applied sciences

MDPI

Article

Evaluating Ensemble Learning Mechanisms for Predicting Advanced Cyber Attacks

Faeiz Alserhani [1,*] and Alaa Aljared [2]

[1] Department of Computer Engineering and Networks, Jouf University, Sakaka 72388, Saudi Arabia
[2] Department of Computer Sciences, Jouf University, Sakaka 72388, Saudi Arabia; 431204009@students.ju.edu.sa
* Correspondence: fmserhani@ju.edu.sa

Abstract: With the increased sophistication of cyber-attacks, there is a greater demand for effective network intrusion detection systems (NIDS) to protect against various threats. Traditional NIDS are incapable of detecting modern and sophisticated attacks due to the fact that they rely on pattern-matching models or simple activity analysis. Moreover, Intelligent NIDS based on Machine Learning (ML) models are still in the early stages and often exhibit low accuracy and high false positives, making them ineffective in detecting emerging cyber-attacks. On the other hand, improved detection and prediction frameworks provided by ensemble algorithms have demonstrated impressive outcomes in specific applications. In this research, we investigate the potential of ensemble models in the enhancement of NIDS functionalities in order to provide a reliable and intelligent security defense. We present a NIDS hybrid model that uses ensemble ML techniques to identify and prevent various intrusions more successfully than stand-alone approaches. A combination of several distinct machine learning methods is integrated into a hybrid framework. The UNSW-NB15 dataset is pre-processed, and its features are engineered prior to being used to train and evaluate the proposed model structure. The performance evaluation of the ensemble of various ML classifiers demonstrates that the proposed system outperforms individual model approaches. Using all the employed experimental combination forms, the designed model significantly enhances the detection accuracy attaining more than 99%, while false positives are reduced to less than 1%.

Keywords: intrusion detection systems (IDS); ensemble learning; advanced attacks

Citation: Alserhani, F.; Aljared, A. Evaluating Ensemble Learning Mechanisms for Predicting Advanced Cyber Attacks. *Appl. Sci.* **2023**, *13*, 13310. https://doi.org/10.3390/app132413310

Academic Editor: Shen Wang

Received: 27 October 2023
Revised: 7 December 2023
Accepted: 14 December 2023
Published: 16 December 2023

1. Introduction

1.1. Background

In today's technologically advanced world, computer networks have become integral to our daily activities. However, this growing dependence on networks has also given rise to a surge in cyber-attacks. While security defenses like intrusion detection systems (IDS), firewalls, and anti-malware software play a crucial role in identifying attacks, their effectiveness largely relies on the detection models they employ. Unfortunately, conventional mechanisms often fall short in detecting advanced and sophisticated attacks, necessitating the integration of intelligent models. To combat these emerging threats, the use of machine learning, data analytics, and threat intelligence has become essential. By harnessing the power of these technologies, hidden relationships among intrusion events can be extracted, potential vulnerabilities identified, and indicators of compromise detected. This enables proactive measures to anticipate and prevent cyber-attacks before they occur [1].

In network security infrastructure, NIDS have emerged as vital components. NIDS are designed to monitor and analyze network traffic to identify suspicious and malicious activities. To enhance the accuracy and efficacy of NIDS, machine learning algorithms have proven to be essential to create intelligent security mechanisms [2]. One such powerful technique is ensemble machine learning, which combines multiple homogenous or heterogeneous models to achieve superior detection performance.

The potential of machine learning to enhance the accuracy and responsiveness of NIDS is immense [3]. By leveraging historical data, these models can effectively learn patterns of normal and anomalous network behavior, enabling them to classify incoming traffic as either benign or potentially malicious. This real-time detection capability empowers security teams to respond swiftly to potential attacks, thereby preventing or mitigating their impact [4]. Ultimately, integrating machine learning with NIDS can significantly sustain network security and reduce the risk of cyber-attacks.

Ensemble machine learning has attracted significant attention in research due to its ability to create more robust and accurate systems for attack detection [5]. This approach addresses the limitations associated with individual algorithms, such as overfitting, bias, or poor generalization of new data. By combining multiple classifiers, ensemble learning consistently outperforms individual models, resulting in enhanced accuracy [6]. The collaborative nature of ensemble methods can be classified into three categories: bagging, boosting, and stacking. Bagging involves training multiple models on different input subset samples and averaging their predictions, effectively reducing variance and addressing overfitting [7]. Boosting, on the other hand, integrates multiple weak learners, each learning from the mistakes of its predecessors, resulting in a stronger learner capable of making improved predictions and reducing bias in the training set [8]. Stacking, a two-level ensemble learning technique, combines the predictions of different single learning algorithms to make more accurate predictions on new data [8]. Various combination methods of ensemble algorithms including bagging, stacking, and boosting can be configured and designed to improve the model performance. It is also essential for leveraging the strengths and capacities of individual algorithms employed in different settings. ML algorithms offer a distinct set of characteristics and operational efficiencies in different contexts with diverse dimensionalities. Ensemble learning entails the combination of these available capacities, resulting in a system with precise prediction of potential unknown attacks. Moreover, limitations associated with a certain algorithm can be avoided by facilitating the efficiency features of another algorithm. To ensure valid experimental evaluation, a benchmark dataset such as UNSW-NB15 is utilized, which consists of numerous features related to network traffic and intrusion detection. The dataset features are analyzed, evaluated, and processed to improve detection performance, reduce computational complexity, and enhance the overall prediction accuracy.

1.2. Problem Statement

Current state-of-the-art network intrusion detection systems (NIDS) still face significant challenges in accurately detecting cyber threats whether new or evolving. Traditional NIDS approaches rely on pattern recognition and are not capable of detecting sophisticated attacks such as zero-day attacks and advanced persistent threats (APT). On the other hand, NIDS that use individual ML algorithms to generate detection models are criticized for numerous drawbacks such as overfitting, bias, and poor generalization of new data. These limitations can cause false negatives, where attacks go undetected, or false positives, where legitimate traffic is misclassified as an attack. Ensemble machine learning has emerged as a promising approach to address these limitations and improve the accuracy of attack detection. However, there is a need to explore the effectiveness of different ensemble methods and algorithm combinations for NIDS, as well as their potential limitations in terms of interpretability, complexity, and computational resources.

1.3. Research Objectives and Contributions

Ensemble machine learning techniques, which leverage the diversity and complementary strengths of multiple models, have the potential to significantly improve the accuracy, efficiency, and robustness of NIDS for detecting known and unknown attacks. We hypothesize that an ensemble approach, which combines multiple machine learning algorithms or models, will outperform a single-model approach in terms of detection accuracy, false positive rate, and detection time, across different types of network attacks scenarios. This

hypothesis emphasizes the advantages of using an ensemble approach for NIDS, which include the ability to handle complex and dynamic attack patterns, mitigate the impact of noisy or conflicting data, and adapt to evolving threats. It also highlights the key evaluation metrics that would be used to test the hypothesis, such as accuracy (the ability to correctly classify benign and malicious traffic), false positive rate (the frequency of misclassifying benign traffic as malicious), and detection time (the speed of identifying and reporting attacks). The hypothesis could be validated through empirical experiments that compare the performance of different ensemble techniques with that of a baseline single-model approach on benchmark datasets or real-world traffic traces.

The primary goal of this research paper is to evaluate the performance of ensemble machine learning techniques in the analysis of Network Intrusion Detection systems (NIDS) data. The predictive models created by different algorithms are utilized to recognize sophisticated cyber-attacks and advanced persistent threats (APT).

The following specific objectives will be tackled:

- Examining the current state of NIDS systems and machine learning for cyber-attack prediction. The application of ensemble machine learning to NIDS is reviewed to identify the limitations and challenges that need to be addressed.
- Assessing and contrasting various machine learning algorithms for predicting cyber-attacks using different NIDS datasets.
- Applying various stand-alone and ensemble machine learning mechanisms to develop and implement a system as a predictive model for the detection of unknown and advanced cyber intrusions.
- Selecting a realistic dataset that reflects real-world network traffic. The dataset contains advanced attack and APT patterns in a spatial-temporal distribution.
- Conducting a feature engineering process to extract the most effective attributes from the dataset to provide an accurate prediction.
- Evaluation of the efficacy of the proposed system using different performance criteria.

Despite the existence of few research efforts to evaluate ensemble learning approaches in several applications, there is a demand for more technical studies using distinct ensemble mechanisms, different tuning parameters, and various collections of ML algorithms. The study's contributions to the network security field can be presented in the following:

- Implementation and evaluation of distinct ensemble machine learning techniques for improving attack detection in Network Intrusion Detection Systems (NIDS).
- Development and implementation of different structures of ensemble-based NIDS trained in real-world attacks.
- Investigation of the impact of different ensemble techniques, including stacking, boosting, and bagging, on the detection accuracy, false positive rate, and detection time of NIDS.
- Evaluation of the proposed system using a well-known and documented benchmark dataset:
    - In terms of popularity and usage, the KDD Cup 1999 [7] dataset is one of the most widely used datasets for evaluating intrusion detection systems. However, this dataset is quite old (it was released in 1999) and may not reflect the current landscape of network attacks and traffic. Moreover, the NSL-KDD dataset [9], which is a modified version of the KDD Cup 1999 dataset, is also quite popular and has been used in many studies. However, it has been criticized for containing duplicate and irrelevant records, and for not representing realistic network traffic patterns.
    - In our work, we decided to use the UNSW-NB15 dataset [10–12], which has been selected as it is a modern dataset that was released and later on was continuously maintained and updated. It includes over 16 million records, including both benign and malicious traffic, and covers a wide range of attack types and scenarios. Furthermore, it has been used in several recent studies for ML-based NIDS evaluation which allows comparison of our system design with others.

- Identification of the strengths and limitations of ensemble-based NIDS, and insights into the factors that affect their performance and scalability.
- Demonstration of the potential of ensemble-based NIDS for handling a wide range of network attacks, including DoS, port scanning, malware infections, and multistage sophisticated attacks.
- Contribution to the field of network security and machine learning by presenting work that combines the benefits of ensemble techniques with the strengths of NIDS, and by identifying new research directions and challenges.

2. Related Work

The use of machine learning techniques in NIDS has gained significant attention in the research community [13,14]. NIDS plays a crucial role in monitoring network traffic to identify potential cyberattacks. Machine learning algorithms help in identifying patterns and behaviors in network traffic that may indicate malicious activities. These algorithms learn from historical data and are capable of detecting new and previously unseen attacks. One approach involves developing predictive models based on historical data to identify specific attack behaviors. By categorizing network traffic into different classes such as legitimate, suspicious, and malicious, machine learning algorithms assist in reducing false positives and improving the accuracy of NIDS.

Numerous works have focused on using machine learning techniques, including ensemble methods, for cyberattack prediction [15,16]. Supervised machine learning algorithms have been widely employed to classify network traffic into different classes. By training rule-based models on diverse datasets, NIDS can achieve more accurate classification and reduce false positives. On the other hand, unsupervised machine learning techniques have been utilized to analyze clustering and association patterns in network data, which can assist in identifying unknown attack patterns. However, in this paper, we primarily focus on supervised machine learning approaches based on binary classification.

In the systematic review [17], authors discussed different ensemble techniques, including bagging, boosting, and stacking. Bagging involves training multiple classifiers on different subsets of the dataset and aggregating their predictions. Boosting, on the other hand, focuses on sequentially training weak learners, with each subsequent learner learning from the mistakes of the previous ones. Stacking is a two-level ensemble approach that combines the predictions of multiple base classifiers using another meta-classifier. The review also emphasized the advantages of ensemble learning in NIDS. By combining multiple classifiers, ensemble methods can reduce the number of false positives and enhance the overall detection accuracy. They can effectively handle imbalanced datasets, improve robustness against adversarial attacks, and handle concept drift, where the underlying data distribution changes over time. The review paper [18] presents a comprehensive review of ensemble learning techniques for intrusion detection. They explored various ensemble strategies and their applications in the context of NIDS. The study highlighted that ensemble methods can effectively handle complex and diverse data, improving the detection of both known and unknown attacks.

Authors in [19] present an adaptive ensemble machine learning model for NIDS. The combination of individual models is dynamically modified, and the model weight is also adjusted in a flexible way. Multiple decision trees, random forest, kNN, and DNN, models are employed as base classifiers. The averaging mechanism is an adaptive voting method. The proposed design has been evaluated using the NSL-KDD dataset, and it has achieved an accuracy of 84.2%, while an adaptive voting mechanism has an accuracy of 85.2%. For improved outcomes, the authors recommend optimizing feature selection and preprocessing. In the [20] research paper, the ensemble-based ML model is proposed using several individual models including RF, GB, and AGX. Bagging and simple stacking averaging techniques are utilized. A port scanning dataset called SIMARG21 has been used and the obtained results illustrate that the RF model has achieved improved performance metrics reaching 99% in accuracy.

Ref. [21] provides an ensemble-based model that employs logistic regression, naive Bayes, and a decision tree with a voting classifier. The efficacy of the model was assessed using the CICIDS2017 dataset, which revealed considerable gains in accuracy in binary and multi-class classification scenarios. Ref. [22] presents a cloud-based technique for tasks to fog nodes. When tested on the NSL-KDD dataset, the technique performed well in the fog layer and took less time to execute than stacking. Ref. [23] focused on the application of ensemble techniques based on Random Forest for intrusion detection in network traffic. Their study demonstrated the effectiveness of the ensemble model in improving the detection accuracy of various network attacks. Moreover, references [24–28] have utilized ensemble learning models for NIDS in different applications and various domains such as IoT, cloud computing infrastructure, and sensor security. Distinct datasets have been used to train different combinations of employed ML models.

These publications exemplify the growing interest in utilizing ensemble machine learning approaches, such as Random Forest, Multilayer Perceptron, and other ensemble techniques, to enhance the accuracy and performance of NIDS [29,30]. By leveraging the strengths of multiple classifiers, these ensemble models contribute to the ongoing research in network security and intrusion detection. Nevertheless, the majority of research publications evaluate the designed model with some older datasets and focus on a specific design or on a particular method. Therefore, in our work, we aim to use different designs of ensemble ML as a solution for cyber-attack prediction. The proposed approach involves combining multiple machine learning algorithms to improve the accuracy and reliability of the prediction. Ensemble techniques such as bagging, boosting, and stacking are used to train multiple models and then aggregate their outputs to make a final prediction. By utilizing a variety of models, ensemble techniques can effectively handle complex and diverse datasets, and improve the detection of both known and unknown attacks.

3. Application of Ensemble Learning in NIDS

The rapid proliferation of computer networks has transformed the way we live and interact, bringing unprecedented convenience and connectivity to our daily lives. However, this growing reliance on interconnected systems has also exposed us to an escalating wave of cyber-attacks, posing serious threats to data security and privacy. In response, various security defenses such as IDS, firewalls, and anti-malware software have been deployed to safeguard networks from malicious activities. The inner workings of these systems range from simple rules matching to sophisticated intelligent models.

3.1. Overview of Intrusion Detection Systems (IDS)

In today's interconnected and digitally driven world, network security has become a paramount concern for individuals, businesses, and organizations alike. As the volume and complexity of cyber threats continue to grow, the need for robust IDS has become more critical than ever before. Intrusion detection plays a pivotal role in safeguarding computer networks and systems against a wide range of cyber-attacks, ranging from common malware and denial-of-service (DoS) attacks to APTs and zero-day exploits. The primary objective of intrusion detection is to detect and respond to unauthorized, malicious, or suspicious activities within a network or host environment. An intrusion can be an external attack launched from outside the network, an internal attack originating from within the organization, or even unintentional actions that may pose security risks. As such, IDS serve as a vigilant and watchful eye, continuously monitoring network traffic, system logs, and various other data sources to identify anomalies and potential signs of intrusion.

There are two primary types of IDS: Host-based Intrusion Detection Systems (HIDS) and Network-based Intrusion Detection Systems (NIDS). Host-Based Intrusion Detection Systems (HIDS) are deployed on individual host machines or endpoints and focus on monitoring activities specific to that host. These systems analyze various host-related events, such as file system changes, login attempts, and process activities. HIDS are effective in detecting attacks targeting a single host or instances where malicious activities

are confined to a particular system. They can provide granular visibility into the security of individual machines, making them valuable tools for endpoint security and forensic investigations. However, HIDS have limitations, especially when it comes to detecting attacks that span across multiple hosts or are distributed across the network. Since HIDS only analyze activities within the scope of a single host, they may miss attacks that manifest at the network level.

Unlike HIDS, Network-Based Intrusion Detection Systems (NIDS) operate at the network level, analyzing network traffic as it traverses the network. NIDS are strategically positioned at critical points within the network to capture and analyze data packets, headers, and payload information. By inspecting network traffic, NIDS can detect a wide range of attacks that may target multiple hosts or exploit vulnerabilities at the network level. NIDS are particularly valuable for monitoring large-scale networks where numerous hosts are interconnected, as they can provide a centralized view of potential threats across the entire network. They are capable of detecting suspicious patterns, unauthorized access attempts, malware propagation, and other network-based anomalies.

However, NIDS also face challenges. They may encounter performance bottlenecks when handling high network traffic, and they can be susceptible to encrypted traffic, which may conceal malicious activities from inspection. Moreover, NIDS may produce a significant number of false positives if not appropriately configured or if they lack the intelligence to differentiate between genuine threats and benign network activities. To address these challenges, the integration of machine learning techniques into NIDS has gained prominence. Machine learning enables NIDS to learn from historical network data, recognize patterns of normal behavior, and detect deviations that may indicate potential attacks. Moreover, by leveraging ensemble machine learning, NIDS can enhance their accuracy and resilience against sophisticated cyber-attacks, offering a more effective defense mechanism for safeguarding modern computer networks and critical infrastructure.

3.2. Machine Learning in Intrusion Detection Systems

Machine learning has emerged as a powerful and versatile approach to enhancing the capabilities of IDS. Traditional rule-based and signature-based IDS approaches have limitations in detecting unknown and sophisticated attacks. Machine learning techniques, on the other hand, offer a data-driven and adaptive approach that can effectively identify complex patterns and anomalies in network traffic, enabling IDS to stay ahead of evolving cyber threats. The main advantage of machine learning in IDS is its ability to learn from historical data and adapt to new attack techniques, making it particularly well-suited for dynamic and ever-changing cybersecurity landscapes. By training on vast datasets containing both normal and anomalous network behavior, machine learning models can discern subtle deviations and recognize novel attack patterns that may evade traditional rule-based systems.

Ensemble machine learning has garnered significant attention in the field of IDS due to its ability to mitigate the limitations of individual algorithms and improve overall detection performance. By combining multiple machine learning models, ensemble methods create a robust and collaborative system that outperforms the individual classifiers. There are three main categories of ensemble learning: bagging, boosting, and stacking. Bagging, such as the Random Forest algorithm, involves training multiple models on different subsets of the data and combining their predictions through averaging or voting. This approach reduces variance and enhances the overall accuracy of the IDS. Boosting, on the other hand, as exemplified by algorithms like AdaBoost and Gradient Boosting Machines (GBM), focuses on sequentially training multiple weak learners, with each learner learning from the mistakes of its predecessors. The resulting ensemble model is a strong learner that can generalize better to new and unseen data, effectively reducing bias in the detection process. Stacking is a two-level ensemble technique where the predictions of multiple individual models serve as inputs to a higher-level model, known as the meta-classifier.

The meta-classifier learns to make final predictions based on the outputs of the individual models, taking advantage of their diverse perspectives and expertise.

In the context of NIDS, ensemble machine learning can significantly improve the accuracy and reliability of attack detection. By combining the strengths of decision trees, support vector machines, and deep neural networks, the proposed ensemble model aims to achieve superior performance in categorizing network traffic and identifying potential security threats. Through this research, we envision that the integration of ensemble machine learning in NIDS will lead to a more robust and proactive defense against cyber-attacks. By harnessing the collective power of multiple algorithms, this approach holds the potential to fortify network security, reduce false positives, and enhance the overall resilience of computer networks against the ever-growing spectrum of cyber threats.

3.3. Research Methodology

The proposed methodology for this research involves creating an ensemble of eight different classifiers, including a Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), K Nearest Neighbors (KNN), Support Vector Machine (SVM), XGBoost (XG), CatBoost (CB), and Artificial Neural Networks (ANNs), to improve the accuracy and robustness of NIDS. Each classifier will be trained on different techniques based on the combination design specification. Some mechanisms require training on the same set of labeled data while others use distinct samples of the dataset. The constructed trained models are required to produce a separate prediction for each incoming network packet (packet record). The ensemble model will combine these predictions using different methods such as bagging, boosting, and stacking, in addition to simpler averaging and voting mechanisms, where the majority vote determines the final classification.

To evaluate the effectiveness of the proposed methodology, common performance metrics will be used, such as accurate detection rate, false alert rate, and processing time. Confusion matrix consisting of Accuracy, Precision, F-Measure and Recall is employed, to evaluate the final model. The Receiver Operating Characteristic (ROC) is also utilized to measure the classifier's efficiency. The evaluation will be conducted on a large dataset, UNSW-NB15 representing real-world network traffic, including both normal and attack traffic. The dataset is preprocessed to extract relevant features and remove noise and redundancy. Cross-validation and parameter-tuning techniques will be used to optimize the performance and avoid overfitting. The results are analyzed and visualized using various tools and techniques, such as statistical analysis, data visualization, and report generation. The research methodology can be briefed in the following stages, and it is illustrated in Figure 1:

1. Data Preprocessing: the UNSW-NB15 dataset will be preprocessed to ensure its quality and usability. This involves removing duplicate and irrelevant records, handling missing values and outliers, and normalizing or scaling the data as needed. Additionally, feature selection will be performed to reduce the dimensionality of the dataset and improve the performance of the models.
2. Feature Engineering: feature selection is performed using a correlation matrix to identify the most relevant features for detecting network attacks. The correlation matrix is used to compute the pairwise correlations between all pairs of features in the dataset. Features with high correlation coefficients are considered redundant and will be removed. The remaining features will be evaluated based on their discriminative power and ability to capture different types of attacks.
3. Model Implementation: different machine learning models will be used to build the proposed NIDS system. These models are selected based on their performance in previous studies and their suitability for handling complex and high-dimensional data. Implementation is performed using the Jupyter Notebook environment.
4. Model Training: the dataset is split into training and testing sets using different methods such as 70:30 ratio and 10-fold cross-validation. The training set is used to train the models, while the testing set is used to evaluate their performance.

5. System Evaluation: the proposed system is evaluated using a Confusion Matrix, which is a common and effective method for evaluating the performance of binary classifiers. The Confusion Matrix is used to compute various performance metrics such as accuracy, precision, recall, and F1-score. The results will be compared with those of existing approaches to determine the efficacy of the proposed system.

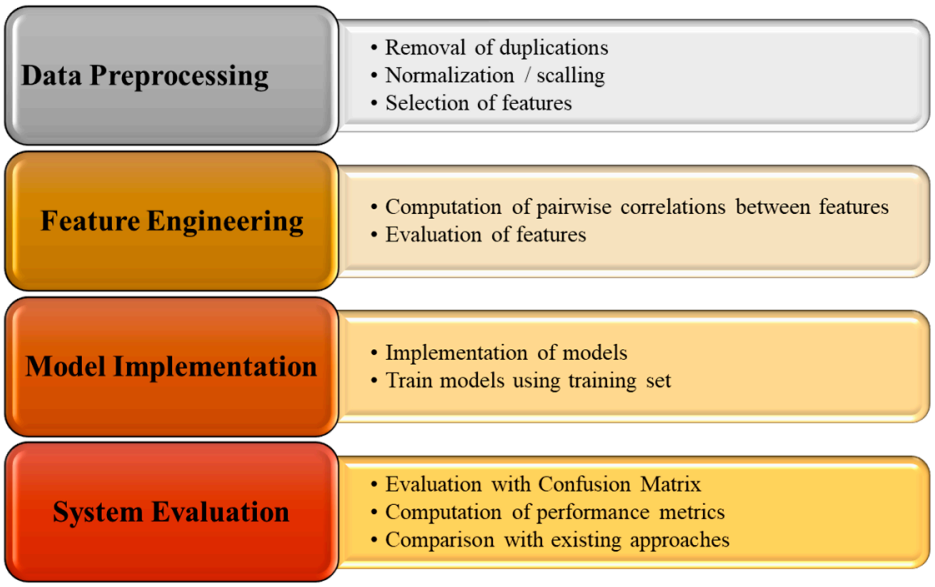

Figure 1. Research Methodology.

4. Implementation and Pre-Dataset Processing

4.1. Dataset

The evaluation of the proposed models has been conducted using the UNSW-NB15 [10–12] dataset, which is widely recognized in the field of network intrusion detection. The UNSW-NB15 dataset was created from real traffic data captured in an Australian university network, providing a more realistic representation of network activity. This dataset includes various attack types, normal traffic, and encrypted traffic, making it a comprehensive resource for evaluating IDS performance. Nonetheless, some researchers criticize the UNSW-NB15 dataset for its skewed class distribution [31], which can lead to biased performance evaluations, especially when dealing with rare or novel attack types.

While the aforementioned datasets have significantly contributed to the advancement of intrusion detection research, they all have their limitations that researchers should be mindful of [32]. To ensure robust and accurate evaluation of intrusion detection systems, it is crucial to select datasets that reflect the latest cyber-threat landscape, encompassing a diverse set of attack scenarios, and avoid biases in class distribution. Additionally, researchers should consider the specific requirements of their study and carefully validate the suitability of the chosen dataset for their research goals. By utilizing the UNSW-NB15 dataset, the experimental methodology aims to assess models' ability to accurately detect and classify different attacks based on real-world network traffic patterns. The inclusion of a wide range of attack types in the dataset ensures that the models are exposed to diverse and realistic attack scenarios, enabling a comprehensive evaluation of their performance in cyber-attack prediction. UNSW-NB15 is a comprehensive dataset commonly used to evaluate IDS models, created by the Cyber Range Lab of UNSW Canberra [10–12]. The dataset is a combination of modern real-world normal network traffic and simulated modern attack behaviors that reflect different security attack scenarios. A total of 2.5 million traffic records are stored in several files, and the ground truth table is given.

The UNSW-NB15 dataset has been selected for its realistic and sophisticated attack scenarios, a broad feature set that can be adapted based on the model functionality, variety of attack types accompanied with their corresponding vectors, and enhanced performance.

Data records contain normal and malicious activities that have been extracted from three distinct virtual servers. Additionally, it contains 49 features that have been obtained from network packets using the Bro-IDS and Argus. Before using this dataset to develop classifier models, however, issues like overlap, extreme values and class imbalance need to be resolved. Authors in [33] have conducted an in-depth analysis of the UNSW-NB15 dataset compared to the KDD99, focusing on the relevance of the features to model performance. They have concluded that the accuracy of the model is influenced by a limited set of features. Every record in the UNSW-NB15 is classed as either normal or attacked, with the attack label additionally split into ten categories for further granularity. This allows the classification to be performed in binary or multiclass fashion.

4.2. Ensemble Learning Mechanism

The design of the proposed model is illustrated in Figure 2, which begins with a data analysis phase. To ensure consistent data representation, numerical attributes are scaled to eliminate the mean and achieve unit variance. Categorical features are encoded to facilitate the sampling method, which addresses the challenge of imbalanced classification. Identifying a precise and accurate selection of features that directly affect the dependent variable is a critical stage in the process.

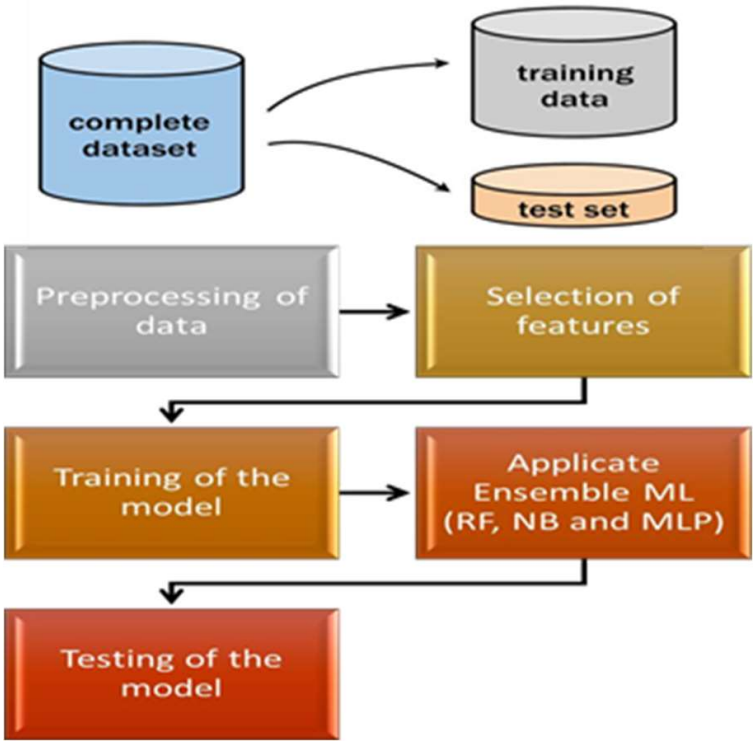

Figure 2. The proposed Model Process.

In Ensemble learning, a number of ML algorithms are selected, therefore, we have selected the following algorithms: Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), K Nearest Neighbours (KNN), Support Vector Machine (SVM), XGBoost (XG), CatBoost (CB), and Artificial Neural Networks (ANNs). Utility functions are used for model tuning by selecting model hyperparameters such as learning rate to obtain accurate prediction in an acceptable learning time. This can be performed automatically by trying different combinations of parameters and values. Grid search and random search methods are used for hyperparameter optimization.

Essentially, ensemble learning design entails building a set of multiple machine learning (ML) algorithms that may be used in various ways depending on how the learning process is organized along with the dataset used. We have implemented the proposed

system using three approaches: bagging, boosting, and stacking to compare the obtained results from each learning procedure. As a result, we have to select which algorithms to use for base models and which ones are used to create the meta-model. Figure 3 illustrates the implementation and evaluation process. It's important to note that the ensemble model maintains multiple models, which might lead to increased complexity and higher processing requirements.

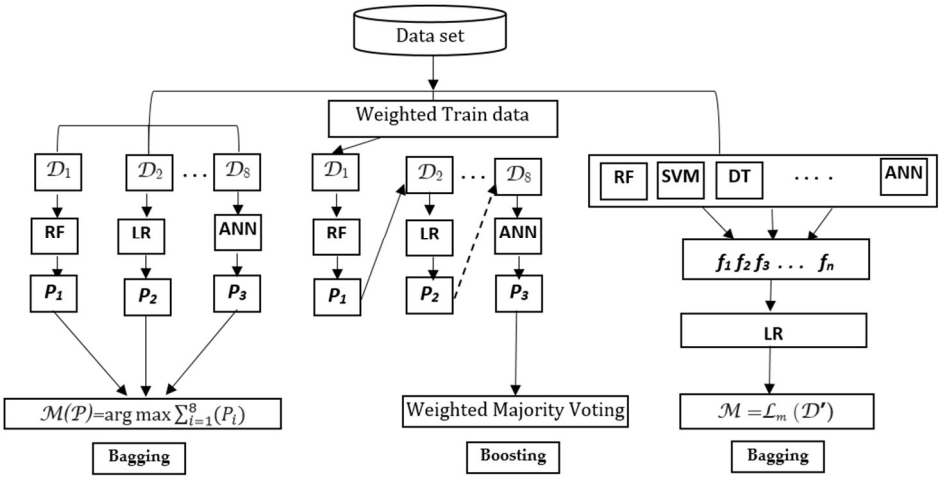

Figure 3. Ensemble Learning Mechanisms (Bagging, Boosting, and Stacking).

Following the selection of ML algorithms, the model structure is designed by arranging these algorithms and how to obtain the final prediction. There are several effective methods for ensemble learning that can be utilized in different structures as shown in Figure 3. Begging (Bootstrapping) ensemble learning is performed by generating multiple samples (bags) of the main datasets. Learning algorithms, also known as base models, are applied independently to each subset of the data. The obtained outcomes are combined using an averaged activity, maximum voting, or any aggregation mechanisms. The goal is to decrease the prediction variance and to improve the predictor force.

On the other hand, boosting ensemble learning, also referred to as forward additive modeling, uses a similar concept but applies a sequential procedure to each subset adaptively rather than conducting the training process in parallel. Each training model is based on the results of the previous one in order to boost the performance of every model. Subsets are not created randomly, yet it depends on the performance of the previous model.

In the ensemble stacking approach, two levels of the training process are performed, in level 1. The main dataset is used to train the model using heterogeneous learning algorithms (base models). The outcome of level 1 training is used as an input to the level 2 training, also known as a prediction vector, which is obtained from base models. The new dataset with its features is considered as an input for the meta-model which is created to generate the final prediction. The stacking algorithm is shown in Algorithm 1. as an instance of an ensemble mechanism.

To implement and evaluate the proposed model, several tools and methodologies were employed. The models were developed using Python and implemented with the Anaconda package and Jupyter Notebook as a development environment. The evaluation process involved several steps. Firstly, the dataset was pre-processed to ensure its compatibility with the models. This includes tasks such as data cleaning, feature selection, and normalization to enhance the quality and consistency of the input data. Next, the dataset was divided into training and testing sets to assess the models' performance. The training set has been used to train the models on known network traffic patterns and attack instances, while the testing set has been used to evaluate the models' ability to accurately predict unseen attacks.

Algorithm 1. Ensemble Learning—Stacking

Input: UNSW-NB15 Dataset D = {(x_1,y_1), (x_2,y_2), ..., (x_n,y_n)}
Base models L_1,L_2,. . .. . .L_k
Meta model L_m
Output: Trained Model M
Initialize k base models L_1,L_2,. . .. . .L_k
For each L_i do
train L_i (D)
End For
create a new training dataset D′ for the meta-model L_m
For each x_i in D do
create a new instance (z_i, y_i) where features are predictions of L_i
add (z_i, y_i) to D′
End For
train L_m using D′
M = L_m (D′)

Therefore, we have trained individual ML algorithms, using the same processes, and that is to give a thorough comparison between various learning methods. Second, we have implemented the ensemble learning model using the three approaches: bagging, boosting and stacking. The implementation stages can be divided into two parts: (1) Implementation of individual learning algorithms and (2) creating an ensemble model based on a combination of these individual models. The stacking technique involves training the model using the base learners and then a meta-model is generated which combines the outputs of the base learners. All base learners are trained on the entire dataset and all resulting predictions are collected to form the meta-learner. We have selected the LR model to represent the meta-model for its scalability and consistency.

4.3. Dataset Pre-Processing

The availability of reliable datasets to evaluate various IDSs, has been a challenge for many years as the quality of the dataset plays a critical role in building IDS models. The dataset UNSW-NB15 and similar datasets have been created to overcome some issues such as unbalanced data, and missing values. However, to obtain accurate and high-performance security models it is required to construct a data pre-processing framework to utilize the dataset features efficiently. The dataset instances are labeled as normal or attack and the attack records are classified further into 11 categories as shown in Table 1. The attack classes can be used to represent intrusion scenarios that reflect sophisticated and multi-stage attacks seen in real life.

Table 1. Attack Classes and counts.

Attack Class	Count	Attack Class	Count
Generic	215,481	Analysis	2677
Exploits	44,525	Backdoor	2329
Fuzzers	49,576	Shellcode	1511
DoS	16,353	Worms	174
Reconnaissance	12,228		

The dataset contains 94 various features that specify different traffic characteristics including traffic flow details, content information, time data, connection statistics, and instance classification as normal or attack, in addition to attack categories. These features can be also recognized as flow-based features obtained from the details of sequences of packets directed from source to destination and packet-based features extracted from the packet header and payload fields. Categorical, binary, and numerical datatypes are the three

types of network traffic data. The distribution of attack categories creates an imbalance in the dataset. For instance, the number of worm instances is only 174 which is 0.007%, whereas normal traffic constitutes 87% of the total data. To obtain a reliable evaluation of intelligent models, a comprehensive preprocessing is required. Moreover, the dataset has been partitioned into training and testing subsets. However, the splitting operation will be conducted directly from the main dataset file using 30% obtained randomly to be the test subset and 70% as the training subset. Therefore, we start with the preprocessing stage and all operations are applied to the training set only.

4.3.1. Data Cleaning and Preprocessing

Irrelevant features dropped include "id" as this feature is just an index and does not have any role in building the designed model. In addition, all features which are strongly correlated should be removed from the feature subset because of no contribution to the generalization problem. For instance, the feature "attack_cat" is strongly linked to the record label giving an accurate prediction but without any support to construct high quality model. We have checked null values in the label feature and replaced them with "normal" values.

A clamping process is applied to limit values in a range to be between two certain data points to avoid extreme values in the dataset and to minimize the skewness of some distributions. We applied clamping using the rule of 10 times the median, in other words, if the value exceeds 10 times the median. It is pruned to 95%, which is close to the maximum, and small values are treated similarly. In order to achieve normal distribution for numerical values and to handle skewed distribution, log transformation is used. Every value in the feature will be logged, and replaced by its exponent value, and as a result, the X scale's value grows exponentially.

4.3.2. Class Distribution

The class of each observation in the dataset can be one of the two available Labels: attack category (11 attack classes) and a binary label designating whether the observation is "Normal" indicated by "0" or "Attack" indicated by "1". Binary classification has been used for attack prediction. Figure 4 shows the data distribution in both training and test samples.

```
In Train: there are 87.3360% % of class 0:Normal and 12.6640% % of class 1:Attack
In Test: there are 87.3870% % of class 0:Normal and 12.6130% % of class 1:Attack
```

Figure 4. Class Distribution training and testing data.

Exploratory analysis is an essential stage in gaining a good grasp of the data. We have implemented several steps to extract important features and to determine the best parameter settings. First, we have created a heatmap to determine the degree of correlation between different features as shown in Figure 5. The Pearson correlation coefficient is used

to generate the correlation matrix of the features, which shows the degree of correlations. The correlation strength is calculated within the [−1 to +1] range using the following formula, where r is the coefficient, x_i and y_i are the features, $\overline{x}$ and $\overline{y}$ are the means, and n is the number of observations [34].

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - x)^2(y_i - yx)^2}} \quad (1)$$

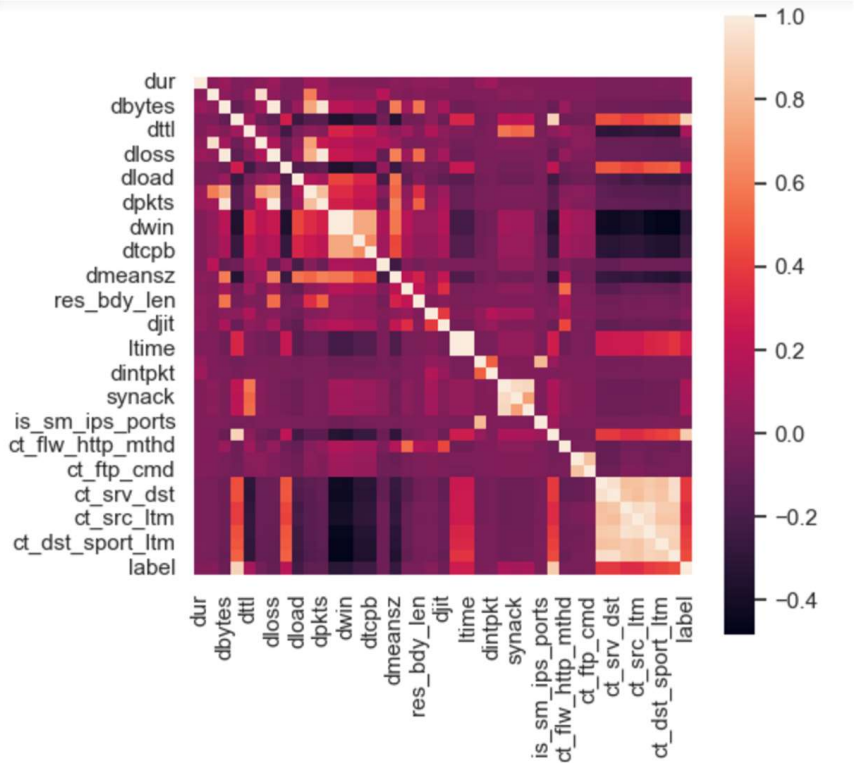

Figure 5. Features Heatmap.

We determined the most correlated features are shown in the list shown in Table 2.

Table 2. The most correlated features.

First Feature	Second Feature	Correlation Strength
sbytes	sloss	95.15%
dbytes	dloss	99.13%
dbytes	dpkts	97.06%
sttl	ct_state_ttl	90.58%
sttl	Label	90.43%
dloss	Dpkts	99.22%
swin	Dwin	99.72%
stime	Ltime	100.00%
tcprtt	Synack	93.32%
tcprtt	Ackdat	92.02%
ct_srv_src	ct_srv_dst	95.67%
ct_srv_src	ct_dst_src_ltm	94.21%
ct_srv_dst	ct_dst_src_ltm	95.10%
ct_dst_ltm	ct_src_ltm	93.85%
ct_dst_ltm	ct_src_dport_ltm	96.01%
ct_src_ltm	ct_src_dport_ltm	94.53%
ct_src_dport_ltm	ct_src_dport_ltm	92.14%
ct_src_dport_ltm	ct_dst_src_ltm	91.09%

Secondly, we have processed all features with a correlation of more than 95% from the list. Alternatively, using the visualized figure, the values of any two features that when plotted form a straight line are regarded as with high a strong correlation. We will explain two examples to show how highly correlated features can be detected by visualization.

(i) The two features sbytes and sloss have a correlation strength of more than 95%, and as shown in Figure 6a there is a straight line that represents this correlation.

(ii) The three features: ct_srv_src, ct_srv_dst and ct_dst_src_ltm ranged from 0–60, however, feature values tend to be close to 0 and less than 10. Some values are scattered around the origin point, but we can see a straight line to represent the feature correlation as shown in Figure 6b.

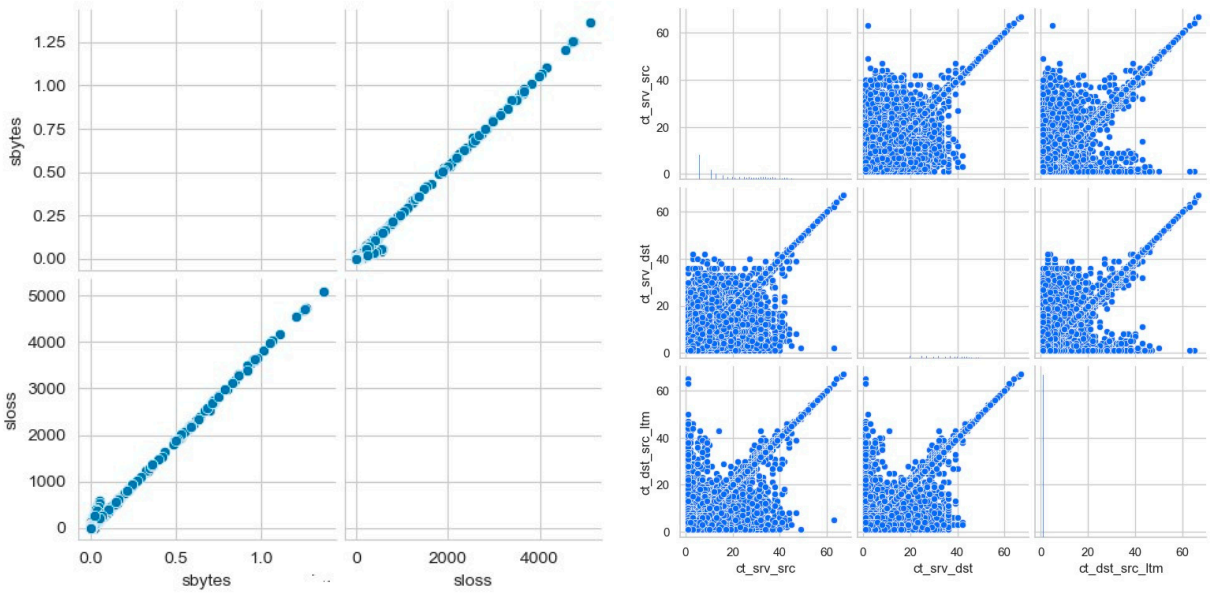

Figure 6. Correlated Features: (**a**) Line only, (**b**) Line with scattered points.

The next step is to deal with categorical values including attack category(attack_cat), protocol(proto), service(service), and protocol state(state). Additionally, some features with numerical values are considered categorical such as is_sm_ips_ports, ct_state_ttl, and is_ftp_login. The attack category involves nine classes that are assigned to the records labeled as 'Attack', while other records are considered 'Normal' as shown in Table 3. The dataset is considered unbalanced because the number of non-attack records is dominant.

Table 3. Label Distribution in (i) Train Data. (ii) Test Data.

(i) Train Data		
label	**count**	**percent**
0	1,552,862	87.335998
1	225,170	12.664002
(ii) Test Data		
label	**count**	**percent**
0	665,902	87.386994
1	96,113	12.613006

To minimize the unbalanced effect in the dataset, we reduced the rare values of certain features. The values of protocol, services, and state features are reduced to include [proto='tcp', 'udp', 'unas', 'arp', 'ospf'], [service='dns', 'http', 'smtp', 'ftp-data', 'ftp', 'ssh',

'pop3'], and [state='int', 'fin', 'con', 'req']. The probability density function (pdf) indicates the probability of continuous features taking on a specific value. Log transformation [35] 'log1p' is used for PDF curves for handling skewed data and, hence, resulting in a higher level of data symmetry.

$$X_{new} = \log(1 + X) \quad (2)$$

For Numerical features, the density figure function is utilized to display the values of every data record. Feature values are analyzed to normalize the existing data in order to remove the effect of dataset outliers. For instance, ct_dst_ltm feature represents the number of concurrent connections to the same destination IP address in the number of connections (100 connections). It is noticed that the range of this feature is from 0–70 connections. For normal traffic, the range is between 0 and 10 while it is from 0 to 30 for attack traffic. Figure 7 shows the density of this feature using log transformation.

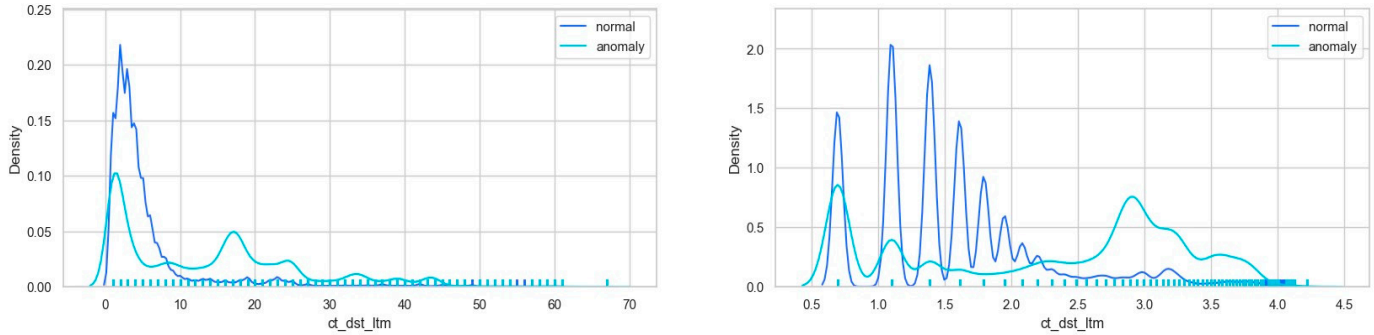

Figure 7. Density function of 'ct_dst_ltm' feature.

Another example of numerical features is 'ct_flw_http'_mthd which represents the number of flows associated with HTTP methods such as GET and POST. For attack traffic, the value of this feature is commonly 0. Furthermore, the feature of ct_srv_src measures the number of connections of the same service as shown in Figure 8.

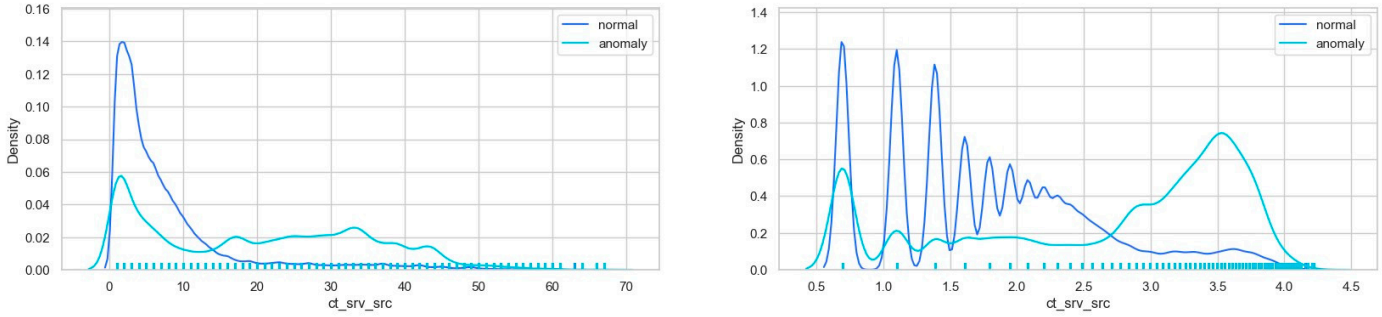

Figure 8. Density function of 'ct_flw_http' feature.

4.3.3. Feature Engineering

We have performed several measures to improve the features' representation of the required model, involving:

1. Removing highly correlated features: to increase efficiency and to avoid unnecessary cost of memory and as a result more accurate prediction model. The features ['sloss', 'dloss', 'dpkts', 'dwin', 'ltime', 'ct_srv_dst', 'ct_src_dport_ltm', 'ct_dst_src_ltm'] have been removed from the training data.
2. Adding New Features: total network bytes as a sum of 'sbytes' (Source to Destination bytes) and 'dbytes' (Destination to Source bytes).
3. Normalization: to keep feature values in a certain range and to avoid extreme values we apply stadardscaler to have these values in the range [0, 1].

4. Encoding of categorical features: convert text data in categorical features into numerical ones using onehotencoder. For instance, in the binary category, we assign 0 for True and 0 for False, and for several values, we can assign numbers starting from 1. Table 4 shows an excerpt of the application of log transformation to data columns.

Table 4. *log* transformation of some data features.

First Feature	Correlation	log_Correlation
dur	0.0019274028701131475	−0.03254413756460623
sbytes	0.010344749695229565	−0.35616315558984374
dbytes	−0.07641408324436148	−0.5193868283741477
sload	0.19211948100086756	0.3474660145034949
dload	−0.21978094390126515	−0.6033545881626365
spkts	−0.12200425437154418	−0.3163533826967563
stcpb	−0.23365153315010911	−0.3135563222142905
dtcpb	−0.23346071773809843	−0.3134006479812102
smeansz	−0.06517990378993671	−0.15111450989648337
dmeansz	−0.27230605607442226	−0.564640212740172
res_bdy_len	−0.0268968526601601	−0.06794901821505799

5. Evaluation and Discussion

5.1. Evaluation Results

In order to assess and compare the performance of the constructed models, various evaluation metrics such as accuracy rate, false positive rate, and true negative rate need to be calculated. These metrics are derived from a confusion matrix, which presents a tabular representation of predicted and actual classes [36]. Instead of an image, Table 5 below illustrates the structure of the confusion matrix, such that:

- True Positives (TP): Represent the instances where the classifier correctly predicts positive classes.
- True Negatives (TN): Indicate the instances where the classifier correctly predicts negative classes.
- False Positives (FP): Refer to the instances where the classifier incorrectly predicts positive classes.
- False Negatives (FN): Represent the instances where the classifier incorrectly predicts negative classes.

Table 5. The structure of the confusion matrix.

	Predicted Positive	Predicted Negative
Actual Positive	True Positives (TP)	False Negatives (FN)
Actual Negative	False Positives (FP)	True Negatives (TN)

To assess the performance of the models, several key metrics can be calculated based on the terms. These metrics provide valuable insights into the accuracy and reliability of the models. The following calculations can be applied:

- Accuracy (TPR):

Accuracy, also known as the true positive rate, measures the overall correctness of the predictions. It is computed using the following formula:

$$\text{TPR} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad (3)$$

The accuracy metric takes into account both the true positives and true negatives and provides an overall assessment of the model's performance.

- Precision (P):

Precision refers to the percentage of positive predictions that are correct. It is determined by the following formula:

$$P = TP / (TP + FP) \quad (4)$$

Precision focuses on the correctness of positive predictions and provides insights into the model's ability to accurately identify positive instances.

- Recall (R):

Recall, also known as sensitivity or true positive rate, measures the proportion of positive instances that are correctly identified by the model. It is calculated as follows:

$$R = TP/(TP + FN) \quad (5)$$

Recall is crucial in scenarios where detecting all positive instances is of utmost importance. It helps evaluate the model's ability to capture positive instances.

- F1-score:

The F1-score is a weighted average of precision and recall, providing a balanced measure of the model's performance. It can be computed using the following formula:

$$\text{F1-score} = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall}) \quad (6)$$

The F1-score combines precision and recall, giving equal importance to both metrics. It is particularly useful when there is an imbalance between positive and negative instances in the dataset. By calculating these metrics, we can gain a comprehensive understanding of the model's performance in terms of accuracy, precision, recall, and overall balance between precision and recall using the F1-score.

Table 6 illustrates performance metrics extracted from the confusion matrix for distinct eight individual models in addition to the three implementations of the ensemble model. The application of an ensemble model based on bagging, boosting, and stacking is compared with standard individual algorithms. Moreover, Table 7 illustrates the detailed performance metrics for Bagging, Boosting, and Stacking mechanisms.

Table 6. Performance results of Individual and ensemble models.

Model	Accuracy	Precision	Recall	F1-Score
RF	0.94	0.95	0.95	0.93
LR	0.94	0.95	0.92	0.93
DT	0.93	0.99	0.93	0.94
KNN	0.88	0.92	0.90	0.90
SVM	0.89	0.91	0.91	0.92
XG	0.90	0.93	0.92	0.92
CB	0.90	0.94	0.91	0.91
ANN	0.91	0.94	0.91	0.90
Bagging	0.99	0.97	0.98	0.99
Boosting	0.99	0.98	0.98	0.98
stacking	0.99	0.99	0.99	0.99

Table 7. Detailed Performance Results of the Three Ensemble mechanisms.

Ensemble Mechanism		Precision	Recall	F-Score	Support
Bagging	**Normal**	1.00	0.99	0.99	665,902
	Attack	0.95	0.97	0.96	96,113
	Accuracy			0.99	762,015
	Macro avg	0.97	0.91	0.92	762,015
	Weighted avg	0.99	0.92	0.92	762,015
Boosting	**Normal**	1.00	1.00	1.00	665,902
	Attack	0.98	0.98	0.98	96,113
	Accuracy			0.99	762,015
	Macro avg	0.99	0.99	0.99	762,015
	Weighted avg	0.99	0.99	0.99	762,015
Stacking	**Normal**	1.00	0.99	0.99	665,902
	Attack	0.95	0.97	0.96	96,113
	Accuracy			0.99	762,015
	Macro avg	0.98	0.98	0.98	762,015
	Weighted avg	0.99	0.99	0.99	762,015

The methods of bagging and boosting have also performed well with a percentage that is almost identical to the stacking mechanism. Furthermore, great accuracy is accomplished by balanced precision-recall trade-offs. We have analyzed various models to predict accurately unknown security events. Experimental results of the proposed model combination demonstrate that ensemble algorithms achieve better performance compared with regular processing of ML algorithms. However, the selection of base models and feature space size plays a significant role in achieving higher performance. It is also noticeable that individual algorithms also have more accurate detection compared to other studies in the literature. This indicates that our method in feature engineering and dataset pre-processing has contributed to producing effective learning operations. The development of optimal IDS capable of detecting unknown and advanced attacks can be constructed using reliable datasets in addition to an extensive analysis of traffic data.

5.2. Discussion

The findings of this experimental study indicate that ensembles can improve performance and eliminate misclassifications that are present in individual classifiers. The essence of the machine learning algorithm is to search for the most accurate hypothesis in the possible hypothesis space ($\mathcal{H}$), that is, the hypothesis closest to the unknown function ($\mathcal{F}$). When the hypothesis space is large, large amounts of training data are required to limit the search for good approximations. Each training example eliminates any hypothesis in ($\mathcal{H}$), that misclassifies it, or makes it less credible. In binary classification problems, each training example can eliminate half of the hypotheses in ($\mathcal{H}$). Another reason why an ensemble is useful is that the training data may not provide enough information to learn from ($\mathcal{H}$). Most learning algorithms consider a very large hypothesis space, so even after eliminating the hypothesis of misclassifying training samples, there are still many observations. From an existing set of hypotheses, an ensemble of classifiers can be easily constructed and combined using the methods described throughout this study. A third reason for integration is that the hypothesis space ($\mathcal{H}$) may not contain the real function ($\mathcal{F}$). Instead, it can contain several equally good approximations of ($\mathcal{F}$). By using weighted combinations of these approximations, we may be able to represent classes other than ($\mathcal{H}$) [37].

In the cybersecurity context, network traffic analysis is critical for protection against cyberattacks. Therefore, it is vital to improve the performance of the prediction models

to obtain more accurate detection results and at the same time, false positives should be lower to the minimum. Throughout this research, three heterogeneous ensemble-learning approaches—stacking, boosting, and bagging—are used to combine four ML algorithms. According to our findings, ensemble-learning algorithms are able to achieve greater prediction accuracies than base classifiers. The employment of meta-classifiers by the stacking bagging, and boosting ensembles to reduce errors made during the learning phase led to greater prediction performance. The stacking mechanism can improve the accuracy of base classifiers by increasing the internal diversity of the stacking model and reducing bias and variance. The structure of heterogeneous ensemble-learning models brings more reliable and robust results to support IDS functionality to detect unknown attacks. An overall performance metric across all potential classification criteria is measured by AUC as illustrated in Figure 9.

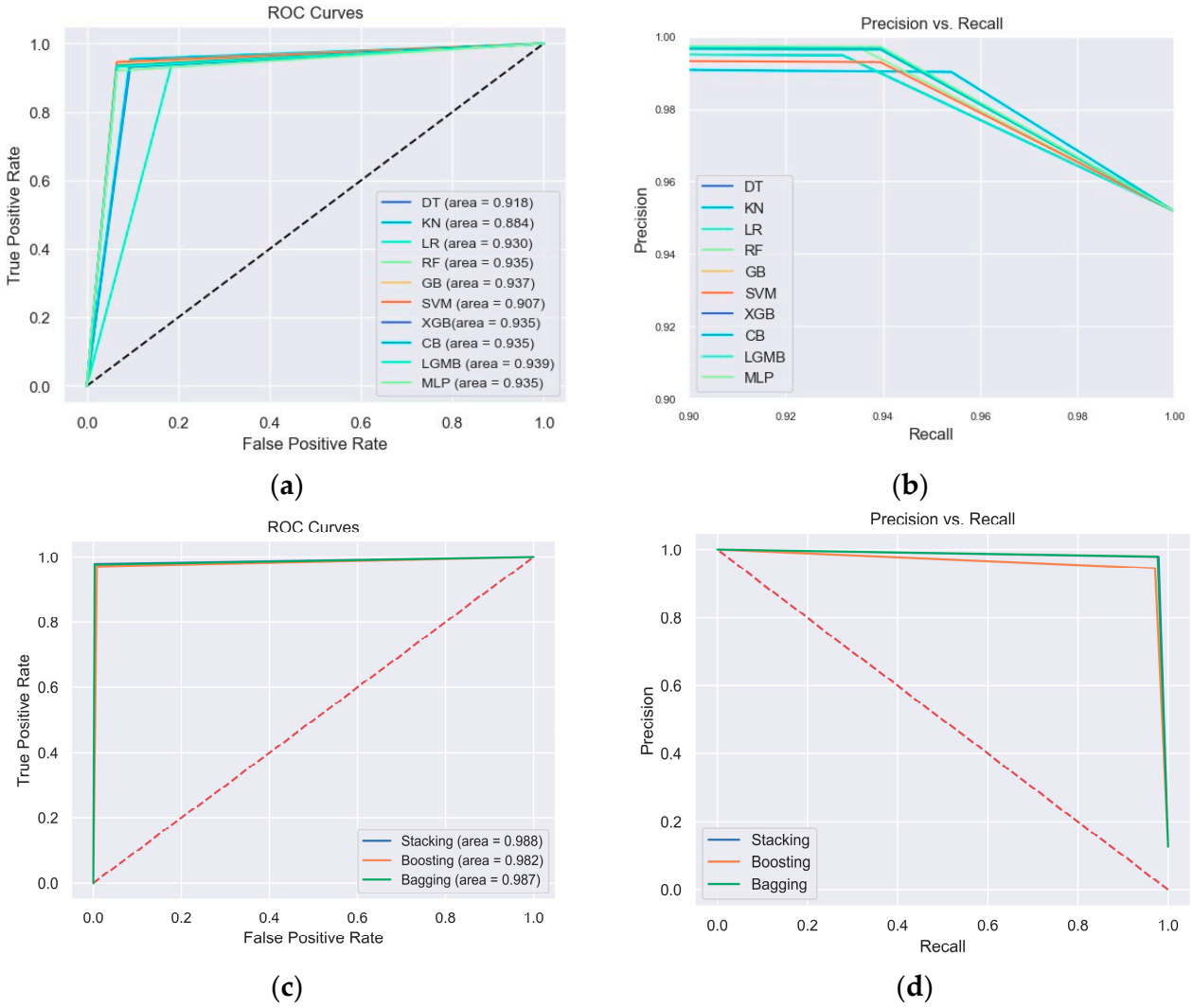

Figure 9. Performance Metrics. (**a**) False Positive Rates-Individual Algorithms, (**b**) Precision vs. Recall-Individual Algorithms, (**c**) False Positive Rates-Ensemble Model, (**d**) Precision vs. Recall-Ensemble Model.

The confusion matrices of the three applied mechanisms are displayed in Figure 10, where "0" indicates Normal and "1" is used for Attack.

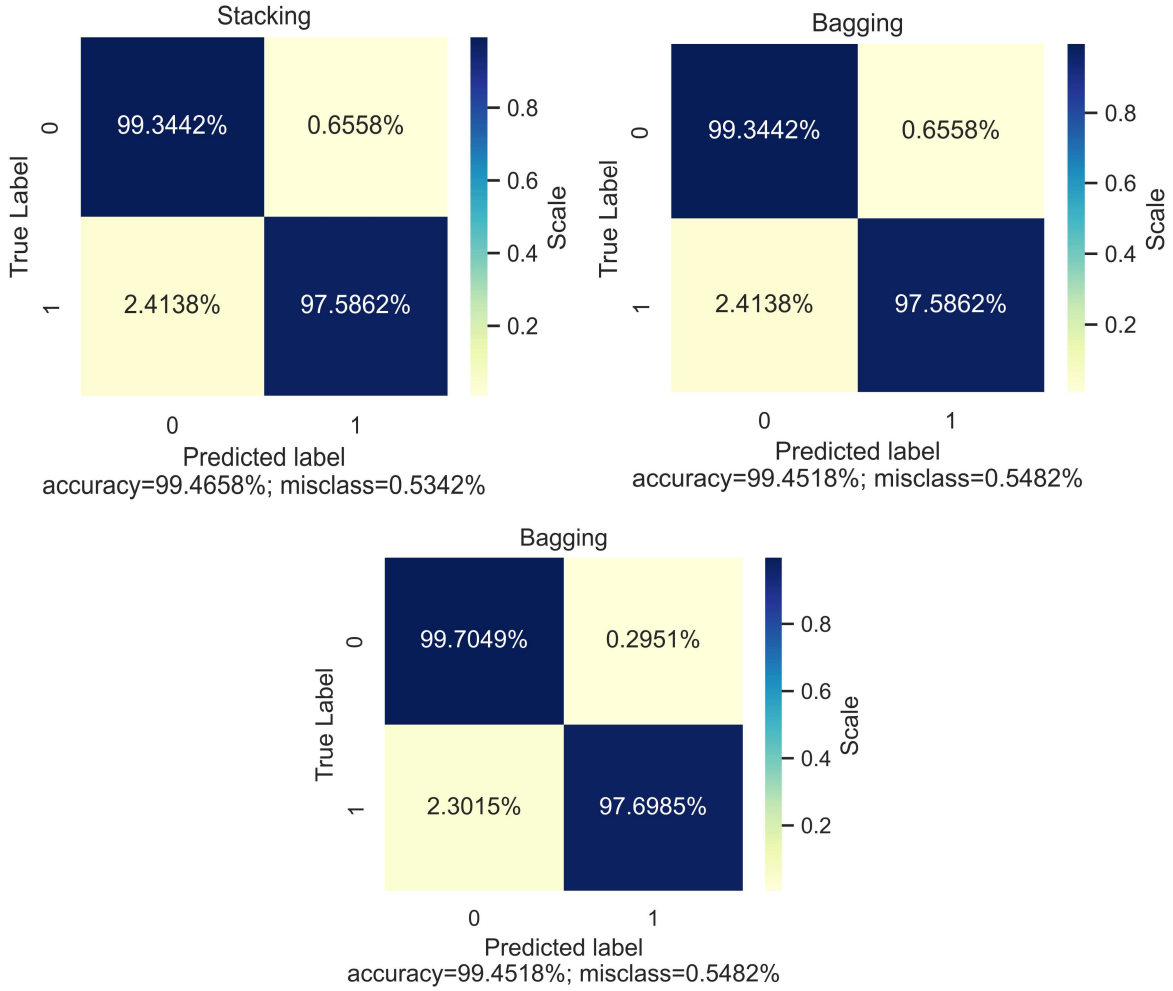

Figure 10. Confusion Matrices of Ensemble Mechanisms.

6. Conclusions

Traditional security solutions are incapable of detecting advanced, sophisticated, and novel attacks, whereas ensemble learning models have demonstrated promising outcomes in predicting new unknown attack vectors. We have proposed an architecture of an ensemble model using several classification algorithms incorporated into various system architectures. The UNSW-NB15 dataset has been used to evaluate the proposed system. The selected algorithms have been combined together and integrated into different strategies to construct an ensemble model. The experimental findings demonstrate that ensemble methods outperform individual algorithms achieving more than 99% accuracy in all settings. Stable predictions and more balanced data are also obtained and that incorporates effectively in generating accurate models. The study has also added a new source of analysis and applications of an ensemble model in IDS design methodologies. The selection of ML algorithms to construct ensemble learning models is one of the most significant issues that require comprehensive experimental assessments. The challenge is to combine base classifiers that are able to improve different aspects of the learning process. In the future, we will explore how to integrate optimization models to construct ensemble learning, in addition, to applying the constructed model to additional datasets.

Author Contributions: Conceptualization, F.A.; Methodology, A.A.; Validation, F.A.; Formal analysis, F.A.; Investigation, A.A.; Resources, A.A.; Writing—original draft, A.A.; Writing—review & editing, F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Deanship of Scientific Research at Jouf University through the Fast-track Research Funding Program.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Albasheer, H.; Siraj, M.M.; Mubarakali, A.; Tayfour, O.E.; Salih, S.; Hamdan, M.; Khan, S.; Zainal, A.; Kamarudeen, S. Cyber-Attack Prediction Based on Network Intrusion Detection Systems for Alert Correlation Techniques: A Survey. *Sensors* **2022**, *22*, 1494. [CrossRef] [PubMed]
2. Alqahtani, H.; Sarker, I.H.; Kalim, A.; Minhaz Hossain, S.M.; Ikhlaq, S.; Hossain, S. Cyber intrusion detection using machine learning classification techniques. In Proceedings of the Computing Science, Communication and Security: First International Conference, COMS2 2020, Gujarat, India, 26–27 March 2020; Springer: Singapore, 2020; pp. 121–131.
3. Pradeep, K.R.; Gowda, A.S.; Dakshayini, M. Hybrid-Network Intrusion Detection (H-NID) Model Using Machine Learning Techniques (MLTs). In *Lecture Notes in Electrical Engineering*; Kumar, A., Senatore, S., Gunjan, V.K., Eds.; ICDSMLA 2021; Springer: Singapore, 2023; Volume 947. [CrossRef]
4. Alzahrani, A.O.; Alenazi, M.J.F. Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks. *Future Internet* **2021**, *13*, 111. [CrossRef]
5. Sathya, R. Ensemble Machine Learning Techniques for Attack Prediction in NIDS Environment. *Iraqi J. Comput. Sci. Math.* **2022**, *3*, 78–82.
6. Abirami, M.S.; Yash, U.; Singh, S. Building an ensemble learning based algorithm for improving intrusion detection system. In *Artificial Intelligence and Evolutionary Computations in Engineering Systems*; Springer: Singapore, 2020; pp. 635–649.
7. KDD Cup 1999. Available online: http://Kdd.Ics.Uci.Edu/Databases/Kddcup99.html (accessed on 10 October 2023).
8. Polikar, R. Ensemble learning. In *Ensemble Machine Learning*; Springer: Boston, MA, USA, 2012; pp. 1–34.
9. NSL-KDD Dataset. Available online: http://nsl.cs.unb.ca/nsl-kdd/ (accessed on 10 October 2023).
10. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 MilCIS*; IEEE: Piscataway, NJ, USA, 2015; pp. 1–6. [CrossRef]
11. Moustafa, N.; Creech, G.; Slay, J. Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models. In *Data Analytics and Decision Support for Cybersecurity*; Springer: Cham, Switzerland, 2017; pp. 127–156.
12. Sarhan, M.; Layeghy, S.; Moustafa, N.; Portmann, M. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. In *Big Data Technologies and Applications: 10th EAI International Conference, BDTA 2020, and 13th EAI International Conference on Wireless Internet, WiCON 2020, Virtual Event, 11 December 2020, Proceedings*; Springer Nature: Berlin/Heidelberg, Germany, 2021; p. 117.
13. Jmila, H.; Khedher, M.I. Adversarial machine learning for network intrusion detection: A comparative study. *Comput. Netw.* **2022**, *214*, 109073. [CrossRef]
14. Otoum, Y.; Wan, Y.; Nayak, A. Transfer learning-driven intrusion detection for Internet of Vehicles (IoV). In Proceedings of the 2022 International Wireless Communications and Mobile Computing (IWCMC), Dubrovnik, Croatia, 30 May–3 June 2022; pp. 342–347.
15. Rashid, A.; Siddique, M.J.; Ahmed, S.M. Machine and Deep Learning Based Comparative Analysis Using Hybrid Approaches for Intrusion Detection System. In Proceedings of the 2020 3rd International Conference on Advancements in Computational Sciences (ICACS), Lahore, Pakistan, 17–19 February 2020; pp. 1–9.
16. Salih, A.; Zeebaree, S.T.; Ameen, S.; Alkhyyat, A.; Shukur, H.M. A Survey on the Role of Artificial Intelligence, Machine Learning and Deep Learning for Cybersecurity Attack Detection. In Proceedings of the 2021 7th International Engineering Conference "Research & Innovation amid Global Pandemic" (IEC), Erbil, Iraq, 24–25 February 2021; pp. 61–66. [CrossRef]
17. Tama, B.A.; Lim, S. Ensemble learning for intrusion detection systems: A systematic mapping study and cross-benchmark evaluation. *Comput. Sci. Rev.* **2021**, *39*, 100357. [CrossRef]
18. Kumar, G.; Thakur, K.; Ayyagari, M.R. MLEsIDSs: Machine learning-based ensembles for intrusion detection systems—A review. *J. Supercomput.* **2020**, *76*, 8938–8971. [CrossRef]
19. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An adaptive ensemble machine learning model for intrusion detection. *Ieee Access* **2019**, *7*, 82512–82521. [CrossRef]
20. Hossain, M.A.; Islam, M.S. Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. *Array* **2023**, *19*, 100306. [CrossRef]
21. Abbas, A.; Khan, M.A.; Latif, S.; Ajaz, M.; Shah, A.A.; Ahmad, J. A New Ensemble-Based Intrusion Detection System for Internet of Things. *Arab. J. Sci. Eng.* **2021**, *47*, 1805–1819. [CrossRef]
22. Tomer, V.; Sharma, S. Detecting IoT Attacks Using an Ensemble Machine Learning Model. *Futur. Internet* **2022**, *14*, 102. [CrossRef]

23. Almomani, A.; Akour, I.; Manasrah, A.M.; Almomani, O.; Alauthman, M.; Abdullah, E.; Al Shwait, A.; Al Sharaa, R. Ensemble-Based Approach for Efficient Intrusion Detection in Network Traffic. *Intell. Autom. Soft Comput.* **2023**, *37*, 2499–2517. [CrossRef]
24. Alotaibi, Y.; Ilyas, M. Ensemble-Learning Framework for Intrusion Detection to Enhance Internet of Things' Devices Security. *Sensors* **2023**, *23*, 5568. [CrossRef] [PubMed]
25. Luo, C.; Tan, Z.; Min, G.; Gan, J.; Shi, W.; Tian, Z. A Novel Web Attack Detection System for Internet of Things via Ensemble Classification. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5810–5818. [CrossRef]
26. Abu Al-Haija, Q.; Al-Badawi, A. Attack-Aware IoT network traffic routing leveraging ensemble learning. *Sensors* **2021**, *22*, 241. [CrossRef] [PubMed]
27. Naz, N.; A Khan, M.; Alsuhibany, S.A.; Diyan, M.; Tan, Z.; Khan, M.A.; Ahmad, J. Ensemble learning-based IDS for sensors telemetry data in IoT networks. *Math. Biosci. Eng.* **2022**, *19*, 10550–10580. [CrossRef] [PubMed]
28. Khan, F.; Jan, M.A.; Alturki, R.; Alshehri, M.D.; Shah, S.T.; Rehman, A.U. A Secure Ensemble Learning-Based Fog-Cloud Approach for Cyberattack Detection in IoMT. *IEEE Trans. Ind. Informatics* **2023**, *19*, 10125–10132. [CrossRef]
29. Ahmad, Z.; Khan, A.S.; Shiang, C.W.; Abdullah, J.; Ahmad, F. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Trans. Emerg. Telecommun. Technol.* **2020**, *32*, e4150. [CrossRef]
30. Ahmad, R.; Alsmadi, I. Machine learning approaches to IoT security: A systematic literature review. *Internet Things* **2021**, *14*, 100365. [CrossRef]
31. Rani, M. Gagandeep Effective network intrusion detection by addressing class imbalance with deep neural networks multimedia tools and applications. *Multimedia Tools Appl.* **2022**, *81*, 8499–8518. [CrossRef]
32. Yang, Z.; Liu, X.; Li, T.; Wu, D.; Wang, J.; Zhao, Y.; Han, H. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Comput. Secur.* **2022**, *116*, 102675. [CrossRef]
33. Al-Daweri, M.S.; Ariffin, K.A.Z.; Abdullah, S.; Senan, M.F.E.M. An Analysis of the KDD99 and UNSW-NB15 Datasets for the Intrusion Detection System. *Symmetry* **2020**, *12*, 1666. [CrossRef]
34. Cohen, I.; Huang, Y.; Chen, J.; Benesty, J.; Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 1–4.
35. Changyong, F.E.N.G.; Hongyue, W.A.N.G.; Naiji, L.U.; Tian, C.H.E.N.; Hua, H.E.; Ying, L.U. Log-transformation and its implications for data analysis. *Shanghai Arch. Psychiatry* **2014**, *26*, 105.
36. Alpaydin, E. *Introduction to Machine Learning*; MIT Press: Cambridge, CA, USA, 2020.
37. Dietterich, T.G. Machine-learning research. *AI Mag.* **1997**, *18*, 97.