International Journal of Advanced Technology and Engineering Exploration, Vol 11(114)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2023.10101904

Research Article

Evaluating the efficacy of decision tree-based machine learning in classifying intrusive behaviour of network users

Ashalata Panigrahi[1*] and Manas Ranjan Patra[2]
Associate Professor, Department of Computer Science & Engineering , NIST University, Institute Park, Berhampur, 761 008, India[1]
Professor, Department of Computer Science & Engineering, NIST University, Institute Park, Berhampur, 761 008, India[2]

Received: 28-June-2023; Revised: 21-May-2024; Accepted: 23-May-2024

Abstract
Building network intrusion detection models to detect intrusive behaviour of malicious users has been a major challenge to protect network resources. In this study, decision tree (DT) based machine learning (ML) classification techniques, namely, best first tree (BFT), functional tree (FT), J48, naïve Bayes tree (NBT), random forest (RF), random tree (RT), reduced error pruning tree (REPT), simple classification and regression tree (Simple CART) have been employed to build an anomaly-based network intrusion detection model. Further, in order to remove irrelevant features from the intrusion data three different categories of feature selection techniques, namely, (i) entropy based (gain ratio (GR), information gain (IG) and symmetrical uncertainty (SU)), (ii) statistical based (chi-squared, one-r, and relief-f), and (iii) search based exploratory data analysis (EDA), feature subset harmony search (FSHS), linear forward search (LFS), feature vote harmony search (FVHS)) have been applied. The proposed method was evaluated using the widely recognized NSL-KDD dataset. The efficacy of various combinations of eight classifiers and ten feature selection methods (eighty models) was analysed based on seventeen evaluation metrics such as sensitivity, false positive rate (FPR), Matthew's correlation coefficient (MCC), Kappa coefficient (KC), geometric mean (GM), and discriminant power (DP). Experimental results showed that LFS+RF model achieved the highest accuracy of 0.9989, sensitivity 0.9982, F-value 0.9988, specificity 0.9994, false negative rate (FNR) 0.0018, MCC 0.9977, GM 0.9988, and DP 7.6156 on the NSL-KDD dataset. The proposed model demonstrated its superiority over the other existing models such as support vector machine (SVM), JRip, bagging, deep learning, and neural network (NN).

Keywords
Machine learning, Cross-validation, Discriminant power, Geometric mean, Random forest, Naïve bayes tree.

1.Introduction
Machine learning (ML) techniques have pervaded into many application areas wherever human-like behavior is mandated, be it learning, decision-making, prediction, and/or classification. Typically, ML techniques fall into four categories, namely, supervised, unsupervised, semi-supervised, and reinforcement learning [1]. While the supervised ML attempts to link the input attributes with a target attribute [2], the unsupervised ML derives conclusions from input data in the absence of any labelled data [3]. On the other hand, semi-supervised ML combines features of supervised and unsupervised techniques to build classification models [4].

Reinforcement learning relies on a trial-and-error method, wherein a particular action is taken based on some input data, if the action turns out to be acceptable, a reward point is granted, and if the action does not yield the expected result, then the system learns that such an action would not be effective in future [5].

The world-wide acceptance of internet as the most popular communication medium for all kinds of online activities is quite evident in the present times. Along with many of its benefits there has been security challenges due to growing cases of cyber-attacks. Though several approaches like data encryption, user authentication, access control, firewalls, etc. have been tried to prevent attacks but none of the approaches have succeeded in providing

*Author for correspondence

complete protection [6]. It has been a real challenge to find a clear distinction between normal users and intruders. Intrusion detection systems (IDS) are being developed as a possible solution to provide early warning of an intrusion so that appropriate preventive measures can be taken. Normally, two approaches are adopted, to identify unauthorized users, namely, signature-based and heuristic or anomaly based. Signature-based IDS (SIDS) utilize simple pattern-matching to find a known attack (also known as knowledge-based detection. Though SIDS gives good detection rate for known intrusions [7] but fail to detect new or unknown attacks or variants of existing attacks. With the increasing rate of zero-day attacks [8] Symantec, (2017), SIDS prove to be less effective because of unavailability of signature for unknown attacks. The heuristic intrusion detection approach tries to classify the behaviour of network users into one of the categories such as good or benign, suspicious, or unknown [9]. Anomaly-based intrusion detection system (AIDS) uses ML to train the detection system to recognize normal network activities and creates a baseline. Any significant deviation from the baseline is interpreted as an intrusion because an intrusion creates distinguishable patterns within the network traffic [10].

An AIDS continuously monitors and analyzes different activities and raises alarms whenever any abnormal activity is observed. Researchers have tried to build AIDS to detect abnormal network activities using various ML techniques such as support vector machine (SVM), rule-based systems, logistic model tree, k-nearest- neighbour (k-NN) and neural network (NN) but they have not succeeded to a large extent as real attacks go undetected without any alarms and non-attacks are signalled as attacks. As a result, there is increase in false negatives (FN) as well as false positives (FP). The aim of this research is to propose an effective AIDS using tree-based ML techniques which can demonstrate high detection accuracy and low false alarms as far as possible.

The key contributions of this research are:
1.The most appropriate features have been selected from the NSL-KDD dataset using entropy based, statistical based and search-based feature selection methods. This is required to enhance the performance of the model as many of the features negatively impact the performance and increase the model building time.
2. The selected feature subsets are evaluated using eight decision tree-based ML techniques, namely best first tree (BFT), functional tree (FT), J48, naïve bayes tree (NBT), random forest (RF), random tree (RT), reduced error pruning tree (REPT), and simple classification and regression tree (Simple CART)
3. The performance of the classifiers are evaluated with the help of confusion matrix using different evaluation criteria such as accuracy, sensitivity, precision, FPR, F-value, balanced classification rate (BCR), specificity, negative predictive value (NPV), false negative rate (FNR), Matthew's correlation coefficient (MCC), error rate (ER), kappa statistic (KC), geometric mean (GM), Youden's index (YI), Jaccard, efficiency, and discriminant power (DP). To the best of the authors' knowledge, there has been no prior performance evaluation of models based on seventeen different criteria, constructed using all conceivable combinations of ten feature selection methods and eight classifiers.

The rest of the paper is organized as follows: Section 2 briefly describes previously published related work on intrusion detection. Section 3 covers the methodology, details of the proposed model, and experimental setup. Section 4 describes experimental results and are discussed in section 5. Conclusion and future work are presented in section 6.

2.Literature review

In the past many researchers have applied varieties of ML algorithms be it single, hybrid, or ensemble techniques on intrusion data.

In [11], Taher et al. have built an IDS model using artificial neural network (ANN) and SVM. Wrapper feature selection method applied for selection of relevant features. They perform the experiment on NSL-KDD intrusion detection data set and reported that ANN based model is better than SVM model and the model is efficient in terms of detection rate. However, the limitation of the proposed model is that they have used only two techniques, namely, SVM and ANN.

Louati and Ktata [12] presented a deep-learning based multi-agent system for IDS. Evaluating the method KDD CUP 99 is used. At the pre-processing stage converted symbolic features into numeric values and autoencoder technique was applied for selection of important features. For evaluation of parameters two classifiers k-NN and multilayer perceptron were applied on the dataset. According to the findings, k-NN classifier achieves 99.95% accuracy and multilayer perceptron gives an accuracy of 99.73%. KDD CUP 99 dataset is unbalanced. Computation of accuracy is not sufficient to

determine the effectiveness of model. Moreover, only 10% of the total dataset is used fo their experimentation.

Rakshe and Gonjari [13] proposed an IDS model based on SVM and RF techniques. They implemented techniques on NSL-KDD data. They report both models have accuracy more than 95% and performance of RF technique is better than SVM. The limitation of the work being no feature selection method has been applied and the model is built using only two classification techniques and by considering all the 41 features.

Benisha and Ratna [14] suggested long short-term memory convolutional neural network (LSTM-CNN) feature –fusion based cross layer IDS for detecting attacks of internet. Both NSL-KDD and KDD CUP 99 dataset are applied for experiment. Findings show that high accuracy of 99.795 and low false alarm rate. They have considered only two evaluation parameters such as accuracy and FAR. Singh et al. [15] have applied online sequential extreme learning machine (OS-ELM) algorithm to build an IDS. Their experimental results with NSL-KDD 2009 dataset achieved 98.66% detection accuracy and 1.74% ER. However, no feature selection method has been applied.

In [16], Patel et al. (2015) discussed an approach that combines two techniques association rule mining and random particle swarm optimization. For performance analysis of algorithm NSL-KDD dataset used. The model successfully detected normal and attack records. The proposed approach achieved overall accuracy of 95.46%. In [17], Sharma and Gaur proposed hybrid IDS model random ant colony optimization (R-ACO) for detect internet-based attacks. For experimental work NSL-KDD dataset used. At the pre-processing stage normal data filtration and attack data filtration applied on the dataset. For better classification result R-ACO algorithm applied. The results suggest that model is capable of classifying all 4 categories of attack which are present in the dataset and achieves good accuracy value. Though the proposed model reports good result but the authors have tried with only one model. Belgrana et al. [18] have proposed artificial intelligence-based algorithms to find network attacks with an objective to reduce FNR and processing time. They have applied condensed nearest neighbour (CNN) algorithm on NSL-KDD and have compared with k-NN algorithm. They have reported improvement in detection rate and reduced processing time as compared to k-NN. But their work is restricted to only two approaches. AI-Safi et al. [19] presented the model by combining artificial bee colony (ABC) algorithm and optimization-cuckoo search (OCS) for optimizing SVM parameters for classification of network data. Further information gain (IG) method is applied on NSL-KDD to select important features. They reported accuracy of 94.21% and detection rate of 98.25%. The limitation being only one feature selection and one classification technique have been used. Gurung et al. [20] in 2019 proposed a deep learning-based approach to detect network attacks. For experimental study NSL-KDD dataset used. During the preprocessing stage converted symbolic features into numeric values and dataset is normalized using max-min normalization. The overall accuracy was 82.7%, precision was 84.6%, detection rate was 92.8%, and specificity was 80.7%.

Sharon et al. [21] proposed a system combining a sparse autoencoder and stacked contractive autoencoder (S-SCAE) along with a bi-directional LSTM followed by a dense layer , a dropout layer, and a layer with attention mechanism (Bi-DLDA) for detecting intrusions in a cloud environment. Using NSL-KDD dataset the model achieves recall rate of 98%, accuracy of 98%, and precision 99%. The limitation being only three parameters are calculated for performance analysis. In [22] a model has been proposed using meta-heuristic and ML algorithms, namely, RF, CART, SVM, and multi-layer perceptron (MLP). The performance of the model has been evaluated using the metrics F-value, precision, recall and accuracy on NSL-KDD dataset. Though the authors have highlighted the capability of meta-heuristic algorithms in optimising IDS models but no feature selection has been done. In [23], Pandey et al. have proposed generative adversarial network and bayesian optimization in multi-class SVM (GAN-BMSVM) for IDS model to overcome imbalance and overfitting problems in intrusion detection. They have experimented with two imbalanced datasets, namely, NSL-KDD and UNSW-NB15. The min-max normalization method has been applied to normalize the input data with a view to reduce the differences in features. The GAN-BMSVM model could achieve 99.58% and 85.38% accuracy for NSL-KDD and UNSW-NB15 datasets, respectively. However, they have considered accuracy as the only comparison parameter.

In [24] deep-learning-based feed-forward neural network (FFNN) algorithm was proposed and the

model was tested with two datasets NSL-KDD and UNSW-NB15. The model achieved 91.29% accuracy, 91.38% detection rate for the UNSW-NB15 dataset, and 89.03% accuracy, 95.65% detection rate for NSL-KDD dataset. But they have used only FFNN for classification. In [25], Jiang et al. have proposed the use of hybrid intrusion detection utilizing three models namely, feature reduction adjustment parameter particle swarm optimization (FR-APPSO), feature reduction bi-directional long short term memory (FR-Bi-LSTM), and adjustment parameter particle swarm optimization -BiLSTM (APPSO-BiLSTM). Evaluation of the model has been carried on three datasets, viz., NSL-KDD, UNSW-NB15, and CICIDS-2017. The FR-APPSO-BiLSTM model has yielded higher classification accuracy and better detection effect on the three datasets as compared to other two models.

In [26], an improved IDS has been proposed based on feature selection and integrated approach. The NSL-KDD dataset was first balanced using synthetic minority over-sampling edited nearest neighbor (SMOTE-ENN) method and then principal component analysis (PCA) was applied on the dataset for feature extraction. Next, bagging technique was applied for classification. The model achieved accuracy of 84.68% and FPR, FP rate of 1.81%. The limitation being only one classification technique was applied. In [27], Shiravani et al. have proposed a correlation feature selection procedure based on fuzzy numbers and scoring methods. The proposed method was applied on two datasets, viz., NSL-KDD and CICIDS2017. A Performance analysis of the proposed model which is based on four classifiers, viz., RF, ANN, k-NN, and SVM show that SVM gives an accuracy of 96.89% and k-NN gives the lowest FPR of 0.302%. Various researchers have proposed different ML models with an objective of detecting unknown attacks and have evaluated the models using few metrics such as accuracy, detection rate, and false alarm rate. Some have proposed hybrid models by combining two or three methods and have also employed different feature selection methods to select potential features for classification. But, none of the work employed appropriate metrics to evaluate the NSL-KDD unbalanced dataset. Accuracy has been taken as a criterion for performance measurement for almost all the studies though it is not suitable for unbalanced datasets. Therefore, in our study the models have been evaluated using metrics such as MCC, Kappa static, GM, and, DP which are more suitable for unbalanced datasets. In this work, an exhaustive study has been carried out using possible combinations of ten feature selection methods belonging to three different categories, and eight classification techniques. Moreover, none of the works reported so far have used EDA search, HS, LFS, and VHS for feature selection. These search-based methods have the potential to deal with imbalanced datasets, thereby enhancing the detection rate and reduce the FP and FNR.

3.Materials and methods

3.1Decision tree (DT) based ML method

DTs are more concerned with supervised classification and it handles the dataset with large number of records and each record belong to one of the given classes. The techniques are so powerful that they are capable of capturing decision-making knowledge from the given dataset and also handle categorical attributes.

DT is constructed using three phases: construction phase, pruning phase, and processing phase. DT consists of one root node, internal and leaf nodes and alternative branches. DT can be generated from the training set of data. The unknown input data are classified by traversing the tree from root node to leaf node. When a test record enters at the root node, a test is applied to find which child node will be traversed next. Always a unique path exists from the root to each leaf and every path represents a rule.

DT can handle both numerical and categorical attributes. The hierarchical structure of tree-based models handles imbalanced dataset. DT creates a structure resembling a tree by dividing the feature space into regions according to feature values. DT can capture complex relationships between features and the target variable without requiring extensive pre-processing. It can detect anomalies easily and the model building time is also minimum. Further, the number of hyperparameters is less. However, the limitations of DT are: the structure becomes unstable whenever there is a small change in data, and the tree becomes more complex as it gets deeper. DT requires few resources for creation of a tree structure. The complexity of DT depends on the number of nodes in the tree, the height of the tree, and the expected path length. While the time complexity of a DT depends on its depth, the space complexity depends on the number of nodes in the tree.

DT Parameters in Weka

Parameter tuning plays a vital role in DT based learning processes. One can tune these parameters to

improve the overall performance of a model. Followings are the important parameters:
MaxDepth: maximum depth of each tree. By default the value is-1 which means the algorithm will automatically control the depth.
NumFolds–NumFolds value specifies number of folds of data used for pruning the DT. The rest will be used for growing the rules.
MinNum –Minimum number of instances per leaf. If not mentioned, the tree will keep splitting till all leaf nodes have only one class associated with it.

A brief description of some tree-based classifiers that are applied in this work on the NSL-KDD dataset are presented below.

3.1.1Best first tree (BFT)

BFT [28] is built following the divide-and-conquer principle wherein "Best" node is selected at each step for splitting. The technique is capable of splitting both numeric and nominal features. The steps are as follows:

Step 1: The best feature is found to split at each node.
Step 2: Determine the node to be expanded next.
Step 3: Finally decide on the stopping criteria for the tree to grow.

Parameter tuning:
Set minimal number of instances at the terminal node = 2
Gini index (GI) is used for splitting criterion.
Seed = 1 (The random number seed to be used.)
Batch size = 100
NumFoldsPruning = 5 (Number of folds in internal cross-validation)

3.1.2Functional tree (FT)

The FT algorithm [29] follows a top-down recursive partitioning approach to build a DT. Though splitting at each node is univariate; however, both the original attributes in the data and new attributes created by an attribute constructor function are considered. Multiple linear regression is used in the regression setting and linear discriminants or multiple logistic regression is applied in the classification setting. Logistic regression functions are used both at the inner and leaf levels. The algorithm is capable of handling both binary and multi-class target variables.

Parameter tuning:
MinNum = 15 (Set the minimum number of instances at which a node is considered for splitting)
WeightTrimBeta = 0.0 (Set the beta value used for weight trimming in LogitBoost)
Number of iterations for Logitboot = 15
Batch size = 100 (The preferred number of instances to process if batch prediction is performed)
Split = 1 (The seed value for randomization)

3.1.3J48

J48 [30] builds DT using the concept of information entropy.
The tree consists of one root node, internal nodes, branches, and leaf nodes. Leaf nodes indicate class labels. The classifier consists of two phases:
Phase 1 : Growth phase
Phase 2 : Pruning phase

Steps of the approach:
Step1: In case the instances belong to the same class the tree denotes a leaf so the leaf is returned by labelling with the same class.
Step 2: The potential information is calculated for every attribute, given by a test on the attribute. Gain in information is considered that would result from a test on the attribute.
Step 3: The best attribute is found on the basis of present criterion and that attribute is selected for branching [31].
J48 model can handle both continuous and discrete features. It uses the data from a set of examples with incomplete information and gives good accuracy. J48 eliminates branches that do not provide information to the model. Overfitting happens if exceptional attributes are present in the dataset.

Parameter tuning:
MinNum = 2 (Minimum number of instances for leaf = 2)
Batch size = 100 (The preferred number of instances to process if batch prediction is performed)
Confidence factor = 0.25 (The confidence factor used for pruning)
NumFolds = 3 (Determines the amount of data used for reduced error pruning)
Seed = 1 (The seed is used for randomizing the data when reduced-error pruning is used)

3.1.4Naïve bayes tree (NBT)

NBT is a combination of Naïve Bayesian classifier and DT [32]. The classification techniques divides the dataset by applying an entropy-based algorithm and then use Naïve Bayesian technique at the leaf node to analyze the attributes.

Given a collection S, if the target attribute can take m different values, then the entropy of S relative to m-wise classification is defined as (Equation 1):
Entropy (S) = $-\sum_{i=1}^{m} pilog(\text{pi})$ (1)

where pi is the proportion of S belonging to class i.

The information gain, Gain (S, A) of an attribute A relative to S is defined as (Equation 2):

Gain (S, A) = Entropy (S) $-\sum_{v\varepsilon Value(A)}$ [| Sv | / | S |] / Entropy (Sv) (2)

where value (A) is the set of all possible values for attribute A and Sv is the subset of S for which attribute A has value v.

Parameter tuning:

Batch size = 100

3.1.5Random forest (RF)

RF combines multiple DTs where each DT is constructed based on the values of independent random vectors. The results of the RF can be controlled by the majority or weighted voting [33].

RF combines the outcomes of several DTs. When the number of trees is sufficiently large, the upper bound of the generalization error converges according to the following formula [34] (Equation 3):

Generalization Error ≤ (α (1 –s2)) / s2 (3)

Where α is the average correlation among the trees and s is a measure of strength of the tree classifiers. The strength refers to the average performance of the classifiers measured probabilistically as (Equation 4):

Margin, M (X, Y) = P ($\hat{C}\varphi$ = Y) – max P($\hat{C}\varphi$ = Z) $Z\neq Y$ (4)

Where $\hat{C}\varphi$ is the predicted class of X according to a classifier built from some random vector φ. The higher the margin, the more likely the classifier correctly predicts an example X.

The algorithm is very robust as it uses multiple DT to arrive at the result. The RF model does not explicitly perform feature selection and may perform poorly when large number of irrelevant features are present in the dataset.

Parameter tuning:

Batch size = 100

MaxDepth = 0 (The maximum depth of the tree, 0 for unlimited)

NumFeatures = 0 (Sets the number of randomly chosen attributes)

Number of trees in the RF = 100

3.1.6Random tree (RT)

RT algorithm constructs the tree taking M random features at each node and employ bagging concept [35].

It does not perform pruning.

Three main choices to be made when constructing a RT are [36]:

i) The method for splitting the leaves.

ii) The type of predictor to be used at each leaf.

iii) The method for injecting randomness into the trees.

Parameter tuning:

MaxDepth = 0 (The maximum depth of the tree, 0 for unlimited.)

Kvalue = 0 (Sets the number of randomly chosen attributes)

Batch size = 100

MinNum = 1.0 (The minimum total weight of the instances in a leaf)

MinVarianceProp = 0.001 (The minimum proportion of the variance on all the data that needs to be present at a node in order for splitting to be performed in regression trees.)

NumFolds = 0 (Determines the amount of data used for backfitting)

3.1.7Reduced error pruning tree (REPT)

REPT builds a DT using maximum IG as the splitting criterion and prunes it using reduced error pruning [35]. The pruning process in the REPT algorithm deals with the backward over-fitting problem. The algorithm constructs many trees but selects the optimal one. The REPT Algorithm [37] follows:

Step 1: Select the dataset

Step 2: Split the input dataset into two subsets, a growing set and a validation set.

Step 3: Repeat the pruning phase i.e., Step 4 and Step 5 for every node in the tree.

Step 4: Evaluate the impact on the validation set i.e., ER for each node.

Step 5: Remove the node which maximally improves the accuracy of the validation set i.e., the node with highest reduced ER.

The main advantage of this method is its linear computational complexity. As every node is visited only once, this method works relatively fast, especially in comparison to others. This method requires separate pruning dataset, which can be a problem for small datasets.

Parameter tuning:

Batch size = 100

MaxDepth = -1 (The maximum tree depth, -1 for no restrictions)

MinNum = 2.0 (The minimum total weight of the instance in a leaf)

MinVarianceProp = 0.001 (The minimum proportion of the variance on all the data that needs to be present at a node in order for splitting to be performed in regression trees.)

NumFolds = 3 (Determines the amount of data used for pruning.)

3.1.8Simple classification and regression tree (Simple CART)

Simple CART generates a binary DT by splitting the instances at each node [38]. This technique divides the data into homogeneous subsets using binary recursive partitions. CART uses the GI for determining the best split [39].

A child is created for each subcategory, only two children are created. At each step, an exhaustive search is used to determine the best split, where best is defined by (Equation 5):

$$\Phi(u / v)=2\, P_{Ls}\, P_{Rs} \sum_{j=1}^{m} |P(C_j \mid t_L) - P(C_j \mid t_R)| \quad (5)$$

where Ls and Rs are used to indicate the left and right sub trees of the current node in the tree.

PLs and PRs are the probability that a tuple in the training set will be on the left or right side of the tree. The formula is evaluated at the current node, v, and for each possible splitting attribute and criterion, u.

CART is simple, non-linear and easy to understand. It can be implemented with little data preparation as the normalization or scaling of input data is not required. The model becomes complex when the data size is very large, which can lead to overfitting.

Parameter tuning:

Batch size = 100

MinNumObj: The minimal number of observations at the terminal node (Default 2).

NumFoldsPruning = 5 (The number of folds in the internal cross-validation.)

3.2Proposed model

The proposed ML based IDS framework:

Our work focuses on anomaly-based network IDS. Implementation of the proposed anomaly-based NIDS consists of following steps (Present in *Figure 1*):

-For selection of potential features NSL-KDD dataset is used which is described in section 3.3.4.

-Three categories of feature selection methods are implemented on the dataset before training. This step selects the most important subset of features. In this work entropy based (gain ratio (GR) [40], information gain (IG) [40] and symmetrical uncertainty (SU) [41]), statistical based (chi-squared [42], one-R [43] , and relief-F [44]) and search based (EDA [45], feature subset harmony search (FSHS) [45], linear forward selection (LFS) [46] , feature vote harmony search (FVHS) [45]) feature selection techniques have been employed to select the optimum number of relevant features.

- The classification stage evaluates the performance of eight DT based classifiers namely BFT, FT, J48, NBT, RF, RT, REPT, simple CART on the selected subset of features. K-fold (K = 10) cross-validation procedure is used to validate the performance of the model presented in section 3.3.2. The performance of the classification techniques is compared using different metrics described in section 3.3.1.

3.3Experimental setup

All experiments have been conducted with a machine configuration having an AMD processor, 8 GB RAM, Windows 7 professional operating system, and the Weka open-source tool for implementation of ML algorithms.

3.3.1Performance measure

Selection of the suitable criteria is a key issue for performance analysis of classification techniques. Different methods are used for testing the performance of the techniques. A major issue for the imbalanced dataset is selection of suitable criteria for performance analysis. NSL-KDD dataset is an imbalanced dataset because instances of Normal, denial of service (DoS), and Probe categories represent the majority classes and minority classes are remote to local (R2L) and user to root (U2R).

In this study confusion matrix [47] is applied to compare the relative efficacy of the tree-based classification techniques.

Confusion matrix is a P × P table which depicts four aspects of a classifier, namely, T_{TP} (true positive), R_{FP} (false positive) , S_{FN} (false negative) , and Q_{TN} (true negative) as shown in *Table 1*.

where

Q_{TN}: The model detected normal connections correctly and does not raise any alarm.

R_{FP}: The model detected wrongly as an attack and false alarm raised.

S_{FN}: The model failed to detect an attack and did not raise alarm.

T_{TP}: The model detected the attack successfully and raised alarm.

Table 1 Confusion matrix

		Predicted	
		Normal	**Attack**
Actual	Normal	Q_{TN}	R_{FP}
	Attack	S_{FN}	T_{TP}

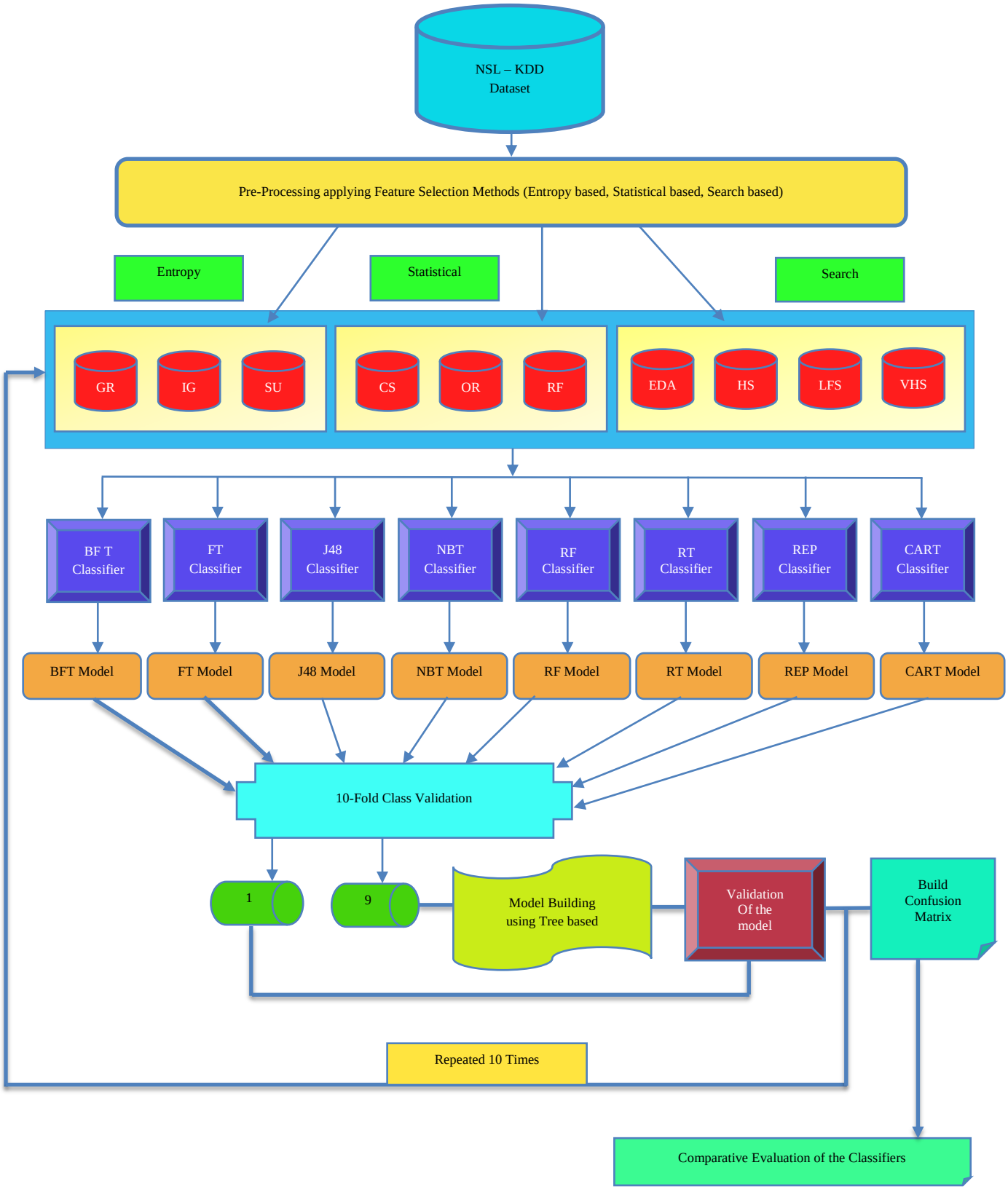

Figure 1 Proposed model

The formulas for the performance measures have been shown by Equations 6 to Equation 22.

Accuracy = $(T_{TP} + Q_{TN}) / [(T_{TP} + R_{FP} + Q_{TN} + S_{FN})]$ (6)

Accuracy Є [0, 1][48]. High accuracy value indicates the model is better.

Sensitivity: Sensitivity can be defined mathematically as [49] (Equation 7):

Sensitivity or TPR or Recall = $(T_{TP})/(T_{TP} + S_{FN})$ (7)

Sensitivity Є [0, 1]. If the sensitivity is high then the ML model is better.

Precision (or PPV): Precision measures how good model is correctly predicted positive instances to the total positive class. Mathematically, precision is calculated as (Equation 8):

Precision = PPV=$(T_{TP}) /(T_{TP} + R_{FP})$ (8)

Precision Є [0, 1]. If value of precision is high then the ML model is better.

False Positive Rate (FPR): FPR is defined as [48] (Equation 9):

FPR = $(R_{FP}) / (Q_{TN} + R_{FP})$ (9)

FPR Є [0, 1]. If the FPR is very low then the ML is better.

F-value calculates using the formula (Equation 10):

F-value = 2×(PPV × TPR/(PPV+TPR) (10)

F-value Є [0, 1]. F-value approximately 1 indicate the performance of the classifier is good.

Balanced Classification Rate (BCR): BCR is calculated using the formula (Equation 11):

BCR=(Sensitivity+Specificity)/ 2 (11)

BCR Є [0, 1]. If the BCR is high then the ML is better.

Specificity (or TNR) : Specificity defined as ratio of number of true negatives to total negative events in the dataset. Mathematically specificity is calculated as (Equation 12) :

Specificity = $(Q_{TN}) / (Q_{TN} + R_{FP})$ (12)

Specificity Є [0, 1]. If the specificity is high then the ML model is better.

Negative Predictive Value (NPV) : NPV is calculated using the formula (Equation 13):

NPV=$(Q_{TN}) / (Q_{TN} + S_{FN})$ (13)

NPV Є [0, 1]. If the NPV is high then the ML model is better.

False Negative Rate (FNR): FNR is calculated using the formula (Equation 14):

FNR = $(S_{FN}) / (S_{FN} + T_{TP})$ (14)

FNR Є [0, 1]. If the FNR is very low then the ML model is better.

Matthews Correlation Coefficient (MCC) : The MCC value varies from – 1 to + 1 i.e. MCC Є (– 1, 1)[49]. The result produce approximately 1 if the prediction results are good. Mathematically MCC is calculated as (Equation 15):

MCC = $[(T_{TP} \times Q_{TN}) – (R_{FP} \times S_{FN})] / SQRT [(T_{TP} + R_{FP}) (T_{TP} + S_{FN}) (Q_{TN} + R_{FP}) (Q_{TN} + S_{FN})]$ (15)

ER is calculated using the formula (Equation 16):

Error Rate = $(R_{FP} + S_{FN})/ (T_{TP}+R_{FP}+Q_{TN} +S_{FN})$ (16)

ER Є [0, 1]. If the ER value is very low then the ML model is better.

Cohen's Kappa Coefficient(KC): A kappa value of 1 represents perfect agreement between model prediction and the actual classes, value of 0 represents no agreement. Kappa is calculated using the formula [50] (Equation 17):

Kappa = [Total Accuracy (Ta) − Random Accuracy (Ra)] / [1 – Random Accuracy (Ra)] (17)

Where

Ta= $(T_{TP} + Q_{TN}) / [(T_{TP} + R_{FP} + Q_{TN} + S_{FN})]$

and

Ra = $[(Q_{TN} + R_{FP}) (Q_{TN} + S_{FN}) + (S_{FN} + T_{TP}) (R_{FP} + T_{TP})] / [(T_{TP} + R_{FP} + Q_{TN} + S_{FN})^2]$

Kappa Є [0 , 1].

Geometric Mean (GM) : Geometric Mean is defined as [51] (Equation 18):

GM = $\sqrt{(Sensitivity \times Specificity)}$ (18)

Geometric Mean Є [0, 1]. The best value is 1 and the worst value is 0.

Youden's Index (YI) is calculated using the formula [49] (Equation 19)

YI = Sensitivity + Specificity –1 (19)

YI Є [– 1, 1]. Higher value of YI indicates good performing of a classifier.

Jaccard: Jaccard metric ignores the negative instances which are correctly classified and is calculated using the formula[47] (Equation 20)

Jaccard = $(T_{TP})/(T_{TP} + R_{FP} + S_{FN})$ (20)

The value of Jaccard ranges from 0 to 1.

Efficiency: Efficiency is calculated using the formula (Equation 21):

$$\text{Efficiency} = \frac{\text{SumofNumberof particular category of attackdetectedcorrectly}}{\text{Totalnumberofattacks}} \quad (21)$$

Discriminant Power (DP): DP is a measure that summarizes specificity and sensitivity , calculated using the formula [49] (Equation 22):

DP = $(\sqrt{3} / \pi)(\log A + \log B)$ (22)

Where A=(TPR)/(1 –TPR)

B =(TNR) /(1 –TNR)

3.3.2Cross-validation

This method involves randomly partitioning total instances into k different groups of equal size. In this study 10-fold cross-validation applied on the dataset. All instances are partition into 10 subparts SP1, SP2, …….SP10 (Present in *Figure 2*). Each time one partition is used for testing and rest nine partitions are used for training.

First partition SP1 is used for validation purpose and the remaining nine partitions SP2 to SP10 are used for training purpose and computing mean squared error (MSE1). Next, partition SP2 is used for validation purpose and the remaining nine partitions SP1 and SP3 to SP10 are used for training purpose and computing MSE2. Repeating this approach 10 times produced 10 squared errors MSE1, MSE2, ……MSE10. The error estimation is average of all the 10 times to get total effectiveness of the model [52] (Equation 23).

$$CV_{10} = (1/10)\left(\sum_{i=1}^{10} MSEi\right) \quad (23)$$

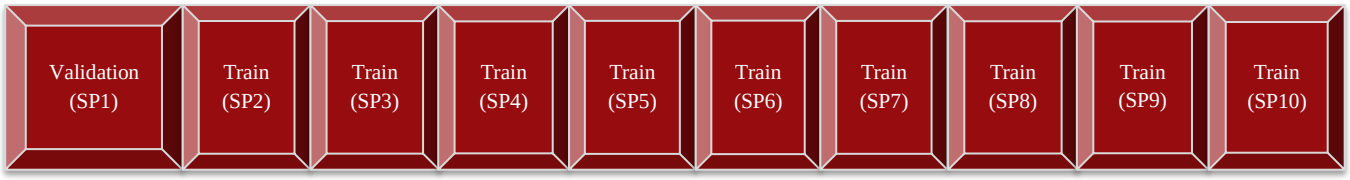

Figure 2 10-Fold cross-validation

3.3.3Feature selection

The aim of feature selection techniques is to improve ML performance. Feature selection methods identify and remove the features from the original dataset which are not required for classification purpose.

For a particular dataset if T input features are present then the feature selection process select the most important P features such that P < T. In this work, entropy based (IG, GR, and SU), statistical based (chi-squared, one-R, and relief-F), search based (EDA, FSHS, LFS, FVHS) feature selection techniques have been applied to select a minimum set of relevant features.

3.3.4Dataset description

In this work, NSL-KDD dataset which was proposed by [53] has been used to build a model. The dataset has 41 features and one class label which identify whether particular instance is harmful or normal. Each connection record contains about 100 bytes.

Each connection instance is a row vector of n features and defined as follows (Equation 24):

$$CR = (F_1, F_2, F_3, \ldots\ldots, F_n, CL) \quad (24)$$

Where F_1 , F_2 , F_3, ……, F_n are features of the dataset and CL denote the class label.

In this data set the number of instances of normal, DoS and Probes are much larger than U2R and R2L. All attacks are classified into four categories: DoS, R2L, probe, U2R. The preparation of data for training and testing of IDS model is critical. NSL-KDD dataset is multiclass and contains normal as well as four types of attack class labels. Normal label has 53.48%, DoS label has 36.45%, Probe has 9.25%, R2L has 0.78%, and U2R has 0.04%. All the attack labels of the dataset have been merged to generate one label known as attack label to reduce the imbalances of the dataset. Thus, the multiclass dataset is reduced to binary class where the two class labels are normal and attack. Normal class label has 53.48% instances and attack class label has 46.52% instances.

4.Results

Comparing classifiers' performance is the most critical task in ML. Quantitative IDS performance measurement results are essential to compare the proposed system. In this study performance of tree-based classifiers are evaluated using confusing matrix which are describe in section 3.3.1.

The metrics, namely, accuracy, sensitivity, precision, FPR, F-value, and BCR of eight tree-based classifiers are compared with three categories of feature selection methods, namely, entropy based, statistical based and search based that are reported in *Table 2*, *3*, and *4* respectively. In the tables, values in boldface represent the highest value as compared to the rest.

Table 2 provides the values of evaluation metrics namely accuracy, sensitivity, precision, FPR, F-value, and BCR for different models. The model GR feature selection + RF model achieves highest accuracy of 0.9943, precision 0.9983, F-value 0.9939, BCR 0.9940, and lowest FPR of 0.0015. GR + Simple Cart model reports highest sensitivity of 0.9897. GR + FT tree model reports lowest accuracy of 0.9931, sensitivity 0.9983, precision 0.9968, F-value 0.9925, BCR of 0.9928, and highest FPR of 0.0027. GR + RF model gives best result for all most

all the metrics whereas GR + FT tree model reports worst result as compared to other models. IG + RF model reports highest accuracy of 0.9988, sensitivity of 0.9982, precision of 0.9991, F-value of 0.9987, BCR of 0.9987 and lowest FPR of 0.0008. IG + FT model gives lowest accuracy of 0.9976, sensitivity of 0.9971, precision of 0.9977, F-value of 0.9974 and BCR of 0.9975, and highest FPR of 0.002.

Table 2 Accuracy, Sensitivity, Precision, FPR, F-value, and BCR for different Tree-based classifiers with Entropy-based feature selection (The values in **boldface** represent the highest value as compared to other values)

Feature selection method	Classifier techniques	Evaluation metric						
		Accuracy	**Sensitivity**	**Precision**	**FPR**	**F-value**	**BCR**	**Time taken to build model (in seconds)**
GR Attribute Evaluator	BF Tree	.9942	.9896	.9977	.0018	.9937	.9939	63.21.
	FT Tree	.9931	.9883	.9968	.0027	.9925	.9928	226.34
	J48	.9940	.9894	.9977	.002	.9935	.9937	8.88
	NB Tree	.9941	.9892	.9981	.0017	.9936	.9937	88.73
	RF	**.9943**	.9895	**.9983**	**.0015**	**.9939**	**.9940**	20.75
	RT	.9938	.9896	.9972	.0024	.9934	.9936	1.95
	REP Tree	.994	.9891	.9978	.0017	.9935	.9937	2.87
	Simple Cart	.9942	**.9897**	.9979	.0019	.9937	.9939	53.81
IG Attribute Evaluator	BF Tree	.9982	.9976	.9984	.0014	.998	.9981	198.21
	FT Tree	.9976	.9971	.9977	.002	.9974	.9975	703.84
	J48	.9982	.9978	.9984	.0014	.9981	.9982	13.24
	NB Tree	.9985	.9978	**.9991**	.0008	.9984	.9985	111.85
	RF	**.9988**	**.9982**	**.9991**	**.0008**	**.9987**	**.9987**	16.02
	RT	.9979	.9975	.998	.0017	.9977	.9979	1.36
	REP Tree	.9977	.9971	.998	.0017	.9975	.9977	4.4
	Simple Cart	.9982	.9977	.9984	.0013	.9981	.9982	191.77
SU	BF Tree	.998	.9972	.9984	.0013	.9978	.9979	210.94
	FT Tree	.9973	.9969	.9974	.0023	.9971	.9973	685.04
	J48	.9981	.9976	.9984	.0014	.9981	.9981	14.38
	NB Tree	.9985	.9979	.999	.0009	.9984	.9985	124.61
	RF	**.9988**	**.9982**	**.9993**	**.0006**	**.9987**	**.9988**	16.27
	RT	.9981	.9979	.9979	.0018	.9979	.9981	1.42
	REP Tree	.9977	.9970	.998	.0018	.9975	.9976	4.2
	Simple Cart	.9981	.9975	.9984	.0014	.9979	.9980	201.85

Table 3 shows that the Chi-squared + RF model achieves the highest values of accuracy (0.9987), sensitivity (0.9982), precision (0.9994), F-value (0.9988), BCR (0.9988), and the lowest FPR (0.0005). In contrast, the Chi-squared + FT model reports the lowest accuracy (0.9973), sensitivity (0.9964), precision (0.9978), F-value (0.9971), BCR (0.9972), and the highest FPR (0.0019).

One-R +RF model gives highest accuracy of 0.9987, precision of 0.9991, F-value of 0.9986, BCR of 0.9987. One-R + J48 model gives highest sensitivity of 0.9982. One-R + NBT gives lowest FPR of 0.0003. One-R + FT model reports lowest accuracy of 0.9973, sensitivity 0.9966, precision 0.9974, F-value of 0.9971, BCR of 0.9972, and highest FPR of 0.0022. Relief-F + RF model gives highest accuracy of 0.9958, sensitivity 0.9946, precision 0.9965, F-value 0.9955, BCR of 0.9958, and lowest FPR of 0.003. Both Relief-F + J48 model and Relief-f + NBT model report second highest accuracy of 0.9952. Relief-f + REPT model gives lowest accuracy of 0.9943, F-value of 0.9939, and BCR of 0.9943. Relief-f + RT model reports lowest sensitivity of 0.9936.

In *Table 4*, four different feature selection methods have been combined each of the eight classifiers. Results show that EDA with RF reports highest accuracy of 0.9980, precision of 0.9987, F-value of 0.9979, and BCR of 0.998. EDA search with NB tree gives highest sensitivity of 0.9972. EDA search with RF gives lowest false alarm rate of 0.0011. EDA with NB tree gives the second highest value of accuracy 0.9979, sensitivity of 0.9971, precision of 0.9983, FPR of 0.0014, F-value of 0.9978, and BCR of 0.9979. EDA with FT gives lowest Accuracy of 0.9966, sensitivity of 0.9959, precision of 0.9968, F-

value of 0.9964, BCR of 0.9966 and the highest FPR of 0.0028. EDA with FT gives the worst value for all the metrics as compared to other models but while combined with RF classifier it gives the best result for almost all the metrics when compared with other models.

FSHS + RF model gives highest accuracy of 0.9988, sensitivity of 0.9981, precision of 0.9993, F-value of 0.9987, BCR value of 0.9987 and lowest FPR of 0.0006. *Table 4* shows that the second-best combination is FSHS with NBT which reports accuracy of 0.9986, sensitivity of 0.9979, precision of 0.9991, FPR value is 0.0008, F-value of 0.9985, and BCR of 0.9986. However, HS with FT model gives lowest accuracy of 0.9970, sensitivity of 0.9961, precision 0.9975, F-value of 0.9968, and BCR of 0.997. HS with FT model reports highest FPR of 0.0021. HS with RF classifier gives the best result for all the metrics as compared to other classifiers whereas HS with FT gives worst result as compared to other models.

LFS + RF model has best performance with accuracy of 0.9989, sensitivity of 0.9982, precision of 0.9994, FPR of 0.0005, F-value of 0.9988, and BCR of 0.9988. LFS + NBT model reports second highest accuracy of 0.9986, sensitivity of 0.9981, precision of 0.9989, FPR is 0.0009, F-value of 0.9985, BCR of 0.9986. LFS + FT model reports lowest accuracy of 0.9972, sensitivity of 0.9964, precision of 0.9977, F-value of 0.997, and BCR of 0.9972. But it has FPR of 0.0009 which is high compared to other models. FVHS + RF model yielded highest accuracy of 0.9985, sensitivity 0.9979, F-value of 0.9984, BCR 0.9985. FVHS + NBT model reports highest precision of 0.999. FVHS + RF model and FVHS + NBT model both reports lowest FPR of 0.0009. It is evident from *Table 4* that FVHS + RF model reports best performance values as compared to other models. VHS + FT model gives lowest accuracy of 0.9967, sensitivity 0.9955, precision 0.9974, F-value 0.9965, BCR 0.9966, and highest FPR of 0.0022. Thus, it is observed that FVHS + FT model gives worst result as compared to other models.

Table 3 Accuracy, sensitivity, precision, FPR, F-value, and BCR for different tree-based classifiers with statistical-based feature selection (The values in boldface represent the highest value as compared to other values)

Feature selection method	**Classifier techniques**	**Evaluation metric**						
		Accuracy	**Sensitivity**	**Precision**	**FPR**	**F-value**	**BCR**	**Time Taken to Build Model (in seconds**
ChiSquared Attribute Evaluator	BF Tree	.9979	.9973	.9982	.0015	.9977	.9979	225.62
	FT Tree	.9973	.9964	.9978	.0019	.9971	.9972	844.93
	J48	.9979	.9973	.9981	.0016	.9977	.9978	16.4
	NB Tree	.9986	.9980	.9989	.0009	.9985	.9985	178.11
	RF	**.9987**	**.9982**	**.9994**	**.0005**	**.9988**	**.9988**	18.8
	RT	.9982	.9978	.9983	.0015	.9980	.9981	1.5
	REP Tree	.9978	.9972	.9981	.0017	.9977	.9978	5.04
	Simple Cart	.9981	.9976	.9982	.0015	.9979	.9980	219.34
One-R Attribute Evaluator	BF Tree	.9982	.9976	.9985	.0013	.9981	.9982	191.77
	FT Tree	.9973	.9966	.9974	.0022	.9971	.9972	590.71
	J48	.9984	.9982	.9984	.0013	.9983	.9984	12.51
	NB Tree	.9985	.9979	.9989	**.0003**	.9984	.9985	101.53
	RF	**.9987**	**.9981**	**.9991**	.0008	**.9986**	**.9987**	12.79
	RT	.9979	.9976	.9979	.0019	.9977	.9979	1.84
	REP Tree	.9977	.9971	.998	.0017	.9975	.9976	3.78
	Simple Cart	.9982	.9977	.9985	.0013	.9981	.9982	195.22
Relief- Attribute Evaluator	BF Tree	.9946	.9938	.9947	.0046	.9942	.9946	248.76
	FT Tree	.9947	.9940	.9946	.0047	.9943	.9947	680.43
	J48	.9952	.9944	.9952	.0041	.9948	.9951	11.45
	NB Tree	.9952	.9945	.9952	.0042	.9949	.9952	95.8
	RF	**.9958**	**.9946**	**.9965**	**.003**	**.9955**	**.9958**	13.07
	RT	.9947	.9936	.995	.0042	.9943	.9946	1.67
	REP Tree	.9943	.9937	.9941	.0051	.9939	.9943	3.53
	Simple Cart	.9951	.9943	.9952	.0041	.9948	.9951	206.97

Table 4 Accuracy, Sensitivity, Precision, FPR, F-value, and BCR for different Tree-based classifiers with Search-based feature selection (The values in **boldface** represent the highest value as compared to other values)

Feature selection method	Classifier technique	Evaluation metric						
		Accuracy	Sensitivity	Precision	FPR	F-value	BCR	Time taken to build model (in seconds)
EDA Search	BF Tree	.9974	.9964	.9980	.0017	.9972	.9973	161.9
	FT Tree	.9966	.9959	.9968	.0028	.9964	.9966	757.88
	J48	.9976	.9969	.9979	.0018	.9974	.9976	7.6
	NB Tree	.9979	**.9972**	.9983	.0014	.9978	.9979	56.52
	RF	**.9980**	.9971	**.9987**	**.0011**	**.9979**	**.998**	10.91
	RT	.9974	.9968	.9976	.0021	.9972	.9973	1.56
	REP Tree	.9974	.9966	.9978	.0019	.9972	.9973	2.79
	Simple Cart	.9976	.9968	.9980	.0017	.9974	.9976	142.99
FSHS	BF Tree	.9982	.9976	.9986	.0012	.9981	.9982	201.63
	FT Tree	.9970	.9961	.9975	.0021	.9968	.997	823.07
	J48	.9983	.9978	.9985	.0013	.9981	.9982	15.73
	NB Tree	.9986	.9979	.9991	.0008	.9985	.9986	155.6
	RF	**.9988**	**.9981**	**.9993**	**.0006**	**.9987**	**.9987**	16.08 Sec.
	RT	.9979	.9975	.9981	.0017	.9978	.9979	1.72 Sec.
	REP Tree	.9978	.997	.9982	.0016	.9976	.9977	4.99 Sec.
	Simple Cart	.9983	.9977	.9986	.0012	.9982	.9983	186.33 Sec.
LFS	BF Tree	.9979	.9971	.9984	.0014	.9977	.9978	189.84 Sec.
	FT Tree	.9972	.9964	.9977	.0020	.997	.9972	879.72 sec
	J48	.9983	.9978	.9985	.0013	.9982	.9983	12.11 Sec.
	NB Tree	.9986	.9981	.9989	.0009	.9985	.9986	98.2 Sec.
	RF	**.9989**	**.9982**	**.9994**	**.0005**	**.9988**	**.9988**	13.1 Sec.
	RT	.9979	.9977	.9977	.002	.9977	.9979	1.98 Sec.
	REP Tree	.9981	.9976	.9983	.0015	.998	.9981	4.04 Sec.
	Simple Cart	.998	.9973	.9983	.0014	.9978	.9979	181.87 Sec.
FVHS	BF Tree	.9978	.9971	.9981	.0017	.9976	.9977	175.64 Sec.
	FT Tree	0.9967	.9955	.9974	.0022	.9965	.9966	734.96 Sec.
	J48	.9979	.9971	.9984	.0014	.9978	.9978	7.05 Sec.
	NB Tree	.9983	.9974	**.999**	**.0009**	.9982	.9982	58.56 Sec.
	RF	**.9985**	**.9979**	.9989	**.0009**	**.9984**	**.9985**	12.26 Sec.
	RT	.9981	.9978	.9961	.0035	.9969	.9979	1.17 Sec.
	REP Tree	.9975	.9965	.9980	.0017	.9973	.9974	3.92 Sec
	Simple Cart	.9981	.9976	.9981	.0017	.9978	.9979	158.87 Sec.

The Specificity, NPV, FNR, MCC, ER, and KC of eight tree-based classifiers are compared with three categories of feature selection methods namely, entropy based, statistical based and search based that are presented in *Tables 5*, *6* and *7* respectively. In the tables, values in **boldface** represent the highest value as compared to the rest.

Table 5 reveals that the GR + RF model achieved the highest specificity (0.9985), NPV (0.991), MCC (0.9886), KC (0.9899), and the lowest ER (0.0057). The GR + Simple CART model recorded the lowest FNR (0.0103). Conversely, the GA + FT model reported the lowest specificity (0.9972), NPV (0.9899), MCC (0.9862), KC (0.9861), and the highest ER (0.0069). Both the IG + RF model and the IG + NBT model showed the highest specificity (0.9992). The IG + RF model also achieved the highest NPV (0.9985), MCC (0.9975), KC (0.9974), the lowest FNR (0.0018), and ER (0.0012). The SU + RF model demonstrated the highest specificity (0.9994), NPV (0.9985), MCC (0.9976), KC (0.9974), and the lowest FNR (0.0018) and ER (0.0012).

Table 6 indicates that the ChiSquared + RF model achieves the highest specificity (0.9994), NPV (0.9984), MCC (0.9977), KC (0.9977), and the lowest FNR (0.0018) and ER (0.0011). In contrast, the ChiSquared + FT model records the lowest specificity (0.9981), NPV (0.9968), and MCC (0.9947), while ChiSquared + REPT reports the

lowest KC (0.9939). When One-R was combined with all eight classifiers, it was observed that One-R + RF achieved the highest specificity (0.9992), NPV (0.9984), MCC (0.9974), KC (0.9972), and the lowest ER (0.0013). The One-R + J48 model achieved the lowest FNR (0.0018). Additionally, when Relief-f feature selection was combined with all classifiers, the Relief-f + RF model reached the highest specificity (0.9969), NPV (0.9953), MCC (0.9917), and KC (0.9922).

Table 5 Specificity, NPV, FNR, MCC, ER, and KC for different Tree-based classifiers with Entropy-based feature selection

Feature selection method	Classifier techniques	Evaluation metric					
		Specificity	**NPV**	**FNR**	**MCC**	**Error rate (ER)**	**Kappa coefficient (KC)**
GR Attribute Evaluator	BF Tree	.9982	.9910	.0104	.9883	.0058	.9896
	FT Tree	.9972	.9899	.0117	.9862	.0069	.9861
	J48	.9980	.9908	.0106	.988	.0061	.9893
	NB Tree	.9983	.9906	.01081	.9881	.0059	.9894
	RF	**.9985**	**.991**	.0105	**.9886**	**.0057**	**.9899**
	RT	.9976	.991	.0104	.9877	.0061	.9891
	REP Tree	.9982	.9906	.0109	.9879	.006	.9892
	Simple Cart	.9981	.9911	**.0103**	.9883	.0058	.9896
IG Attribute Evaluator	BF Tree	.9986	.9979	.0023	.9963	.0018	.9965
	FT Tree	.9981	.9975	.0029	.9951	.0024	.9954
	J48	.9986	.9981	.0022	.9964	.0018	.9962
	NB Tree	.9992	.9981	.0022	.9971	.0014	.996
	RF	**.9992**	**.9985**	**.0018**	**.9975**	**.0012**	**.9974**
	RT	.9983	.9978	.0025	.9958	.0021	.9951
	REP Tree	.9982	.9975	.0029	.9954	.0023	.9937
	Simple Cart	.9986	.998	.0023	.9964	.0018	.9965
SU	BF Tree	.9986	.9976	.0028	.996	.002	.9962
	FT Tree	.9977	.9973	.0031	.9947	.0026	.9949
	J48	.9986	.9979	.0024	.9962	.0019	.996
	NB Tree	.9991	.9981	.0021	.9973	.0014	.9959
	RF	**.9994**	**.9985**	**.0018**	**.9976**	**.0012**	**.9974**
	RT	.9982	.9982	.0021	.9961	.0019	.9954
	REP Tree	.9982	.9974	.0029	.9953	.0023	.9936
	Simple Cart	.9986	.9978	.0025	.9961	.0019	.9964

Table 7 shows that the EDA search + RF model achieves the highest specificity (0.9989), NPV (0.9988), MCC (0.9967), KC (0.9944), the lowest FNR (0.0029), and ER (0.002). The EDA search + FT model reports the lowest specificity (0.9989), NPV (0.9964), KC (0.9922), MCC (0.9932), the highest ER (0.002), and FNR (0.0041). When combined with all classifiers, the HS + RF model achieves the highest specificity (0.9994), NPV (0.9983), MCC (0.9975), KC (0.9976), the lowest FNR (0.0019), and ER (0.0012). The LFS + RF model records the highest specificity (0.9994), NPV (0.9984), MCC (0.9977), KC (0.9973), the lowest FNR (0.0018), and ER (0.0011). Both the VHS + RF and VHS + NBT models report the highest specificity (0.9991), with the VHS + RF model also achieving the highest NPV (0.9982), MCC (0.9971), KC (0.9914), the lowest FNR (0.002), and ER (0.0014). Additionally, the GM, YI, Jaccard, Efficiency, and DP values of eight tree-based classifiers are compared across three categories of FS methods, presented in Tables 8, 9, and 10. Boldfaced values in these tables indicate the highest metrics compared to the others. *Table 8* indicates that the GR + RF model achieves the highest GM (0.994), YI (0.998), Jaccard (0.9878), DP (6.0977), and an efficiency of 98.9323%. The GR + FT model records the lowest GM (0.9928), YI (0.9856), Jaccard (0.9852), DP (5.695), and an efficiency of 98.8044%. The IG + RF model presents the highest GM (0.9987), YI (0.9975), Jaccard (0.9974), DP (7.444), and an efficiency of 99.7692%. The SU + RF model reports the highest GM (0.9988), YI (0.9976), Jaccard (0.9975), efficiency (99.7544%), and DP (7.5433). The results clearly demonstrate that the SU + RF model outperforms the other models in terms of efficiency and performance metrics.

Table 6 Specificity, NPV, FNR, MCC, ER, and KC for different Tree-based classifiers with Statistical-based Feature Selection

Feature selection method		Classifier techniques	Evaluation metric					
			Specificity	NPV	FNR	MCC	Error rate	Kappa coefficient
ChiSquared Evaluator	Attribute	BF Tree	.9984	.9976	.0027	.9958	.0021	.996
		FT Tree	.9981	.9968	.0036	.9947	.0027	.9948
		J48	.9982	.9976	.0027	.9957	.0021	.9961
		NB Tree	.9991	.9983	.002	.9971	.0014	.9973
		RF	**.9994**	**.9984**	**.0018**	**.9977**	**.0011**	**.9977**
		RT	.9985	.9981	.0022	.9971	.0018	.9958
		REP Tree	.9983	.9976	.0028	.9956	.0022	.9939
		Simple Cart	.9984	.9979	.0023	.9961	.0019	.9963
One-R Evaluator	Attribute	BF Tree	.9987	.9979	.0024	.9964	.0018	.9965
		FT Tree	.9978	.9971	.0033	.9945	.0027	.9948
		J48	.9986	.9984	**.0018**	.9968	.0016	.9965
		NB Tree	.9991	.9982	.0021	.997	.0015	.996
		RF	**.9992**	**.9984**	.0019	**.9974**	**.0013**	**.9972**
		RT	.9981	.9979	.0024	.9957	.002119	.9955
		REP Tree	.9982	.9974	.0029	.9954	.0023	.9937
		Simple Cart	.9987	.9981	.0023	.9964	.0018	.9966
Relief-f Evaluator	Attribute	BF Tree	.9954	.9946	.0062	.9892	.0053	.99
		FT Tree	.9953	.9948	.0059	.9893	.0053	.9902
		J48	.9958	.9952	.0056	.9903	.0048	.9909
		NB Tree	.9958	.9952	**.0054**	.9904	.0048	.99
		RF	**9969**	**.9953**	**.0054**	**.9917**	**.0041**	**.9922**
		RT	.9956	.9944	.0064	.9893	.0053	.9896
		REP Tree	.9949	.9945	.0063	.9886	.0057	.9875
		Simple Cart	.9958	.9951	.0057	.9902	.0048	.991

Table 7 Specificity, NPV, FNR, MCC, Error Rate, and KC for different tree-based classifiers with Search-based feature selection

Feature selection method	Classifier Technique	Evaluation metric					
		Specificity	NPV	FNR	MCC	Error rate (ER)	Kappa coefficient (KC)
EDA Search	BF Tree	.9983	.9982	.0036	.9955	.0026	.9938
	FT Tree	.9972	.9964	.0041	.9932	.0034	.9922
	J48	.9982	.9973	.0031	.9952	.0024	**.9944**
	NB Tree	.9986	.9976	.0028	.9958	.0021	.9931
	RF	**.9989**	**.9988**	**.0029**	**.9967**	**.002**	.9943
	RT	.9979	.9972	.0032	.9947	.0026	.9931
	REP Tree	.9981	.9970	.0034	.9947	.0026	.9907
	Simple Cart	.9983	.9973	.0031	.9952	.0022	.9941
HS	BF Tree	.9988	.9979	.0024	.9964	.0018	.9966
	FT Tree	.9978	.9966	.0092	.994	.003	.9944
	J48	.9987	.9981	.0022	.9965	.0017	.9963
	NB Tree	.9992	.9982	.0021	.9972	.0014	.9973
	RF	**.9994**	**.9983**	**.0019**	**.9975**	**.0012**	**.9976**
	RT	.9983	.9978	.0025	.9959	.0020	.9953
	REP Tree	.9984	.9974	.0030	.9954	.0023	.9937
	Simple Cart	.9988	.9980	.0023	.9966	.0017	.9968
LFS	BF Tree	.9986	.9970	.0029	.9956	.0021	.9956
	FT Tree	.9981	.9968	.0036	.9958	.00277	.9944
	J48	.9987	.9981	.0022	.9963	.0017	.9961
	NB Tree	.9991	.9983	.0019	.9972	.0014	.9961
	RF	**.9994**	**.9984**	**.0018**	**.9977**	**.0011**	**.9973**

Feature selection method	Classifier Technique	Evaluation metric					
		Specificity	NPV	FNR	MCC	Error rate (ER)	Kappa coefficient (KC)
	RT	.9980	.9981	.0023	.9957	.0021	.9948
	REP Tree	.9985	.9979	.0024	.9962	.0019	.9945
	Simple Cart	.9985	.9977	.0027	.9959	.0020	.9959
VHS	BF Tree	.9983	.9975	.0029	.9955	.0022	.9953
	FT Tree	. 9978	.9961	.0045	.9934	.0033	.9933
	J48	.9986	.9975	.0029	.9958	.0021	.9952
	NB Tree	**. 9991**	.9978	.0026	.9966	.0017	**.9964**
	RF	**.9991**	**.9982**	**.002**	**.9971**	**.0014**	.9959
	RT	.9981	.9981	.0022	.995	.0020	.995
	REP Tree	.9931	.9970	.0034	.9949	.0025	.9914
	Simple Cart	.9983	.9979	.0024	.9959	.0020	.9957

Table 8 GM, YI, MK, Jaccard, Efficiency, and DP for different tree-based classifiers with entropy-based feature selection

Feature selection method	Classifier technique	Evaluation metric				
		GM	YI	Jaccard	Efficiency in %	DP
GR Attribute Evaluator	BF Tree	.9939	.9878	.9879	98.9391	6.0977
	FT Tree	.9928	.9856	.9852	98.8044	5.695
	J48	.9937	.9874	.9872	98.9186	5.9344
	NB Tree	.9937	.9875	.9873	98.8948	6.0087
	RF	**.994**	**.988**	**.9878**	98.9323	**6.0977**
	RT	.9936	.9871	.9868	98.934	5.8303
	REP Tree	.9937	.9873	.9871	98.8743	5.9814
	Simple Cart	.9939	.9878	.9875	**98.9493**	5.9643
IG Attribute Evaluator	BF Tree	.9981	.9963	.9961	99.722	6.9681
	FT Tree	.9975	.9951	.9948	99.664	6.6395
	J48	.9982	.9964	.9962	99.693	6.9994
	NB Tree	.9985	.997	.9969	99.5992	7.3035
	RF	**.9987**	**.9975**	**.9974**	**99.7692**	**7.4444**
	RT	.9979	.9957	.9955	99.5992	6.7969
	REP Tree	.9977	.9953	.9951	99.4286	6.7122
	Simple Cart	.9982	.9964	.9962	99.7305	6.9903
SU	BF Tree	.9979	.9959	.9957	99.6862	6.8857
	FT Tree	.9973	.9946	.9943	99.6282	6.5374
	J48	.9981	.9962	.996	99.6674	6.9485
	NB Tree	.9985	.997	.9969	99.5923	7.2774
	RF	**.9988**	**.9976**	**.9975**	**99.7544**	**7.5434**
	RT	.9981	.9961	.9959	99.6384	6.8894
	REP Tree	.9976	.9953	.9950	99.4167	6.7011
	Simple Cart	.9984	.9961	.9959	99.71	6.9235

Table 9 compares the models built with all possible combinations of three statistical based feature selection and eight classification techniques. The result clearly shows that Chisquared + RF model gives highest GM of 0.9988, YI of 0.9976, Jaccard of 0.9976, DP of 7.6157, and efficiency of 99.7766%. Relief-F + REP model gives lowest GM of 0.9946, Jaccard of 0.9879, DP of 5.6918, efficiency of 99.0517, and DP of 5.6918. *Table 10* compares the models built with all possible combinations of three statistical based feature selection and eight classification techniques. The result clearly shows that LFS + RF model gives highest GM of 0.9988, YI of 0.9976, and DP of 7.6156.

Table 9 GM, YI, MK, Jaccard, Efficiency, and DP for different Tree-based classifiers with Statistical-based feature selection

Feature selection method	Classifier technique	Evaluation metric				
		GM	YI	Jaccard	Efficiency in %	DP
ChiSquared Attribute Evaluator	BF Tree	.9979	.9957	.9955	99.681	6.8222
	FT Tree	.9972	.9945	.9942	99.5838	6.5484
	J48	.9978	.9957	.9954	99.7049	6.7995
	NB Tree	.9985	.9971	.9969	99.768	7.2767
	RF	**.9988**	**.9976**	**.9976**	**99.7766**	**7.6157**
	RT	.9981	.9963	.9961	99.6589	6.9551
	REP Tree	.9978	.9956	.9953	99.4423	6.9743
	Simple Cart	.998	.9961	.9959	99.7186	6.9005
One-R Attribute Evaluator	BF Tree	.9982	.9963	.9961	99.7152	6.9909
	FT Tree	.9972	.9933	.9958	99.6179	6.5079
	J48	.9984	.9968	.9966	99.7254	7.1147
	NB Tree	.9985	.997	.9968	99.6128	7.2431
	RF	**.9987**	**.9973**	**.9972**	99.4759	**7.3872**
	RT	.9979	.9957	.9954	99.6555	6.7831
	REP Tree	.9976	.9953	.9951	99.4286	6.709
	Simple Cart	.9982	.9964	.9962	**99.7288**	7.0102
Relief-F Attribute Evaluator	BF Tree	.9946	.9829	.9886	99.299	5.7603
	FT Tree	.9947	.9893	.9887	99.3348	5.7694
	J48	.9951	.9903	.9897	99.3621	5.8776
	NB Tree	.9952	.9904	.9898	99.2512	5.8859
	RF	**.9958**	**.9915**	**.9911**	**99.3911**	**6.0646**
	RT	.9946	.9892	.9886	99.2273	5.771
	REP Tree	.9943	.9834	.9879	99.0517	5.6918
	Simple Cart	.9951	.9902	.9896	99.3706	5.8695

Figure 3 presents a comparison of DP values for all the classifiers. Analysis indicates that RF classifier suggests highest DP value irrespective of the search-based feature selection methods used. LF search with RF classifier presents highest DP value of 7.6156. EDA search + RF achieves, DP value of 6.9547, FSHS + RF achieves DP value of 7. 5157, and FVHS + RF achieves DP value of 7.2627. Hence high DP value indicates that the proposed model is stronger and is capable of detecting unknown attacks.

Table 10 GM, YI, Jaccard, Efficiency, and DP for different tree-based classifiers with search-based feature selection

Feature selection method	Classifier techniques	Evaluation metric				
		GM	YI	Jaccard	Efficiency in %	DP
EDA Search	BF Tree	.9973	.9947	.9945	99.4286	6.6127
	FT Tree	.9966	.9931	.9927	99.3553	6.2893
	J48	.9975	.9951	.9949	99.5122	6.6685
	NB Tree	.9972	.9958	.9955	99.3211	6.8793
	RF	**.998**	**.9959**	**.9958**	**99.5241**	**6.9547**
	RT	.9973	.9946	.9943	99.3877	6.5516
	REP Tree	.9973	.9946	.9887	99.0824	6.5653
	Simple Cart	.9976	.9951	.9949	99.473	6.6829
FSHS	BF Tree	.9982	.9963	.9961	99.7254	7.005
	FT Tree	.997	.9939	.9936	99.5616	6.4346
	J48	.9982	.9965	.9963	99.693	7.0173
	NB Tree	**.9986**	.9971	.9970	99.7629	7.3404
	RF	.9982	**.9975**	**.9974**	**99.7766**	**7.5157**
	RT	.9979	.9958	.9956	99.6077	6.8224
	REP Tree	.9972	.9954	.9951	99.4081	6.7388
	Simple Cart	.9983	.9965	.9964	99.7407	7.0724
LF Search	BF Tree	.9978	.9957	.9955	99.6265	6.8382

	FT Tree	.9972	.9943	.9941	99.5446	6.5122
	J48	.9983	.9965	.9963	99.635	7.0449
	NB Tree	.9986	.9971	.9970	99.6265	7.2897
	RF	**.9988**	**.9976**	**.9976**	**99.7288**	**7.6156**
	RT	.9979	.9957	.9954	99.5889	6.7809
	REP Tree	.9981	.9961	.9959	99.4917	6.918
	Simple Cart	.9979	.9959	.9956	99.6657	6.862
FVHS	BF Tree	.9977	.9954	.9952	99.6196	6.7425
	FT Tree	.9966	.9933	.993	99.4269	6.3452
	J48	.9978	.9957	.9955	99.5736	6.8382
	NB Tree	.9982	.9964	**.9964**	99.6606	7.115
	RF	**.9985**	**.9973**	.9938	99.6008	**7.2627**
	RT	.9979	.9959	.9938	**99.6043**	6.8274
	REP Tree	.9974	.9949	.9946	99.1404	6.6391
	Simple Cart	.9979	.9959	.9956	99.6623	6.8339

Comparative Analysis

The performance of the proposed model with other ML techniques has been compared and the findings are reported in *Table 11* based on the experiments performed on the NSL-KDD dataset. The results reported that the proposed model yields a high accuracy which is 99.89%, high sensitivity rate of 99.82%, and much lower FPR of 0.03% in comparison to other approaches.

Table 11 Comparison of our proposed model with other existing ML methods (N/R: Not Reported)

Work	**Model**	**Dataset**	**Accuracy in %**	**FAR or FPR (%)**	**Sensitivity or DR (%)**
[54]	C5.0 + One class SVM	NSL-KDD	98.20	1.40	95
[55]	SU +JRip	NSL-KDD	99.83	0.14	99.80
[56]	Intelligent water drops feature selection + SVM	KDD Cup 99	N/R	1.40	99.40
[57]	FSSL-EL-CART	KDD CUP	84.54	20.35	N/R
[58]	TVCPSO-SVM	NSL-KDD	98.30	0.87	97.05
[59]	LMDRT-SVM	KDD CUP	99.93	0.10	99.94
[60]	GR + J48 + Bagging	NSL-KDD	84.25	2.79	N/R
[61]	Ensemble (REP Tree, IBk, RT, RF, and J48graft)	NSL-KDD	99.72%	0. 003	99.70
[47]	Deep-learning based feedforward neural network	NSL-KDD	89.03	17.59%	95.65
[62]	Spider Monkey Optimization (SMO) and Hierarchical Particle Swarm Optimization (HPSO) + RF	NSL-KDD	99.17	N / R	98.33
[63]	Neural Network	NSL-KDD	97.5	N / R	96.7
[64]	Kernel-based principal component analysis (KPCA) + Deep Neural Network	KDD CUP 99	96.0	N / R	N / R
[65]	MLP	NSL-KDD	97.8	N / R	98
[66]	Logistic Model Tree	NSL-KDD	99.40	0.32	N/R
Proposed model	**LFS + RF**	**NSL-KDD**	**99.89**	**0.0005**	**99.82**

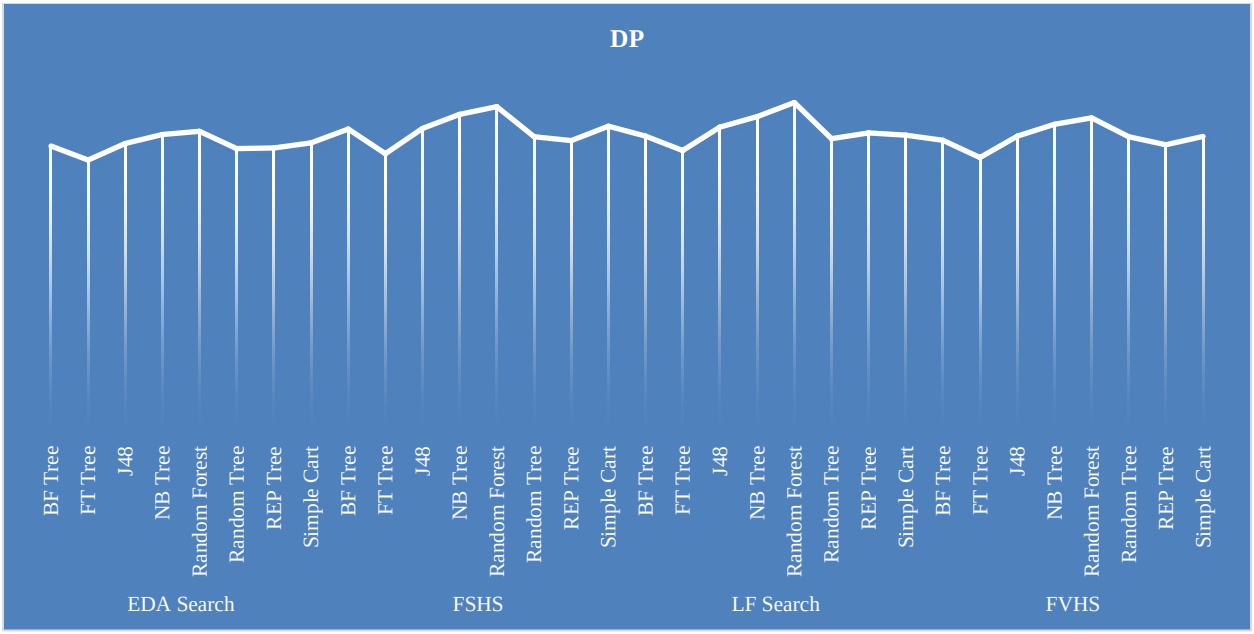

Figure 3 Performance comparison of DP

5.Discussion

None of the earlier works referred in *Table 11*, employed appropriate metrics to evaluate their performance. Invariably they have used accuracy as the only evaluation parameter but it is not suitable in dealing with unbalanced data. To address this limitation, seventeen metrices have been used for evaluating the performance of the proposed model. The results show that high accuracy, high sensitivity and very low FPR for all the classifiers with respect to the three categories of FS methods. All the classifiers have approximately the same accuracy value (i.e., more than 0.99) which indicates effectiveness of the proposed model. Further, the sensitivity value is close to 1 for all the classifiers, and particularly the RF classifier gives the highest sensitivity value for almost all feature selection methods, the maximum being 0.9982. This shows the effectiveness of RF. Thus, the proposed method is able to reduce the FPR considerably. RF is an efficient method due to its speed and ability to handle high dimensional data and its robustness against overfitting. As per the results, it is inferred that NB Tree with One-R attributer evaluator gives lowest FPR rate of 0.0003. The higher precision value is an indication of a good classifier. It is observed that the F-value is more than 0.99 for all the classifiers. High value of F-value indicates high classification performance. High value of BCR (close to 1) for all the classifiers indicates classification performance is good. It is inferred that the RF classifier gives good results for almost all the metrics and feature selection methods. It is further inferred that the performance indicators are close to each other in terms of NPV, FNR, MCC, ER, and KC. A fairly acceptable scalar value of specificity (Approximately 1) and very low FNR indicate that the proposed model correctly classified normal instances for all the classifiers. Similarly, a good NPV score indicates that the proposed model perfectly predicted negative instances. High score of MCC (approximately 1) for all classifiers irrespective of feature selection methods indicates that the correlation coefficient between the exact and predicted values is quite acceptable. The low ER value achieved through the proposed model proves the effectiveness of the model in classifying attacks in an IDS. The KC value which compares the accuracy of a system to that of a random system serves as a measure of efficiency of a model. Here, the high value of KC which ranges between 0.981 to 1.0 indicates a perfect classification model. It is observed that RF classifier gives good result for almost all the metrics which proves its superiority. The geometric mean measures the balance between majority and minority classification performance. Here, the high value of geometric mean (approximately 1) for all the models indicates the effectiveness of the classification model. Both FPR and FNR values being very low indicates that YI value is high (between 0.98 to 0.9978) which indicates that the proposed model is able to accurately distinguish between normal and attack

instances. Higher the Jaccard score higher the accuracy of the classifier. From the *Table 8* to *10*, it is inferred that Jaccard score of all the classifiers are high (between 0.98 to 0.998). The DP metric (which depends on TPR and TNR) evaluates the ability of a classification model to differentiate between positive and negative instances. It is observed that the DP value lies between 5.5 and 8 for all classifiers which shows the efficiency of the model. The highest DP value is obtained for RF classifier with Chi-squared attribute evaluator with 7.6157. RF classifier gives highest DP value for all classifiers irrespective of the feature selection used. *Figure 3* present Discriminant Power of classifiers with search-based FS methods. In the Table values in boldface represent the highest value as compared to other values. Finally, the results suggest that RF classifier outperforms other classifiers, and the proposed model can distinguish between positive and negative instances correctly.

The limitation of our work is that the proposed model is tested only on the NSL-KDD dataset. In future, other available datasets will be used to test the efficacy of the model. A complete list of abbreviations is listed in *Appendix I.*

6.Conclusion and future work

Though many ML models have been proposed to increase the efficacy of IDS but the existing intrusion detection techniques are yet to achieve the expected performance level. In this work, three categories of feature selection methods were employed to select the best features. Following this, eight decision tree-based ML classifiers were applied to build the classification model. The proposed model was evaluated using 10-fold cross-validation. Confusion matrix was used to analyze and compare the performance of the classifiers. Experimental results showed that RF classifier emerged as the best model with sensitivity or detection rate of 99.82%, FPR as 0.03%, KC as 0.9977, GM value as 0.9988 and DP as 7.6157 with respect to the NSL-KDD dataset. In future studies, the aim is to investigate the performance of the proposed model on individual classes contained within the datasets and to increase the model performance of minority classes. Further, it is planned to experiment with different hybrid models in order to enhance the accuracy of intrusion detection.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

Data availability

The dataset used in this study is publicly available at http://nsl.cs.unb.ca/nsl-kdd/.

Author's contributions statement

Ashalata Panigrahi: Contributed to the design of the models and execution of experiments. She also critically analysed the results and presented the performance aspects. **Manas Ranjan Patra:** Conceptualized the study and designed the overall approach for the research. He also contributed to the preparation and revisions of the manuscript to its final form.

References

[1] Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction. New York: springer; 2009.

[2] Maimon O, Rokach L. Data mining and knowledge discovery handbook. New York: Springer; 2005.

[3] Zhang D, Nunamaker JF. Powering e-learning in the new millennium: an overview of e-learning and enabling technology. Information Systems Frontiers. 2003; 5:207-18.

[4] Levatić J, Kocev D, Ceci M, Džeroski S. Semi-supervised trees for multi-target regression. Information Sciences. 2018; 450:109-27.

[5] Géron A. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc."; 2022.

[6] Kim DS, Nguyen HN, Ohn SY, Park JS. Fusions of GA and SVM for anomaly detection in intrusion detection system. In advances in neural networks–ISNN: second international symposium on neural networks, Congqing, China, Proceedings, Part III 2005 (pp. 415-20). Springer Berlin Heidelberg.

[7] Kreibich C, Crowcroft J. Honeycomb: creating intrusion detection signatures using honeypots. ACM SIGCOMM Computer Communication Review. 2004; 34(1):51-6.

[8] https://docs.broadcom.com/doc/istr-22-2017-en. Accessed 29 November 2020.

[9] Pfleeger CP. Security in computing. Prentice-Hall, Inc.; 1988.

[10] Fadlullah ZM, Tang F, Mao B, Kato N, Akashi O, Inoue T, et al. State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems. IEEE Communications Surveys & Tutorials. 2017; 19(4):2432-55.

[11] Taher KA, Jisan BM, Rahman MM. Network intrusion detection using supervised machine learning technique with feature selection. In international conference on robotics, electrical and signal processing techniques 2019 (pp. 643-6). IEEE.

[12] Louati F, Ktata FB. A deep learning-based multi-agent system for intrusion detection. SN Applied Sciences. 2020; 2(4):1-13.

[13] Rakshe T, Gonjari V. Anomaly based network intrusion detection using machine learning techniques.

International Journal of Engineering Research and Technology. 2017; 6(5):216-20.
[14] Benisha RB, Ratna SR. Detection of data integrity attacks by constructing an effective intrusion detection system. Journal of Ambient Intelligence and Humanized Computing. 2020; 11(11):5233-44.
[15] Singh R, Kumar H, Singla RK. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. Expert Systems with Applications. 2015; 42(22):8609-24.
[16] Patel R, Bakhshi D, Arjariya T. Random particle swarm optimization (RPSO) based intrusion detection system. International Journal of Advanced Technology and Engineering Exploration. 2015; 2(5):60-6.
[17] Sharma N, Gaur B. An approach for efficient intrusion detection based on R-ACO. International Journal of Advanced Technology and Engineering Exploration. 2016; 3(20):98-104.
[18] Belgrana FZ, Benamrane N, Hamaida MA, Chaabani AM, Taleb-ahmed A. Network intrusion detection system using neural network and condensed nearest neighbors with selection of NSL-KDD influencing features. In IEEE international conference on internet of things and intelligence system 2021 (pp. 23-9). IEEE.
[19] Al-safi AH, Hani ZI, Zahra MM. Using a hybrid algorithm and feature selection for network anomaly intrusion detection. Journal of Mechanical Engineering Research and Developments. 2021; 44(4):253-62.
[20] Gurung S, Ghose MK, Subedi A. Deep learning approach on network intrusion detection system using NSL-KDD dataset. International Journal of Computer Network and Information Security. 2019; 11(3):8-14.
[21] Sharon A, Mohanraj P, Abraham TE, Sundan B, Thangasamy A. An intelligent intrusion detection system using hybrid deep learning approaches in cloud environment. In international conference on computer, communication, and signal processing 2022 (pp. 281-98). Cham: Springer International Publishing.
[22] Dinesh K, Kalaivani D. Enhancing performance of intrusion detection system in the NSL-KDD dataset using meta-heuristic and machine learning algorithms-design thinking approach. In international conference on sustainable computing and smart systems 2023 (pp. 1471-9). IEEE.
[23] Pandey AK, Singh P, Jain D, Sharma AK, Jain A, Gupta A. Generative adversarial network and bayesian optimization in multi-class support vector machine for intrusion detection system. International Journal of Intelligent Engineering and Systems. 2023; 16:110-9.
[24] Ghani H, Virdee B, Salekzamankhani S. A deep learning approach for network intrusion detection using a small features vector. Journal of Cybersecurity and Privacy. 2023; 3(3):451-63.
[25] Jiang H, Ji S, He G, Li X. Network traffic anomaly detection model based on feature reduction and bidirectional LSTM neural network optimization. Scientific Programming. 2023; 2023:1-18.
[26] Liu Q, Tong Z, Wang S, Yang Z. Research on intrusion detection method based on feature selection and integrated learning. In journal of physics: conference series 2022 (pp. 1-6). IOP Publishing.
[27] Shiravani A, Sadreddini MH, Nahook HN. Network intrusion detection using data dimensions reduction techniques. Journal of Big Data. 2023; 10(1):1-25.
[28] Shi H. Best-first decision tree learning. Doctoral Dissertation, The University of Waikato. 2017.
[29] Gama J. Functional trees. Machine learning. 2004; 55:219-50.
[30] Quinlan JR. C4. 5: programs for machine learning. Elsevier; 2014.
[31] Salzberg SL. Book Review: C4. 5: programs for machine learning. Machine Learning. 1994; 16(3):235-40.
[32] Kohavi R. Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In KDD 1996 (pp. 202-7).
[33] Buczak AL, Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials. 2015; 18(2):1153-76.
[34] Tan PN, Steinbach M, Kumar V. Introduction to data mining. Pearson Education India; 2016.
[35] Witten IH, Frank E. Data mining: practical machine learning tools and techniques with java implementations. ACM Sigmod Record. 2002; 31(1):76-7.
[36] Denil M, Matheson D, De FN. Narrowing the gap: random forests in theory and in practice. In international conference on machine learning 2014 (pp. 665-73). PMLR.
[37] Dhakar M, Tiwari A. A new model for intrusion detection based on reduced error pruning technique. International Journal of Computer Network and Information Security. 2013; 5(11):51-7.
[38] Loh WY. Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2011; 1(1):14-23.
[39] Dunham MH. Data mining: introductory and advanced topics. Pearson Education India; 2006.
[40] Han J, Pei J, Tong H. Data mining: concepts and techniques. Morgan Kaufmann; 2022.
[41] Hall MA, Smith LA. Practical feature subset selection for machine learning. Proceedings of the 21st Australasian computer science conference ACSC'98, Perth 1998 (pp. 181-91). Springer
[42] Liu H, Setiono R. Chi2: feature selection and discretization of numeric attributes. In proceedings of 7th international conference on tools with artificial intelligence 1995 (pp. 388-91). IEEE.
[43] Holte RC. Very simple classification rules perform well on most commonly used datasets. Machine Learning. 1993; 11:63-90.
[44] Robnik-sikonja M, Kononenko I. Theoretical and empirical analysis of reliefF and RreliefF. Machine Learning. 2003; 53:23-69.
[45] Liu H, Setiono R. A probabilistic approach to feature selection-a filter solution. In ICML 1996 (pp. 319-27).

[46] Gutlein M, Frank E, Hall M, Karwath A. Large-scale attribute selection using wrappers. In symposium on computational intelligence and data mining 2009 (pp. 332-9). IEEE.
[47] Tharwat A. Classification assessment methods. Applied Computing and Informatics. 2020; 17(1):168-92.
[48] Powers DM. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies. 2011; 2(1): 37-63.
[49] Sokolova M, Japkowicz N, Szpakowicz S. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In Australasian joint conference on artificial intelligence 2006 (pp. 1015-21). Springer Berlin Heidelberg.
[50] https://thedatascientist.com/performance-measures-cohens-kappa-statistic/.Accessed 29 November 2020.
[51] Kubat M, Matwin S. Addressing the curse of imbalanced data sets: one-sided sampling. In proceedings of the fourteenth international conference on machine learning 1997 (pp. 179-86).
[52] James G, Witten D, Hastie T, Tibshirani R. An introduction to statistical learning. New York: Springer; 2013.
[53] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In the symposium on computational intelligence for security and defense applications 2009 (pp. 1-6). IEEE.
[54] Rani MS, Xavier SB. A hybrid intrusion detection system based on C5. 0 decision tree and one-class SVM. International journal of Current Engineering and Technology. 2015; 5(3):2001-7.
[55] Panigrahi A, Patra MR. Performance evaluation of rule learning classifiers in anomaly based intrusion detection. In proceedings of the international conference on computational intelligence in data mining 2016 (pp. 97-108). Springer India.
[56] Acharya N, Singh S. An IWD-based feature selection method for intrusion detection system. Soft Computing. 2018; 22:4407-16.
[57] Gao Y, Liu Y, Jin Y, Chen J, Wu H. A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system. IEEE Access. 2018; 6:50927-38.
[58] Bamakan SM, Wang H, Yingjie T, Shi Y. An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. Neurocomputing. 2016; 199:90-102.
[59] Wang H, Gu J, Wang S. An effective intrusion detection framework based on SVM with feature augmentation. Knowledge-Based Systems. 2017; 136:130-9.
[60] Pham NT, Foo E, Suriadi S, Jeffrey H, Lahza HF. Improving performance of intrusion detection system using ensemble methods and feature selection. In proceedings of the Australasian computer science week multiconference 2018 (pp. 1-6). ACM.
[61] Dua M. Attribute selection and ensemble classifier based novel approach to intrusion detection system. Procedia Computer Science. 2020; 167:2191-9.
[62] Ethala S, Kumarappan A. A hybrid spider monkey and hierarchical particle swarm optimization approach for intrusion detection on internet of things. Sensors. 2022; 22(21):1-18.
[63] Zakariah M, Alqahtani SA, Alawwad AM, Alotaibi AA. Intrusion detection system with customized machine learning techniques for NSL-KDD dataset. Computers, Materials & Continua. 2023; 77(3):4025-54.
[64] Ravi V, Chaganti R, Alazab M. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. Computers and Electrical Engineering. 2022; 102:108156.
[65] Türk F. Analysis of intrusion detection systems in UNSW-NB15 and NSL-KDD datasets with machine learning algorithms. Bitlis Eren University Journal of Science. 2023; 12(2):465-77.
[66] Dada EG, Bassi JS, Adekunle OO. An investigation into the effectiveness of machine learning techniques for intrusion detection. Arid Zone Journal of Engineering, Technology and Environment. 2017; 13(6):764-8.

Dr. Ashalata Panigrahi completed her Ph.D. in Computer Science from Berhampur University, India and presently working as Associate Professor in the Department of Computer Science & Engineering, NIST University, Berhampur, India. Her research interests are Information Security, Intrusion Detection, Artificial Intelligence, Machine Learning. She has published more than 20 papers in referred journals and reputed conferences.
Email: ashalata.panigrahi@nist.edu

Dr. Manas Ranjan Patra holds a Ph.D. in Computer Science from the Central University of Hyderabad, India. Currently, he is a Professor in the Department of Computer Science & Engineering, NIST University, Berhampur, India. He was a visiting United Nations Fellow at the International Institute of Software Technology, United Nations University, Macau. His research interests include Artificial Intelligence, Machine Learning, Cloud Computing, and applications of Data Mining. He has successfully guided 18 Ph.D. students and more than 100 Master theses. To his credit, he has 200+ research publications in reputed journals and conferences. He has extensively travelled to many countries in USA, Europe, Australia, Africa, and South-East Asia on various academic assignments. He is a member of Editorial boards and Programme committees of many international journals and conferences.
Email: mrpatra@nist.edu

Appendix I

S. No.	Abbreviation	Description
1	AIDS	Anomaly-based Intrusion Detection System
2	ABC	Artificial Bee Colony
3	ANN	Artificial Neural Network
4	BCA	Balanced Classification Accuracy
5	BCR	Balanced Classification Rate
6	BF	Best First
7	Bi-LSTM	Bidirectional Long short Term Memory
8	CART	Classification and Regression Trees
9	CNN	Condensed Nearest Neighbour
10	DT	Decision Tree
11	DoS	Denial of Service
12	DP	Discriminant Power
13	ER	Error Rate
14	FN	False Negative
15	FNR	False Negative Rate
16	FP	False Positive
17	FPR	False Positive Rate
18	FR-APPSO-BiLSTM	Feature Reduction Adjustment Parameter Particle Swarm Optimization Bi-directional Long Short Term Memory (FR-Bi-LSTM),
19	FVHS	Feature Vote Harmony Search
20	FFNN	Feed Forward Neural Network
21	FT	Functional Tree
22	GR	Gain Ratio
23	GAN-BMSVM	Generative Adversarial Network and Bayesian Optimization in Multi-class SVM
24	GM	Geometric Mean
25	GI	Gini Index
26	IG	Information Gain
27	IDS	Intrusion Detection Systems
28	KC	Kappa Coefficient
29	K-NN	k-Nearest- Neighbour
30	LFS	Linear Forward Selection
31	LSTM-CNN	Long Short-Term Memory Convolutional Neural Network
32	ML	Machine Learning
33	MCC	Matthews Correlation Coefficient
34	MLP	Multi Layer Perceptron
35	NB	Naïve Bayes
36	NPV	Negative Predictive Value
37	NN	Neural Network
38	OS-ELM	Online Sequential Extreme Learning Machine
39	OCS	Optimization-Cuckoo Search
40	PPV	Positive Predictive Value
41	R-ACO	Random Ant Colony Optimization
42	RF	Random Forest
43	RT	Random Tree
44	REPT	Reduced Error Pruning Tree
45	R2L	Remote to Local
46	Q_{TN}	True Negative
47	SIDS	Signature-based Intrusion Detection System
48	S-SCAE	Stacked Contractive Autoencoder
49	SVM	Support Vector Machine
50	SU	Symmetrical Uncertainty
51	TNR	True Negative Rate
52	T_{TP}	True Positive
53	TPR	True Positive Rate
54	YI	Youden's Index
55	SU	Symmetrical Uncertainty