A tree-based stacking ensemble technique with feature selection for network intrusion detection

Mamunur Rashid ; Joarder Kamruzzaman ; Tasadduq Imam ; Santoso Wibowo ; Steven Gordon

Noname manuscript No.
(will be inserted by the editor)

A Tree-based Stacking Ensemble Technique with Feature Selection for Network Intrusion Detection

Md. Mamunur Rashid · Joarder Kamruzzaman · Tasadduq Imam · Santoso Wibowo · Steven Gordon

Received: date / Accepted: date

Abstract Several studies have used machine learning algorithms to develop intrusion systems (IDS), which differentiate anomalous behaviours from the normal activities of network systems. Due to the ease of automated data collection and subsequently an increased size of collected data on network traffic and activities, the complexity of intrusion analysis is increasing exponentially. A particular issue is, due to statistical and computation limitations, a single classifier may not perform well for large scale data as existent in modern IDS contexts. Ensemble methods have been explored in literature in such big data contexts. Although more complicated and requiring additional computation, literature has note that ensemble methods can result in better accuracy than single classifiers in different large scale data classification contexts, and it is interesting to explore how ensemble approaches can perform in IDS. In this research, we introduce a tree-based stacking ensemble technique (SET) and test the effectiveness of the proposed model on two intrusion datasets (NSL-KDD and UNSW-NB15). We further enhance incorporate feature selection techniques to select the best relevant features with the proposed SET. A comprehensive performance analysis shows that our proposed model can better identify the normal and anomaly traffic in network than other existing IDS models. This implies the potentials of our proposed system for cybersecurity in Internet of Things (IoT) and large scale networks.

Keywords Machine learning · ensemble techniques · anomaly detection · cybersecurity · Intrusion detection seystem

Md. Mamunur Rashid, Santoso Wibowo, Steven Gordon
School of Engineering and Technology, CQUniversity, Australia
E-mail: {md.rashid, s.wibowo, s.d.gordon}@cqu.edu.au

Joarder Kamruzzaman
School of Engineering and Information Technology Federation University, Australia.
E-mail: { joarder.kamruzzaman}@federation.edu.au

Tasadduq Imam
School of Business and Law, CQUniversity, Australia.
E-mail: { t.imam}@cqu.edu.au

1 Introduction

Recent technical advancements has led to exponential growth of Internet-of-Things (IoT) applications in several areas including smart city, smart transportation and smart grids [1]. The adoption of Industry 4.0 employing IIoT (industrial IoT) is poised to boost productivity through smart industries and utilities as well as agriculture services through precision agriculture. However, this has exposed such IoT based infrastructure and enterprise systems to increased vulnerability and attacks, creating huge demand for cybersecurity to tackle various types cyber-attacks [1]. A study by AV-TEST [2] revealed that, in 2010, the businesses and organizations experienced 50 million malware attacks which reached to 900 million by 2019 and this number is still growing. These cyber-attacks have caused severe harm and economic losses to organizations as well as individuals. Recently, Juniper research [3] has reported that the number of data breaches will triple in the next 5 years and the annual cost of these breaches will cost over 5 trillion USD by 2024 worldwide.

In addition, Juniper Research [3] states that threat actors will continue to target healthcare providers and vaccine makers. Therefore, it is critical to build a strong cybersecurity mechanism which will be able to detect different cyber-attacks in a timely manner and ensure the security of the relevant systems [4].

Usually, a cybersecurity system is a combination of networks as well as computer security systems. Different components (i.e., firewall) and cryptography mechanisms are introduced to intercept the cyber-attacks, and an IDS is used to prevent the external attacks respectively [5]. Here, the main objective of an IDS is to recognise different modes of abnormal activities in a network traffic, and then use the available security systems for prevention. An IDS is also used to define, evaluate, and recognise unauthorised activities in the system, such as unauthorised access or alteration and destruction [6, 7]. To promote the security of a system, we need to first identify the cyber-attacks and then create a robust IDS for securing both enterprise networks and IoT-based systems.

Depending on the usage spectrum, IDSs could be of distinct types such as host-based IDS (HIDS) and network-based IDS (NIDS) [7]. A HIDS operates on a single computational device and tracks suspicious or malicious software components or unknown malicious codes affecting its operating system; whereas, a NIDS analyses and tracks data for irregular traffic within a network [7]. Also, depending on how IDS classify network irregularities, they can be categorised into two types: misuse or signature-based and anomaly-based [8,9]. A misuse or signature based IDS uses signatures and patterns, like, byte sequence of a network traffic, for detecting compromised systems during attacks. Such signature based systems are often used by anti-virus software, especially to match against known attacks. For example, byte sequence of a network traffic utilised by malware may be regarded as a signature. These bit sequences are used as a signature by anti-virus software and can identify attacks by matching them. The main advantage of signature-based IDS is that it can easily identify the known attacks but it is not well suitable for dealing with new attacks due to no signature or pattern guiding the detection. However, it is not suitable for dealing with new attacks as there is no signature or pattern that can be extracted from previous cases. On the contrary, an anomaly-based IDS analyses the network's activity and identifies abnormalities in the event of any anomalies based on their deviation from normal profiles. The key benefit of using

anomaly-based IDS is that they are capable of identifying new and unseen vulnerabilities. However, while an anomaly-based IDS may consider an unseen activity as an anomaly, in real scenario this might not be the case and there can be high false alarm rates which is also undesirable. It can be seen that developments of IDS for improving the defense of data breaches remains a major research effort in the security domain [10]. Research has, hence, focused on improving the defense against data breaches in IDS [10]. This paper develops an efficient detection method based on artificial intelligence (AI), particularly machine learning (ML), for addressing such security issues.

Much research has been conducted in the application of machine learning techniques for improving the performance of IDS [11, 12]. However, these systems are limited due to their use of single classifiers and consequently being unable to detect and prevent severe attacks. To deal with this issue, Hansen and Salamon [13] have developed multiple ensemble techniques that result in a better performance than single classifiers. Pham et al. [14], in this regard, point to several factors such as feature selection, base classifier, and ensemble algorithms that can influence the performance of IDS [14]. This implies the importance of exploring ensemble-based IDS from various respects.

In predictive analytics, tree-based ML methods have shown promising results [1]. Further, tree-based classifiers can be trained much faster than many other types of machine learning models [23]. With sophisticated network monitoring system and IoT, network administrators tend to collect huge amount of data continually for updating the model. As such, tree-based models, with their high detection performance and less computational time, can offer added advantages in an IDS. Considering their suitability, in this article we propose a tree-based stacking ensemble technique for improving the performance of IDS.

Another essential factor to consider for IDS's accuracy is identifying security sensitive features for model building because today's security datasets of high dimensions may include features that are less or not relevant from security perspective. A model with these redundant features triggers several problems. Firstly, the high variance normally causes a tree-based model to over-fit as it can only learn from one decision direction. Secondly, complexity for model training (in terms of computational cost and time) is higher for high dimension. Thirdly, the model's ability to generalise on unseen data suffer with redundant features. Then, the research question is how we can mitigate these problems and build an efficient data-driven IDS model.

To address these research issues, further to introducing a tree-based stacking ensemble model, we incorporate feature selection. We first calculate the rating of the available features according to their significance in modelling and choose the most relevant features. Based on the selected significant features, we subsequently construct a tree-based stacking ensemble model for IDS.

The contributions of this paper are summarised as follows:

- Measure the significance of the features to reduce the feature dimensions in an ML-based IDS.
- Develop a tree-based stacking model for intrusion detection which considers the ranking of features based on a score and then creates a stacking model built on those features. To reduce model over-fitting cross-validation, scaling of the input feature and model hyperparameter fine-tuning have been employed.

- Finally, experiments demonstrate the efficacy of the proposed tree-based stacking model and show that it outperforms the existing models in detecting intrusion in terms of accuracy and false alarm rate (FAR). Statistical significance test confirms the proposed model's superiority over the existing competing models.

The remainder of the paper is organised as follows. Section 2 presents the recent works in hybrid and ensemble ML techniques for IDS. Section 3 presents the development of the tree-based stacking model. Section 4 presents the experimental results, and this is followed by Section 5 which concludes the paper.

2 Related Works

Researchers have introduced various methods [15,16,17] in recent years to increase the performance of an IDS. This section presents these works that have used hybrid and ensemble ML techniques for IDS.

Panigrah et al. [15] presented a fuzzy and rough set theory based hybrid IDS model to identify the normal and anomalous behavior by analysing the network data. Their model consists of two stages. It, firstly, identified the most relevant features using the rank and search based feature selection techniques and secondly, used the five classifiers - fuzzy nearest neighbour (FNN), fuzzy-rough nearest neighbour (FRNN), fuzz-rough ownership nearest neighbours (FRONN), vaguely quantified nearest neighbours (VQNN) and ordered weighted average nearest neighbours (OWANN)) to distinguish between the normal and anomalous states of the network dataset. They performed their experiments on the NSL-KDD dataset [27]. Performance analysis demonstrate that FRNN classification technique provide better result which achieved 99.61% detection rate and 99.61% FAR.

In [16], Bayes net classifier-based IDS has been introduced by the same authors where they used search (K2, Tabu and Hill Climbing), and tree augmented naive-bayes (TAN) technique. Moreover, to reduce the feature dimensionality, they used entropy and statistical based feature selection techniques. They evaluated their proposed model on the NSL-KDD dataset and the outcomes demonstrate that TAN classifier with One-R feature selection method produces better performance and model achieved 99.74%, 99.76% and 0.279% accuracy, DR, and FAR, respectively.

In [17], Tama et al. proposed an ensemble-based IDS which is a combination of a hybrid feature selection method and a two-stage classifier. The feature selection techniques consists of particle swarm optimization, ant colony algorithm, and genetic algorithm. After selecting the relevant features, they applied two meta learner-based ensemble techniques where in the first and second stage, they used rotation forest and bagging, respectively. They considered the NSL-KDD and UNSW-BC15 datasets and the results show that this technique achieved 96.38% and 81.53% accuracy for the datasets, respectively. In [18], Toma el al. further extended this study comparing various ensemble classifiers such as bagging, boosting, majority voting, and stacking applied to IDS. They used two intrusion datasets (NSL-KDD and GPRS) to perform experiments. The results showed that stacking performed superior to other ensemble classifiers and achieved 99.28%, 0.9915, 0.995 and 0.9933 in terms of measured through accuracy, precision, recall, and F1-score, respectively.

Smitha et al. [19] proposed a stacking ensemble-based IDS where they used Linear Regression (LR), Random Forest (RF), and K-Nearest Neighbour (KNN) as base classifiers and Support Vector Machine (SVM) as the meta-classifier. They used entropy-based feature selection approach to identify important features. Experiments were performed on UNSW-NB15 and UGR'16 heterogeneous datasets. The results showed that the proposed model achieved 94% accuracy and 5.2% FAR on the UNSW-NB15 dataset.

Paulauskas et al. [20] analysed the data pre-processing effect to attack detection accuracy by using Decision Trees (DT), Naïve Bayes (NB) and Rule-Based classifiers on NSL-KDD dataset.They also introduced an ensemble model-based IDS consisting of four base classifiers, namely, J48, C5.0, Naive Bayes, and Partial Decision Tree (PART) where they combined multiple classifiers to create a stronger learner. Experiments t showed that the ensemble approach performed better than individual classifier for the NSL-KDD dataset and achieved 84% accuracy when the model was tested with KDDTest+ datset.

In [21], Moustafa et al. presented an Adaboost ensemble technique consisting of DT, NB, and Artificial Neural Network (ANN) to detect malicious activities in IoT networks where the DNS, HTTP, and MQTT protocols were considered to generate new statistical flow features. The UNSW-NB15 and NIMS botnet [51] datasets were used to evaluate the proposed technique and experiment outcomes showed that their ensemble technique yielded a higher DR and a lower FAR compared to other state-of-the-art techniques.

Salo et al. [22] introduced a hybrid IDS where they used two feature selection methods(information gain and principal component analysis (PCA)) and then combined with an ensemble method using SVM, KNN, and ANN. The model was evaluated on NSL-KDD, and Kyoto 2006+ [50]) datasets and achieved 98.24 % and 98.95% accuracy on both datasets respectively.

In [23], Zhou et al. presented an ensemble based IDS with feature selection. For feature selection, they combined correlation-based feature selection (CFS) and Bat algorithm [24]. After selecting the features they used a voting ensemble technique on NSL-KDD, AWID [46] and CIC-IDS2017 [47] datasets to perform the experiment. Results shows that proposed model archived 99.8%, 99.5% and 99.8% accuracy respectively for the mentioned datasets.

In [52], Ahmed et al. introduced a new intrusion dataset called game theory and cybersecurity (GTCS) where they used Ostinato and Kali Linux to generate benignity and attack traffic, respectively. This dataset contains eighty-four features and recent network attacks. They used six ML algorithms to identify the benign and attack classes. Finally, they used the majority voting ensemble technique to enhance the performance further.

In [25], Rashid et al investigated different machine learning algorithms to detect cyber-attacks from IoT based smart city applications with feature selection technique. The experiments were performed on two recent intrusion datasets: UNSW-NB15 and CICIDS2017. The results show that the stacking ensemble performs well compared to other classifiers and achieved 96.83% and 99.9% accuracy for UNSW-NB15 and CICIDS2017 datasets respectively. Our proposed stacking model, which also involves feature selection and ensemble similar to [25] in the IoT network, is different in the following ways. Firstly, the proposed model considers only the tree-based algorithms as the base classifiers whereas the stacking model in [25] used tree as well as other algorithms as the base classifier. Secondly, the

proposed model used recently widely used XGBoost classifier, and thirdly, the proposed model used selectkbest feature selection model to select the best 20 features whereas [25] used information-gain model and used 25 features. Overall, our algorithm chooses classification approach and feature selection models that are less computationally intensive than the existing technique, implying the model can be built faster.

3 Materials and Methods

In this section, we introduce a stacking-based intrusion detection system which consists of the following steps: exploring intrusion datasets, pre-processing of data, ranked the features based on the feature importance, and design a stacking model. We present these steps in the following sub-sections.

3.1 Exploring intrusion datasets

Intrusion datasets contain a number of records that consist of several network related features and facts. These features and facts can be used to build a data-driven IDS [25]. Therefore, it is very essential to explore raw intrusion data to differentiate the normal and abnormal behavior. In this research, two publicly available intrusion datasets, namely, NSL-KDD and UNSW-15NB are used and considered two categories for detection purpose - normal and anomaly.

The NSL-KDD dataset was created using the KDDcup99 dataset [28] is a very popular intrusion datset. An issue of the KDDcup99 dataset is the presence of duplicated records in both the training and test data and which can bias classifiers. The issue was solved in the NSL-KDD dataset. The number of records in the training (KDDTrain++) dataset is 125,973 with 67,343 and 58,630 samples, respectively, representing the benign and anomaly conditions. The distribution the of benign and anomaly records on the NSL-KDD datasets is shown in Table 1.

The UNSW-NB15 dataset is a widely used intrusion dataset that contains modern attacks. This dataset was developed in the Australian Centre for Cyber Security (ACCS) in 2015 where the raw network packets were generated by the IXIA PerfectStrom tool. This dataset contains over 2.5 million records. In this work, we have used a random portion of a subset that contains 175,341 records among which 56,000 are normal and 119,241 anomalies. This dataset was split into two parts: a training set and a test set, each with 140,272 and 35,069 records, respectively. The training and test sets have the same distribution of the benign and anomalous classes as the original dataset and shown in Table 1.

3.2 Pre-processing of data

Data pre-processing consist of feature encoding and scaling based on the intrusion dataset's characteristics.

Feature encoding: The used datasets in this research contain the numeric as well as categorical feature values where the majority of the features are numeric and only a few are categorical. For example , the UNSW-NB15 dataset contains

Table 1 The distribution of the benign and anomaly class in NSL-KDD and UNSW-NB15 datasets

Class	NSL-KDD	UNSW-NB15
Benign	67,343	56,000
Anomaly	58,630	119,241
Total	1,25,973	175,341

three categorical features which are protocol, service, and flag. Therefore, all the categorical features must be converted into vectors to suit these data to the ML algorithm for training.

Different techniques are available to convert the categorical data into vectors. Among them the mostly used techniques are 'Label Encoding' and 'One Hot Encoding'. Here we have used the first one, because in later technique, the number of feature dimensions significantly increases [30]. It transformed the feature values into numeric values in a straight forward way. Label encoding, for example, will convert the values [icmp, http, tcp] into vectors[0,1,2] when it comes to protocol-related features.

Feature scaling: A mechanism for normalizing the range of feature values is called feature scaling that is important for the ML techniques that compute distances between data. Different features have different values, and feature scaling ensures that the range of features is normalized. As a result, each feature equally contributes to the final distance calculation. In our experiment, we have used the minimum-maximum method [31] which is defined as,

$$F = \frac{F - F_{min}}{F_{max} - F_{min}} \tag{1}$$

where F_{min} and F_{max} show the min and max values of feature F.

3.3 Determining the best feature set

One of the most important machine learning tasks that can influence a model's effectiveness is feature selection. The main objective of feature selection is to determine the most important features for IDS that would contribute to its ability in identifying anomalies. In this paper, we used selectkbest model from sklearn [32] to identify the significant feature set. In this feature selection approach, at first the variance of each feature is calculated, and then a subset of features is selected based on a user-specified threshold where it assumes that features with a higher variance may contain more useful information. Selectkbest model selects k features according to the highest scores.

3.4 Stacking based model

Ensemble techniques are a common machine learning methodology that combines multiple base classifier to create a single optimal predictive model [33, 34]. An ensemble approach takes a number of models into account and incorporates them

to create a single model. The final model will remove the weakness of each individual learner and create a strong model that will increase the model performance. Algorithm 1 describes the steps involved in training our proposed model.

Algorithm 1 Algorithm 1 - Stacking

Input: Training data $D = \{A_i, B_i\}_{j=1}^{p}$ where $A = A_i \epsilon R_q$ is given records set and $B = B_i \epsilon N$ is labels set
Output: Ensemble E's predictions
Begin
Step 1: Divide D into 'p' equal-sized subsets randomly, i.e., $D = D_1, D_2, ..., D_p$
Step 2:
for p **do** ← 1 to P
  Learn base classifiers namely DT, RF and Xboost
  for q **do** ← 1 to Q
    Learn a base classifier C_{pq} for T or T_m
  end for
  Step 3: Derive a training set for meta-classifier (XGBoost)
  for e **do**ach $A_i \epsilon D_p$
    Extract a new instance (a_i, b_i) where $x_i = C_{p1}(A_i), C_{p2}(A_i), \ldots, C_{pQ}(A_i)$
  end for
end for
End

The stacking ensemble is a generic framework that consists of two levels of classifier: base classifier and meta-classifier. In the case of the base classifier, the base (initial) classifiers are trained with the training dataset and a new dataset is created for the meta-classifier. The meta-classifier is then trained with this new dataset. Finally, the trained meta-learner is used to predict the test dataset. The most important phase in stacking is to select the best base learner where many base classifiers are selected instead of just one to train the dataset. Here, we introduced a tree-based stacking ensemble model where decision tree (DT), random forest (RF) and Extreme gradient boosting (XGBoost) [35] are used as base classifiers and XGBoost used as a meta classifier.

In the stacking ensemble technique, multiple types of base classifiers are used instead of one, as a result, this can increase the model computational cost and complexity. However, the reason for using stacking ensemble are two folds: firstly, it can achieve better performance by reducing the variance component of prediction errors, and secondly, it can enhance robustness by reducing the dispersion of the predictions [56]. Also, with increasingly powerful computational resources at our disposal at lowering hardware cost, higher model performance outweighs the computational cost for model building. Our proposed stacking model consists of three tree-based classifiers, namely, DT, RF, and XGBoost. The reasons for using tree-based classifiers are as follows:

- A tree-based model is computationally fast compared to other ML models (e.g., ANN, SVM or KNN),
- In the real-time scenario where a model update is required to capture the recent trend/knowledge, tree-based models are easier to adjust,
- A tree-based method is ideal for data resampling and feature subsampling, both of which assure the ensemble technique's diversity, and

– A tree-based method can handle any form of data without converting or normalizing it because each split in the tree employs the best single variable [36].

We also justified the use of the tree classifiers in this research. In [37], Fernández-Delgado et al. highlight that random forest is a highly effective classification technique across a range of real world problems. Random forest has also been noted to show promising outcomes in other research (e.g. [38]–[40]). However, as [41] notes, a random forest can produce biased outcomes for categorical variables with various levels. Existing research further promotes decision tree, especially with its outcomes being easily interpretable and its training having robustness against outliers [42]. XGBoost is a newer tree classifier which can scale to large-scale data [43] and is achieving increasing acceptance across different domains, including cyber security, for its high performance (e.g. [44]–[46]).

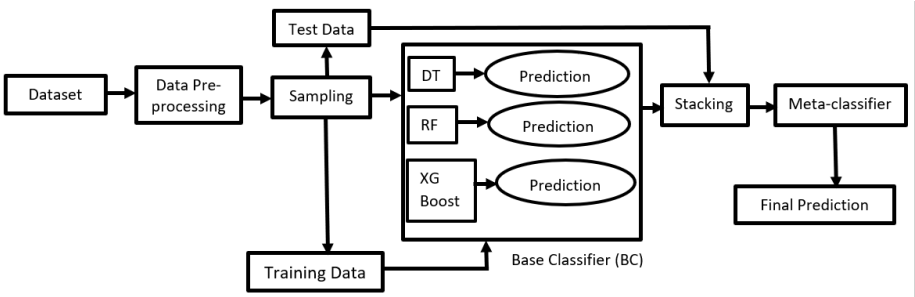


Fig. 1 Stacking ensemble based intrusion detection model

Fig 1 shows the proposed framework which involves pre-processing to choose the most important features by using the selectkbest feature selection technique. After that, the tree-based classifiers were used to construct the stacking ensemble model. In this model, at first, we have used DT, RF and XGBoost [35- 38] as base classifiers. We used the training dataset to train the base classifier to build multiple learners and their outputs are combined to generate a new dataset for the meta-classifier. The base classifier can be homogeneous (the combination of the same classifiers) or heterogeneous (the combination of different classifiers).

The meta-classifier takes base classifiers' outputs as inputs and trains on that input set and adjusts the error of the base classifiers for the optimal outcome. To perform k-fold cross-validation, repeat this phase k times to decrease the error and enhance prediction accuracy. At this stage, the meta-classifiercan generalize for the input data. A single classifier-based IDS model may perform well on the training dataset, but may perform poorly on the unseen test dataset. The stacking ensemble can reduce this risk by averaging outcomes from the constituent classifiers. The reason is that, if one of the selected classifiers in the ensemble is not perform well, the risk of relying on a single classifier can be reduced by averaging all of them. Although a stacking ensemble not ensure the best output for every problem, it does eliminate the risk of weak classifier selection.

To avoid the over-fitting, the proposed model used feature scaling, cross-validation, and hyperparameter tuning to select the best model parameters. We have scaled all

the input features, into a range between 0 and 1 using the min-max feature scaling technique that reduce the model complexity [58]. We have used cross-validation as a preventative measure against overfitting where the initial training dataset is used to generate multiple mini train-test splits. Then we used these splits to tune the model. We have used 10-fold cross-validation where we partitioned the data into 10 subsets referred to as fold. Then, we iteratively trained the algorithm on 9 folds while using the remaining fold as the test set. Finally, we have tuned the algorithm's parameters to find the best parameters of a specific model. For that, we have used the GridSearchCV() class from the scikit-learn package which exploits grid search method for parameter optimization within the setting of cross-validation.

4 Experimental results

This section evaluates the performance of the proposed models. At first, we setup the experimental environment and then presented results analysing various aspects of the model in relation to intrusion detection.

Extensive performance analyses have been performed on the intrusion datasets which contain the normal and anomaly classes. Details of these datasets are presented the Section 3. The Python programming language and several libraries such as Pandas, Numpy, and sklearn, were used to evaluate performance, and the program run on an HP (ELITEBOOK) laptop with Windows 10 Education 64-bit OS, core-i5 processor with 16 GB RAM. The hyperparameter values of the base classifiers chosen through grid search are shown in Table 2. The adopted evaluation metrics are detailed in the next subsection.

Table 2 Hyperparameters for the classifiers

Classifier	Parameter
DT	max_depth=2, min_samples_leaf=1, min_samples_split=2,random_state=42
RF	max_depth=8, min_samples_split=2, random_state=1, n_estimators=200
XGboost	learning_rate=0.1, max_depth=2, n_estimators=100, random_state=1

4.1 Evaluation Metrics

We compare the performance of our proposed stacking model against existing approaches in terms of accuracy, precision, recall, F1- score, and FAR, especially since these measures are commonly used in intrusion detection applications to assess models [51]. We further consider the receiver operating characteristic (ROC) curve, generated by comparing the model's true positive rate (TPR) against the false positive rate (FPR). The performance metrics are defined below:

TP = true positive: an intrusion sample is correctly identified as intrusion
TN = true negative: a normal sample is correctly identified as normal
FP = false positive: a normal sample is incorrectly identified as intrusion
FN = false negative: an intrusion sample is incorrectly identified as normal
P = total positive = $TP + FN$
N = total negative = $TN + FP$

$$Accuracy = \frac{TP + TN}{P + N} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Detection\ rate\ (Recall) = \frac{TP}{TP + FN} \tag{4}$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

$$FAR = \frac{FP}{FP + TN} \tag{6}$$

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

$$FPR = \frac{FP}{TP + FN} \tag{8}$$

Table 3 The Score values of top 20 features with for UNSW-NB15 dataset.

Feature Number	Feature Name	Score
34	ct_dst_sport_ltm	79668.978511
35	ct_dst_src_ltm	44099.458770
40	ct_srv_dst	43680.205607
33	ct_src_dport_ltm	43389.357240
30	ct_srv_src	41524.830057
32	ct_dst_ltm	33706.259215
9	sttl	24139.759037
39	ct_src_ltm	21538.902873
31	ct_state_ttl	12059.042096
3	state	8915.590932
19	swin	8841.986805
22	dwin	8484.672936
10	dttl	7216.384158
8	rate	5403.235763
20	stcpb	4541.218248
21	dtcpb	4535.311419
27	dmean	3844.496739
23	tcprtt	3749.259173
12	dload	3575.933585
25	ackdat	3541.980700

Table 4 The Score values of top 20 features with for NSL-KDD dataset.

Feature Number	Feature Name	Score
4	src_bytes	5.290205e+07
5	dst_bytes	9.779682e+06
1	duration	6.685718e+05
32	dst_host_srv_count	1.965635e+05
22	count	1.013430e+05
31	dst_host_count	3.729063e+04
23	srv_count	1.699837e+04
2	service	8.070633e+03
26	rerror_rate	1.118459e+03
27	srv_rerror_rate	1.097293e+03
11	logged_in	1.025882e+03
40	dst_host_srv_rerror_rate	1.017707e+03
39	dst_host_rerror_rate	9.998519e+02
33	dst_host_same_srv_rate	6.521661e+02
38	dst_host_srv_serror_rate	4.449203e+02
25	srv_serror_rate	4.411172e+02
24	serror_rate	4.263775e+02
15	num_root	4.224041e+02
37	dst_host_serror_rate	4.213963e+02
28	same_srv_rate	3.792745e+02

4.2 Effect of selecting best features

All features of an IDS dataset do not contribute equally to the design of effective an IDS. In this experiment, we used the selectkbest model [32] to find the k best features where $k = 20$, which gives us the top 20 significant features based on their calculated score. The scores for the features of NSL-KDD and UNSW-NB15 datasets are shown in Tables 3 & 4. Out of the 41 and 42 features for both datasets, we only consider the 20 features based on their score. A feature with higher score has more effect in distinguishing between the normal and anomaly applications. For example, the feature *ct_dst_sport_ltm* score (79668.978) in the UNSW-NB15 dataset, indicating that this feature has a vital impact on our proposed model. Tables 3 and 4 indicate that the feature importance score can simplify our model building by cutting down on the feature number. Also, excluding the redundant features enhances the detection capability of the model.

4.3 Intrusion detection performance

We use a 80%/20% split of data into training and test sets to evaluate the proposed model's performance. We use the 10-fold cross-validation (CV) approach for finding the best parameters to train models, and report the trained models' classification performance on the test set

Tables 4 and 5 demonstrate the detection results of a base classifier and the proposed ensemble model. They detect either a class is a normal or an anomaly for the given dataset. Table 5 shows that for the NSL-KDD dataset the accuracy, precision, recall, f1-score, and FAR performance metrics of stacking ensemble for normal and anomaly class are 0.9990 and 0.9990, 0.9986 and 0.9995, 0.9996 and

0.9984, 0.9991 and 0.9990 and 0.0015 and 0.0003, respectively. On the other hand, Table 6 shows that for the UNSW-NB15 dataset the accuracy, precision, recall, f1-score, and FAR for normal and anomaly class are 0.9513 and 0.9513, 0.9386 and 0.9570, 0.9100 and 0.9700, 0.9223 and 0.9644 and 0.028 and 0.0092 respectively. Table 5 and 6, thus, reflect that the proposed model provides with improved performance in terms of adopted metrics and is highly effective in detecting anomalies.

Table 5 Proposed model's detection results in NSL-KDD dataset.

Method	Class	Accuracy	Precision	Recall	F1-Score	FAR
DT	Normal	0.998	0.998	0.998	0.998	0.001
	Anomaly	0.998	0.998	0.998	0.998	0.001
RF	Normal	0.998	0.997	0.999	0.998	0.002
	Anomaly	0.998	0.999	0.997	0.998	0.0003
XGboost	Normal	0.998	0.998	0.998	0.998	0.001
	Anomaly	0.998	0.998	0.998	0.998	0.001
Stacking	Normal	0.9990	0.9986	0.9996	0.9991	0.001
	Anomaly	0.9990	0.9995	0.9984	0.9990	0.0003

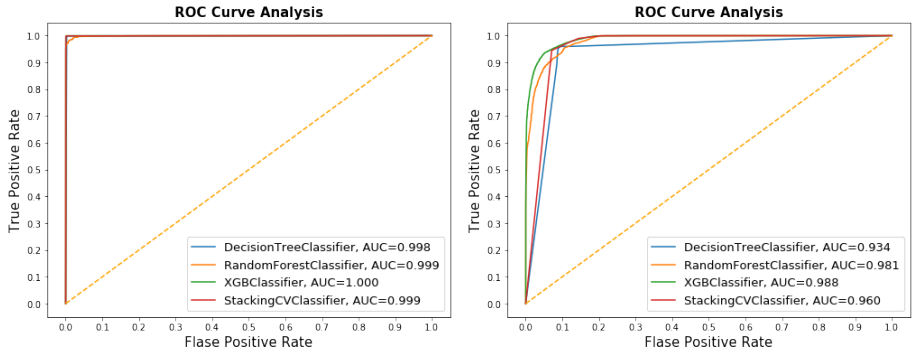
Table 6 Proposed model's detection results in UNSW-NB15 dataset.

Method	Class	Accuracy	Precision	Recall	F1-Score	FAR
DT	Normal	0.917	0.992	0.75	0.854	0.002
	Anomaly	0.917	0.89	0.997	0.942	0.25
RF	Normal	0.934	0.998	0.80	0.886	0.0005
	Anomaly	0.934	0.912	0.999	0.953	0.20
XGboost	Normal	0.934	0.979	0.813	0.880	0.008
	Anomaly	0.934	0.918	0.991	0.953	0.18
Stacking	Normal	0.937	0.98	0.816	0.89	0.007
	Anomaly	0.937	0.919	0.992	0.954	0.18

Figure 2 presents the ROC curve for our proposed model on both datasets and shows that the TPR rate is high which is close to 1, while the FPR is low as desired. Therefore, from the performance outcome presented in Table 5-5 and Fig 2, we may draw the conclusion that an IDS system based on our proposed model can effectively identify the normal or anomaly class based on their recurrence trends in the intrusion dataset.

4.4 Performance comparison with recent methods

Table 7 shows a performance comparison of our stacking model with recent works in intrusion detection (binary classification of anomaly or normal application) using NSL-KDD, and UNSW-BC15 datasets. The table indicates that the proposed



(a) (b)

Fig. 2 The ROC curve: (a) *NSL-KDD* and (b) *UNSW-NB15*

Table 7 Performance comparison with existing models

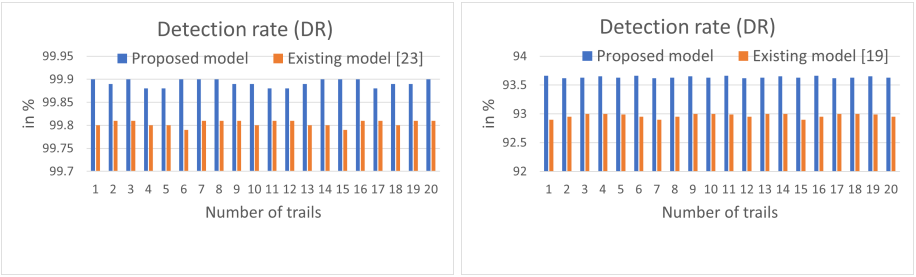
Method	Dataset	Feature selection	Features	Acc(%)	DR(%)	FAR(%)
Fuzzy Ownership NN [15]	NSL-KDD	Greedy Stepwise	11	0.9963	0. 9961	0.00309
Boosting (CART) [18]	NSL-KDD	CFS+PSO	11	0.997285	0. 9977	
BN +TAN [16]	NSL-KDD	OneR		0. 997412	0. 997646	0.002792
Two-stage	NSL-KDD	Hybrid	37	0.96388		
Ensemble Voting(C4.5,RF, ForestPA) [23]	NSL-KDD	CFS-BA	10	0.9981	0.998	0.001
Ensemble (SVM, KNN, MLP) [22]	NSL-KDD	IG+PCA	12	0.9824		0.017
Proposed (Stacking-DT, RF, XGBoost)	NSL-KDD	SelectKbest	20	**0.9990**	**0.9990**	**0.0009**
DT	UNSW-NB15	Sigmoid _PIO [48]	14	0. 9130	-	0.052
Stacking (RF,LR, KNN, SVM) [19]	UNSW-NB15		42	0. 9400	0.9300	0.052
Proposed (Stacking-DT, RF, XGBoost)	UNSW-NB15	SelectKbest	20	**0.94**	**0.94**	**0.06**

model outperforms other similar ensemble classifiers proposed in [19, 22, 23], which use 10-fold CV and consider intrusion detection as a binary classification problem. As evident, our stacking model performs better than these existing methods in terms of accuracy, DR, and FAR for both datasets.

We postulate some reasons of our proposed model's better performance than existing models. First, our model identifies important features and then builds model using only the selected features. Arguably, this step minimizes variance and

Table 8 Model building and per sample test time for the base and Stacking ensembles. The standard deviation of ten trails is indicated by the value inside the bracket.

Algorithm	NSL-KDD Time to build model (s)	Test time (µs)	UNSW-NB15 Time to build model (s)	Test time (µs)
DT	0.678 (± 0.002)	0.396	1.11 (± 0.003)	0.48
RF	0.795 (± 0.0018)	1.2	1.62 (± 0.0004)	1.68
XGBoost	1.53 (± 0.003)	3.21	2.42 (± 0.006)	1.62
Stacking	8.21 (± 0.003)	5.55	11.65(± 0.020)	8



(a) (b)

Fig. 3 Detection rate for proposed and existing model: (a) *NSL-KDD* and (b) *UNSW-NB15*

over-fitting and, thereby, enhances the model's generalization ability, as reflected from improved performance. Moreover, the proposed stacking model takes the advantage of multiple heterogeneous classifiers and overcomes potential limitations of classifiers involving single or homogeneous models. As a result, the proposed model demonstrates better performance in identifying unseen data.

For the given datasets, we also take into account the model building period calculated over ten runs for each of the base classifiers as well as the stacking ensemble technique. The outcomes are presented in Table 8. From Table 8, we can see that the model building time for the base and stacking classifiers are 0.678s, 0.795s, 1.53s, and 8.21s for the NSL-KDD dataset and 1.11s, 1.62s, 2.42s, and 11.65s for the UNSW-NB15 dataset, respectively. Among the classifier, DT takes the least amount of time, while stacking involves largest computational time to build models for both datasets. This additional computation time arises due to the stacking model combining several base classifiers, each of which involve some computational time to build. We also consider the time it takes classifiers to predict intrusion for the test dataset and shown in Table 8. We find that DT and RF are the fastest classifiers in this respect.

We also assess the statistical significance of comparative performance of the models using Wilcoxon rank-sum test [57] with continuity correction. In this regard, we consider 20 trials of our proposed model for each dataset with similar number of trials for the models proposed in the literature for the respective dataset. For NSL-KDD and UNSW-NB15 datasets respectively, among the models shown in Table 7, models reported in [23] and [19] showed better performance than oth-

ers, and therefore, we compared our model with those two models. The detection rate (recall) of our model and existing model for all the 20 trials for both datasets is illustrated in Fig 3. Wilcoxon rank-sum test [57] suggests that our proposed models perform statistically significantly better than the existing models for both respective datasets at 99.99% confidence level with p-values <0.0001. Notably, we have used hyperparameter tuning to ensure each base model utilises the right parameter to reduce overfitting. This consideration further ensures the base models are consistent with the hyperparameter tuning we used for our proposed models.

However, despite some increases in complexity and consequently time requirement for the stacking model, the finding that the stacking model outperform existing approaches for intrusion detection, as shown in the earlier result, is a notable consideration. Such high performing, albeit computationally expensive, classification approaches have a substantial practical implication for emerging services like IoT based smart city applications. In such a system, the cost of missing an intrusion can be very high. Thus, the trade-off for some extra time, which still lies within seconds for the experimented datasets and thus, arguably well scalable compared to existing methods, is justified. Further, research has suggested efficient allocation of resources in fog computing environment [52], and the proposed model can be implemented using of resources in the fog. Overall, the proposed model, hence, has a notable practical value.

Indeed intrusion detection has been an issue of consideration with the task being identified as one of the top applications of data mining in businesses [54]. In business areas like smart city and financial institutions, sustainability of services largely depends on timely detection of unusual activity in the network. Attacks left undetected in such areas can be costly yet manual identification of the attacks can be very difficult [55]. In such systems, which often adopt high computational resources for automatically detecting attacks, the focus is on accurate detection of intrusion. As notable from the results presented, the proposed tree-based stacking approach shows a notable promise in this respect compared to other algorithms.

5 Conclusions

Even though various ML approaches have been introduced to improve the performance of IDSs, currently available techniques for intrusion detection continue to struggle to perform well. In this research work, we introduce a tree-based stacking model with selectkbest feature selection approach to detect intrusion. We used a feature selection approach to reduce the dimensionality of the network data as well as identify the most relevant features. We, then, presented a stacking ensemble technique that consists of DT, RF, and XGBoost. We use the widely used NSL-KDD and UNSW-NB15 datasets to test the effectiveness of the proposed model. Extensive performance evaluation shows that the proposed model achieved 99.9% and 95.26% accuracy and 0.09% and 4.7% FAR for NSL-KDD and UNSW-NB15 datasets and outperforms other recent works in terms of accuracy and false alarm rate. This research, however, focuses on intrusion detection as a binary classification problem, similar to other existing works. In future, we will explore other hybrid methods to increase accuracy of the prediction and categorize the different forms of attacks.

Appendix: Abbreviations

IDS Intrusion detecsion system
FAR False alarm rate
SET Stacking ensemble technique
TAN Tree augmented naive-bayes
ACCS Australian centre for cybersecurity
DT Decision tree
RF Random forest
XGBoost Extreme gradient boosting
CV Cross valaidation
TPR True positive rate
FPR False positive rate
DDoS Distributed denial of service
HIDS Host-based IDS
NIDS Network-based IDS
ROC Receiver Operating Characteristics

References

1. Sarker I.H., Kayes A.S.M., Badsha S., Alqahtani H., Watters P. and Ng, A.: Cybersecurity data science: an overview from machine learning perspective, *Journal of Big Data*, 7(1), pp.1-29 (2020)
2. Av-test institute, germany, https://www.av-test.org/en/statistics/malware/. Accessed 19 Jan 2021.
3. Juniper research. https://www.juniperresearch.com/. White paper: Cybercrime & the Internet of Threats 2019. Accessed on 19 Jan 2021.
4. Md. Mamunur Rashid, Joarder Kamruzzaman, Mohiuddin Ahmed, Nahina Islam, Santoso Wibowo and Steve Gordon: Performance Enhancement of Intrusion detection System Using Bagging Ensemble Technique with Feature Selection, *7thIEEE Asia-Pacific Conference on Computer Science and Data Engineering*, 16-18 December, 2020, Gold Coast, Australia (2020)
5. Tsai C.F., Hsu Y.F., Lin C.Y., Lin W.Y: Intrusion detection by machine learning: A review, *Expert Syst. Appl.* vol 36, pp.11994–12000 (2009)
6. Buczak A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection, *IEEE Commun. Surv. Tutor*, 18, 1153–1176 (2015)
7. Xin Y., Kong L., Liu Z., Chen Y., Li Y., Zhu, H., Gao, M., Hou H, Wang C.: Machine learning and deep learning methods for cybersecurity, *IEEE Access* 6, 35365–35381 (2018)
8. Sommer R., Paxson V.: Outside the closed world: On using machine learning for network intrusion detection, *In Proceedings of the 2010 IEEE Symposium on Security and Privacy*, Berkeley/Oakland, CA, USA, 16–19 May 2010; pp. 305–316 (2010)
9. Garg A., and Maheshwari P.: A hybrid intrusion detection system: A review, *10th International Conference on Intelligent Systems and Control (ISCO)*, pp. 1-5 (2016)
10. Biswas S.K.: Intrusion detection using machine learning: A comparison study, *International Journal of Pure and Applied Mathematics*, vol. 118, no. 19, pp. 101-114 (2018)
11. Saxena A. K., Sinha S. and Shukla P.: General study of intrusion detection system and survey of agent-based intrusion detection system, *2017 International Conference on Computing Communication and Automation (ICCCA)*, pp. 471-421 (2017)
12. Sarker, I.H., Abushark, Y.B., Alsolami, F. and Khan, A.I.: IntruDTree: A Machine Learning Based Cyber Security Intrusion Detection Model. *Symmetry*, 12(5), p.754 (2020)
13. Hansen L. K. and Salamon P.: Neural network ensembles, *IEEE transactions on pattern analysis and machine intelligence* vol. 12 (10), pp. 993–1001 (1990)
14. Pham N. T., Foo E.,Suriadi S., Jeffrey H., and Lahza H.F.M.: Improving performance of intrusion detection system using ensemble methods and feature selection, *in Proceedings of the Australasian Computer Science Week Multiconference*, pp. 1–6 (2018)

15. Panigrah A. and Patra M.R.: Fuzzy rough classification models for network intrusion detection, *Transactions on Machine Learning and Artificial Intelligence*, 4(2), pp.07-07 (2016)
16. Panigrahi A. and Patra M.Anomaly based network intrusion detection using bayes net classifiers. *International Journal of Scientific and Technology Research*, 8(9), pp.481-485 (2019)
17. Tama B.A., Comuzzi M. and Rhee K.H.: TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system, *IEEE Access*, 7, pp.94497-94507 (2019)
18. Tama B.A. and Rhee K.H.: An extensive empirical evaluation of classifier ensembles for intrusion detection task. *Computer Systems Science and Engineering*, 32(2), pp.149-158 (2017)
19. Smitha R., Kundapur P.P. and Hareesha K.S.: A Stacking Ensemble for Network Intrusion Detection Using Heterogeneous Datasets, *Hindawi security and communication networks*, pp. 1-9 (2020)
20. Paulauskas N. and Auskalnis J.: Analysis of data pre-processing influence on intrusion detection using NSL-KDD dataset, *In 2017 open conference of electrical, electronic and information sciences (eStream)* pp. 1-5 (2017)
21. Moustafa N., Turnbull B. and Choo K.K.R.: 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), pp.4815-4830 (2019)
22. Salo F., Nassif A.B., Essex A.: Dimensionality reduction with ig-pca and ensemble classifier for network intrusion detection, *Computer Networks* 148, 164–175 (2019)
23. Zhou Y., Cheng G., Jiang S. and Dai M.: Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Computer Networks*, p.107247 (2020)
24. Yang X.S. A new metaheuristic bat-inspired algorithm. In Nature Inspired Cooperative Strategies for Optimization (NICSO 2010); Springer: Madrid, Spain, 2010; pp. 65–74, ISBN 978-3-642-24094-2.
25. Rashid, M. M., Kamruzzaman, J., Hassan, M. M., Imam, T., & Gordon, S. :Cyberattacks Detection in IoT-Based Smart City Applications Using Machine Learning Techniques. International Journal of Environmental Research and Public Health, 17(24), 9347 (2020)
26. Tavallaee M., Bagheri E.,Lu W., and Ghorbani A.A.: A detailed analysis of the kdd cup 99 data set, *in 2009 IEEE symposium on computational intelligence for security and defense applications*, pp. 1–6 (2009)
27. NSL-KDD dataset. Available on http://www.unb.ca/cic/research/datasets/
28. Moustafa N., Slay J.: The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. J. A Glob. Perspect*vol 25, pp. 18–31 (2016)
29. Moustafa, N. Designing an Online and Reliable Statistical Anomaly Detection Framework for Dealing with Large High-Speed Network Traffic. Ph.D. Thesis, University of New South Wales, Canberra, Australia (2017)
30. Scikit-Learn Developers. Available online: sklearn.preprocessing.LabelEncoder accessed on 10 June 2020 (2020)
31. Kotsiantis S., Kanellopoulos D., Pintelas P.: Data preprocessing for supervised leaning. International Journal of Computer Science 1, 111–117 (2006)
32. Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V.nd Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P.and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E: Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research*, pp. vol 12, 2825–2830 (2011)
33. Hansen L.K., Salamon P: Neural network ensembles, *IEEE Trans. Pattern Anal. Mach. Intell.* vol. 12, 993–1001 (1990)
34. Wolpert D.H: Stacked generalization, *Neural Netw.*, 5, 241–259 (1992)
35. Bansal A., Kaur S.:Extreme gradient boosting based tuning for classification in intrusion detection systems, *International Conference on Advances in Computing and Data Sciences*, Springer, pp. 372–380 (2018)
36. Pham N. T, Foo E., Suriadi S., Jeffrey H., Lahza H. F. H.: Improving performance of intrusion detection system using ensemble methods and feature selection, *in Proceedings of the Australasian Computer Science Week Multiconference*, pp. 1–6 (2018)

37. Fernández-Delgado M, Cernadas E., Barro S., Amorim D.: Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?, *Journal of Machine Learning Research*, vol. 15, no. 90, pp. 3133–3181 (2014) Accessed: Mar. 21, 2021. [Online]. Available: http://jmlr.org/papers/v15/delgado14a.html.
38. Esmaily H., Tayefi M., Doosti H., Ghayour-Mobarhan M., Nezami H., Amirabadizadeh A.: A Comparison between Decision Tree and Random Forest in Determining the Risk Factors Associated with Type 2 Diabetes, *J Res Health Sci*, vol. 18, no. 2, p. 412 (2018) Accessed: Mar. 21, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7204421/.
39. Ali J., Khan R., Ahmad N., Maqsood I: Random Forests and Decision Trees, *International Journal of Computer Science Issues*, vol. 9, no. 5, pp. 272–27 (2012)
40. Berhane T. M. et al.: Decision-Tree, Rule-Based, and Random Forest Classification of High-Resolution Multispectral Imagery for Wetland Mapping and Inventory, *Remote Sens (Basel)*, vol. 10, no. 4, p. 580 (2018) doi: 10.3390/rs10040580.
41. Prajwala T.R.: A Comparative Study on Decision Tree and Random Forest Using R Tool, *IJARCCE*, vol. 4, no. 1, pp. 196–199 (2015) doi: 10.17148/IJARCCE.2015.4142.
42. Chen T., Guestrin C.: XGBoost: A Scalable Tree Boosting System, *in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA (2016) pp. 785–794, doi: 10.1145/2939672.2939785.
43. Dhaliwal S. S., Nahid A., Abbas R.: Effective Intrusion Detection System Using XGBoost, *Information*, vol. 9, no. 7, Art. no. 7 (2018), doi: 10.3390/info9070149.
44. Chen Z., Jiang F., Cheng Y., Gu X., Liu W., Peng J.: XGBoost Classifier for DDoS Attack Detection and Analysis in SDN-Based Cloud, *in 2018 IEEE International Conference on Big Data and Smart Computing (BigComp)* pp. 251–256 (2018), doi: 10.1109/BigComp.2018.00044.
45. Law A. et al.: Secure Collaborative Training and Inference for XGBoost, *in Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice*, New York, NY, USA, pp. 21–26 (2020) doi: 10.1145/3411501.3419420.
46. Kolias C., Kambourakis G., Stavrou A., Gritzalis S.: Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset.*IEEE Commun. Surv. Tutor.*,vol. 18, pp. 184–208 (2015)
47. Sharafaldin I., Lashkari A.H., Ghorbani A.A.: Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization,*ICISSP*, pp. 108–116, Jan 22–24, Funchal, Portugal (2018)
48. Alazzam H., Sharieh A. and Sabri K.E.: A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer, *Expert Systems with Applications*, 148, p.113-249 (2020)
49. Shiravi A., Shiravi H., Tavallaee M. and Ghorbani A.A.: Toward developing a systematic approach to generate benchmark datasets for intrusion detection, *Comput. Secur.,* 31 (3), pp. 357-374 (2012)
50. Song J.,Takakura H., Okabe Y.,Eto M.,Inoue D., Nakao K.: Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation, *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, ACM (2011)*, pp. 29-36 (2011)
51. The-NIMS-Dataset, Available: https://projects.cs.dal.ca/projectx/Download.html.
52. Mahfouz, A., Abuhussein, A., Venugopal, D. and Shiva, S.: Ensemble Classifiers for Network Intrusion Detection Using a Novel Network Attack Dataset. Future Internet, 12(11), p.180 (2020)
53. Taneja M., A. Davy: Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm, *in 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, May 2017, pp. 1222–1228 (2017) doi: 10.23919/INM.2017.7987464.
54. L. W. Chao, K. Shih-Wen, and T. Chih-Fong: 10 data mining techniques in business applications:brief survey, *Kybernetes*, vol. 46, no. 7, pp. 1158–1170, Jan. 2017, doi: 10.1108/K-10-2016-0302.
55. U. Noor, Z. Anwar, T. Amjad, and K.-K. R. Choo: A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise. *Future Generation Computer Systems*, vol. 96, pp. 227–242, Jul. 2019, doi: 10.1016/j.future.2019.02.013.
56. Džeroski, S. and Ženko, B.: Is combining classifiers with stacking better than selecting the best one?. Machine learning, 54(3), pp.255-273, 2004.
57. Wilcoxon Rank-Sum Test, https://www.stat.auckland.ac.nz/ wild/ChanceEnc/Ch10.wilcoxon.pdf
58. Ying, X. An overview of overfitting and its solutions. In Journal of Physics: Conference Series, Vol. 1168, No. 2, p. 022022. IOP Publishing.(2019)