorithm will interpret any other model in the same manner.

Evaluation metrices

The system used the UNSW-NB15 data set to evaluate the results of the perfomance metrics such as Confusion Matrix, F1-Score,Precision,Recall,PR curve.

Result

The experiments were conducted on Google Colab, utilizing a NVIDIA K80 GPU and 12 GB of RAM provided by Google. Specifically, the runtime environment in Google Colab used Python version 3.7, Keras version 2.5.0, and the TensorFlow version 2.5.0 framework. A distinct splitting was utilized to separate the dataset into training and testing sets. Looking into the dataset on the basis of number of samples per class, dataset was not balanced.

Random Forest

1. *K-Fold Result*: The model attained an average training accuracy of 99.233% across 10 folds, as indicated in Table 1.

Fold	*Training Accuracy*
K1	99.3
K2	99.21
K3	99.21
K4	99.34
K5	99.12
K6	99.26
K7	99.18
K8	99.39
K9	99.24
K10	99.08
Average	99.233

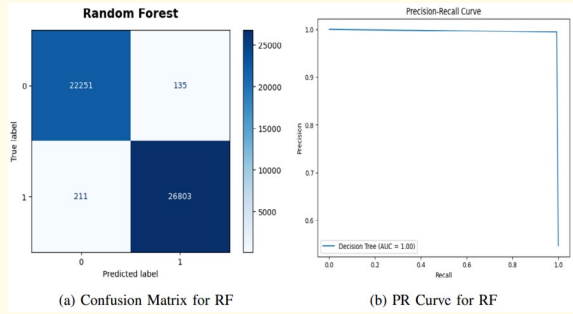
Table 1: K-Fold result of RF for dataset.

2. *Confusion Matrix*: Figure 2 (a) shows that 135 normal samples were miss-classified where as 211 intrusion samples were miss-classified.
3. *Classification Report*: Table 2 was obtained as the classification report of DT, showing the Precision, Recall, and F1- Score, as well as Accuracy, Macro Avg, and Weighted Avg. Intrusion samples were highly precise than normal sample with precision value of 99.51%. The model have achieved almost equal recall as well as equal f1-score.

	Precision	*Recall*	*F1-score*	*Support*
0	0.9905	0.9941	0.9923	22386
1	0.9951	0.9921	0.9936	27014
Accuracy			0.9930	49400
Macro Avg	0.9928	0.9931	0.9930	49400
Weighted Avg	0.9930	0.9930	0.9930	49400

Table 2: Classification report of RF.

4. *PR curve*: Figure 2 (b) shows the precision recall relation with AUC value of 100%.



(a) Confusion Matrix for RF (b) PR Curve for RF

Figure 2: Confusion Matrix and PR Curve of RF.

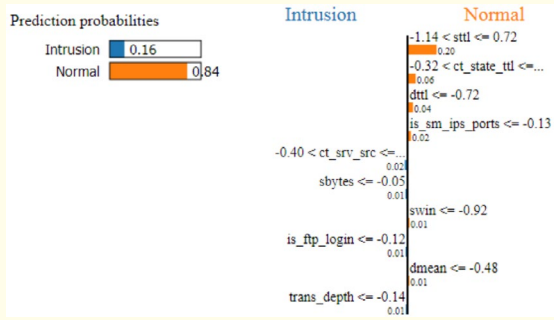


Figure 3: RF LIME Tabular Plots.

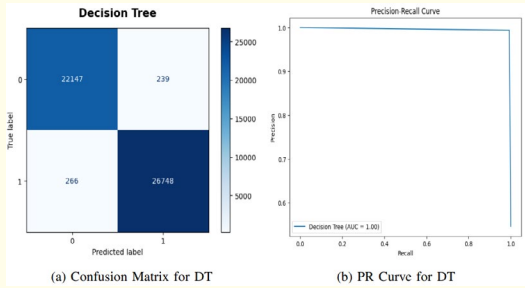
5. *RF Lime*: Test record with index 0 is predicted as Normal by 84% where as it is predicted as Intrusion by 16% as shown in prediction probabilities in Figure 3. Features like sttl, ct_state_ttl, is_sm_ips_ports, swin and dmean mean predict the instance as Normal by 20%, 6%, 4%, 2%, 1% and 1% respectively. It's interesting to see that the likelihood of Normal prediction is reduced by the feature ct_srv_src, sbytes, is_ftp_login and trans_depth by 2%, 1%, 1% and 1% respectively 1% each to prove Intrusion.

Decision Tree

1. *K-Fold Result*: For K=10 folds, the model achieved the average training accuracy of 98.78% as shown in Table 3.
2. *Confusion Matrix*: Figure 4 (a) shows that 239 normal samples were miss-classified where as 266 intrusion samples were miss-classified.
3. *Classification Report*: Table 4 was obtained as the classification report of DT, showing the Precision, Recall, and F1- Score, as well as Accuracy, Macro Avg, and Weighted Avg. Intrusion samples were highly precise than normal sample with precision value of 99.11%. The model have achieved higher recall and f1-score for intrusion samples in comparision to normal samples.
4. *PR Curve*: Figure 4 (b)shows the precision recall relation for dataset with AUC value of 100%.
5. *DT Lime*: Test record with index 0 is predicted Normal by 100% as shown in prediction probabilities in Figure 3. Features like sttl, is_ftp_login, is_sm_ips_ports, djjt and ct_src_ltm predict the instance as normal by 64%, 6%, 5%, 4% and 3% respectively. It's interesting to see that the likelihood of Intrusion is 0%.

Fold	*Training Accuracy*
K1	98.83
K2	98.86
K3	98.58
K4	98.8
K5	98.76
K6	99
K7	98.88
K8	98.85
K9	98.62
K10	98.62
Average	98.78

Table 3: K-Fold result of DT.



(a) Confusion Matrix for DT (b) PR Curve for DT

Figure 4: Confusion Matrix and PR Curve of DT.

	Precision	*Recall*	*F1-score*	*Support*
0	0.9881	0.9893	0.9887	22386
1	0.9911	0.9902	0.9906	27014
Accuracy			0.9898	49400
Macro Avg	0.9896	0.9897	0.9897	49400
Weighted Avg	0.9898	0.9898	0.9898	49400

Table 4: Classification report of DT.

Comparative Analysis of Statistical Results

Two Algorithms, RF and DT were implemented to predict whether the instance is Normal or Instrusion. Table 5 shows the training and testing accuracy of DT and RF where the results show that RF slightly outperformed DT in training accuracy and we saw that testing accuracy was same for both the algorithms.

Algorithms	*Training Accuracy*	*Testing Accuracy*
RF	99.23	99.3
DT	98.78	99.3

Table 5: Comparative Analysis of Training and Testing Accuracy.

Table 6 provides data outlines a comparative analysis of two algorithms, Random Forest (RF) and Decision Tree (DT), in terms of their performance metrics for two categories: ”Normal” (0) and ”Intrusion” (1). The data indicates that, for both the ”Normal” and ”Intrusion” categories, the Decision Tree algorithm generally performs slightly better than the Random Forest algorithm in terms of F1 Score, Precision, and Recall. In the ”Intrusion” category, both algorithms demonstrate high precision, recall, and F1 Score, with the Decision Tree algorithm having a slightly higher F1 Score and Precision.

The given result also presents a comparative analysis of two algorithms RF and DT, focusing on their performance in terms of the False Positive Rate. The False Positive Rate measures the proportion of instances that were incorrectly classified as positive (in this case, the ”Intrusion” category) when they were actually negative (in this case, the ”Normal” category). Result suggests that the Random Forest algorithm has a lower False Positive Rate compared to the Decision Tree algorithm i.e. when using the Random Forest algorithm, there were fewer instances that were wrongly classified as ”Intrusion” when they should have been classified as ”Normal.” This indicates that the Random Forest algorithm may be better at minimizing the occurrence of false alarms, which could be crucial in scenarios where accurately identifying true intrusions is important while avoiding unnecessary alerts.

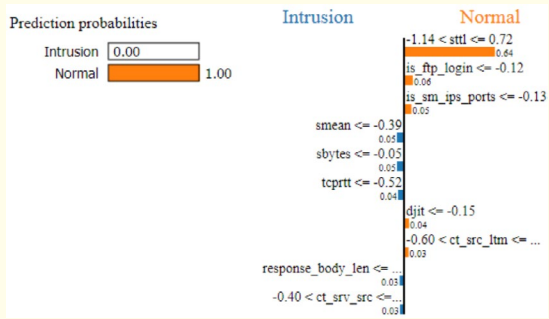


Figure 5: DT LIME Tabular Plots.

Algorithms	*Category*	*F1 Score*	*Precision*	*Recall*	*False Positive Rate*
RF	Normal (0)	0.9887	0.9881	0.9893	125
	Intrusion (1)	0.9906	0.9911	0.9902	
DT	Normal (0)	0.9923	0.9905	0.9941	229
	Intrusion (1)	0.9936	0.9921	0.9921	

Table 6: Comparative Analysis of Precision, Recall and F- Score and False Postive Rate.

Comparative Analysis of XAI Lime Results

Different features from RF and DT were compared to find the common pattern in prediction which is shown in the tabulated form in the Table 7. From DT, Features like sttl, is_ftp_login, is_sm_ips_ports, djjt and ct_src_ltm predict the instance as Normal and features like smean, sbytes, tcprtt, response_body_len and ct_srv_src contribute to the instance to be as Intrusion. Similarly, From RF, Features like Features like sttl, ct_state_ttl, is_sm_ips_ports, swin and dmean mean predict the instance as Normal and features like ct_srv_src, sbytes, is_ftp_login and trans_depth contribute to the instance as Intrusion.

Upon comparing the selected features for both algorithms, we can identify that, common features selected by both RF and DT algorithms for the ”Normal” category are ”is_ftp_login” and ”is_sm_ips_ports”, both algorithms have identified these features in the ”Normal” category implies that ”is_ftp_login” and ”is_sm_ips_ports” hold consistent importance across their decision- making processes. Similarly, the common feature selected by both RF and Decision Tree DT algorithms for the ”Intrusion” category is ”ct_srv_src” which

a common feature and suggests that this feature carries significant information for distinguishing potentially harmful network instances. It indicates that both algorithms recognize the importance of analyzing the frequency and patterns of connections to specific services as a key factor in identifying network intrusions. Also from the result we can see that, the common features that appear in both the ”Normal” and ”Intrusion” categories is ”is_ftp_login”. Both algorithms seem to recognize the importance of this features for distinguishing between the two categories, suggesting that they play a significant role in identifying network intrusions and normal network behavior.

Conclusion

This study presented the data results that offers a comprehensive evaluation of the RF and Decision DT algorithms' effectiveness in both training and testing phases. RF demonstrates notable accuracy with a training accuracy of 99.23% and a closely aligned testing accuracy of 99.3%. Similarly, DT showcases robust performance, achieving a training accuracy of 98.78%, along with a testing accuracy of 99.3%. This proficiency is expected to contribute to RF's exceptional performance in both classifying new data and minimizing errors when faced with unseen instances. Such substantial accuracy levels signify their potential for real-world application in network intrusion detection and related domains. The algorithms exhibit promising results in accurately classifying data for both ”Normal” and ”Intrusion” categories. RF achieves an F1 score of 0.9906 and a precision of 0.9911 for the ”Intrusion” category, highlighting its proficiency in precisely identifying intrusive network behavior. Additionally, DT excels in the ”Intrusion” category, attaining an impressive F1 score of 0.9936 and a precision of 0.9921, underscoring its capability in distinguishing malicious activities. Also, using XAI Algorithm i.e. LIME we found out the attributes that play a major contributing role to predict the instance as Normal or as Intrusion.

These findings emphasize the algorithms' potential for practical applications, particularly in the realm of network intrusion detection. Their consistent accuracy, combined with strong F1 scores and precision values, suggests their reliability in identifying both normal and intrusive network behaviors, thus contributing to enhanced cybersecurity measures.

	Normal	***Intrusion***
RF	sttl, ct_state_ttl, is_sm_ips_ports, swin ,dmean	Ct_srv_src, sbytes, is_ftp_login, trans_depth
DT	sttl, is_ftp_login, is_sm_ips_ports, djjt, ct_src_ltm	smean, sbytes, tcprtt, response_body_len, ct_srv_src

Table 7: LIME Comparative Analysis of Normal and Intrusion for DT and RF.

References

1. Margaret Bearman and Rola Ajjawi. “Learning to work with the black box: Pedagogy for a world with artificial intelligence”. British Journal of Educational Technology (2023).
2. PV Sai Charan., et al. “DKaaS: DARK-KERNEL as a Service for Active Cyber Threat Intelligence”. Computers & Security (2023): 103329.
3. Asma Alotaibi and Ahmed Barnawi. “IDSoft: A Federated and Softwarized Intrusion Detection Framework for Massive Internet of Things in 6G Network”. Journal of King Saud University-Computer and Information Sciences (2023): 101575.
4. Neelam Gupta, Sarvesh Tanwar and Sumit Badotra. “Review of Software-Defined Network-Enabled Security”. Computational Intelligence for Engineering and Management Applications: Select Proceedings of CIEMA 2022. Springer (2023): 427-441.
5. Robert Appiah. “MultiVul-GCN: automatic smart contract vulnerability detection using multi-graph convolutional net- works”. PhD thesis. University of British Columbia (2023).
6. Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin. “Why should i trust you?” Explaining the predictions of any classifier”. Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (2016): 1135-1144.
7. Samanta Knapic., et al. “Explainable artificial intelligence for human decision support system in the medical domain”. Machine Learning and Knowledge Extraction 3.3 (2021): 740-770.

8. Shruti Patil., et al. "Explainable artificial intelligence for intrusion detection system". Electronics 11.19 (2022): 3079.
9. Saabas Ando. "Interpreting random forests". (2019).
10. Matus Cavojsky, Gabriel Bugar, and Dusan Levicky. "Comparative Analysis of Feed-Forward and RNN Models for Intrusion Detection in Data Network Security with UNSW-NB15 Dataset". 2023 33rd International Conference Radioelektronika (Radioelektronika). IEEE. (2023): 1-6.
11. Souhail Meftah, Tajjeeddine Rachidi and Nasser Assem. "Network based intrusion detection using the UNSW-NB15 dataset". International Journal of Computing and Digital Systems 8.5 (2019): 478-487.
12. Sydney Mambwe Kasongo. "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework". Computer Communications 199 (2023): 113-125.
13. Brandon Bowen., et al. "BLoCNet: a hybrid, dataset-independent intrusion detection system using deep learning". International Journal of Information Security (2023): 1-25.
14. S Suhana, S Karthic and N Yuvaraj. "Ensemble based Dimensionality Reduction for Intrusion Detection using Random Forest in Wireless Networks". 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT) (2023): 704-708.
15. Nour Moustafa and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW- NB15 network data set)". 2015 military communications and information systems conference (MilCIS). IEEE (2015): 1-6.
16. Nour Moustafa, Benjamin Turnbull and Kim-Kwang Raymond Choo. "An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things". IEEE Internet of Things Journal 6.3 (2018): 4815-4830.
17. Ekaba Bisong and Ekaba Bisong. "Google colaboratory". Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners (2019): 59-64.
18. Pronab Ghosh., et al. "Efficient prediction of cardiovascular disease using machine learning algorithms with relief and LASSO feature selection techniques". IEEE Access 9 (2021): 19304-19326.
19. Nahla Ben Amor, Salem Benferhat and Zied Elouedi. "Naive bayes vs decision trees in intrusion detection systems". Proceedings of the 2004 ACM symposium on Applied computing (2004): 420-424.
20. Muhammad Rehman Zafar and Naimul Khan. "Deterministic local interpretable model-agnostic explanations for stable explainability". Machine Learning and Knowledge Extraction 3.3 (2021): 525-541.
21. Denis Rothman. Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps. Packt Publishing Ltd (2020).

Volume 5 Issue 3 September 2023
© All rights are reserved by Satish Kumar Karna., et al.