# Detection of IoT Botnet Cyber Attacks Using Machine Learning

Alaa Dhahi Khaleefah[1], Haider M. Al-Mashhadi[2]
[1, 2] College of Computer Science and Information Technology, Computer Information Systems Department, University of Basrah, Basrah, Iraq
Email: alaa.dahy.2021.2022@gmail.com[1], mashhad01@gmail.com[2]

**Keywords:** IoT, botnet, malware, machine learning, intrusion detection, anomaly detection, classification

**Received:** February 8, 2023

*As of 2018, the number of online devices has outpaced the global human population, a trend expected to surge towards an estimated 80 billion devices by 2024. With the growing ubiquity of Internet of Things (IoT) devices, securing these systems and the data they exchange has become increasingly complex, especially with the escalating frequency of IoT botnet attacks (IBA). The extensive data quantity and pervasive availability provided by these devices present a lucrative prospect for potential hackers, further escalating cybersecurity risks. Hence, one of the paramount challenges concerning IoT is ensuring its security. The primary objective of this research project is the development of a robust, machine learning algorithm-based model capable of detecting and mitigating botnet-based intrusions within IoT networks. The proposed model tackles the prevalent security issue posed by malicious bot activities. To optimize the model's performance, it was trained using the BoT-IoT dataset, employing a diverse range of machine learning methodologies, including linear regression, logistic regression, K-Nearest Neighbor (KNN), and Support Vector Machine (SVM) models. The efficacy of these models was evaluated using the F-measure, yielding results of 98.0%, 99.0%, 99.0%, and 99.0% respectively. These outcomes substantiate the models' capacity to accurately distinguish between normal and malicious network activities.*

*Povzetek: Razvit je model strojnega učenja za zaznavanje in ublažitev napadov IoT botnetov, ki se je izkazal na domeni BoT-IoT.*

## 1 Introduction

Sensitive user data in large quantities is vulnerable to different internal and external threats. As technology has developed, cyberattacks have grown along with the complexity of algorithms [1]. Cyberattacks primarily target computers that process, store crucial data, or services that rely on those systems [2]. For the identification of malicious cyberattacks that represent a security risk, a unique intrusion detection system (IDS) is needed. IDS is an intrusion detection system that automatically detects and categorizes intrusions, security policy violations, and attacks on host and network infrastructures [1]. The constantly changing nature of threats has made it necessary to significantly tune and modify IDS performance by adding Machine Learning (ML) [3]. Artificial intelligence (AI) has a branch called machine learning (ML), which enables computerized learning without the requirement for outside programming [4]. ML techniques use historical data to learn and create predictions. The ultimate objective of ML is to create an effective technique that takes incoming data and produces a prediction using statistical analysis [5]. Two classes of machine learning techniques are recognized: Supervised learning and unsupervised learning are the first two.

A well-labeled training dataset with both normal and attack samples is necessary for supervised learning. This kind of learning involves giving the learning model the input and the target output so it can predict the future [6]. The datasets utilized to train these ML models directly affect the amount of training necessary [7]. Biases in data or algorithms that are ignored or concealed might provide skewed predictions and impair the effectiveness of AI applications [8]. in this situation, ML is among the most effective computational methods. to provide embedded smartness in the Internet of Things context. For a variety of network security tasks, including network traffic analysis [9-12], intrusion detection [12] and botnet identification [13], machine learning algorithms have been utilized. Figure 1 shows the IDS using ML in network and IoT environment.

The Internet of Things (IoT) has been multiplying in recent years all over the world. By the year 2030, there could be 125 billion IoT devices that are connected. The management of IoT networks has become increasingly difficult as a result of embedding these Iot systems with numerous alternative architectures, services, and protocols. As a result, the internet is exposed to significant risks and cyberattacks that could put users of such devices in danger [14].

The UNSW-NB15 network security dataset become available in 2015 [15]. 2,540,044 actual instances of both typical and abnormal behavior are included in this collection (often known as attack) functioning of networks in the electronic age. IXIA traffic generator employed three virtual servers to get this information. Two servers were set up to distribute standard network traffic, Table 1, shows the Summarization of the Related Works.

Table 1: Summarization table on the related works.

| Ref | Methodology | Performance/Results |
|---|---|---|
| [23] | • Genetic Algorithm | • The work used a combination of the Genetic Algorithm (GA) to remove unimportant characteristics and the Self-Organizing Map (SOM) classifier, which was optimized by GA's selected features, to find the high detection rates. |
| [24] | • Support Vector Machine | • Performance of IDS employing reduction features beats that of competitors using all features. The Support Vector Machine (SVM) classifier was used as a multiclass detection approach, and Mutual Information with Linear Correlation Coefficient (MI-LCC) was used to identify the best features. |
| [22] | • UNSW-NB15 | • They presented a hybrid system for IDS based on a Genetic Algorithm (GA) and Support Vector Machine for each assault in the UNSW-NB15 dataset (SVM). They transformed the traits into chromosomes and chose the ones with the best degree of correctness. (They suggested the Least Squares Support Vector Machine as a detection technique (LSSVM). The accuracy, true positive rate, and false-positive rate of the results were evaluated. |
| [25] | • Logistic Regression (LR), Support Vector Machines (SVM), and Random Forest (RF) | • The dataset is used by the authors to categorize botnet traffic in the Iot infrastructure. Nine operational IoT devices that were attacked by the Mirai and BASHLITE botnets provided the data for this dataset, which contains genuine network traffic information. Three classification techniques, Logistic Regression (LR), Support Vector Machines (SVM), and Random Forest (RF), are used to examine the data and classify it by botnet, attack, and device. |
| [15] | • UWSNs | • the authors suggest a novel routing protocol for the ocean floor that integrates two-dimensional UWSNs with sleep-scheduling routing to detect and report oil traces to the sink as soon as possible. |
| [25] | • K-NN | • By combining the K-NN algorithm with the clustering technique, the authors of suggest a new routing strategy that may significantly cut down on both latency and power consumption throughout the whole network. This proposal shows how to create clusters using node classifications and the shortest possible distances between them. |
| [16] | • Network Performance | • In order to learn about network performance through the identification of lost and transmitted packets, and to keep the cost of monitoring and communications infrastructure to a minimum, our system employs the placement of packet probes in passive monitoring devices on strategic links within the network. This work, which includes a user-friendly graphical user interface (GUI) and various data, metrics, and statistics related to network outcomes, can serve as a helpful manual for network researchers or other programmers wishing to analyze their networks and gain an understanding of how to calculate network performance. |
| [26], | • Extreme Learning Machine | • The approaches with various steps based on supervised ML were suggested. It begins by using the Synthetic Minority Oversampling Technique (SMOTE) to address the issue of imbalanced classes in the dataset before using the Extremely Randomized Trees Classifier to choose the crucial features for each class that already exists in the dataset according to the Gini Impurity criterion (Extra Trees Classifier). The detection of each attack is then done independently by a pretrained Extreme Learning Machine (ELM) model using "One-Versus-All" as a binary classifier. |

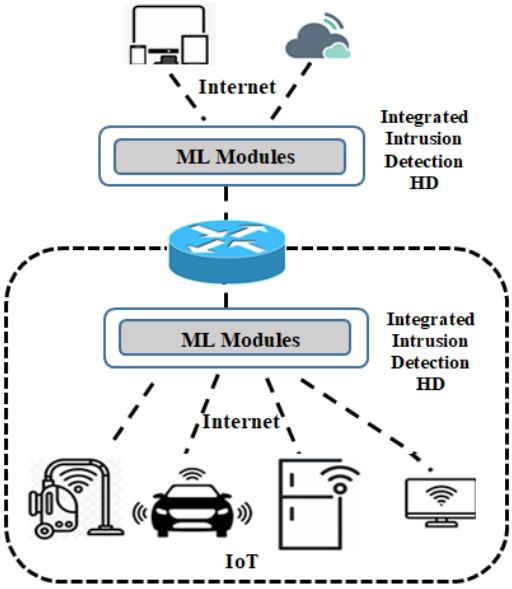


Figure 1: Integrated intrusion detection using ML

The Argus and Bro-IDS tools were used to break down the original network packets into a total of 49 attributes, including both flow-based and packet-based features. For packet-based features, the payload and header of the packet are mined. Sequencing packets as they travel from source to destination over the network in turn produces flow-based features. The direction, inter-packet length and inter-arrival times are the most important properties in the flow-based feature formulation: Two examples of flow-based characteristics are total duration (dur) and destination-to-source-time-to-live (dttl). The features are divided into three groups basic (6 to 18), content (19 to 26), and time (27 to 35). The terms "connection features" and "general-purpose features" refer to features 36 through 40 and 41 through 47, respectively. General purpose features are those qualities intended to illustrate the purpose of a particular record, whereas connection features show the characteristic of the interaction of 100 records in sequence, consecutively. The final two features are labels and attack categories.

The types of attacks include Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, and Worms. 2,218,761 records are used to represent typical attacks, whereas 24246, 2677, 2329, 16535, 44525, 215481, 13987, 1511, and 174 records, respectively, are used to represent fuzzers, analysis, backdoors, DoS, exploits, generic, reconnaissance, shellcode, and worm's signatures. As a result, the dataset shows a large imbalance in its distribution. since Normal records make up 87% of it while Worms records make up just 0.007%. The dataset's creators additionally subsampled and divided it into training and testing subsets, the testing set has 82,332 records from the attack and normal classes, whereas the training set contains 175,341 records of each type. as shown in Table 2, which other researchers have used [16-18]. In contrast to existing benchmark datasets as DARPA98 [19], KDDCUP 99 [16, 20], and NSL-KDD [21], among others, the UNSW-NB15 dataset has a more complex structure. As a result, the UNSW-NB15 is enhanced to provide a more thorough assessment of the current network intrusion detection technologies [16].

We used a number of preprocessing procedures to get the data ready for visual analysis. We first determine whether UNSW-NB15 has any redundant features, convert the nominal input features to numerical ones, rescale them, and then choose the pertinent input features.

In the UNSW-NB15 Dataset, nine different attack types have been identified.

1. Fuzzers: are attempts by the attacker to exploit security flaws in the operating system, network, or program in order to temporarily halt or even crash these resources.
2. Analysis: There is a category of intrusions that target online applications by scanning their ports, sending spam emails, and other means.
3. Backdoor: a method through which an attacker can acquire remote access to a system without being authenticated.
4. DoS: a kind of intrusion when the hacker makes an effort to overburden computational resources to prevent unauthorized access to them.
5. Exploit: a terminology used to characterize intrusions that profit from bugs, mistakes, or malfunctions in software or operating systems (OS).
6. Generic: By using a cryptographic system, this attack aims to decrypt the security system's key.
7. Reconnaissance: Also known as a probe, this type of attack gathers details about the victim computer system in order to get over its protection measures.
8. A malware attack known as a shellcode involves the hacker controlling the compromised machine by infiltrating a little piece of code beginning with a shell.
9. Worms are malicious software programs that reproduce themselves and spread to other computers via a network, relying on security flaws on the target computer that they are trying to reach.

To project the preprocessed data into a low-dimensional space, the research use binary classification and the classes of the dataset are lastly visualized using multi-class classification. After that data normalization, label encoding, correlations between features of dataset are performed. Also, the machine learning techniques are used such as Linear regression, logistic regression to predict the attacks.

Table 2: Shows the number of records for each group in the training and testing subsets.

| Classes | Training Subset | Testing Subset |
|---|---|---|
| Normal | 56,000 | 37,000 |
| Analysis | 2,000 | 677 |
| Backdoor | 1,746 | 583 |
| DoS | 12,264 | 4,089 |
| Exploits | 33,393 | 11,132 |
| Fuzzers | 18,184 | 6,062 |
| Generic | 40,000 | 18,871 |
| Reconnaissance | 10,491 | 3,496 |
| Shellcode | 1,133 | 378 |
| Worms | 130 | 44 |
| Total Number of Records | 175,341 | 82,332 |

# 2 Machine learning

Passive modern security techniques rely heavily on mathematical analysis models, which frequently do not reflect the correctness of the systems. Suitable defense in wireless environments necessitates weighty mathematical answers, that takes a long duration to compute and adds complexity [27]. Since machine learning algorithms are effective at modeling techniques that aren't able to be expressed by mathematical formulas, they will consequently play a vital role in IoT security solutions. The area of computer science known as machine learning allows machines to utilize previous instances and experience. The development of a ground-breaking new anomaly detection methodology based on machine learning allows for the discovery of anomalous traffic that may point to attempted network breaches [28]. The following list of machine learning algorithms (MLAs) adds the capability for computers to make decisions without being explicitly taught. Each MLA is formulated using sample data. Based on the sort of supervision provided during training [29], there are four different groups of MLAs. supervised learning, Unsupervised learning, semi-supervised learning and reinforcement learning [30].

**A. Supervised learning:**

In its simplest form, supervised learning describes teaching methods that involve a supervisor. It includes learning and prediction, as well as sample data with defined outcomes that make it easier for the algorithm to move from input to output [31]. Examples of supervised learning include classification techniques like KNN, SVM, Naive Bayes, Decision Tree, and Random Forest [32].

**B. Unsupervised learning:**

Unsupervised learning is the process of evaluating data without labels. It's also referred to as clustering. Similar to a self-directed learning method, Finding the unexpected data points is the aim of unsupervised learning [31].

**C. Semi-supervised learning:**

Machine learning techniques that blend a bigger sample of unlabeled data with a smaller amount of labeled data are referred to as semi-supervised approaches [31]. Between training data with labels and training data without labels, these learning fall. With more unlabeled data and fewer labeled data, these algorithms perform better [32].

**D. Re-enforcement learning:**

The area of machine learning known as reinforcement learning is centered on the agent, action, state, reward, and environment [32]. It does not presuppose mastery of any precise mathematical model; instead, it trains an agent composed of learning algorithms and policy through trial and error in an unsupervised setting.

# 3 Supervised ML algorithms

**A. Linear regression**

Model of variable x's linear function of dependence with respect to one or more independent variables (factors, regresses). As a straightforward forerunner to non-linear techniques utilized to teach neural networks, linear regression is the process of identifying the "best fit line" through a collection of data points. The technique entails decreasing the Euclidean distance between two vectors—a vector of the dependent variable's restored values and a vector of its actual values as in Eq. (1). The premise of linear regression is that parameters affect function f in a linear fashion. The linear dependence does not, however, always rely on a free variable x [33].

$$\boldsymbol{p(y|x) = \alpha(W.x + b)} \quad (1)$$

The logistic function produces probabilistic labels y for input data x.

The function first linearly transforms the input data x with the model's learned weights (W)and bias (b) parameters. The function then applies the nonlinear sigmoid (α) transformation to the linear result to produce the probability labels y, Eq. (2).

$$p(\boldsymbol{y|x}) = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b \quad (2)$$

**B. Logistic regression**

LR is a recognized statistical method for classifying data [34]. The logistic, or sigmoid, function provides the basis for the model (Eq. 3), and the training objective is to fit the function to optimally divide the training data. The resulting curve can be seen as an S-shape in 2D space in Figure 2.

$$\boldsymbol{f(x) = \frac{1}{1+e^{-x}}} \quad (3)$$

LR can be (i) binary, where the dependent variable (i.e., the output) is a category of two possible choices (for example, benign and anomaly), (ii) multinomial, where the dependent variable can be selected from a number of categories (for example, benign, attack 1, and attack 2), or (iii) ordinal, which is multinomial while the classes have an ordinal relation (for example, attack severity) [35].

Based on a threshold and a decision boundary, LR's output is determined. According to Eq. 4, in the binary situation, for instance, if the output is 0.5, it belongs to class A, and instead, it belongs to class B.

$$\boldsymbol{Y = \begin{cases} A, & f(x) \geq 0.5 \\ B, & Otherwize \end{cases}} \quad (4)$$

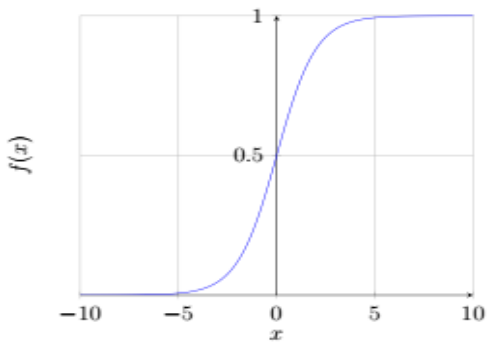


Figure 2: LR Sigmoid function.

### C. KNN

Based on Euclidean distance calculations, K Nearest Neighbor algorithms classify objects by the majority vote of their K neighbors with entities belonging to several classes [36]. K has a positive and typically low value. The number of selected neighbors, or the amount of K, determines how accurate the KNN algorithm is. For binary classification, the value of K is often an odd number to avoid the chance of two classes' labels having the same count. The value of K that is selected should be the best possible number; if it is too small or too large, the model may not fit the data as well Figure 3 shows the KNN model. In Euclidean n-space, the Euclidean distance between two points (X, Y) is written as:

$$\boldsymbol{d(X,Y) = \sqrt{\sum_{a=1}^{n}(Ya - Xa)^2}} \qquad (5)$$

### D. SVM

Based on the margin notation on either side of the hyperplane, SVM divides and separates the two data classes. Figure 5 illustrates the SVM. The margin and separation between the hyperplanes can be increased to improve classification accuracy. Support vector points are the data points that are located on the hyperplane's edge. SVM is divided into two main groups. Depending on the kernel function, it can be both linear and non-linear. Based on the type of detection, it may also be single-class or multi-class [37]. Both memory and time are important considerations when using SVM. In order to achieve better outcomes, SVM needs to be trained at various time intervals to learn the dynamic user's behavior. Eq. (6) represents the SVM [38]:

$$\boldsymbol{\min\|w\|^2 + C\sum_i^N max\,(0, 1 - yif(xi))} \qquad (6)$$

Where C is a regularization parameter that depicts the trade-off between maintaining that xi is on the predicted side of the plane and boosting the margin. In a two-dimensional space, where an SVM operates, the hyperplane appears as a line. When operating in extra dimensionality, it becomes an n-dimensional plane instead of a plane in three dimensions.

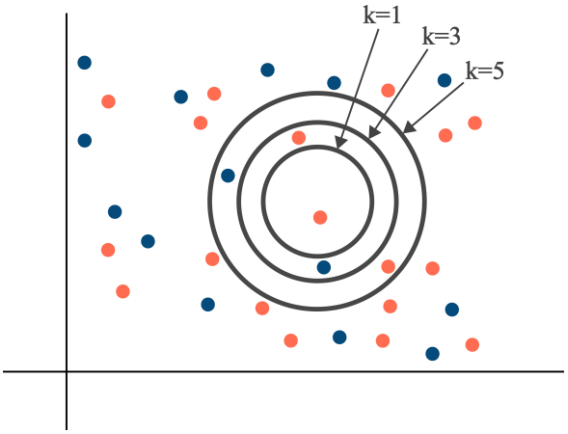


Figure 3: KNN model

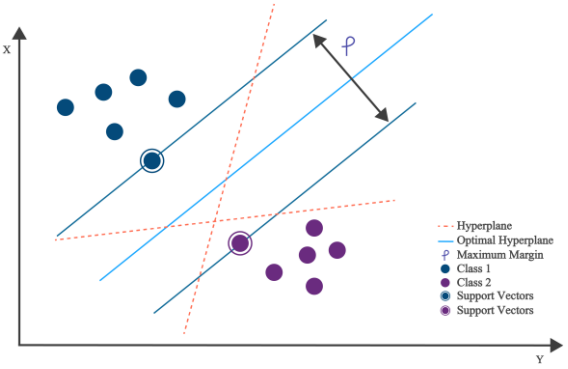


Figure 4: SVM model.

## 4 The proposed system

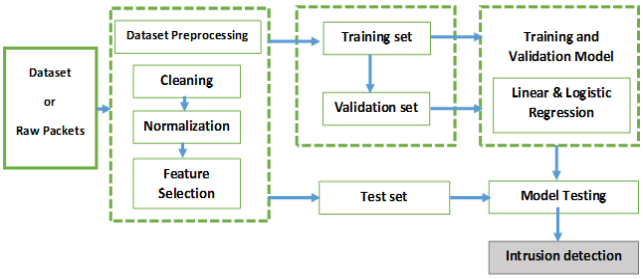


Figure 5: The structure of IDS system.

This section outlines the steps taken to create the botnet detection model, along with the datasets employed, preprocessing phase, experimental environment, outcomes, and justifications. To choose the optimum approach for our model, various supervised ML were applied to various combinations of Botnet dataset and the results were benchmarked. First, we looked at the packet data to examine the botnet behavior.

The dataset was split into two halves, one with regular traffic and the other with botnet traffic, in order to study the behavior of the botnet. This analysis assisted in choosing features with more trustworthy data, Figure (5) shows the structure of the IDS system.

**B. Normalization**

The learning process for ML techniques like Linear Regression and Logistic Regression is impacted by the large numerical value of many attributes. Additionally, a lot of computer resources are needed for the learning

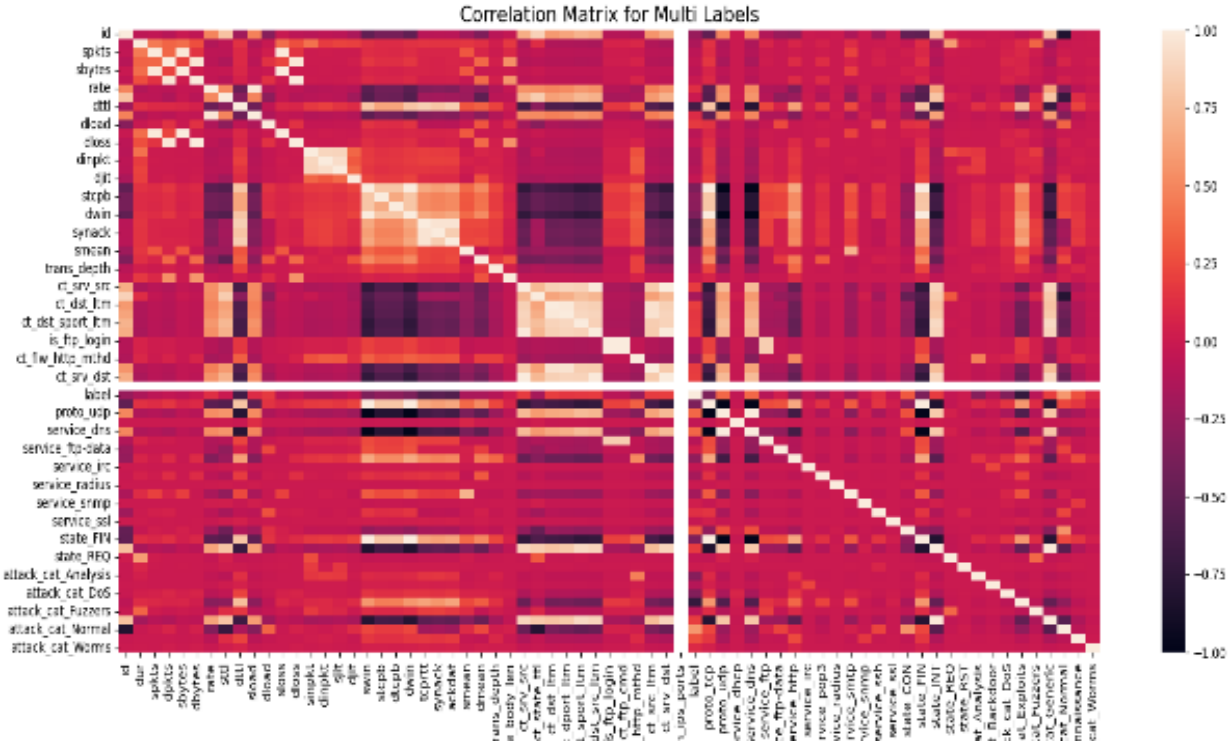


Figure 6: Correlation map of all features.

The term "data engineering" is frequently used to describe this procedure. For the learning process to be successful, this phase is essential. There are three processes in data processing: cleansing, normalization, and feature selection.

**A. Cleaning**

The first phase in the cleaning process is to look for null values. The dataset contains numerous fields with null values, which need to be replaced with the correct values.

Another step in the cleaning process is removing the unnecessary fields, such as "id" represents the first feature to be removed. This feature is not descriptive; it is an index. “attack-cat” is the second functionality to be removed. Since this feature is a continuation of the target feature, utilizing it will result in 100% accurate predictions but not a generalizable model.

The other features that must be removed are those that have excessive correlation. Since the model is first assessed to determine how effectively it can function, none of them were eliminated in the present edition.

Look for any incorrect values that may be included in any of the fields. We must address any issues if there are any. Despite being a binary column in this dataset, "is ftp login" has values other than 0 and 1. remove any values but (0 and 1).

of high dimensional datasets.

Data is frequently scaled using techniques like Z-score standardization, Decimal scaling, Max normalization, and Min-Max scaling to address these difficulties [39]. The application is frequently taken into account while deciding which method to use. In the data processing step, we apply the Min-Max scaling (Eq. 4).

$$F: F_{norm} = \frac{F - F_{min}}{F_{max} - F_{min}} \qquad \textbf{(7)}$$

The standardization calculation occurs as specified in Algorithm 1 with a dataset with an input sequence (feature domain) defined by $U(f_1,...,f_n)U(f_1,...,f_n)$, where $1<n<N$, in which N is the entire number of occurrences (features) in the domain.

Algorithm 1 Min-Max Scaling Algorithm

Input: $U(f1, \dots, fn), where\ 1 < n < N$

Output: $U_{normalized}(f1^{norm}, \dots, fn^{norm})$:

for $i$ from 1 to $k$ do

if ($fi$ a non-numeric input) then

Step 1: encode using one-hot-encode or scikit-learn feature mapping

Step 2 : Compute Min-Max Scaling: $f_n^{norm} = \frac{f_n - (f_n)_{min}}{(f_n)_{max} - (f_n)_{min}}$

end if

Step 1: Compute Min-Max Scaling: $f_n^{norm} = \frac{f_n-(f_n)_{min}}{(f_n)_{max}-(f_n)_{min}}$

**C. Correlation**

To create a histogram of the correlated values for better display, find the correlation between the characteristics in the dataset. If two features have a high correlation with the correlated values, only preserve one feature and discard the other. Correlation matrix is shown in figure 6.

**D. Feature selection**

First, the variance of each feature in the chosen model is calculated, and features that have zero variance are removed because they are not valuable for classification [40, 41] and retains those with a comparatively high variance for the following stage of feature selection. The methods are then saved in the list in descending order of feature importance once the primary elements for each technique has been calculated. Furthermore, the prediction accuracy AC for all attributes is computed and used to set a limit on the size of the subset; for this reason, a subset's performance is unlikely to be as good as it may be if its accuracy is lower than the accuracy utilizing all features. In the final process, features are sequentially added to the subset using the forward floating search method, beginning with the most crucial characteristics, and the effectiveness of each subset is assessed until AC is achieved. This step is based on the ranking list.

Algorithm 1 determines the relative relevance of each attribute and ranks them in descending order using a standard feature selection approach.

**Algorithm 2 Feature ranking algorithm**

1: Input:The training set $F = \{f_i, i = 1,2, \dots, I\}, i$ is the number of features;Feature
Selection method FS
2: Output:The ranked subset of features $F_{rank}$
3: for $i = 1$ $to$ $I$ $do$
4: Var=VarianceThreshoid ($F$)
5: if $F_i = 0$ then
6: $F' =$ Remove $F_i$ from $F$
7: end if
8: end for
9: Initialize FS Method
10: Fit $FS(F')$
11: Get $F_{imp}$ # Get the important of each feature
12: $F_{rank} = sorted(F_{imp})$
13:return $F_{rank}$

# 5 Results and discussions

**A. Performance metrics**

There are numerous measures for evaluating ML-based IDS systems, but the purpose of this study is to increase the proportion of cases in the test dataset that are correctly predicted. The following criteria are used to define these measurement systems:

$T_p$ = True Positive and the percentage of instances that are appropriately classified as attacks.

$T_n$ = True Negative, is the volume of legal traffic considered legal.

$F_p$ = False Positive, is the proportion of valid traffic labeled as attacks (also known as Type I error).

$F_n$ = The percentage of valid traffic that is categorized as intrusions is referred to as Type II error.

p = total positive = $T_p$+$F_n$.

n = total negative = Tn+Fp.

The Accuracy (AC) defined below should be the primary indicator to consider:

AC=(Tn+Tp)/(Fp+Fn+Tp+Tn) (8)

Precision provides information on how many of the specified things are relevant in comparison to the ones that were retrieved and has the following definition:

Precision= Tp/(Tp+Fp) (9)

Recall reveals the number of relevant items that are chosen from the overall pool of relevant objects is described as follows:

Recall=Tp/(Tp+Fn) (10)

From both precision and recall, the F1-Score can be calculated as follows:

$$F1_{Score} = 2\frac{Precision \times Recall}{Precision + Recall} \quad (11)$$

**B. Experiments and results**

The chosen machine learning techniques, Linear Regression, Logistic Regression, KNN, and SVM classifiers, are tested on the unique intrusion detection dataset UNSW-NB15. Python is used for the experimental work on an Intel Core (TM) i5-3210M CPU running at 2.50 GHz with 8GB of RAM. The dataset is separated into training and testing data when preprocessing is completed. Four classifiers are employed for training i.e. Linear regression, Logistic regression, KNN and SVM. Performance is assessed using a variety of factors and the chosen attributes, as shown in Tables 3 respectively. The accuracy of selected classifiers on all characteristics and when utilizing selected features are depicted in Figure 7.

Collate acknowledgements in a separate section at the end of the article before the references and do not, therefore, include them on the title page, as a footnote to the title or otherwise. List here those individuals who provided help during the research (e.g., providing language help, writing assistance or proofreading the article, etc.).

Table 3: The results of 4 methods.

| Classifier | Accuracy | Precision | Recall | F1-Score | MSE |
|---|---|---|---|---|---|
| **Liner R** | **97.8** | **0.97** | **1.00** | **0.98** | **0.021** |
| **Logistic R** | **97.76** | **0.97** | **1.00** | **0.99** | **0.022** |
| **KNN** | **98.3** | **0.99** | **0.99** | **0.99** | **0.016** |
| **SVM** | **97.85** | **0.97** | **1.00** | **0.99** | **0.21** |

The findings in Table 2 demonstrate that the KNN classifier performs better than the other approaches in terms of accuracy (98.3%), Precision (99%), and MSE (0.016). In contrast, among the chosen group of classifiers, the SVM exhibits the greatest MSE of 0.21 and the good accuracy of 97.85%.

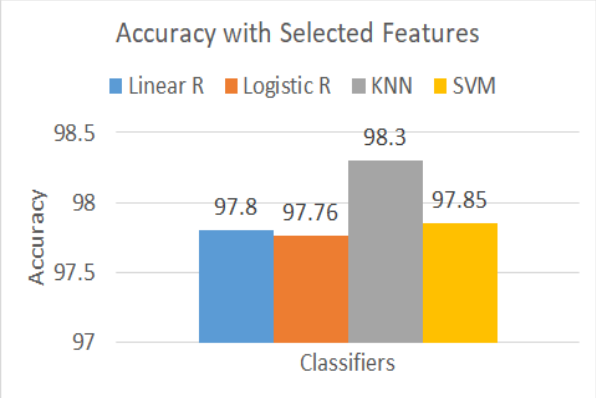


Figure 7. The accuracy for the selected methods.

## 6. Conclusion

For evaluating the performance of the ML classifiers for intrusion detection, experimental work has been done on linear regression, logistic regression, KNN, and SVM. The UNSWNB15 dataset is used to evaluate these models. On the basis of precision, MSE, recall, F1-Score, and accuracy, the classifiers are compared. Using particular parameters, the results demonstrate that the KNN classifier performs better than other classifiers on the UNSW dataset. The accuracy of the KNN classifier is 98.3%, while the accuracy of the Logistic Regression is the lowest of the other classifications at 97.76%.

## References

[1] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep Learning Approach for Intelligent Intrusion Detection System,” IEEE Access, vol. 7, pp. 41525–41550, 2019, doi: 10.1109/ACCESS.2019.2895334.

[2] M. A. Ferrag, L. Maglaras, A. Ahmim, M. Derdour, and H. Janicke, “RDTIDS: Rules and decision tree-based intrusion detection system for internet-of-things networks,” Futur. Internet, vol. 12, no. 3, pp. 1–15, 2020, doi: 10.3390/fi12030044.

[3] M. H. Ali, M. Fadlizolkipi, A. Firdaus, and N. Z. Khidzir, “A hybrid Particle swarm optimization-Extreme Learning Machine approach for Intrusion Detection System,” 2018 IEEE 16th Student Conf. Res. Dev. SCOReD 2018, pp. 2018–2021, 2018, doi: 10.1109/SCORED.2018.8711287.

[4] L. Haripriya and M. A. Jabbar, “Role of Machine Learning in Intrusion Detection System: Review,” Proc. 2nd Int. Conf. Electron. Commun. Aerosp. Technol. ICECA 2018, no. Iceca, pp. 925–929, 2018, doi: 10.1109/ICECA.2018.8474576.

[5] A. Haider, M. A. Khan, A. Rehman, M. Ur Rahman, and H. S. Kim, “A real-time sequential deep extreme learning machine cybersecurity intrusion detection system,” Comput. Mater. Contin., vol. 66, no. 2, pp. 1785–1798, 2020, doi: 10.32604/cmc.2020.013910.

[7] C. Song, T. Ristenpart, and V. Shmatikov, “Machine learning models that remember too much,” Proc. ACM Conf. Comput. Commun. Secur., pp. 587–601, 2017, doi: 10.1145/3133956.3134077.

[8] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, “A survey on bias and fairness in machine learning,” arXiv, 2019.

[9] I. Arnaldo, A. Cuesta-Infante, A. Arun, M. Lam, C. Bassias, and K. Veeramachaneni, “Learning representations for log data in cyberse-curity,” International Conference on Cyber Security Cryptography and Machine Learning, pp. 250–268, 2017.

[10] M. Du, F. Li, G. Zheng, and V. Srikumar, “Deeplog: Anomaly detection and diagnosis from system logs through deep learning,” Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp. 1285–1298, 2017.

[11] B. J. Radford, B. D. Richardson, and S. E. Davis, “Sequence aggregation rules for anomaly detection in computer network traffic,” arXiv preprint arXiv:1805.03735, 2018.

[12] Lambert, Glenn M. II, "Security Analytics: Using Deep Learning to Detect Cyber Attacks" (2017). *UNF Graduate Theses and Dissertations*. 728. https://digitalcommons.unf.edu/etd/728

[13] M. Stevanovic and J. M. Pedersen, “Detecting bots using multi-level traffic analysis.” IJCSA, vol. 1, no. 1, pp. 182–209, 2016.

[14] I. Alrashdi, A. Alqazzaz, E. Aloufi, R. Alharthi, M. Zohdy, and H. Ming, “AD-IoT: Anomaly detection of IoT cyberattacks in smart city using machine learning,” 2019 IEEE 9th Annu. Comput. Commun. Work. Conf. CCWC 2019, pp. 305–310, 2019, doi: 10.1109/CCWC.2019.8666450.

[15] Moustafa, N., & Slay, J. “UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)”. IEEE military communications and information systems conference (MilCIS) 2015. DOI: 10.1109/MilCIS.2015.7348942

[16] J. Alkenani and K. A. Nassar, “Network Performance Analysis Using Packets Probe For Passive Monitoring,” *Informatica*, vol. 46, no. 7, 2022 .

[17] Kumar, V., Sinha, D., Das, A. K., Pandey, S. C., & Goswami, R. T. “An integrated rule based intrusion detection system: Analysis on UNSW-NB15 data set and the real time online dataset,” Cluster Computing, 23(2), 2020, 1397-1418.

[18] Moustafa, N., & Slay, J. “The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set,” Information

Security Journal: A Global Perspective, 25(1-3), 2016, 18-31.

[19] LABORATORY, L. (1998). 1998 DARPA Intrusion Detection Evaluation Dataset. Retrieved from https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset

[20] Janarthanan, T., & Zargari, S. "Feature selection in UNSW-NB15 and KDDCUP'99 datasets," IEEE 26th international symposium on industrial electronics (ISIE), 2017. DOI: 10.1109/ISIE.2017.8001537.

[21] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. "A detailed analysis of the KDD CUP 99 data set," IEEE symposium on computational intelligence for security and defense applications, 2009. DOI: 10.1109/CISDA.2009.5356528.

[22] H. Gharaee and H. Hamid, "A new feature selection IDS based on genetic algorithm and SVM," 8th International Symposium on Telecommunications (IST), IEEE, Tehran, Iran, 2016. DOI:10.1109/ISTEL.2016.7881798

[23] M. Moukhafi, "Artificial neural network optimized by genetic algorithm for intrusion detection system," Advanced Intelligent Systems for Sustainable Development Conference, Springer, Berlin, Germany, 2018. DOI: 10.1007/978-3-030-11928-7_35.

[24] B. A. Manjunatha, P. Gogoi, and M. T. Akkalappa, "Data mining based framework for effective intrusion detection using hybrid feature selection approach," International Journal of Computer Network & Information Security, vol. 11, p. 8, 2019.

[25] Sikha Bagui, Xiaojian Wang, and Subhash Bagui, "Machine Learning Based Intrusion Detection for IoT Botnet," International Journal of Machine Learning and Computing, Vol. 11, No. 6, pp. 399-406, 2021.

[26] Soulaiman Moualla , Khaldoun Khorzom , and Assef Jafar, "Improving the Performance of Machine Learning-Based Network Intrusion Detection Systems on the UNSW-NB15 Dataset," Computational Intelligence and Neuroscience, Volume 2021, Article ID 5557577, 13 pages https://doi.org/10.1155/2021/5557577

[27] S. Ali, W. Saad, N. Rajatheva, K. Chang, D. Steinbach, B. Sliwa, C. Wietfeld, K. Mei, H. Shiri, H. J. Zepernick, T. M. C. Chu, I. Ahmad, J. Huusko, J. Suutala, S. Bhadauria, V. Bhatia, R. Mitra, S. Amuru, R. Abbas, B. Shao, M. Capobianco, G. Yu, M. Claes, T. Karvonen, M. Chen, M. Girnyk, and H. Malik, "6G White paper on machine learning in wireless communication networks," arXiv, pp. 1–29, 2020.

[28] J. Lam and R. Abbas, "Machine learning based anomaly detection for 5G networks," arXiv, pp. 1–12, 2020.

[29] J. Alkenani and K. A. Nassari, "Enhance work for java based network analyzer tool used to analyze network simulator files," vol. 29, no. 2, pp. 954–962, 2023, doi: 10.11591/ijeecs.v29.i2.

[30] S. Ray, "A Quick Review of Machine Learning Algorithms," in Proc. Int. Conf. Mach. Learn. Big Data, Cloud Parallel Comput. Trends, Prespectives Prospect. Com. 2019, pp. 35–39, Institute of Electrical and Electronics Engineers Inc., feb 2019.

[31] G. Shaheamlung, H. Kaur, and M. Kaur, "A Survey on machine learning techniques for the diagnosis of liver disease," Proc. Int. Conf. Intell. Eng. Manag. ICIEM 2020, pp. 337–341, 2020.

[32] G. Shaheamlung, H. Kaur, and M. Kaur, "A Survey on machine learning techniques for the diagnosis of liver disease," Proc. Int. Conf. Intell. Eng. Manag. ICIEM 2020, pp. 337–341, 2020.

[33] Veselska Olga , Ziubina Ruslana , Finenko Yuriy, Nikodem Joanna, "Big Data Analysis Methods Based on Machine Learning to Ensure Information Security," 25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems, Procedia Computer Science 192 (2021) 2633–2640.

[34] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, "*Applied Logistic Regression," Book First edition*, John Wiley & Sons, 2013.

[35] H. Hegre, "Logistic regression: Binomial, multinomial and ordinal." *Universitetet i Oslo*, pp. 1–35, r 2011. [Online]. Available: https://havardhegre.files.wordpress.com/2014/03/logisticregression2011.pdf.

[36] J. Huang, Y. Wei, J. Yi, and M. Liu, "An improved knn based on class contribution and feature weighting," Proc. - 10th Int. Conf. Meas. Technol. Mechatronics Autom. ICMTMA 2018, vol. 2018-Janua, pp. 313–316, 2018.

[37] Kamran Shaukat, Suhuai Luo, Vijay Varadharajan1, Ibrahim A. Hameed, Min Xu "A Survey on Machine Learning Techniques for Cyber Security in the Last Decade, IEEE Access" VOLUME 8, pp. 222310-222354, 2020. DOI:10.1109/ACCESS.2020.3041951

[38] J. Ngiam, D. Peng, V. Vasudevan, S. Kornblith, Q. V. Le, and R. Pang, "Domain adaptive transfer learning with specialist models." arXiv preprint arXiv:1811.07056, 2018.

[39] Liu Z, et al. A method of SVM with normalization in intrusion detection. Procedia Environ Sci. 11:256–62, 2011.

[40] D. K. Altmemi, A. A. Abdulzahra, and I. S. Alshawi, "A New Approach Based on Intelligent Method to Classify Quality of Service," *Informatica*, vol. 46, no. 9, 2022.

[41] Vaca, F.D.; Niyaz, Q. "An Ensemble Learning Based Wi-Fi Network Intrusion Detection System (WNIDS)." IEEE 17th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 2018. doi**:** 10.1109/NCA.2018.8548315.