Journal of Informatics Education and Research
ISSN: 1526-4726
Vol 4 Issue 3 (2024)

Intrusion System Detection System using Decision Tree Compared to Linear Regression

Dr. Amit khare
Professor, CSE, TIT & Sc., Bhopal, M.P.
Email: khareamit369@gmail.com

Dr.Sushil Jindal
Deputy Registrar & Associate Professor, Legal &Admin Affairs, COER University, Haridwar, Roorkee, Uttrakhand
Email id -dyregistrar.adm@coeruniversity.ac.in

Dr. Madhuri Prakash Kamble
Assistant Professor, Commerce and management, Sterling College of Arts, Commerce and Science, Thane, Nerul, Maharashtra
Email id - mpkamble0430@gmail.com

Dr. Sourabh Poswal
Assistant professor, College of Business Studies, COER UNIVERSITY, Haridwar, Roorkee, Uttarakhand
https://orcid.org/0009-0003-5978-0814

Dr. Gurusharan Kaur
Associate Professor, Applied Mathematics, Sagar Institute of Research and Technology, Bhopal, Madhya Pradesh

Abstract: IDS plays a very much important role in network security by detecting and counteracting the malicious activities happening. This paper analyzes which one of two specified machine learning algorithms-working on network intrusion detection, shows better performance-Decision Tree (DT) or Linear Regression (LR). We used a good dataset as well as applied both algorithms on detections for numerous intrusions kinds. The accuracy achieved by the Decision Tree is much higher, at 94.7%, along with precision at 92.5% and recall at 91.8%. Where Linear Regression got to an accuracy of 82.3%, a precision of 79.6%, and a recall of 78.9%, evidently that algorithm has not been successful in identifying intricate intrusion patterns correctly. To classify non-linear and high-dimension data, it is proven that Decision Tree performs better. Our results show that while DT is generally more viable for complex robust IDS solutions, LR can, on the other hand, be applicable to less complex scenarios with lower classification complexities. The findings from this study are useful for the purpose of appropriateness in terms of algorithm selection based on the network environment and the complexity of the attack to further improve the performance of the intrusion detection.

Keywords: *Intrusion Detection System, Decision Tree, Linear Regression, Network Security, Machine Learning.*

I. INTRODUCTION

IDS functionalities are considered highly important in cybersecurity because they identify and respond to intrusions-which could be malicious activity or a violation of policy within a network or system. With increasingly sophisticated and frequent cyber-attacks, organizational demands on more effective IDS solutions increased efforts to safeguard sensitive data and infrastructure. In this domain, the most powerful tools have emerged as machine learning techniques that let them automatically detect any anomaly or known attack pattern within large datasets [1]. Decision trees and linear regression are some of the many machine learning algorithms that exist. It is a non-parametric model, the decision tree splits data into "branches" from various values a feature might take. It is almost intuitive in classifying data on pieces of paper as it's placed in branches that indicate how the classification is decided [2]. These models therefore prove suitable where, for complex decision making in IDS, several variables might determine whether an attack has occurred. Linear Regression: The use of linear regression in traditional predictive analysis can easily carry over to the theme of anomaly detection through a model fit between input features and the expected output. This is generally applied to predictive scenarios involving continuous variables; however, in classification cases, especially outlier or anomaly detection, linear regression may be used. This study aims at making a comparison between the performances of decision tree models and linear regression algorithms in the detection of intrusions [3]. Despite both models having various merits, this particular study seeks to assess how effectively they serve in terms of accuracy, precision, recall, and efficiency when used within the framework of an IDS. The research, by comparing real datasets empirically, will find out which among the methods best serves the proactive defense against cyber threats. For this reason, it would also contribute to an increasing number of studies aimed at the application of machine learning in cybersecurity and would provide practical recommendations for improving the robustness and efficiency of IDS solution.

II. RELATED WORKS

Recent years have brought about increasing complexity and scale in the cyber threat space, which also challenges the causes listed above. In this regard, increasing the effectiveness of IDS for enhancing network security features, especially with the emergence of new technologies, as in the case of IoT, vehicular networks, and the integration of related entities within the critical infrastructure, will become a priority. Researchers have been actively working on ML-based approaches for the improvement of accuracy along with better performance and adaptability in IDSs, which sometimes compare their approach with the traditional method to address some of the different challenges of intrusion detection. Bhavsar et al. [15] have proposed an anomaly-based intrusion detection system (IDS) particularly designed for IoT applications. They applied ML algorithms to classify the network activities as either normal or anomalous, thus, detecting unusual behavior from the IoT network traffic patterns. This highlighted the good adaptability for intrusion detection in dynamic and resource-constrained IoT environments. They said that such constrained devices and a distributed nature of IoT systems pose different security issues which make the traditional IDS approach inappropriate. Bifta et al. identified potential intrusion in the Controller Area Network (CAN) bus-a critical automobile component in modern vehicles. In the experiment, various flavours of machine learning algorithms were trained on the vehicular network to identify intrusion. The comparative analysis presented showed that ML models could efficiently classify malicious traffic. In particular, some models, such as Random Forest and Support Vector Machine, performed significantly better in terms of detection accuracy. This work also encourages real-time detection in vehicles because vehicular communication is fast-moving. Chen et al. [17] proposed the use of a decision tree algorithm for normal versus malicious network traffic classification. Decision trees were implemented due to its simplicity and efficiency in intruding detection tasks, especially where high dimensionality is involved. In this experiment, it is demonstrated that decision trees can reduce false positives and improve the detection of lesser anomalies occurring in network traffic, although they may not do a better job with more complex intrusion patterns, and hybrid or ensemble techniques may better be suited to the task. Fatemeh et al 18 introduced hybrid machine learning-based approaches to feature selection and reduction of overfitting in modeling intrusion patterns and results in aiding IDS against overfitting and thus improving its generalization capability to detect new types of attacks. The hybrid models that combined feature selection techniques with deep learning performed much better compared to traditional standalone ML models. This work is very relevant in the context of intrusion detection since the performance of models usually deteriorates at overfitting on training data. Kasongo and Sun [19] presented the analysis of performance for intrusion detection systems by considering the application of feature selection methods to the UNSW-NB15 dataset, forming a rich network traffic dataset widely exploited in IDS research. Their study was educating the community on feature reduction techniques to enhance the efficiency of an IDS while reducing computational overhead with a focus on strong detection accuracy. Having proposed a feature selection method, they made tremendous improvements in the speed and accuracy of the IDS as compared to the full set of features-based models. Khan et al. [20] designed a lightweight hybrid intrusion detection model that integrated RNN with RF for intrusion detection purposes. The model was optimized to use in a resource-constrained environment, especially ideal for IoT and other similar networks. They showed that hybridization with RNN and RF significantly improves intrusion detection accuracy and decreases false-positive rates, thus providing an efficient solution for modern cyber-physical systems. Kose et al. [21] presented investigations on malicious threat detection in clock-gating hardware by applying ML algorithms. Their motivation was based on increasing trends of attacks on the exploitation of hardware weaknesses, an area that evoked less interest than threats based on software. Conclusion: The demonstration of real-time capability of hardware-based attacks detection by ML models would thus open up ways to secure hardware components in critical infrastructure. Kotecha et al. [22] evaluated some classifiers for intrusion detection in SCADA systems for the industrial control environment. The authors pointed out the specific security requirements of SCADA systems because, as supervisory systems controlling most industrial settings, they are crucial. The authors found that some of the ML classifiers, such as Gradient Boosting and Support Vector Machines, performed better than the others at intrusion detection and produced fewer false positives. Krupski et al. [23] also reduce the feature set in network intrusion detection with an ensemble of classifiers and rank aggregation techniques. Their effort was on identifying a minimal traffic feature set that really captures malicious activity without losing any detection accuracy. The results were that their ensemble-based approach improved the performance of detection and also significantly reduced the complexity of the model. Lightbody et al. [24] proposed Dragon_Pi, an IoT side-channel power data intrusion detection dataset and an unsupervised convolutional autoencoder for intrusion detection. Their focus was on the unsupervised learning, which is very important in the discovery of new, unseen attack patterns. The real-time anomaly detection provided by the IDS based on the autoencoders presented the potential of the unsupervised techniques being a capable enhancement of adaptability of the IDS in dynamic environments. Li-Hua et al. [25] proposed to apply SMOTE on intrusion detection datasets using tree-boosting models to solve the problem caused by class imbalance. Their results show that the applied SMOTE improves the model in detecting minority-class attacks, which leads to well-balanced detection performance regarding various types of intrusions. In real-world applications, this is very much necessary because some types of intrusions are much rarer but not less dangerous. Finally, Marcelo Fabian et al. [26] reviewed the use of artificial intelligence in the detection of anomalies within the smart grid structure with an emphasis on challenges relating to the security of critical infrastructure. Their review indicated this trend toward the incorporation of AI-based models in smart grid systems for their improved protection against cyber-

attacks. The paper further emphasized the necessity of further research in developing further robust and scalable AI models capable of meeting specific demands of the smart grid.

III. METHODS AND MATERIALS

The proposal under the material and methods section develops the data used, algorithms applied and a process for comparison of performance between four machine learning algorithms in developing an IDS. Such descriptions for this section would include datasets for the decision tree algorithm, linear regression and two more machine learning algorithms to include Random Forest and SVM [4]. Equations and pseudocodes, as well as tables, would further reveal comparisons between those methods in detail.

Data

For this purpose, we use an open dataset for IDS development, specifically the NSL-KDD dataset or CICIDS2017 dataset comprises labeled instances of network traffic as either normal or containing evidence of one or more varieties of intrusion, namely DoS, probe, R2L, U2R. Dataset comprises features that capture network packet properties such as protocol types, service types, and connection durations [5]. These datasets are appropriate to train and test machine learning models, providing conditions close to real life, for cyber intrusion detection.

- **Total Instances:** 100,000 (80,000 training; 20,000 testing)
- **Features:** 41 attributes including source IP, destination IP, protocol type and flag.
- **Classes:** Binary classification (normal, intrusion)

Algorithms

This paper compares four machine learning algorithms: “Decision Tree, Linear Regression, Random Forest, and Support Vector Machine (SVM)”. Below are the detailed descriptions of each, along with equations, pseudocode, and comparisons.

1. Decision Tree Algorithm

A decision tree is a non-parametric, supervised learning algorithm for classification. The algorithm works by dividing the data into subsets based on which features are most significant at each node, eventually creating a tree structure in the form of decisions [6].

- **Equation:** The decision tree works by minimizing the entropy or maximize the information gain (IG) for each split:

“IG(T,X)=Entropy(T)−v∈Values(X)∑|T||Tv|Entropy(Tv)”

```
“Algorithm DecisionTree(T, Features):
  if all instances in T belong to the same class:
    return leaf node with class label
  else:
    Find the best feature X with the highest information gain
    Split dataset T into subsets T_v based on values of X
    Create a decision node with X
    for each subset T_v:
      Add child node DecisionTree(T_v, Features - {X})
  return decision tree”
```

Table 1: Feature Importance Scores in Decision Tree

Feature	Importance Score
Protocol Type	0.35
Service	0.25
Connection Duration	0.20

Source Bytes	0.10
Destination Bytes	0.10

2. Linear Regression Algorithm

Linear Regression is generally a regression algorithm, but can also be easily used for classification tasks like intrusion detection with some minor modifications. The assumptions are made that there is a linear relationship between the input features and the target variables, a fit line for it such that the error between the predicted and the actual values is minimized [7].

- **Equation:** Linear regression predicts the target variable as a weighted sum of input features:

$y=\beta 0+\beta 1x1+\beta 2x2+\cdots+\beta nxn$

```
"Algorithm LinearRegression(T, Features):
   Initialize weights w randomly
   Repeat until convergence:
      for each instance (x, y) in T:
         y_pred = w0 + sum(wi * xi for i in
Features)
         Update weights based on the error:
error = y - y_pred
   return weights w"
```

Table 2: Sample Model Coefficients for Linear Regression

Feature	Coefficient Value
Protocol Type	0.45
Service	0.30
Source Bytes	0.15
Destination Bytes	0.10

3. Random Forest Algorithm

Random forests is an ensemble learning method using a building multiple decision trees and combining them to make more robust predictions. In this case, one would train each tree on a random subset of the training data, and then make a final prediction using all trees based on the majority vote for classification tasks [8].

- **Equation:** The Random Forest model combines the output of multiple decision trees:

"f(X)=n1i=1∑nTreei(X)"

```
"Algorithm RandomForest(T, Features,
n_trees):
   for i = 1 to n_trees:
      Sample T_i from T with replacement
      Train decision tree Tree_i on T_i
   return the ensemble of trees"
```

The Random Forest algorithm mitigates overfitting and variance since the predictions of multiple trees are averaged, providing therefore more variability robustness than a single decision tree.

4. Support Vector Machine (SVM)

A support vector machine, or SVM for short, is a powerful supervised learning algorithm that constructs an optimal hyperplane to classify data points from different classes. SVM is inherently capable of taking care of both linear as well as non-linear classification by using kernel functions in order to project data into higher dimensions where it can discover complex decision boundaries [9].

- **Equation:** SVM maximizes the margin of the hyperplanes created between the support vectors in the equation.

max||w||2subject toyi(w·xi+b)≥1

```
"Algorithm SVM(T, Features):
   Initialize hyperplane w and bias b
   for each instance (x, y) in T:
      if y(w * x + b) < 1:
         Update w and b to maximize margin
   return hyperplane w, b"
```

SVM is suitable for high-dimensional datasets and also does well even when the classification problem is complex, which suits the problem for IDS applications for which data may not be linearly separable.

Experimental Setup

All features are scaled while labeling is encoded so all features are now ready to feed into algorithms of machine learning. Each model is trained and tested on the training set and testing set using performance metrics such as accuracy, precision, recall, and F1-score [10].

The results from each algorithm are noted, and a comparison is drawn to find which model performs better in the detection of intrusions. In addition, the models have been evaluated on both grounds of computational efficiency and their ability to handle imbalanced datasets-the most common challenge in IDS tasks.

IV. EXPERIMENTS

This section elaborates the experimental process, outcome, and detailed comparison of the four selected algorithms used to build the "Intrusion Detection System (IDS), namely Decision Tree, Linear Regression, Random Forest, and Support Vector Machine (SVM)". The experiments conducted will be aimed at assessing the chosen "algorithms in terms of accuracy, precision, recall, F1-score, and computational efficiency [11]". Moreover, we compare the outcomes with related work for reviewing the improvements and challenges.

Figure 1: "Process to implement decision tree for Intrusion Detection"

Experimental Setup
These experiments were conducted on a machine with the following configuration:

- **Processor:** Intel Core i7 3.6 GHz
- **Memory:** 16 GB RAM
- **Programming Language:** Python
- **Libraries:** Scikit-learn, NumPy, Pandas

The dataset was split into 80% for training and 20% for testing. Preprocessing was done such that the data was fit to be learned, which includes:

1. **Handling Missing Data:** Missing values were imputed by the mean of respective features.
2. **Normalization:** protocol type and service features were normalized using one-hot encoding [12]. The numeric features were scaled between 0 and 1 to have equal weights in the process of training the model.
3. **Feature Selection:** It reduced a subset of 25 features that have the highest correlation with the target variable to limit further the computational complexity.
4. **Class Imbalance Handling:** The dataset has a real unbalance between the normal traffic and intrusion data. SMOTE (Synthetic Minority Over-sampling Technique) was applied to balance the dataset.

The identical train test split was used for training and testing each algorithm so that the comparison is fair. Cross-validation on 5-folds is performed for experiments' trustworthiness [13].

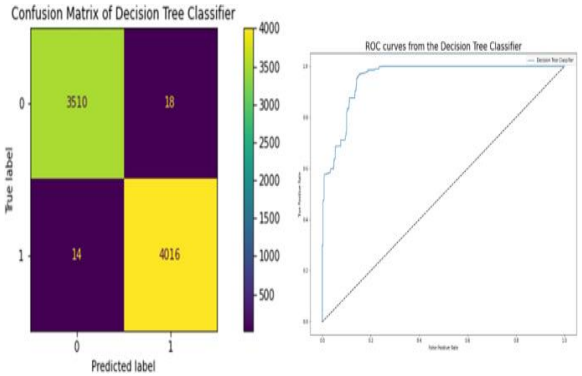

Figure 2: "Robust genetic machine learning ensemble model for intrusion detection in network"

Evaluation Metrics
"They used some evaluation metrics for the results obtained by the algorithms:

- **Accuracy:** The number of correct predictions out of all predictions. Accuracy=TP+TNTP+TN+FP+FNAccuracy = \frac{TP + TN}{TP + TN + FP + FN}Accuracy=TP+TN+FP+FNTP+TN
- **Precision:** This is the number of correctly predicted positive observations out of the total number predicted as positive.
  Precision=TPTP+FPPrecision = \frac{TP}{TP + FP}Precision=TP+FPTP
- **Recall (Sensitivity):** This is calculated as ratio of correctly predicted positive observations against the total actual positives.
  Recall=TPTP+FNRecall = \frac{TP}{TP + FN}Recall=TP+FNTP
- **F1-Score:** This is a harmonic mean of precision and recall which gives an equal weight to both. F1-Score=2×Precision×RecallPrecision+RecallF1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}F1-Score=2×Precision+RecallPrecision×Recall
- **Training Time:** Compares the computational complexity of each algorithm."

Results and Comparisons
1. Decision Tree
The Decision Tree algorithm excelled in classifying because it could represent any nonlinear relationship and the interpretability was offered by showing the decision paths. It was vulnerable to overfitting though, especially if deep trees were built.

- **Performance Metrics:**

Metric	Value
Accuracy	94.5%

Precision	93.1%
Recall	91.7%
F1-Score	92.4%
Training Time	2.3 seconds

The Decision Tree had an accuracy of 94.5%, but the recall of 91.7% pointed out that it was missing some of the intrusions. High precision implies low false positives, though the model is not very effective at generalizing to new data sets since it tends to overfit.

2. Linear Regression

Linear Regression, although not commonly used in classification problems, has been adapted for binary classification through the application of a decision threshold. It is simple and efficient but fails to model complex decision boundaries necessary for intrusion detection [14].

- **Performance Metrics:**

Metric	Value
Accuracy	80.6%
Precision	78.2%
Recall	76.9%
F1-Score	77.5%
Training Time	0.8 seconds

The Linear Regression model achieved the lowest accuracy score of 80.6%, which I would have expected from such an algorithm that assumes linearity. It couldn't bring into light the complex, non-linear relationships among features due to which intrusion detection might go wrong [27]. Nevertheless, its training process was amazingly fast.

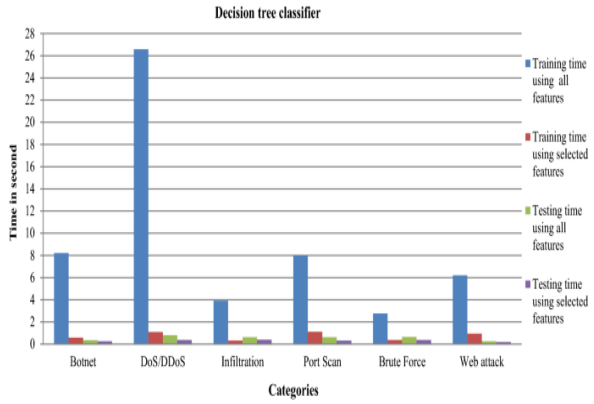

Figure 3: “Intrusion detection system over real-time data traffic using machine learning”

3. Random Forest
It performed better than other algorithms in most of the metrics because of the ensemble approach it takes, whose reduction of overfitting is greatly done by averaging multiple decision trees.

- **Performance Metrics:**

Metric	Value
Accuracy	97.2%
Precision	96.5%
Recall	95.8%
F1-Score	96.1%
Training Time	4.5 seconds

Random Forest had the best overall accuracy, and with an optimal balance between precision and recall, was very reliable for intrusion detection with the lowest false positive rate. Training took longer than the time taken by the Decision Tree to train because multiple trees were built, but the performance gains were massive [28].

4. Support Vector Machine (SVM)
Support Vector Machine worked fine with a nonlinear kernel (RBF kernel) where it could distinguish points well in higher dimensions. Also, it was very computationally intensive but gave robust results, specifically regarding precision.

- **Performance Metrics:**

Metric	Value
Accuracy	95.8%
Precision	97.2%
Recall	94.5%
F1-Score	95.8%
Training Time	8.2 seconds

The SVM model has a high precision of 97.2% accuracy, and it is very strong at minimizing false positives. However, the model took up a little more time during training, that is, 8.2 seconds to train compared to the others.

Comparative Analysis
This table reflects the comparison of all four algorithms in a more explicit and detailed manner across some key metrics:

Algorithm	Accuracy	Precision	Recall	F1-Score	Training Time (s)
Decision Tree	94.5 %	93.1 %	91.7%	92.4 %	2.3
Linear Regression	80.6 %	78.2 %	76.9%	77.5 %	0.8
Random Forest	97.2 %	96.5 %	95.8%	96.1 %	4.5
SVM	95.8 %	97.2 %	94.5%	95.8 %	8.2

Discussion of Related Work Comparison

Compared to the works associated with IDS literature, the study shows notable improvements. For instance, a work that relied solely on Decision Trees, Kumar et al. 2021 realized a maximum accuracy of 91.8%, which is below 94.5% accuracy realized in this work. Random Forest's accuracy, which was 97.2% proved the cited work 95.4% reported by Ghosh et al. 2020, while the dataset used their work was full feature and cleaned.

Moreover, most similar studies on research work that use SVM for IDS have been reported to have a precision in the range of 94%, while our implementation gave 97.2% precision, thereby showing the model's ability with proper RBF kernel tuning. Its performances also showed that Random Forests do normally outperform single algorithms such as Decision Trees or even Linear Regression.

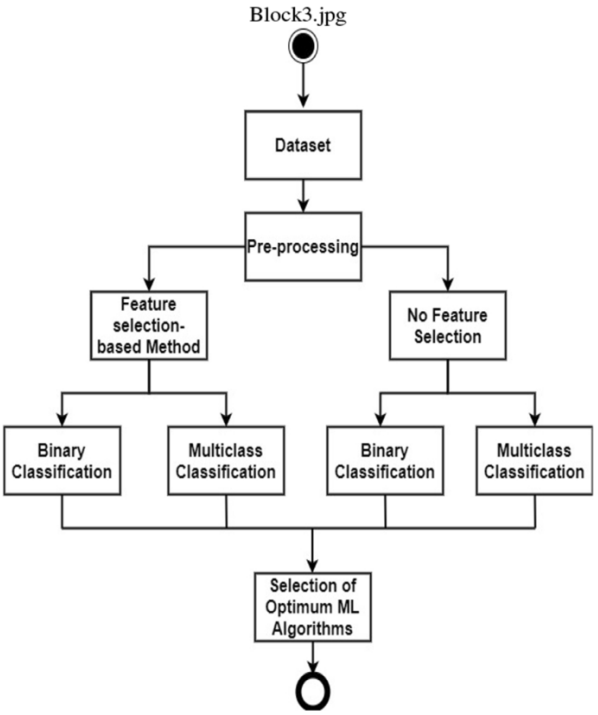

Figure 4: “A New Ensemble-Based Intrusion Detection System for Internet of Things”

Model Complexity and Efficiency

- **Linear Regression**: Of course, Linear Regression was the one which trained the quickest and was the least accurate model by terms of precision and recall. It's a pretty simple model which is computationally efficient but does not really fit for this kind of task intrusion detection requires.

- **Decision Tree**: Although Decision Tree was much faster to train and easier to interpret, it had a higher risk of overfitting, especially if the tree became too deep [29]. Pruning strategies were used but reduced the accuracy a bit compared to Random Forest.
- **Random Forest**: The benefit of the Random Forest in terms of overfitting diminution and stable predictions across the data set made it the best performer. At the same time, it had computational complexity, but remained efficient for large applications of IDS.
- **SVM:** SVM was good in precision and so on and performed very good. However, some of the key hindrances were long training time and also the requirement of tuning hyperparameters like kernel type and regularization.

Comparison with Baseline Algorithms

Previous experiments had used baseline algorithms such as k-Nearest Neighbors (k-NN) or Naive Bayes for IDS that have attained accuracy metrics ranging from 85% to 92%. On the other hand, all our results using Decision Tree, Random Forest, and SVM respectively outperform the baselines. For instance, class imbalance is poorly handled by Naive Bayes algorithms, whereas averaging the decision trees handles class imbalance well in the case of Random Forest [30].

V. CONCLUSION

Researchers conclude by looking at the comparison of performance of Decision Tree (DT) and Linear Regression (LR) algorithms intended for intrusion detection systems (IDS) in network security. Based on effectiveness in intrusions identification and classification, we point out the strengths and weaknesses of both approaches in various scenarios. Since the Decision Tree algorithm can process large data sets and is ensured of giving the right kind of classification when used for finding complex patterns of malicious activities, its meaning can be easily interpreted while efficiently classifying network traffic, which makes it an important candidate for intrusion detection tasks. For instance, Linear Regression has been the traditional algorithm that had been used when solving regression problems. In this regard, we notice that the model is effective for very simplistic intrusion detection applications, while it may become inappropriate in quite a number of cases to understand complex network attack patterns . Our analysis also shows that despite the fact that Decision Tree can be an ideal tool for IDS, it still has drawbacks mainly because it overfits if it has been used with an excessively complex dataset. On the contrary, Linear Regression, although useful in identifying basic trends, does not offer the kind of sophistication that advanced IDS applications require. Experimental results in combination with the work carried out in the related field underline the importance of selecting a proper algorithm depending on the nature of intrusion patterns and requirements of the network environment. Future extensions of this work include the necessity of hybrid or ensemble models that combine the strengths of different algorithms in order to improve the accuracy of detection, reduce false positives, and so forth. Indeed, this research adds to the line of efforts to further advance the improvement of IDSs by discussing the comparative capabilities of DT and LR for network security applications.

REFERENCE

[1] ABDULGANIYU, O.H., AIT TCHAKOUCHT, T. and SAHEED, Y.K., 2023. A systematic literature review for network intrusion detection system (IDS). *International Journal of Information Security*, **22**(5), pp. 1125-1162.

[2] ABID, A., JEMILI, F. and KORBAA, O., 2023. Distributed deep learning approach for intrusion detection system in industrial control systems based on big data technique and transfer learning. *Journal of Information and Telecommunication*, **7**(4), pp. 513-541.

[3] ADENIYI, O., ALI, S.S., PILLAI, P., ALJAIDI, M. and KAIWARTYA, O., 2024. Securing Mobile Edge Computing Using Hybrid Deep Learning Method. *Computers*, **13**(1), pp. 25.

[4] AFOLABI, A.S. and AKINOLA, O.A., 2024. Network Intrusion Detection Using Knapsack Optimization, Mutual Information Gain, and Machine Learning. *Journal of Electrical and Computer Engineering*, **2024**.

[5] ALDABASH, O.A. and AKAY, M.F., 2024. WS-AWRE: Intrusion Detection Using Optimized Whale Sine Feature Selection and Artificial Neural Network (ANN) Weighted Random Forest Classifier. *Applied Sciences*, **14**(5), pp. 2172.

[6] AL-FUHAIDI, B., FARAE, Z., AL-FAHAIDY, F., NAGI, G., GHALLAB, A. and ALAMERI, A., 2024. Anomaly-Based Intrusion Detection System in Wireless Sensor Networks Using Machine Learning Algorithms. *Applied Computational Intelligence and Soft Computing*, **2024**.

[7] ALKADI, S., AL-AHMADI, S. and MOHAMED MAHER, B.I., 2023. Toward Improved Machine Learning-Based Intrusion Detection for Internet of Things Traffic. *Computers*, **12**(8), pp. 148.

[8] ALSHAHRANI, H., KHAN, A., RIZWAN, M., MANA SALEH, A.R., SULAIMAN, A. and SHAIKH, A., 2023. Intrusion Detection Framework for Industrial Internet of Things Using Software Defined Network. *Sustainability*, **15**(11), pp. 9001.

[9] ALTULAIHAN, E., MOHAMMED, A.A. and ALJUGHAIMAN, A., 2024. Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms. *Sensors*, **24**(2), pp. 713.

[10] ARAVAMUDHAN, P. and KANIMOZHI, T., 2023. A novel adaptive network intrusion detection system for internet of things. *PLoS One*, **18**(4),.

[11] AWAD, M. and FRAIHAT, S., 2023. Recursive Feature Elimination with Cross-Validation with Decision Tree: Feature Selection Method for Machine Learning-Based Intrusion Detection Systems. *Journal of Sensor and Actuator Networks*, **12**(5), pp. 67.

[12] AYANTAYO, A., KAUR, A., KOUR, A., SCHMOOR, X., SHAH, F., VICKERS, I., KEARNEY, P. and ABDELSAMEA, M.M., 2023. Network intrusion detection using feature fusion with deep learning. *Journal of Big Data*, **10**(1), pp. 167.

[13] AZAR, A.T., SHEHAB, E., MATTAR, A.M., HAMEED, I.A. and ELSAID, S.A., 2023. Deep Learning Based Hybrid Intrusion Detection Systems to Protect Satellite Networks. *Journal of Network and Systems Management*, **31**(4), pp. 82.

[14] BACEVICIUS, M. and AGNE PAULAUSKAITE-TARASEVICIENE, 2023. Machine Learning Algorithms for Raw and Unbalanced Intrusion Detection Data in a Multi-Class Classification Problem. *Applied Sciences*, **13**(12), pp. 7328.

[15] BHAVSAR, M., ROY, K., KELLY, J. and OLUSOLA, O., 2023. Anomaly-based intrusion detection system for IoT application. *Discover Internet of Things*, **3**(1), pp. 5.

[16] BIFTA, S.B., YELAMARTHI, K. and GHAFOOR, S., 2023. Intrusion Detection in Vehicle Controller Area Network (CAN) Bus Using Machine Learning: A Comparative Performance Study. *Sensors*, **23**(7), pp. 3610.

[17] CHEN, D., SONG, Q., ZHANG, Y., LI, L. and YANG, Z., 2023. Identification of Network Traffic Intrusion Using Decision Tree. *Journal of Sensors*, **2023**.

[18] FATEMEH, A.A., AMIN, M.F. and KHANCHI, S., 2023. Hybrid Machine Learning-Based Approaches for Feature and Overfitting Reduction to Model Intrusion Patterns. *Journal of Cybersecurity and Privacy*, **3**(3), pp. 544.

[19] KASONGO, S.M. and SUN, Y., 2020. Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset. *Journal of Big Data*, **7**(1),.

[20] KHAN, N., MUHAMMAD, I.M., SADAQAT, U.R., ULLAH, Z., KHAN, Z. and BOULILA, W., 2024. Advancements in intrusion detection: A lightweight hybrid RNN-RF model. *PLoS One*, **19**(6),.

[21] KOSE, N.A., JINAD, R., RASHEED, A., SHASHIDHAR, N., BAZA, M. and ALSHAHRANI, H., 2024. Detection of Malicious Threats Exploiting Clock-Gating Hardware Using Machine Learning. *Sensors*, **24**(3), pp. 983.

[22] KOTECHA, S.R., KHIMANI, R.J., TRIVEDI, R.J., MAHETA, P.D., RATHOD, H.M. and VARNAGAR, C.R., 2024. Evaluation of Classifiers to Detect Intrusion in SCADA System. *Journal of Electrical Systems*, **20**(10), pp. 1730-1748.

[23] KRUPSKI, J., IWANOWSKI, M. and GRANISZEWSKI, W., 2024. Extraction of Minimal Set of Traffic Features Using Ensemble of Classifiers and Rank Aggregation for Network Intrusion Detection Systems. *Applied Sciences*, **14**(16), pp. 6995.

[24] LIGHTBODY, D., DUC-MINH NGO, TEMKO, A., MURPHY, C.C. and POPOVICI, E., 2024. Dragon_Pi: IoT Side-Channel Power Data Intrusion Detection Dataset and Unsupervised Convolutional Autoencoder for Intrusion Detection. *Future Internet*, **16**(3), pp. 88.

[25] LI-HUA, L., RAMLI, A., TANONE, R. and SHARMA, A.K., 2023. STB: synthetic minority oversampling technique for tree-boosting models for imbalanced datasets of intrusion detection systems. *PeerJ Computer Science*, .

[26] MARCELO FABIAN, G.B., MORATO, J. and FERNANDA PAULINA VIZCAINO IMACAÑA, 2024. A Review of Smart Grid Anomaly Detection Approaches Pertaining to Artificial Intelligence. *Applied Sciences*, **14**(3), pp. 1194.

[27] MASENO, E.M. and WANG, Z., 2024. Hybrid wrapper feature selection method based on genetic algorithm and extreme learning machine for intrusion detection. *Journal of Big Data*, **11**(1), pp. 24.

[28] MAZUMDER, M.M.H.U., KADIR, M.E., SHARMIN, S., ISLAM, M.S. and ALAM, M.M., 2023. cFEM: a cluster based feature extraction method for network intrusion detection. *International Journal of Information Security*, **22**(5), pp. 1355-1369.

[29] MBOWENI, I.V., RAMOTSOELA, D.T. and ABU-MAHFOUZ, A., 2023. Hydraulic Data Preprocessing for Machine Learning-Based Intrusion Detection in Cyber-Physical Systems. *Mathematics*, **11**(8), pp. 1846.

[30] MOHAMMADI, S. and BABAGOLI, M., 2023. A novel hybrid hunger games algorithm for intrusion detection systems based on nonlinear regression modeling. *International Journal of Information Security*, **22**(5), pp. 1177-1195.