



## Research Article

# Explainable Artificial Intelligence (XAI) to Enhance Trust Management in Intrusion Detection Systems Using Decision Tree Model

**Basim Mahbooba** <sup>1</sup>, **Mohan Timilsina**,<sup>1</sup> **Radhya Sahal** <sup>2</sup>, and **Martin Serrano**<sup>1</sup>

<sup>1</sup>Data Science Institute, Insight Centre for Data Analytics, National University of Ireland Galway, Galway, Ireland

<sup>2</sup>Faculty of Computer Science and Engineering, Hodeidah University, Al Hudaydah, Yemen

Correspondence should be addressed to Basim Mahbooba; [basim.mahbooba@insight-centre.org](mailto:basim.mahbooba@insight-centre.org)

Received 4 December 2020; Accepted 24 December 2020; Published 28 January 2021

Academic Editor: Ahmed Mostafa Khalil

Copyright © 2021 Basim Mahbooba et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Despite the growing popularity of machine learning models in the cyber-security applications (e.g., an intrusion detection system (IDS)), most of these models are perceived as a black-box. The eXplainable Artificial Intelligence (XAI) has become increasingly important to interpret the machine learning models to enhance trust management by allowing human experts to understand the underlying data evidence and causal reasoning. According to IDS, the critical role of trust management is to understand the impact of the malicious data to detect any intrusion in the system. The previous studies focused more on the accuracy of the various classification algorithms for trust in IDS. They do not often provide insights into their behavior and reasoning provided by the sophisticated algorithm. Therefore, in this paper, we have addressed XAI concept to enhance trust management by exploring the decision tree model in the area of IDS. We use simple decision tree algorithms that can be easily read and even resemble a human approach to decision-making by splitting the choice into many small subchoices for IDS. We experimented with this approach by extracting rules in a widely used KDD benchmark dataset. We also compared the accuracy of the decision tree approach with the other state-of-the-art algorithms.

## 1. Introduction

Organizations are increasingly developing more complex cyber-security ecosystems that rely on different peers such as people, technology, and processes to function effectively. Trust management in cyber-security is based on the relationships between these peers including (1) people and groups, (2) people and organizations, (3) organizations, and (4) people and technology. Each of these trusting peers can deploy cyber-security countermeasures that an individual can rely on to prevent and defend against cyber-attacks [1].

Artificial Intelligence (AI) is a set of models and methodologies that are used to extract knowledge from a collection of data. Furthermore, no human can trust an AI system, because it is both possible and desirable to quality of data, complexity of methodology and accountability, and experiences of AI engineer. In regarding to the AI-based

solutions for cyber-security context, other software development techniques can be peers with AI since these do not “trust,” so no one actually can trust AI-based solutions in cyber-security systems. The tricky question is “how we can trust the AI-based solutions in cyber-security systems which are designed based on the data, methodology, and expert accountability?” To answer this question, researchers have captured interpretability and the topic eXplainable Artificial Intelligence (XAI) to justify the AI-based solution reliability, ability, and trustworthy [2]. Further details about trust and AI, explainable AI, and explainable AI and intrusion detection system are addressed in the following subsections.

**1.1. AI and Trust Management.** An open challenge with AI is the lack of understanding and trust compared to traditional model-based optimisation. For example, deep reinforcement

learning is not able to explain the essential features that influence actions [3]. This drawback is being worst, which affects the trust management in cyber-security (e.g., malicious vehicle identification) [4]. Furthermore, for the Bayesian inference, the recent research has shown that they are incredibly brittle to insufficient data. Therefore, the need to develop statistical AI algorithms is increasing to quantify uncertainty, especially mapping big data inputs and algorithm design, to the projected wireless key performance indicators (KPIs) [5]. Rather than trying to create statistical AI algorithms that are inherently interpretable, there has been a recent research direction for explosion of work on “explainable AI” where the statistical AI algorithms are created to explain the AI black-box model [6]. Consequently, trustworthy AI should be able to explain its decisions to allow the human expert to understand the underlying data evidence and causal reasoning.

*1.2. Explainable AI.* In the last few years, AI has achieved a notable success by delivering the best of expectations over many applications. This empirical success of machine learning (ML) and deep learning (DL) models attributes to the combination of efficient learning algorithms and their huge parametric space [5]. The combination of the parametric learning space comprises hundreds of layers and millions of parameters, which makes ML and DL models be considered as complex black-box models [7]. Due to the black-box-ness of the models, AI experts (e.g., engineers and developers) need to search for a direct understanding of the mechanism by which a model works. The transparency, which is the opposite of the black-box-ness of models, is increasingly being demanded to avoid the danger of making decisions that are not justifiable and do not allow obtaining detailed explanations of their behavior. For example, in precision medicine, binary predictions are insufficient due to their sensitivity of medicine prescriptions to the patients. Also, for cyber-security, the misleading predictions can make the system vulnerable for attacks and lead to zero-trust security for critical systems.

Consequently, the focus of the recent research in AI is related to explainable AI field, which plays a vital role in improving practical deployment of AI-based solution. Considering of interpretability during AI-based solution design can empower their implementability because (1) interpretability helps ensure impartiality in decision-making, i.e., to detect, and consequently, correct from bias in the training dataset (i.e., imbalanced dataset); (2) interpretability strengthens the robustness of AI-based solutions by highlighting potential adversarial perturbations that could change the prediction; and (3) interpretability empowers the trust of AI-based solutions by providing the meaningful variable inference and causality of model reasoning [5, 8]. Explanation methods and techniques for AI/ML interpretability can be classified according to different criteria as follows: (1) premodel, in-model, and postmodel, (2) intrinsic and post hoc, (3) model-specific and model-agnostic, (4) feature engineering, and so on [8]. According to this work, feature engineering and the rule-based model will be

addressed to exploring the decision tree method for understanding the characteristics of malicious attacks for enhancing trust in IDS.

*1.3. Intrusion Detection Systems.* Intrusion detection systems (IDS) have been developed rapidly in research and industry in response to the increasing cyber-attacks against governments and commercial enterprises globally. The annual cost of fighting cyber-crime is continuously increasing [9]. The most disastrous cyber-crimes are those caused by malicious insiders, denial of services, and web-based attacks. Industries or companies can lose their intellectual property due to these malicious attacks into the system. To fight back against such acts, organizations deploy a firewall, antivirus software, and an intrusion detection system.

Intrusion detection (ID) is a vital part of cyber-security. It allows us to identify malicious network activity before it compromises information availability, integrity, or confidentiality [10]. It is a process of identifying security breaches by examining events occurring in an information system. In today’s increasingly digital world, lack of network access is unthinkable in both professional and personal life. Nowadays, with a rise of the Internet of Things (IoT) devices connected to the Internet, attackers have maximum possibilities for intrusion attack. Thus, keeping the network devices safe from the intrusion is an important task. This also opens up the issue about how to successfully secure from both known and unknown threats. There is no straightforward answer to this because of the increasing number of threats every year [11].

Any valuable information having network access needs to be permanently protected from all attempts to destroy, expose, alter, disable, steal, or gain unauthorized access and usage. To prevent valuable information from such a malicious act, the intrusion detection system (IDS) is designed. The IDS is a device or software application that collects and analyzes information in a network to identify possible security breaches, including data for both intrusion (i.e., attacks from outside the organization) and misuse (i.e., attacks from within the organization) [11]. The classical IDSs are mainly signature-based, detecting only known attacks, and their major drawback is the inability to detect new attacks [12]. One way to address that kind of problem is to use a “machine learning” (ML) approach that provides computers the ability to learn without being explicitly programmed. ML is mainly applied to improve detection accuracy and low false alarm rate [13].

*1.4. Explainable AI and Intrusion Detection Systems.* Nowadays, due to highly accurate predictions, the deep neural network (DNN) is getting more popular. These kinds of models are useful, but they are hard to interpret. For example, using the DNN model, the control of a self-driving car requires thousands of parameters tuning [14]. In the context of IDS, if such DNN methods are implemented, then it is tough for a network administrator to understand the reasoning provided by the ML system. DNN is also called black-box models [15]. A black-box keeps its decision-making the process challenging to interpret

because DNN primarily edits different features by trial and error until they find the ideal solution [16]. Many studies in the IDS use ML techniques to increase the classification accuracy of known attacks, recognize anomalous network traffic, and automate model construction [17]. However, few of these systems focus on interpreting the results as a means of understanding how the ML algorithms reach into the conclusion for predicting attacks.

If the attacks are explained in the forms of rules, it can be easily interpreted. For example, simple interpreted rules in the form of If, Then statement. Also, the explanation is needed to highlight (i) which parts of the target of the attack, (ii) which parts of the network features, and (iii) what parts of the security policies are violated. The challenge is to link up the IDS, analyze it, explain it, and provide insight to network administrators to enforce security policies for identified attacks. An essential backbone for this process is to have models that can logically interpret results. Thus, we consider decision tree (DT) for such a task. If the scenario demands an explanation over the prediction, then DTs are the perfect models because they can provide an understandable explanation over the prediction. Unlike other methods in supervised machine learning methods, DTs do not require assumptions on the distribution of data. It handles the feature colinearity (highly correlation among features) problem efficiently. Thus, the interpretative model seems to be necessary in the context of IDS.

**1.5. Contribution.** Although previous work has used DT in IDS, their main focus is on the accuracy of benchmark machine learning algorithms. Conversely, we focus on the interpretability in a widely used benchmark dataset called KDD datasets [18]. Our contribution is as follows:

- (1) We addressed XAI concept to enhance trust management that human experts can understand (e.g., the underlying data evidence and causal reasoning). To do so, we have used feature engineering and the rule-based model to explore the decision tree algorithm in the area of IDS.
- (2) We analyzed the importance of feature based on the entropy measure for intrusion detection.
- (3) We interpreted the rules extracted from the DT approach for intrusion classification.
- (4) We compared the accuracy of the decision tree with the state-of-the-art methods.

The rest of this work is organized as follows: a review of relevant works is conducted in the section “Related Work.” In the “Methodology” section, we demonstrated the DT method. Results and experiment settings are mentioned in the sections “Experiments” and “Discussion.” Finally, we conclude our work and describe some direction for future works in the section “Conclusion and Future Work.”

## 2. Related Work

There are different types of solution proposed for IDS such as statistical methods [19], hidden Markov model [20],

artificial neural network [21], and fuzzy logic [22]. Recent studies showed that support vector machines (SVMs) demonstrated high accuracy in developing IDS [23, 24]. However, its main caveat is that it needs a long training time, which makes its usability limited. There are also classical data mining methods used in IDS, such as association rule mining. For example, Ilgun et al. [25] employed rule-based techniques to design and develop IDS, where expert knowledge is considered as a rule set. Similarly, Lee et al. [26] used association rules instead of human experts as an analytical model. The limitation of such methods is an extraction of a large number of association rules, which increases the complexity of the model.

One of the critical aspects of a supervised classification model is feature selection. Selecting essential features will reduce the computational time of the algorithms. Due to the significant features of network data, many IDSs were developed with feature selection [27]. Chebrolu et al. [28] classified primary features in constructing an IDS that is very crucial for real-world intrusion detection. Zaman and Karray [29] implemented a feature selection technique to construct a lightweight IDS. Vimalkumar and Radhika [30] implemented principal component analysis (PCA)-based feature selection technique in the big data framework for IDS. Balakrishnan et al. [31] developed an IDS model with a gain ratio as a feature selection technique and two classification techniques, namely, support vector machine and rule-based classification which were used for identifying the class label and demonstrated higher accuracy levels for denial of service (DoS) attacks. One drawback of this kind of approach is being high computational for employing separate gain ratio and classification algorithm.

Most of the IDS-based studies focused on the performance of the implemented model. Farrahi and Ahmadzadeh [32] explored various algorithms such as k-means clustering, Naïve Bayes, support vector machine, and OneR algorithms. This model resulted in better accuracy for regular traffic and DoS attack. Also, the genetic algorithm (GA) was implemented to enhance the detection of different types of intrusions with an accuracy of 97% [33]. The work by Alkasasbeh et al. [34] focused on different types of attack, such as http flood, smurf, siddos, and UDP flood. They implemented various machine learning algorithms to detect DoS intrusions and demonstrated the high accuracy of 98.36% using multilevel perceptron (MLP). Peng et al. [35] proposed an IDS system based on a decision tree to improve the efficiency of detection. Their method showed better performance over Naïve Bayesian and KNN methods.

The researchers are still investigating to find an effective way to detect the intrusions with high performance, high speed, and low false-positive alarm rate [36]. The majority of the IDS system is focused on accuracy. Less interest is given on the interpretable side of predictive algorithms. In this work, we explore the interpretable side of the classification algorithm implemented for IDS using a decision tree. A decision tree is considered as a highly interpretable model [37]. Interpretability is a promising approach to diagnose issues and verify the correctness of machine learning models in an IDS setting by providing insight into the model's

reasoning. Therefore, we used a decision tree algorithm that ranks the importance of features, provides explainable rules for intrusion detection, and has comparable accuracy with the state-of-the-art algorithms.

### 3. Methodology

**3.1. Decision Tree.** A decision tree is nonparametric among the supervised learning methods. It uses a tree-like approach for decisions, demonstrates the chance of event outcomes. This method provides decision rules which are interpretable and straight-forward and contains a conditional control statement. It allows to see and interpret the logic for the data, unlike another supervised model such as SVM. This method is also termed as a supervised learning model in machine learning because it allows automatically building predictive models via algorithms from a given set of observations (data) as a training dataset [38]. The decision model is built top-down in the form of tree structure from the (top) root node with subsequent decision nodes and leaf nodes. The root nodes are significant predictors, and the leaf nodes provide a final classification, as shown in Figure 1.

The decision tree generates rules based on splitting criteria. There are different algorithms for the construction of the decision tree, but one of the best is called the ID3 algorithm, which stands for Iterative Dichotomiser 3 [39]. ID3 [40] constructs a decision tree by constructing a top-down, greedy search through the given sets of training data to test each attribute at every node. It uses statistical metric call information gain to select which attribute to test at each node in the tree. This metric computes how well a given attribute separates the training examples according to their target classification.

**3.1.1. Entropy.** It is a measure in the information theory, which characterizes the impurity of an arbitrary collection of examples. If the target attribute takes on  $c$  different values, then the entropy  $S$  relative to this  $c$ -wise classification is defined as

$$E(S) = \sum_{i=1}^c -P_i \log_2 P_i, \quad (1)$$

where  $E$  is the entropy and  $p_i$  is the probability of  $S$  belonging to a class  $i$ . The logarithm of base 2 means the entropy is a measure of the encoding length computed in bits.

**3.1.2. Information Gain.** It computes the expected minimization in entropy by splitting the examples according to its attribute. The information gain,  $\text{Gain}(S; A)$  of an attribute  $A$ , concerning the collection of examples  $S$  is defined as

$$\text{Gain}(S, A) = E(S) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} E(S_v), \quad (2)$$

where  $\text{values}(A)$  is the set of all possible values for attribute  $A$  and  $S_v$  is the subset of  $S$  for which the attribute  $A$  has value  $v$ . This metric is used to rank attributes and build the decision tree where at each node is located the attribute with

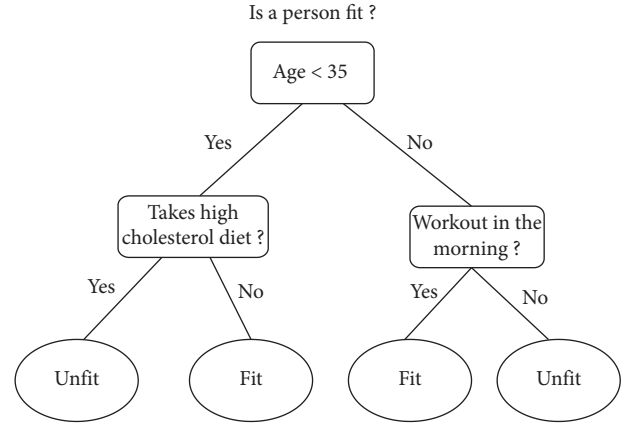


FIGURE 1: Example of a decision tree.

the highest information gain among the attributes not yet considered in the path from the root.

### 4. Experiments

Here, we answer the following questions:

- (i) Q1-Features: are all features important in the prediction process?
- (ii) Q2-Rules: what are the rules extracted by the DT?
- (iii) Q3-Accuracy: how accurate DT is compared to other state-of-the-art methods?

**Datasets:** the DT model is applied to the 1999 KDD Cup network intrusion dataset (<http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>). The data come from DARPA 98 Intrusion Detection Evaluation handled by Lincoln Laboratory at MIT. According to [41], these datasets were collected using multiple computers connected to the Internet to model a small US Air Force base of qualified personnel by using several simulated intrusions. There are 42 attributes used in this dataset. There are five main groups of which four of them are labeled as attacks, and one group is “normal,” which means no threat. The description of the attacks is shown in Table 1.

Previous studies using this data [42–44] have applied the classification algorithm to predict the personal attacks and report the performance of different supervised classification algorithm in multiclass settings. Intrusion detection is usually equivalent to a classification problem, such as a binary classification problem, i.e., identifying whether network traffic behavior is normal or malicious. In this study, we are not considering to make the multiclass prediction. We focus on explaining how algorithms come into a conclusion to predict the normal or malicious in a binary setting.

**4.1. Evaluation Metric.** The following evaluation metrics are used to assess the accuracy of the model.

**4.1.1. Precision.** In a classification problem, precision is defined as the number of true positives ( $t_p$ ) over the number of true positives plus the number of false positives ( $f_p$ ). Formally,



TABLE 1: Types of attack in the datasets.

Types of attack	Examples	Quantity	Proportion (%)
Denial of service (DoS)	smurf, apache2, pod, etc.	229,853	73.90
Remote-to-local (R2L)	imap, worm, phf, etc.	16,189	5.2
User to root (U2R)	perl, rootkit, and so on	228	0.07
PROBING	nmap, portsweep, etc.	4,166	1.34
Normal	Usual traffic patterns	60,593	19.48

$$\text{Precision} = \frac{t_p}{t_p + f_p}, \quad (3)$$

where  $t_p$  is true positives (correctly identified) and  $f_p$  is false positives (incorrectly identified).

**4.1.2. Recall.** In a classification problem, Recall is defined as the number of true positives ( $t_p$ ) over the number of true positives plus the number of false negatives ( $f_n$ ). Formally,

$$\text{Recall} = \frac{t_p}{t_p + f_n}, \quad (4)$$

where  $f_n$  is false negatives (incorrectly rejected).

**4.1.3. F1-Scores.** F1-score captures the tradeoff between precision and recall of a classifier model. It is combined metrics of both precision and recall and computed as the harmonic mean between these metrics:

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (5)$$

**4.2. Q1: Features.** There are 42 attributes in the datasets. Out of these 42 attributes, 41 attributes are classified into four different classes [45]:

- (1) Basic features (BFs) are the attributes of individual TCP connections
- (2) Content features (CFs) are the attributes within a connection suggested by the domain knowledge
- (3) Traffic features (TFs) are the attributes computed using a two-second time window
- (4) Host features (HFs) are the attributes designed to assess

The core concepts in machine learning hugely impact the performance of the model. It allows us to get rid of the irrelevant features from the model that do not contribute to the prediction accuracy. Less redundant features also mean less opportunity to make decisions based on noise. Thus, fewer features also reduce algorithm complexity to train the model faster. DT has a natural method for selecting essential features from the model. It can be obtained from the number and the quality of splits that are generated from a predictor variable [46].

The importance of features can be estimated from data by building a prediction model. DT has a built-in mechanism to report on variable importance. We randomly split

our datasets as follows: 60% data as training, 20% data as validation, and 20% data as the test set. DT considers all features and creates a split on the one that is separating class labels the best in terms of entropy (III-A1) measures using training datasets. The ranking of the important features is demonstrated in Figure 2.

We observed that feature V23 is ranked highest. It is from attacks that last for more than two seconds.

We are not sure by only manual inspection that is the critical feature to keep in the modeling process. Thus, we performed feature selection, which is considered as one of the traffic features (T) category. The name of the feature is “count,” which is computed using a two-second time window and is a numeric attribute. Similarly, we observed that feature V3 is ranked second, which is from the basic features (B) category, and the feature name is “service.” This feature is based on the attributes of the individual TCP connections, such as http and telnet. This feature is of the discrete attribute. The third highest ranked feature is “Flag,” which is also from the basic (B) category. This feature is about the normal or error status of the connection and is a discrete attribute. The description of the top 10 important features is shown in Table 2. From the table, we observed that one traffic (T), two basic (B), three content (C), and four host (H) feature categories are ranked according to their importance.

It implies that a combination of BCTH features gives a better model for intrusion detection.

**4.3. Q2: Rules.** We extracted the rules from our training set data. Figure 3 shows DT created from our dataset. The top node is the root node, and the bottom node is the leaf node. The rules traverse from the top nodes to the bottom nodes. There are a total of 19 steps involved in the construction of this tree. The steps are demonstrated in by depth-first fashion from the node by node. Each traversal from the root node to the leaf node gives the controlling rule for the decision. The leaf nodes are the decision nodes.

We extracted the rules from our model using rattle (<https://cran.r-project.org/web/packages/rattle/vignettes/rattle.pdf>) package in R. For the demonstration purpose, the five rules for predicting normal and malicious nodes are shown in Table 3.

**4.4. Q3: Accuracy.** We compared the performance of the decision tree with two popularly used classification algorithms: (i) support vector machines [47] and (ii) logistic regression [48]. The results of the comparison are shown in Figure 4. From the results, we observe that DT algorithms

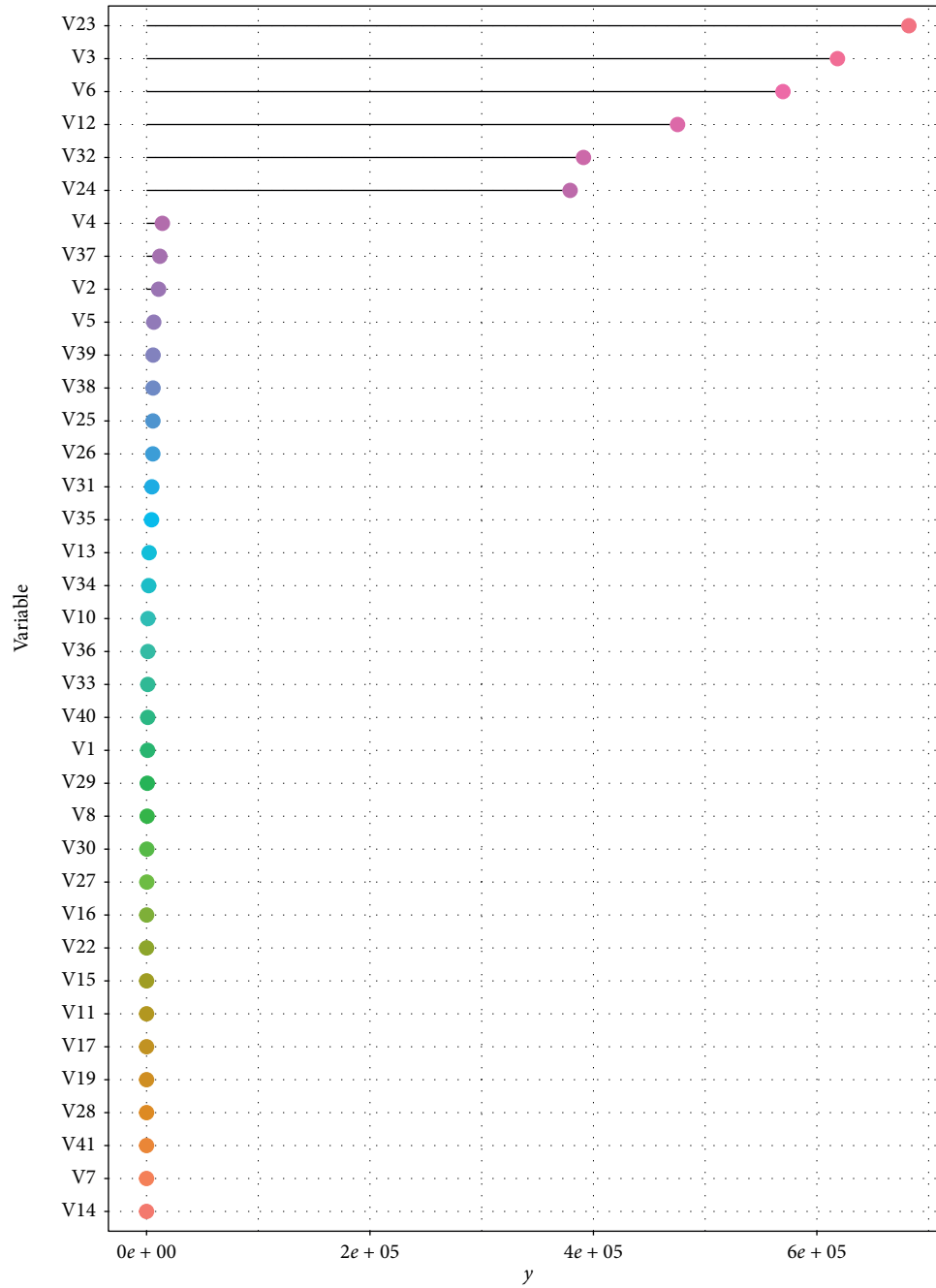


FIGURE 2: Ranking the features based on information gain.

have higher performances, in terms of precision, recall, and  $F1$ -score. However, the improvement is very marginal. DT performed equally in terms of precision and recall for predicting malicious and normal nodes. In contrast, SVM has similar performance for predicting malicious nodes but low performance for predicting normal nodes in comparisons to DT. LR performed better in comparison to SVM in predicting the normal nodes but did not perform better in predicting malicious nodes in comparison to SVM and DT.

SVM has a low recall in comparison to other methods in our data. In terms of the  $F1$ -score, also DT has the best performance. One of the reasons for DT to perform better is that it does not take features to be linear/normal or additive, and the possible interactions do not need to be prespecified. The problem, such as missing values of covariates, multicollinearity, and outlier, is automatically taken into account [49]. Moreover, LR is a linear model; thus, in these datasets, maybe all features are not linear, so we see its performance

TABLE 2: Description of top ten features used in a decision tree.

Sr. No.	Features	Description	Type	Label
V23	Count	Number of connections to the same host as the current connection in the past two seconds	Continuous	T
V3	Service	Network service on the destination, e.g., telnet	Discrete	B
V6	Flag	Normal or error status of the connection	Discrete	B
V12	Logged_in	1 if successfully logged in; 0 otherwise	Discrete	C
V32	Dst_host_count	% of connections that have continuous H “SYN” errors	Continuous	H
V24	Serror_rate	% of connections that have continuous H “SYN” errors	Continuous	H
V37	dst_host_srv_diff_host_rate	The percentage of connection that has different destination machine	Discrete	H
V2	Protocol_type	Type of the protocol, e.g., TCP and UDP	Discrete	C
V5	dst_bytes	Number of data bytes from destination to source	Continuous	C
V39	dst_host_srv_error_rate	Server error rate	Continuous	H

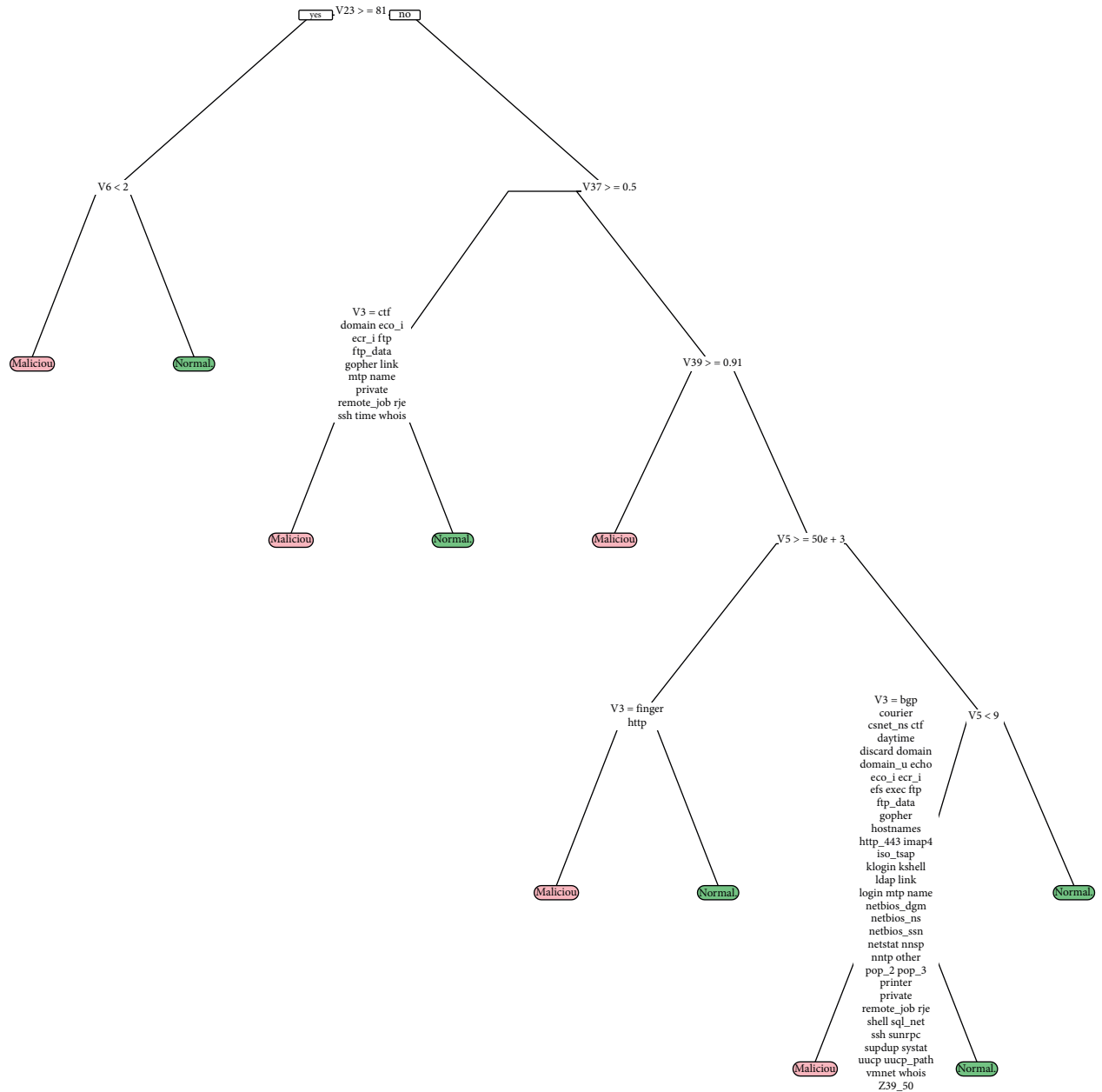


FIGURE 3: Decision tree from the training dataset. Pink nodes are malicious nodes, and green nodes are normal nodes.

TABLE 3: Decision rule description from the training sets for malicious and normal nodes. The bold text is the clauses for the rules.

Leaf numbers	Rules	Interpretation
63	[V42 = normal. Cover = 681132 (47%)]	
	V23 < 79.5	<b>IF</b> V23 < 79.5, V37 < 0.495, V39 < 0.905, V5 ≥ 8.5, <b>THEN</b> the class is normal, which covers 47% of terminal nodes, overall 681132 cases
	V37 < 0.495	
	V39 < 0.905	
	V5 ≥ 8.5	
125	[V42 = normal. Cover = 43147 (3%)]	
	V23 < 79.5	<b>IF</b> V23 < 79.5, V37 < 0.495, V39 < 0.905, V5 < 8.5, V3 = auth or finger or http or IRC or smtp or telnet or tftp_u or time or X11, <b>THEN</b> the class is normal which covers 3% of terminal nodes, overall 43146 cases
	V37 < 0.495	
	V39 < 0.905	
	V5 < 8.5	
	V3 = auth, finger, http, IRC, smtp, telnet, tftp_u, time, X11	
14	[V42 = malicious. Cover = 3559 (0.01%)]	
	V23 < 79.5	<b>IF</b> V23 < 79.5, V37 < 0.495, V39 ≥ 0.905, <b>THEN</b> the class is malicious, which covers very small amount of terminal nodes, overall 3559 cases
	V37 < 0.495	
	V39 ≥ 0.905	
60	[V42 = malicious. Cover = 1538 (0.005%)]	
	V23 < 79.5	<b>IF</b> V23 < 79.5, V37 < 0.495, V39 < 0.905, V5 ≥ 4.995e + 04, V3 = finger or http, <b>THEN</b> the class is malicious, which covers very small amount of terminal nodes, overall 1538 cases
	V37 < 0.495	
	V39 < 0.905	
	V5 ≥ 4.995e + 04	
	V3 = finger, http	
4	[V42 = malicious. Cover = 709860 (49%)]	
	V23 ≥ 79.5	<b>IF</b> V23 ≥ 79.5, V6 < 2, <b>THEN</b> the class is malicious, which covers 49% of terminal nodes, overall 709860 cases
	V6 < 2	

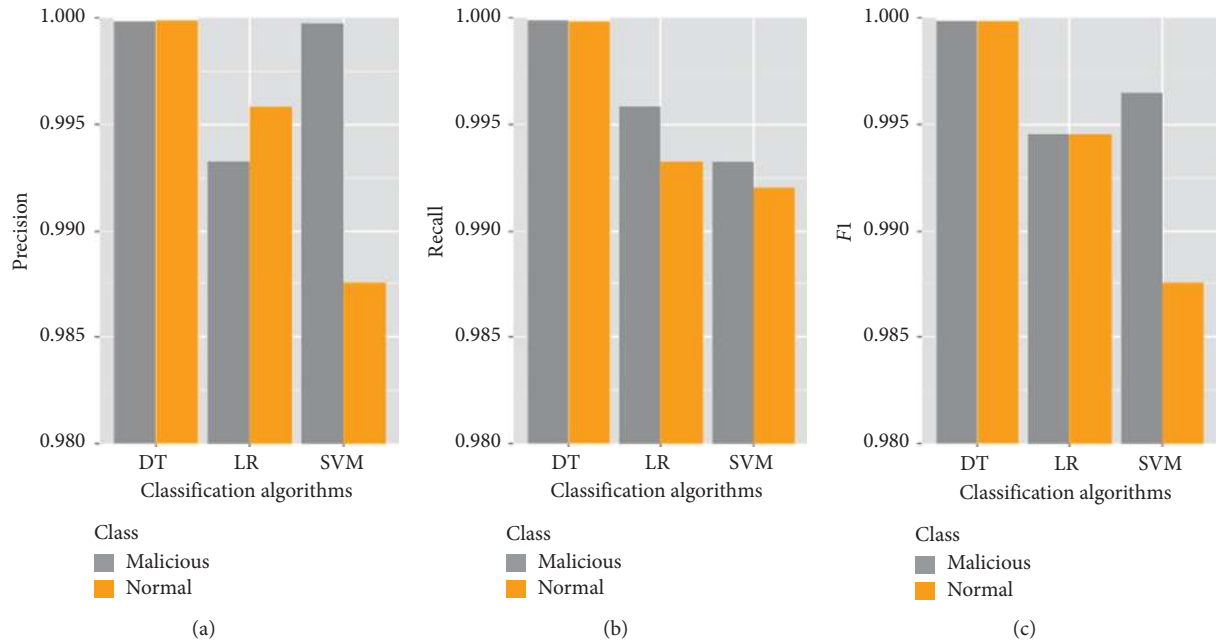


FIGURE 4: Performance of the state-of-the-art methods in predicting the classes between malicious and normal nodes. LR: logistic regression, DT: decision tree; SVM: support vector machines. (a) Precision. (b) Recall, (c) F1.



lower in comparison to DT. Similarly, we employ SVM with a linear kernel, but it performed better than LR but cannot outperform DT.

## 5. Discussion

An advantage of the presented explainable analysis of decision tree algorithm is to provide a trustworthy AI solution for IDS. In particular, the decision tree algorithm has explored its decisions using feature engineering and the rule-based model that human experts can understand (e.g., the underlying data evidence and causal reasoning). Understanding the characteristics of malicious attacks and vulnerability to IDS is essential to the success of future cyber-security systems.

It is interesting to note that a lot of research work based on the original KDD dataset reported high prediction accuracy using KDD intrusion detection data. The prior work benchmarks the supervised classification algorithms to show which algorithms perform better in the datasets. However, our approach is different. Detecting intrusion is important, but knowing and explaining to them how they are caused is an important issue. Machine learning algorithms are complex, and it is sometimes difficult how they derive the conclusion for decision-making, which makes them unusable for many scenarios. It is especially true for cases where the decision might need to be understood, such as in malicious node detection in a communication network. Any network engineers or security personnel who work on the attacks have a “right to an explanation.” A decision tree’s ability for human comprehension is often mimicking the human level thinking, so it is simple to understand the data and make some good interpretations [50].

Decision rules follow a common structure: IF, the conditions are met, and THEN, making a certain prediction. Decision rules are probably the most interpretable prediction models [51]. Their IF-THEN structure semantically simulates human language and the way we think, provided that the condition is built from intelligible features, the length of the condition is short (a small number of feature = value pairs combined with an AND), and there are not too many rules. The decision tree learns these IF-THEN rules from the datasets. In the context of malicious node identification, these rules can play an important role in avoiding large attacks and save the communication infrastructure from further damage. For example, the rules like this can be an alarm to security personnel: [IF  $V_{23} \geq 79.5$ ,  $V_6 < 2$  THEN the class is malicious, with the probability of 97%]. It means there are 97% chances of the malicious nodes if the number of connections to the host is greater than 79.5, error status of the connection is less than two, and then these nodes can be malicious, so higher caution is required. This rule is straightforward and highly informative to deal with the network security personnel to monitor malicious behavior in the network.

Another classification algorithm, such as SVM or even deep neural network, indeed gives us a better prediction. However, in terms of interpretability, the decision tree is considered to be one of the best methods. A tree is the

natural interpretation of humans. When a human constructs a decision tree, the questions and answers are based on their rationale and knowledge. In data science, the creation of these rules is usually administered by an algorithm learning which questions to ask by analyzing the entire dataset. In the context of the malicious node detection, the algorithm will look at entire datasets to figure out a different set of rules to classify “normal” and “malicious” nodes. The decision tree algorithms mathematically split the entire network dataset and judge its class. This whole process of learning network intrusion data through decision trees can classify new network data in a way that any human can understand and do judgment.

## 6. Conclusion and Future Work

AI makes decisions using a complex system of analysis to identify potentially hidden patterns and weak signals from large amounts of data. Given the consideration, interpretability in AI-based solutions becomes essential for enhancing the trust for the in real-life applications. Cyber-security systems (e.g., the IDS system) are one of the crucial sensitive applications where the systems are vulnerable to malicious attacks. Therefore, we explore decision tree algorithm for the malicious node identification by applying an openly available KDD dataset. We performed three important tasks in this dataset: (i) ranking the features, (ii) decision tree rule extraction, and (iii) comparison with the state-of-the-art algorithms. Not all features have an equal contribution to the malicious node prediction. We saw that the network traffic feature is computed using a two-second time window as one of the major predictors in the decision tree and is made root node by the algorithms. The second-highest ranked feature is a basic feature-based network service for individual TCP connections. The brief description of the ranking of the network features is given in IV-A3.

By observing the decision tree rules, we found out what set of feature values could distinguish malicious from regular network traffic. These rules are interpretable and help network security personnel to enhance the trust by taking a possible course of action in the case of malicious traffic identification. In this work, we explained the rules generated by the decision tree algorithms. These rules are simple IF-THEN rules with logical conditions. The beauty of the decision tree algorithm is the rule extraction, and explaining these rules to any network expert can make proper planning in the future to avoid the network intrusions. In Table 3, we interpreted the rules extracted by the decision tree algorithms.

We compared the performance of the decision tree algorithm with the other state-of-the-art algorithm. We found that the decision tree has a marginal improvement with other state-of-the-art methods in terms of precision, recall, and  $F1$ -scores. The decision tree algorithm is computationally cheaper and easy to evaluate and intuitive, unlike other methods such as SVM. SVM is a compelling method, but it is computationally expensive and noninterpretable. There is an essential advantage of the decision tree; for instance, the missing values in the data also do not affect the

process of building a decision tree. This model does not need to be normally distributed. However, there are caveats to these methods. There may be a chance of overfitting when the algorithm captures noise in the dataset, due to this algorithm might perform better in the training set but perform poorly in the test set. Another noteworthy limitation is for data, including categorical variables with a different number of levels, information gain in decision trees is biased in favor of those attributes with more levels. This behavior might impact prediction performance.

## Data Availability

The data used to support the findings of this study are from the KDD benchmark dataset.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This work was supported by Science Foundation Ireland.

## References

- [1] D. Pienta, S. Tams, and J. Thatcher, "Can trust be trusted in cybersecurity?" in *Proceedings of the 53rd Hawaii International Conference on System Sciences*, Maui, HI, USA, January 2020.
- [2] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser et al., "Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020.
- [3] N. C. Luong, D. T. Hoang, S. Gong et al., "Applications of deep reinforcement learning in communications and networking: a survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.
- [4] T. Perarasi, S. Vidhya, M. Leeban Moses, and P. Ramya, "Malicious vehicles identifying and trust management algorithm for enhance the security in 5G-VANET," in *Proceedings of the 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, July 2020.
- [5] W. Guo, "Explainable artificial intelligence for 6G: improving trust between human and machine," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, 2020.
- [6] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.
- [7] D. Castelvechi, "Can we open the black box of AI?" *Nature*, vol. 538, no. 7623, p. 20, 2016.
- [8] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: a survey on methods and metrics," *Electronics*, vol. 8, no. 8, p. 832, 2019.
- [9] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493–501, 2019.
- [10] P. Svenmarck, L. Luotsinen, M. Nilsson, and J. Schubert, "Possibilities and challenges for artificial intelligence in military applications," in *Proceedings of the NATO Big Data and Artificial Intelligence for Military Decision Making Specialists' Meeting*, Bordeaux, France, May 2018.
- [11] M. Stampar and K. Fertalj, "Artificial intelligence in network intrusion detection," in *Proceedings of the 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1318–1323, IEEE, Opatija, Croatia, May 2015.
- [12] W. Lee, S. J. Stolfo, P. K. Chan et al., "Real time data mining-based intrusion detection," in *Proceedings of the DARPA Information Survivability Conference and Exposition II. DISCEX'01*, pp. 89–100, IEEE, Anaheim, CA, USA, June 2001.
- [13] S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting traffic anomaly detection using software defined networking," in *Proceedings of the International Workshop on Recent Advances in Intrusion Detection*, pp. 161–180, Springer, Menlo Park, CA, USA, September 2011.
- [14] M. Bojarski, D. Del Testa, D. Dworakowski et al., "End to end learning for self-driving cars," 2016, <https://arxiv.org/abs/1604.07316>.
- [15] T. Zahavy, N. Ben-Zrihem, and S. Mannor, "Graying the black box: understanding DQNs," in *Proceedings of the International Conference on Machine Learning*, pp. 1899–1908, New York, NY, USA, 2016.
- [16] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
- [17] G. Kumar, K. Kumar, and M. Sachdeva, "The use of artificial intelligence based techniques for intrusion detection: a review," *Artificial Intelligence Review*, vol. 34, no. 4, pp. 369–387, 2010.
- [18] K. Cup, <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 2007.
- [19] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proceedings of the 2003 SIAM International Conference on Data Mining*, pp. 25–36, SIAM, San Francisco, CA, USA, May 2003.
- [20] N. Ye, Y. Zhang, and C. M. Borror, "Robustness of the markov-chain model for cyber-attack detection," *IEEE Transactions on Reliability*, vol. 53, no. 1, pp. 116–123, 2004.
- [21] D. Novikov, R. V. Yampolskiy, and L. Reznik, "Anomaly detection based intrusion detection," in *Proceedings of the Third International Conference on Information Technology: New Generations (ITNG'06)*, pp. 420–425, IEEE, Las Vegas, NV, USA, April 2006.
- [22] A. N. Toosi and M. Kahani, "A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers," *Computer Communications*, vol. 30, no. 10, pp. 2201–2212, 2007.
- [23] M. K. Lahre, M. T. Dhar, D. Suresh, K. Kashyap, and P. Agrawal, "Analyze different approaches for ids using KDD 99 data set," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 1, no. 8, pp. 645–651, 2013.
- [24] Z. Zhang and H. Shen, "Application of online-training SVMs for real-time intrusion detection with different considerations," *Computer Communications*, vol. 28, no. 12, pp. 1428–1442, 2005.
- [25] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: a rule-based intrusion detection approach," *IEEE Transactions on Software Engineering*, vol. 21, no. 3, pp. 181–199, 1995.
- [26] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection model," in

- Proceedings of the IEEE Symposium on Security and Privacy*, May 1999.
- [27] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," *Journal of King Saud University-Computer and Information Sciences*, vol. 29, no. 4, pp. 462–472, 2017.
  - [28] S. Chebroli, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers & Security*, vol. 24, no. 4, pp. 295–307, 2005.
  - [29] S. Zaman and F. Karray, "Lightweight ids based on features selection and ids classification scheme," in *Proceedings of the 2009 International Conference on Computational Science and Engineering*, pp. 365–370, IEEE, Vancouver, BC, Canada, August 2009.
  - [30] K. Vimalkumar and N. Radhika, "A big data framework for intrusion detection in smart grids using Apache spark," in *Proceedings of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 198–204, IEEE, Udupi, India, September 2017.
  - [31] S. Balakrishnan, K. Venkatalakshmi, and K. Arputharaj, "Intrusion detection system using feature selection and classification technique," *International Journal of Computer Science and Application*, vol. 3, no. 4, pp. 145–151, 2014.
  - [32] S. V. Farrahi and M. Ahmadzadeh, "KCMC: a hybrid learning approach for network intrusion detection using k-means clustering and multiple classifiers," *International Journal of Computer Applications*, vol. 124, no. 9, 2015.
  - [33] S. Paliwal and R. Gupta, "Denial-of-service, probing & remote to user (R2L) attack detection using genetic algorithm," *International Journal of Computer Applications*, vol. 60, no. 19, pp. 57–62, 2012.
  - [34] M. Alkasasbeh, G. Al-Naymat, A. Hassanat, and M. Almseidin, "Detecting distributed denial of service attacks using data mining techniques," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 1, pp. 436–445, 2016.
  - [35] K. Peng, V. Leung, L. Zheng, S. Wang, C. Huang, and T. Lin, "Intrusion detection system based on decision tree over big data in fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 4680867, 10 pages, 2018.
  - [36] S. M. Othman, F. M. Ba-Alwi, N. T. Alsohybe, and A. Y. Al-Hashida, "Intrusion detection model using machine learning algorithm on big data environment," *Journal of Big Data*, vol. 5, no. 1, p. 34, 2018.
  - [37] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: an overview of interpretability of machine learning," in *Proceedings of the 2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 80–89, IEEE, Turin, Italy, October 2018.
  - [38] P. Geurts, A. Irtuthum, and L. Wehenkel, "Supervised learning with decision tree-based methods in computational and systems biology," *Molecular Biosystems*, vol. 5, no. 12, pp. 1593–1605, 2009.
  - [39] S. W. Norton, "Generating better decision trees," *IJCAI*, vol. 89, pp. 800–805, 1989.
  - [40] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.
  - [41] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cyber-Security*, vol. 2, no. 1, p. 20, 2019.
  - [42] K. A. P. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of things: a survey on machine learning-based intrusion detection approaches," *Computer Networks*, vol. 151, pp. 147–157, 2019.
  - [43] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, "Selecting features for intrusion detection: a feature relevance analysis on KDD 99," in *Proceedings of the Third Annual Conference on Privacy, Security and Trust*, pp. 1723–1722, New Brunswick, Canada, October 2005.
  - [44] M. Sabhnani and G. Serpen, "Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set," *Intelligent Data Analysis*, vol. 8, no. 4, pp. 403–415, 2004.
  - [45] P. Aggarwal and S. K. Sharma, "Analysis of KDD dataset attributes—class wise for intrusion detection," *Procedia Computer Science*, vol. 57, pp. 842–851, 2015.
  - [46] E. Tuv, A. Borisov, G. Runger, and K. Torkkola, "Feature selection with ensembles, artificial variables, and redundancy elimination," *Journal of Machine Learning Research*, vol. 10, no. 45, pp. 1341–1366, 2009.
  - [47] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.
  - [48] R. E. Wright, *Logistic Regression*, American Psychological Association, Washington, DC, USA, 1995.
  - [49] M. Tandan, M. Timilsina, M. Cormican, and A. Vellinga, "Role of patient descriptors in predicting antimicrobial resistance in urinary tract infections using a decision tree approach: a retrospective cohort study," *International Journal of Medical Informatics*, vol. 127, pp. 127–133, 2019.
  - [50] M. Wu, M. C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez, "Beyond sparsity: tree regularization of deep models for interpretability," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.
  - [51] M. Craven and J. W. Shavlik, "Extracting tree-structured representations of trained networks," in *Advances in Neural Information Processing Systems*, pp. 24–30, MIT Press, Cambridge, MA, USA, 1996.