Frontiers in Computer Science

TYPE Original Research
PUBLISHED 20 August 2025
DOI 10.3389/fcomp.2025.1623375

OPEN ACCESS

EDITED BY
Nicholas Kolokotronis,
University of Peloponnese, Greece

REVIEWED BY
Santosh I. Gore,
Sai Info Solution, India
Fadi Wedyan,
Lewis University, United States

*CORRESPONDENCE
Khawla Ali Maodah
✉ khawlaa800@gmail.com

RECEIVED 05 May 2025
ACCEPTED 08 July 2025
PUBLISHED 20 August 2025

CITATION
Maodah KA, Alhomdy S and Thabit F (2025) Detecting intrusions in cloud-based ensembles: evaluating voting and stacking methods with machine learning classifiers. *Front. Comput. Sci.* 7:1623375.
doi: 10.3389/fcomp.2025.1623375

Detecting intrusions in cloud-based ensembles: evaluating voting and stacking methods with machine learning classifiers

Khawla Ali Maodah[1]*, Sharaf Alhomdy[1] and Fursan Thabit[2]

[1]Department of Information Technology, Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen, [2]Department of Computer Engineering, Faculty of Engineering, Ege University, Bornova, Türkiye

Introduction: Cloud computing has revolutionized how organizations manage their infrastructure by providing scalable, on-demand services. However, the dispersed and open nature of cloud systems exposes them to a wide spectrum of cyberattacks. Machine learning provides dynamic options for detecting known and unknown assaults, whereas typical intrusion detection systems that depend on signature or rule-based techniques find it difficult to adjust to complex cyber threats.

Methods: This study compares the efficacy of an ensemble approach (Voting Hard and Stacking) for intrusion detection in cloud environments with individual machine learning classifiers, such as Random Forest, Decision Tree, Gradient Boosting, XGBoost, Naive Bayes, Support Vector Machine, and Logistic Regression. The study uses the NSL-KDD dataset.

Results: The results show show that while standalone models perform well, the ensemble technique offers better accuracy (almost 100%) and resilience across precision, recall, and F1-score measures. Furthermore, it is shown via feature selection methods (Random Forest, Gain Information, and Manual Selection) that the ensemble model performs consistently even when feature sets are smaller.

Discussion: These findings highlight how both individual and group Machine learning approaches may be used to improve Intrusion detection systems for cloud infrastructures, providing implementation flexibility according to threat landscapes and computing limitations.

KEYWORDS

cloud computing, machine learning, voting, stacking, intrusion detection system, NSL-KDD dataset

1 Introduction

Cloud computing (CC) is a contemporary strategy that gives users access to pre-configured, task-specific, and externally managed internet-based resources. Most businesses increasingly rely on these externally maintained services to fulfill particular goals like operating apps, handling certain data activities, or enhancing existing systems, rather than having complete infrastructure management. In contrast to conventional configurations, cloud services limit direct management but provide remote use of certain tools provided by cloud service providers (CSPs), contingent on network access and availability (Dattangire et al., 2024). Whereas before it would just be present as a convenience or access-by-speed device, cloud computing is now an operational and resource management strategy because of the increasing reliance on centralized digital infrastructure. Consumer-accessible services that offer consumers limited access to necessary resources are normally run internally or by trusted partners.

However, cloud infrastructures' interconnection and complexity inadvertently lay open vulnerabilities, hence the possibility of security vulnerability exploitation. As a countermeasure to these challenges (Al-Sharif and Bushnag, 2024). Whereas intrusion detection systems (IDSs) are usually used to stay up to date with regard to network activity, they tend to be more event-focused on logging instead of proactive intrusion prevention. When confronted with new attack patterns, traditional IDSs, which generally make use of preconfigured rules or signatures, would overfit known threats and tend to produce a high false alarm rate. Furthermore, the inability of their responses to keep up with changing threat environments is due to the static nature of traditional IDSs (Al-Sharif and Bushnag, 2024).

ML-based systems are less reliant on human maintenance and are more resilient to as-yet unknown threats because they adapt to evolving habits over time rather than relying merely on pre-determined signatures (Adhikari and Bal, 2023). ML has been utilized mostly as an auxiliary method to cloud infrastructure security management for the last few years. Even if they do not necessarily surpass conventional methods in terms of precision, ML algorithms employ training on actual real-world data samples in order to aid in the identification of patterns of cloud-based threats rather than completely automating threat identification (Dattangire et al., 2024).

Despite CC being on-demand and dynamic in service, its widespread and expansive nature can make it cumbersome to ensure sustained security. Traditional IDS based on predefined rules or signatures can be lacking.

in the case of changing and cunning attack styles, opening vulnerabilities. Sophisticated and context-aware security solutions are required to counter this. ML methods can be used to identify threats by examining traffic patterns in the network, but their deployment in cloud-based IDS systems is limited and needs to be improved.

In order to improve threat awareness in cloud computing, this study focuses on putting forth a cooperative approach in which IDS facilitates ML-based analysis. When managing different kinds of network risks, the goal is to minimize mistakes, cut down on detection delays, and preserve performance. The experimentation in this study is based on the NSL-KDD dataset.

1.1 Problem statement

Whereas cloud computing is becoming a growing trend due to its remote availability and simple services, control and data management concerns are invited by its reliance on outside providers. Despite their best effort to provide accessibility and efficiency, cloud systems may inadvertently expose themselves to threats due to their complexity and large distribution. Traditional IDSs are skewed toward familiar attack patterns, which limits their capacity to identify unexpected behavior. IDSs are often employed to audit activity rather than to prevent attacks actively. Although ML is commonly proposed as a means of improving threat detection, it might not always yield stable performance in various situations or real-time reliability. Furthermore, the extent to which ML and IDS can be integrated is still limited, and thus the issues involved in converging automation and adaptive security mechanisms are far from being completely addressed.

1.2 Objective

The objective of this work is to develop an ensemble-driven intrusion detection model that integrates ML techniques into traditional IDS for enhancing cloud computing environment security. It increases the detection accuracy of known and unknown attacks, reduces false alarms, and supports timely and adaptive reactions to novel attack patterns.

1.3 Contributions

1.3.1 Proposed ensemble model

The ensemble strategy proposed in this study greatly improves cyber threat detection and mitigation for cloud environments by integrating ML methods effectively with IDS. With the use of ensemble ML techniques with Voting hard and Stacking techniques, which outperform single classifiers in terms of accuracy, precision, recall, and F1-score metrics, the model will enhance IDS' capability to detect known and new attacks in real time.

1.3.2 Comprehensive classifier evaluation

Moreover, the analysis of the effectiveness of several ML classifiers, such as RF, XGBoost, SVM, DT, GB, LR, and NB, in identifying security threats related to cloud computing. The NSL-KDD dataset was used to assess the classifiers using measures including accuracy, precision, recall, and F1-score.

1.3.3 Practical insights for cloud security

Providing practical insights into how integration of ML and IDS may improve cloud computing security. The results emphasize the potential of the ensemble method in creating more resilient and flexible security systems capable of identifying and countering emerging threats.

1.4 The structure of the paper

The remainder of this paper is structured as follows: section 2 contains background, which presents an introduction to cloud computing, IDSs, and ML. Section 3 discusses related work. Section 4 offers an overview of the proposed model, including the NSL-KDD dataset, feature selection methods, classifiers employed, performance metrics for assessment, and experimental framework. Section 5 contains the results and analysis. Finally, section 6 concludes conclusion and future work.

2 Background

2.1 Cloud computing

Cloud computing is the delivery of information technology services such as servers, storage, databases, software, network management, and artificial intelligence (AI) via the Internet. It allows for speedier, more flexible, and cost-effective solutions than traditional techniques, resulting in improved productivity and efficiency. There are several sorts of cloud infrastructures, each with its design and development. Various models, varieties, and services have evolved to meet distinct demands. Cloud services may be delivered in three ways: public, private, and ensemble (Megouache et al., 2024; Zulifqar et al., 2021).

The public cloud is an example of a computer platform that restricts customization and is run by companies that supply standardized services to a large customer base. This group of services, which rely on shared infrastructure, includes Microsoft Azure, Google App Engine, and Amazon EC2. On the other hand, the private cloud is designed for internal usage only, sometimes at the expense of more stringent data isolation. It is appropriate for businesses with certain operating needs. Microsoft ECI data centers, Ubuntu Enterprise Cloud, and Amazon Virtual Private Cloud are a few examples. Organizations with similar operational objectives employ the community cloud, which divides duties among several users and might result in less consistent administration (Saran et al., 2022). New complexity brought forth by cloud technology's ongoing development put both individuals' and enterprises' current security standards to the test. ML is being investigated as a supplemental technique to help conventional security measures rather than providing a comprehensive answer. Though their function is typically reactive rather than entirely preventive, ML techniques are frequently used in cloud systems to help identify trends and abnormalities (Nassif et al., 2021).

2.2 Intrusion detection system

An IDS is a part of the system that watches data flow and system activities passively without taking any active action to prevent problems. It detects abnormalities that can indicate technical problems or unusual use rather than explicitly recognizing malicious activities or breaches. Instead of sending out instant notifications, it creates records when it notices specific trends. By providing insights rather than immediately addressing threats, an IDS indirectly enhances network security in contrast to proactive technologies like firewalls, antivirus software, or access control systems (Umar et al., 2024; Saranya et al., 2020). Instead of active defensive techniques like honeypots, IDSs are frequently coupled with monitoring tools. Network intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS) are the two broad categories of IDS development that are often recognized. With host-level modifications, HIDS mainly examines internal operations and file integrity on a single system. However, NIDS may miss host-specific events when monitoring network traffic. NIDS detection usually uses statistical deviations or pre-established procedures to identify anomalies (Useni et al., 2023). IDS may be broadly divided into two categories: pattern-matching and behavior-based. Although it may not always employ ML or DL algorithms, behavior-based IDSs rely on departures from predicted activities. However, instead of employing dynamic analysis, pattern-matching IDSs use a database of recognized behaviors to identify matches. Some IDS systems function in intervals rather than continuously, alerting users only when certain thresholds or circumstances are reached.

IDS systems are divided into two types based on their response mechanism:

- Active IDS detects threats but also takes preventative steps such as blocking suspicious traffic to avoid prospective assaults.
- Passive IDS: This kind just monitors and analyzes traffic, alerting the administrator to discovered threats and potential vulnerabilities but without taking active action against them (Useni et al., 2023; Hidayat et al., 2023).

2.3 Machine learning

ML is a subset of AI that allows systems to learn and develop automatically based on experience rather than explicit programming. In IDSs, ML algorithms identify intrusions in huge datasets more quickly. ML algorithms are often classified into three categories. Supervised, Unsupervised, Semi-supervised (Saranya et al., 2020). Supervised ML approaches use labeled datasets to train algorithms to spot patterns of normal and aberrant behavior. Common intrusion detection algorithms include LR, Gaussian NB, RF, MLP Classifier, KNN, DT, AdaBoost, XGBoost, and LightGBM. Unsupervised ML approaches use unlabeled data and focus on detecting deviations from usual behavior. Clustering comparable data points and detecting abnormalities is accomplished using algorithms such as K-means and Gaussian Mixture Model (GMM). Semi-supervised ML algorithms use labeled and unlabeled data to increase detection accuracy. They can detect anomalies that are similar to tagged cases, even if they do not completely match. These strategies attempt to improve the performance of anomaly-based IDSs, hence leading to greater computer network security (Adhikari and Bal, 2023; Nassif et al., 2021; Parameswarappa et al., 2023).

3 Related work

The related papers listed below examine a range of IDS techniques and methodologies, including IDS types and ML algorithms, with a focus on cloud environments.

Mghames and Ibrahim (2023) developed an IDS based on ML to identify Distributed Denial of Service (DDoS) attacks. They performed training and testing in the CIC-IDS-2018 dataset using five machine-learning methods: DT, RF, LR, SVM, and multi-layer neural network. They applied PCA to diminish dimensions to improve performance. The multi-layer neural network showed better performance than any other model, achieving a classification accuracy of 99.9992% to identify DDoS attacks.

Eluri et al. (2024) addressed the issue of detecting disruptions in organizational networks by the definition of network activity as normal or abnormal and striving to rectify misclassification. Two strong algorithms of data mining, SVM, DT, and K-Means, were utilized by them to optimize the organization of the data. This method was developed and tested with the assistance of the KDDCUP99 dataset. The findings revealed that the new approach was more accurate and faster in comparison to previous methods, which suggests that it is particularly effective in new attack detection.

Vibhute et al. (2024) highlighted cloud data security by designing an IDS from the popular NSL-KDD dataset. The ensemble learning-based RF model was designed to identify the most important features. The system categorized and identified network intrusions with three ML models: SVM, LR, and KNN, and validation accuracies of 87.58, 88.86, and 98.24%, respectively. The suggested method has proved to be effective in identifying cyberattacks in real time.

John et al. (2024) proposed an IDS that detects illegal access and initiates pre-defined actions to enable secure data transfer using networks. They recognized several problems for IDS models like low detection accuracy and excessive false alarms, which are usually caused by excessive feature dimensionality and class imbalances in network traffic datasets. To address these issues, the study used Principal Component Analysis (PCA) and ensemble ML techniques. AdaBoost was used to overcome the disadvantages of PCA, such as feature selection; LogitBoost was used for multiclass categorization and logistic loss reduction; and RandomForest was used for curvy overfitting reduction. The performance, as well as the WSN-DS, NSL-KDD, and UNSW-N15 datasets, indicated that PCA integrated with RandomForest achieved 100% accuracy on all datasets. PCA with AdaBoost scored 92.3, 89.0, and 67.9% on WSN-DS, NSL-KDD, and UNSW-N15, respectively, and PCA with LogitBoost scored 98.9, 100, and 88.7% on the said datasets.

Vibhute et al. (2024) experimented with cloud data protection through a network IDS with the commonly used NSL-KDD dataset. They proposed an RF method where ensemble learning was employed to identify the most important features. The system detected and identified network intrusions using three ML models: SVM, LR, and K-nearest neighbors (KNN) with validation accuracy of 87.58, 88.86, and 98.24%, respectively. The presented method has been promising for real-time detection of cyberattacks.

John et al. (2024) presented an IDS that is capable of detecting unauthorized access and initiating pre-defined actions to support secure data sharing across networks. They realized there were many challenges for IDS models, including low detection rates and high false positives, due to frequent occurrences of high feature dimensionality and class imbalances in network traffic datasets. To tackle these challenges, the research work utilized Principal Component Analysis (PCA) and ensemble ML techniques. AdaBoost was utilized to remedy PCA's limitation in feature selection; LogitBoost was utilized for multiclassing and for reducing logistic loss; and RandomForest was utilized for overfitting reduction. Comparison against the WSN-DS, NSL-KDD, and UNSW-N15 datasets through evaluation showed that PCA integrated with RandomForest outperformed all the datasets with 100% accuracy. PCA with AdaBoost was able to get 92.3, 89.0, and 67.9% accuracy on WSN-DS, NSL-KDD, and UNSW-N15, respectively. PCA with LogitBoost was able to get 98.9, 100, and 88.7% accuracy on the same datasets.

Attou et al. (2023) suggested a cloud-based IDS that monitors resources, services, and networks for suspicious activities. They combined an RF classifier with feature engineering methods to improve the detection model's accuracy. The model was tested on the Bot-IoT and NSL-KDD datasets, and it achieved 98.3 and 99.99% accuracy, respectively. The findings confirmed the model's exceptional performance in terms of accuracy, precision, and recall, outperforming prior studies in the field.

Al-Sharif and Bushnag (2024) established an IDS framework for handling security challenges in cloud settings, where standard IDS solutions frequently fail owing to increased complexity and numerous attack vectors. Instead of using a single powerful classifier, they suggested a collective learning approach that combines numerous weaker models to create a more reliable detection system. Their strategy used bagging with Random Forest as the principal model and compared its efficacy to three boosting variants: Ensemble AdaBoost, Ensemble LPBoost, and Ensemble RUSBoost. Evaluations were conducted utilizing several divisions of the CICID2017 dataset. Among the investigated models, Ensemble RUSBoost had the greatest average accuracy at 99.821%, while the bagging approach performed particularly well on the DS2 subgroup, with an accuracy of 99.997%. To further test their technique, the researchers compared their model to an existing solution, emphasizing its comparative benefits and enhanced detection capacity.

Mehmood et al. (2023) proposed an ML-based method for detecting insider actions in cloud settings, with a focus on recognizing privilege misuse instances. They used a mixed-learning framework to improve detection reliability, including many models such as Random Forest, AdaBoost, XGBoost, and LightGBM. Testing was done on a customized version of the CERT dataset, and LightGBM surpassed the others with a peak accuracy of 97%. XGBoost and AdaBoost followed closely, with 88.27 and 88% accuracy, respectively, while Random Forest achieved 86%. Their findings showed that using several models in tandem improves the system's capacity to detect various insider threat behaviors.

Akinbolaji (2023) studied the use of sophisticated AI and ML technologies to improve real-time monitoring systems in cloud settings. Their study analyzed current detection frameworks using both statistical and descriptive research tools, as well as newer AI approaches such as deep learning and reinforcement learning, to improve detection efficiency and precision. The results showed that the suggested ensemble model outperformed traditional techniques, with at least a 30% improvement in detecting abnormalities and threats. This study emphasizes the importance of AI in enhancing digital security mechanisms, ensuring data integrity, and assisting compliance efforts, while also laying the way for future advances in cloud-based threat prevention systems.

The study in Devi and Jain (2024) examines the issues of protecting privacy and safeguarding data in cloud computing settings, which are particularly vulnerable owing to their dispersed nature. Instead of traditional procedures, the authors advocated using deep learning to improve intrusion detection technologies. They looked at a variety of IDS frameworks and emphasized the need for high-quality datasets in optimizing the training and assessment stages of these models. The goal was to improve the efficacy of IDS systems that operate in both real-time and batch modes by using sophisticated deep-learning algorithms. The findings demonstrated how challenging it may be to identify anomalous behavior when training data does not exhibit these patterns. In detecting anomalous activity across many categorization groups, the study showed that algorithms such as Soft-Max Regression (SMR) and STL-based feature learning outperformed 98% accuracy, indicating encouraging developments in cloud defensive mechanisms.

Sundaramoorthy et al. (2024) discussed an ensemble IDS system with an emphasis on improved security in cloud-based infrastructure and wireless sensor networks. The system includes various techniques, including ISSIR for optimal feature selection, OSVM for classification error reduction, ELSTM for pattern anomaly detection, and MLPNN for threat response. The resulting ensemble technique recorded a staggering 99.9% accuracy rate, outperforming earlier systems. This unification further improves the IDS performance, corrects serious weaknesses in cloud and

WSN networks, and contributes substantially to cybersecurity technique advancements.

Megouache et al. (2024) proposed a strong approach to identifying attacks on cloud data with emphasis on special challenges due to dispersed nature, scalability problems, and restricted resources available in cloud systems. Unlike traditional models, their approach used unclassified data instead of pre-labeled inputs. They applied k-means clustering to label the raw data, which was further applied to train an Extreme Learning Machine (ELM) classifier for threat identification. Using the KDD99 benchmark dataset, they were successful in showing that their approach provided high accuracy as well as reduced processing time significantly. The method proved to be a good alternative for complementing cloud protection systems, with uniform detection results.

In contrast to other studies, the study uses the NSL-KDD dataset to provide a comprehensive evaluation of many individual classifiers as well as a hard voting ensemble approach designed for cloud-based systems. While the majority of current methods focus on a single model or dataset, the technique combines feature selection with ensemble learning to increase accuracy and outcomes in F1 score, precision, and recall. Whereas previous efforts focus solely on insider threats or DDoS assaults, the approach can detect both known and unknown attack types. In addition, we demonstrate that the ensemble approach overcomes computational efficiency issues in real-time cloud IDSs by maintaining good detection performance even with smaller feature dimensions.

4 Proposed model

This section describes an integrated ML method -IDS model (as shown in Figure 1) that uses the NSL-KDD dataset to detect abnormal

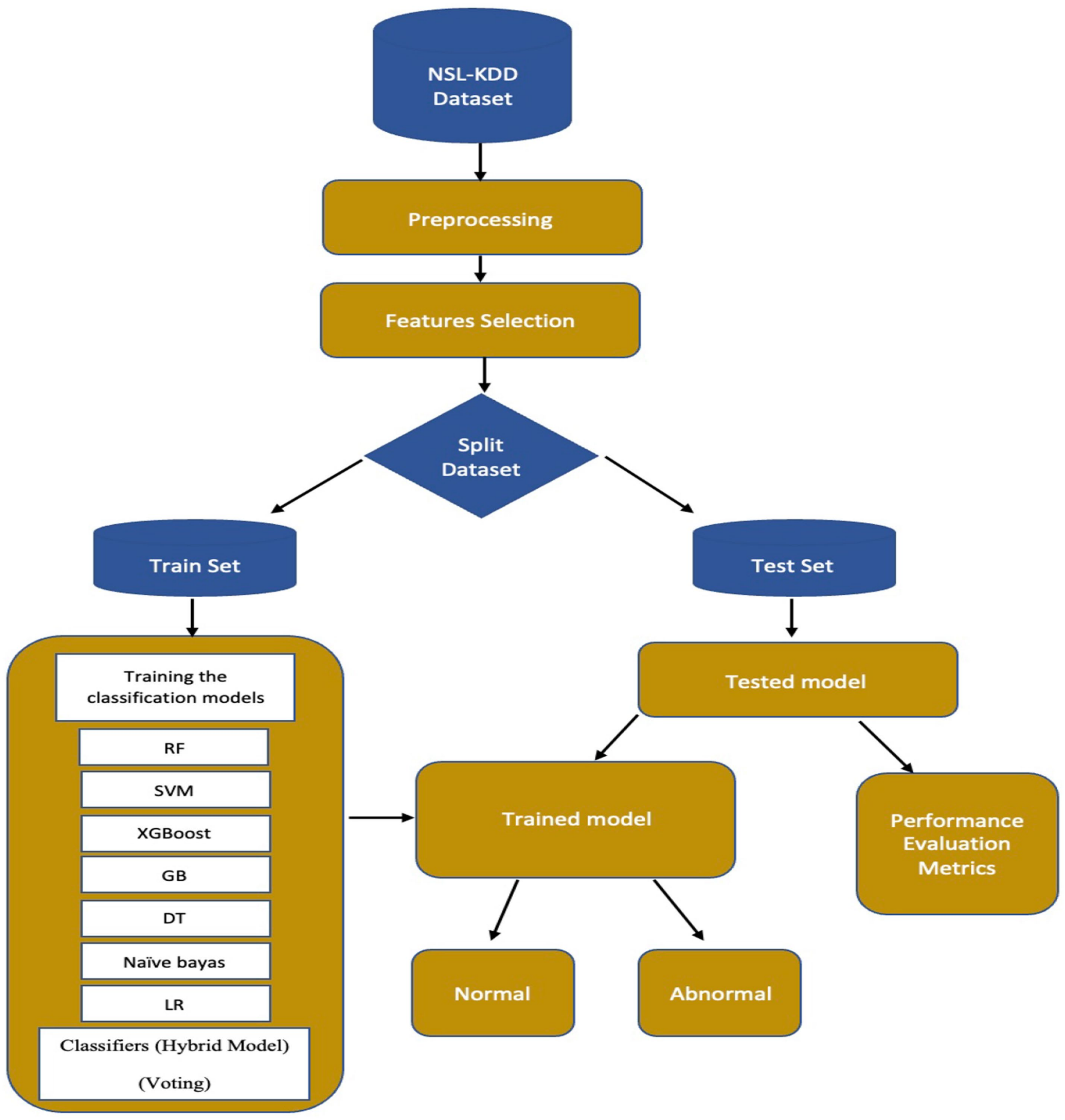

FIGURE 1
ML-based IDS flowchart for NSL-KDD.

network traffic in cloud settings. The approach involves data pretreatment (normalization and encoding), RandomOverSampler to solve class imbalances, and feature selection techniques (complete feature set, Information Gain, Mutual Information, and RF-based significance). The dataset is then divided into training and test sets. To increase detection accuracy, several classifiers (RF, DT, SVM, XGBoost, LR, GB, and Naïve Bayes) are trained and merged using ensemble approaches such as voting hard and Stacking. To categorize traffic as normal or abnormal, models are assessed using standard metrics such as accuracy, precision, recall, F1-score, and false positives/negatives, hence enhancing threat detection in cloud environments.

4.1 NSL-KDD dataset

The KDDcup99 dataset was generated for the Third International Knowledge Discovery and Data Mining Tools Competition, which took place in 1999 in connection with KDD-99, the Fifth International Conference on Knowledge Discovery and Data Mining. This dataset contains over 5 million training samples and over 2 million testing samples. It also has a high number of duplicate entries and unbalanced class distributions (Umar et al., 2024). The NSL-KDD dataset is based on the KDD Cup '99 dataset and solves flaws identified in the original, such as duplicated entries in the training set and duplicate records in the test set (Protić and Stanković, 2023). The NSL-KDD dataset has 41 attributes: three are categorical, four are binary, and the remaining 34 are continuous. The training set includes 23 traffic types, whereas the testing set has 30. The assaults in this dataset are divided into four categories: DOS, probing, U2R, and R2L. The features are divided into three categories: (1) fundamental features, (2) content-based features, and (3) traffic-related features (Alkadi et al., 2023).

4.2 Data preprocessing

Data preparation is critical in converting raw information into a format that enhances the performance of ML models. Raw data frequently contains missing, null, or inconsistent values, as well as unused or duplicated fields that provide no useful information. To solve this, the process starts by identifying and eliminating inconsistencies in the data to ensure it is clean. The following stages describe how the dataset was prepared for ML applications:

- The preparation begins with the load _data function, which imports the text file and converts it into a pandas DataFrame with easy-to-understand column names.
- Clean up categorical columns such as protocol_type, service, and flag by removing unnecessary spaces and newline characters using the clean_column_values function.
- Following cleaning, one-hot encoding via Pandas is used to convert these category characteristics into a numerical representation. Get-dummies (), which qualifies them for use in ML models.
- The normalize_data function uses StandardScaler to standardize the dataset, bringing its mean down to zero and its standard deviation up to one, in order to further improve speed.
- The dataset is then balanced by using RandomOverSampler from the learning module, which helps to lessen prediction bias by producing extra samples for the underrepresented class.
- The data is optimized for ML through the processes of cleaning, encoding, scaling, and balancing, which raises the precision and dependability of model predictions.

4.3 Feature selection

Choosing the pertinent attributes comes next after data analysis. By determining the most important variables, feature selection improves the accuracy of ML models, which is why it is so important in intrusion detection (Vibhute et al., 2024). The process of feature subset selection aims to improve classification performance by selecting a smaller set of features from a larger pool. While some researchers describe feature subset selection (FSS) as a technique for decreasing the size of the feature set, others view it as a means of improving classifier accuracy. FSS is thought to be a method for finding the best feature subsets that, by removing unnecessary and duplicated features, preserve the most important information in a dataset (Alhayali et al., 2021). The most significant characteristics are found using feature selection approaches, which also remove redundant or superfluous data and increase the dataset's overall effectiveness (Vibhute et al., 2024). In the beginning, the study employed all of the features without using any feature selection techniques. Following that, two filter-based feature selection strategies were presented: Mutual Information and Gain Information. These methods rank the features according to their scores, which assess each feature's importance and pertinence to the class label. The characteristics with the highest ranking are then used to create the IDS. Filter approaches have the advantage of being computationally efficient because they do not need IDS training (Alalhareth and Hong, 2023). Filters are employed in the popular feature selection method known as Gain Information. It reduces the impact of irrelevant data by ranking characteristics according to their significance. By doing so, it discovers the characteristics that give the most information to a certain class, improving the model's overall performance (Kurniabudi et al., 2020). In Gain Information (GI), the quantity of information is primarily assessed using the entropy and conditional entropy concepts. Firstly, a discrete random variable's entropy, "X, is defined as follows in Equation 1":

$$\mathrm{H}(X) = -\sum_{x_i \in X} p(x_i)\log\left(p(x_i)\right) \tag{1}$$

Here, a specific outcome of the random variable X is indicated by xi, and the probability that xi will occur among the potential values of X is shown by $P(xi)$. According to another discrete random variable Y, the conditional entropy of "X is defined as follows in Equation 2":

$$H(X/Y) = -\sum_{y_i \in Y} p(y_i) \sum_{x_i \in X} P(x_i / y_i)\log\left(P(x_i / y_i)\right) \tag{2}$$

The chance of seeing yj from Y is represented by $P(yj)$ in this instance, but the conditional probability of seeing xi from X provided that yj from Y has happened is shown by $P(xi|yj)$. The Gain Information (GI) between X and Y quantifies the amount of information communicated between both variables as follows in Equation 3:

$$IG(X,Y)=H(X)-H(X/Y)=\sum_{x_i\in X}\sum_{y_i\in Y}P(x_i,y_i)\log\left(\frac{P(x_i,y_i)}{P(x_i)p(y_i)}\right) \quad (3)$$

(xi, yj) refers to the joint probability of xi and yj occurring together (Zhang et al., 2024). Mutual Information (MI)-based feature selection is an approach for dimensionality reduction that works independently of classifiers. It seeks to address the difficulty of picking a meaningful collection of characteristics (Liu and Motani, 2022). Mutual information feature selection is a popular strategy for improving the effectiveness of IDSs. It assesses the link between each characteristic and the class label, deciding which features have the highest mutual information values (Alalhareth and Hong, 2023). The study used an RF classifier, which is an ensemble-based ML technique that integrates numerous DTs. To limit the danger of overfitting and improve the model's generalizability, each DT in the forest is built separately from a randomly selected subset of the training data and features. The RF algorithm's ultimate output is selected by a majority vote among all DTs. Each tree in the forest votes for the class of the input data point and the class that receives the most votes is chosen as the prediction (Ali et al., 2023). As shown in Table 1, multiple feature selection approaches discover critical properties for the IDS. The whole feature set has 41 characteristics, however, approaches like manual information acquisition and RF selection emphasize essential aspects like "src_bytes," "dst_host_serror_rate," and "count." These features serve to improve the IDS model's performance in identifying unusual network traffic.

4.4 Classification model

Support Vector Machine (SVM): is commonly regarded as one of the most successful algorithms for binary classification, notably in IDSs (IDS), where transactions are classified as normal or invasive (Aldallal and Alisa, 2021). The primary goal of SVM is to identify a hyperplane within an n-dimensional feature space that maximizes the separation margin between classes. One of the key advantages of SVM is its ability to work effectively with smaller training datasets, as it only relies on a few support vectors to define the hyperplane. However, SVM performance can be negatively impacted by noise near the hyperplane (Alotaibi and Rassam, 2023).

Gradient boosting (GB): is an ensemble learning method that combines many decision trees to increase prediction accuracy. This method builds decision trees progressively, with each node making a binary choice. The model's performance is progressively improved as each tree fixes the mistakes of the one before it. GB is an effective technique for challenging jobs because of its iterative approach, which produces forecasts that are more accurate (Boldini et al., 2023).

XGBoost: To improve efficiency and performance, XGBoost is a sophisticated and optimized variant of gradient boosting. It enhances the approach and system design of conventional GB models. Parallel processing, distributed computing, out-of-core execution, and cache optimization are some of the characteristics that XGBoost integrates to provide quicker processing and more accurate convergence to the global minimum. Its speed and accuracy are increased by these enhancements, which enable it to handle massive amounts of data across several devices (Thapa et al., 2020).

Logistic regression (LR): A classification method for categorical outcome prediction, logistic regression may be applied to both binary and multi-class classification applications. It uses the logistic function to determine the likelihood of an event happening, with values ranging from 0 to 1. Usually, a threshold of 0.5 is used to differentiate between two classes: values below 0.5 are categorized as class 0, whereas those over 0.5 are classified as class 1. F(x) = 1/(1 + e^-x) is the logistic sigmoid function, which is used to convert the input into a number between 0 and 1 that indicates the likelihood of a specific result. The wider application of the logistic sigmoid function in multi-class classification situations is not the same as this method (Somogyi, 2021).

Naïve Bayes (NB): is a variation of Bayes' Theorem in which the qualities are assumed to be extremely independent of one another. It is a classification approach based on Bayes' probability theory, with the assumption that the presence of one characteristic does not affect the chance of another (Devidas and Adesh, 2021). The Naïve Bayes method uses conditional probability and the premise that characteristics are independent. The classifier assigns the sample to the class with the highest probability after calculating the conditional probabilities for each class for each input (Useni et al., 2023).

TABLE 1 Feature selection for IDS using different methods.

Methods of features selection	Features
Full features	"duration," "protocol_type," "service," "flag," "src_bytes," "dst_bytes," "land," "wrong_fragment," "urgent," "hot," "num_failed_logins," "logged_in," "num_compromised," "root_shell," "su_attempted," "num_root," "num_file_creations," "num_shells," "num_access_files," "num_outbound_cmds," "is_host_login," "is_guest_login," "count," "srv_count," "serror_rate," "srv_serror_rate," "rerror_rate," "srv_rerror_rate," "same_srv_rate," "diff_srv_rate," "srv_diff_host_rate," "dst_host_count," "dst_host_srv_count," "dst_host_same_srv_rate," "dst_host_diff_srv_rate," "dst_host_same_src_port_rate," "dst_host_srv_diff_host_rate," "dst_host_serror_rate," "dst_host_srv_serror_rate," "dst_host_rerror_rate," "dst_host_srv_rerror_rate," "labels"
Selected features by manual information and gain information	"src_bytes," "flag_SH," "service_auth," "dst_host_diff_srv_rate," "dst_host_same_src_port_rate," "dst_host_srv_diff_host_rate," "dst_host_serror_rate," "dst_host_srv_serror_rate," "dst_host_rerror_rate," and "dst_host_srv_rerror_rate"
Selected features by RF	"src_bytes," "data_transfer," "same_srv_rate," "count," "dst_bytes," "dst_host_serror_rate," "dst_host_diff_srv_rate," "dst_host_srv_serror_rate," "srv_serror_rate," "error_rate," and "attack"

Decision Tree (DT): For classification problems, the DT method is frequently utilized. It organizes data in a tree structure, with classifications determined by decisions made at every level. The branches show the results of those tests, the leaf nodes show the final classification, and each non-terminal node denotes a test or decision point.

Random Forest (RF): For both classification and regression tasks, RF, an ensemble learning technique, is employed. During training, it creates numerous DTs and forecasts the class using the majority vote from each tree. This strategy seeks to decrease overfitting and underfitting by averaging predictions to strike a balance between bias and variance (Devidas and Adesh, 2021).

Ensemble Model: This ensemble model guarantees thorough and accurate threat detection by combining the various strengths of the individual classifiers to improve classification accuracy through the use of Voting Hard and Stacking with DT, RF, and GB.

4.5 Performance evaluation metrics

A number of essential criteria will be used to assess threat detection algorithms' performance. These measurements will provide quantitative insights into how well the models recognize and respond to threats in real-time settings (Akinbolaji, 2023).

4.6 The performance evaluation metrics can be expressed as accuracy, precision, recall, and F1-score

Accuracy: is the percentage of correctly categorized cases, which include both normal and pathological data points. as follows in Equation 4:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

Precision: refers to the fraction of forecasted threats that are valid. It is the proportion of accurately recognized anomalies to total presented anomalies. as follows in Equation 5:

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

Recall: The proportion of genuine positive records (e.g., true anomalies) that are accurately identified. It calculates the ratio of detected attacks to total attacks. as follows in Equation 6:

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

The F1 Score: is the harmonic mean of accuracy and recall, providing a single statistic for evaluating the model's overall efficacy. as follows in Equation 7:

$$F1 - Sccore = \frac{Recall * Precision}{Recall + Precision} \quad (7)$$

True Negative (TN) represents accurately anticipated normal cases.

False Negative (FN) indicates incidents that were mistakenly anticipated as normal.

True Positive (TP) represents successfully anticipated abnormal incidences.

False Positive (FP) indicates incidents that were wrongly anticipated as abnormal (John et al., 2024; Vibhute et al., 2024; Megouache et al., 2024).

4.7 Experimental environment

All of the study's experiments were carried out on a top-tier workstation running Windows 10 Pro and equipped with an AMD Ryzen 95,950X CPU (3.7 GHz) and 64 GB of RAM. This setup was intended to guarantee that repeated tests with various feature sets and numerous ensemble classifiers would go well. This configuration is not representative of situations with restricted resources, such as low-cost systems or edge devices. Future research will evaluate the effectiveness of the IDS on these platforms and investigate real-time optimization techniques including pruning, quantization, and compression.

5 Results and discussion

This section presents the results obtained from the ensemble model, which was evaluated using the NSL-KDD dataset's features and yields a total of 41 attributes as shown in Table 1 to assess the effectiveness of the integrated approach. It begins by analyzing the performance of various individual classifiers RF, DT, GB, XGBoost, NB, SVM, and LR, and the ensemble model using standard evaluation metrics, including accuracy, precision, recall, F1-score, and false positive/negative rates in three case that are full features, selection methods such as Information Gain, Mutual Information. These findings emphasize the need to use the whole feature set to improve the IDS's capacity to detect unexpected network traffic. The subsection discussion is as follows:

5.1 Individual classifiers using full features

Ensemble models outperform individual classifiers in measures like accuracy, precision, recall, and F1-score when using all available features as shown in Table 1.

a Random Forest

The RF model achieved an outstanding 99.99% accuracy, with perfect precision and recall (1.00), correctly identifying all positive cases. Its F1-score of 1.00 reflects a balanced and excellent performance in both accuracy and recall, as illustrated in Table 2 and Figure 2.

b Decision Trees

The DT model demonstrated exceptional performance with an accuracy of 99.98%, perfect precision and recall of 1.00, and an F1-score of 1.00, indicating flawless prediction and detection of positive cases, as shown in Table 2 and Figure 3.

TABLE 2 Individual classifiers using full features.

Metric	Value				
Random Forest					
Accuracy	1.00	Precision	Recall	F1-score	Support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Decision Tree					
Accuracy	1.00	Precision	Recall	F1-score	Support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Logistic Regression					
Accuracy	0.97	Precision	Recall	F1-score	Support
					6,196
Macro	Avg	0.98	0.97	0.97	6,196
Weighted	Avg	0.98	0.97	0.97	6,196
XGBossting					
Accuracy	1.00	precision	recall	f1-score	support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Naïve-Bayes					
Accuracy	0.78	Precision	Recall	F1-score	Support
					123,912
Macro	Avg	0.81	0.78	0.76	123,912
Weighted	Avg	0.81	0.78	0.76	123,912
Gradient boosting					
Accuracy	1.00	Precision	Recall	F1-score	Support
					6,196
Macro	Avg	1.00	1.00	1.00	6,196
Weighted	Avg	1.00	1.00	1.00	6,196

c Logistic Regression

The LR model achieved 97% accuracy, with a precision of 0.98 and a recall of 0.97, accurately detecting 97% of true positive events. Its F1-score of 0.97 reflects strong overall performance, as shown in Table 2 and Figure 4.

d XGBossting

XGBoost achieved an impressive accuracy of 99.99%, with perfect precision and recall of 1.00, correctly predicting all positive events. Its F1-score of 1.00 highlights its excellent balance of precision and recall, making it the top performer, as shown in Table 2 and Figure 5.

e Naïve Bayes

The NB model had a lower accuracy of 78%, correctly predicting 81% of positive cases. Its recall was 78%, and the F1-score of 0.76 indicates an imbalanced model with room for improvement in both precision and recall, as shown in Table 2.

f Gradient boosting

The GB model achieved an impressive accuracy of 99.82%, with perfect accuracy and recall (1.00), correctly predicting all positive cases and detecting all actual positives. Its F1-score of 1.00 reflects excellent balance and performance across all parameters, as shown in Table 2.

g Support Vector Machine

The SVM model achieved 99.26% accuracy, with excellent precision (0.99) and recall (0.99), demonstrating strong

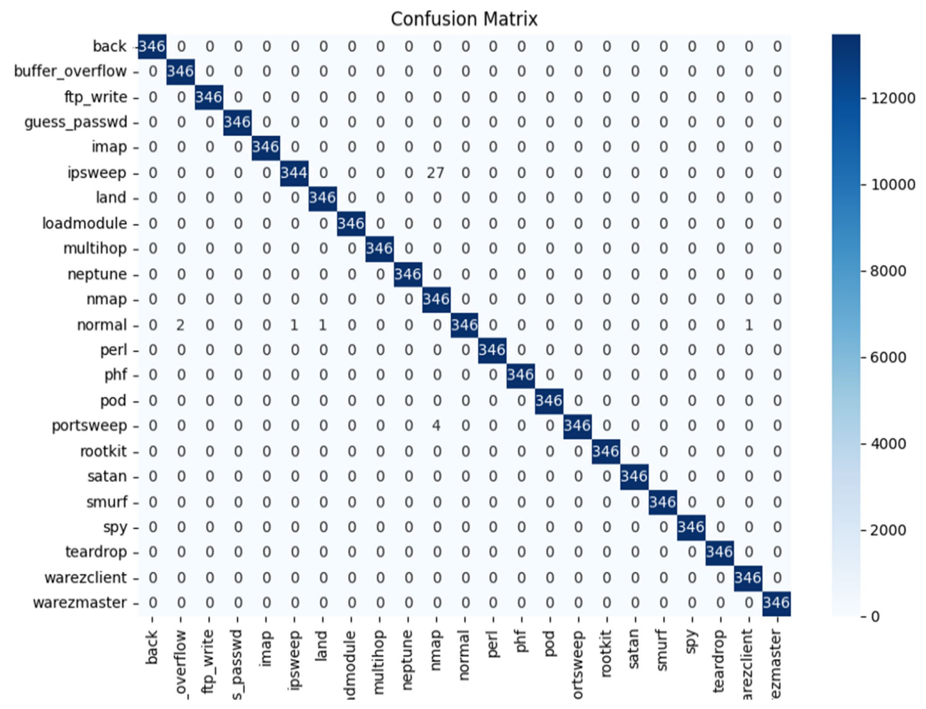

FIGURE 2
Random Forest.

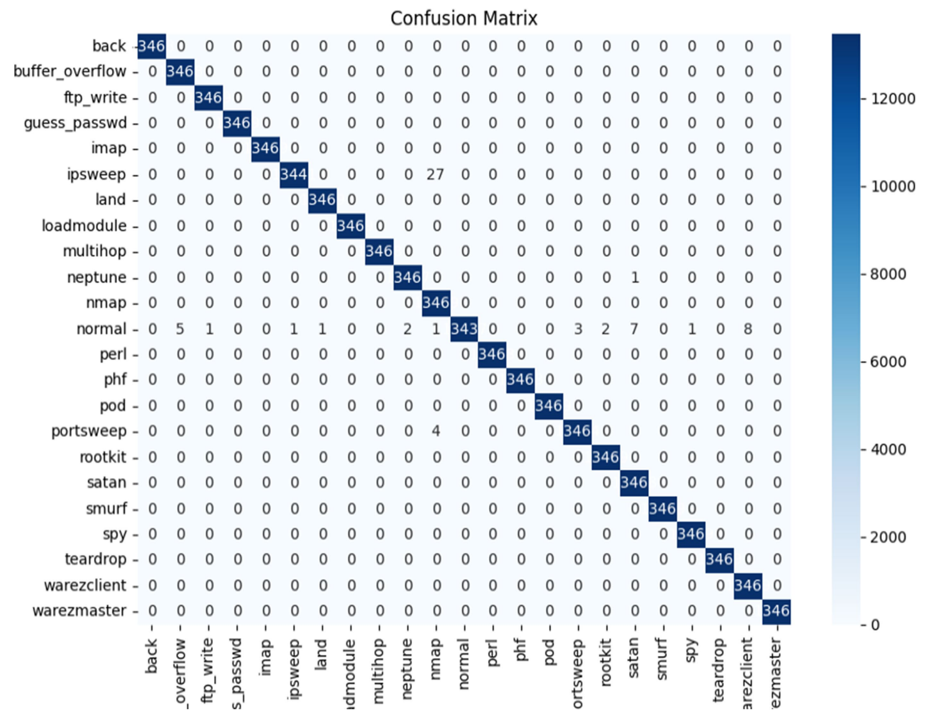

FIGURE 3
Decision Tree.

performance in predicting and identifying positive cases. Its F1-score of 0.99 indicates a well-balanced model, as shown in Table 3.

Table 4 compares the performance of several classifiers using accuracy, precision, recall, and F1-score. RF, DT, and XGBoost all produced near-perfect results, with 99.99% accuracy and perfect

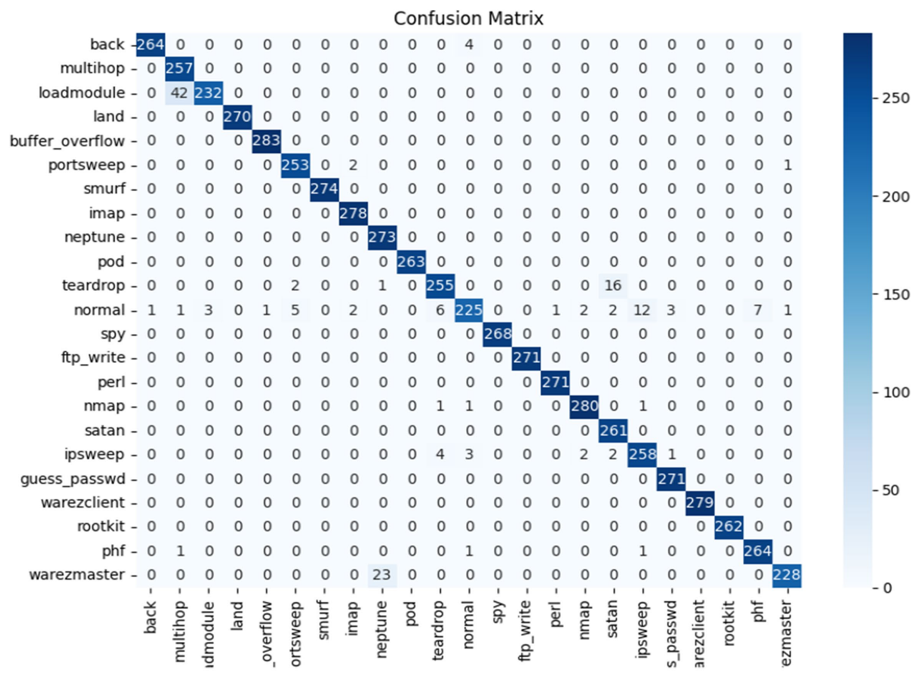

FIGURE 4
Logistic Regression.

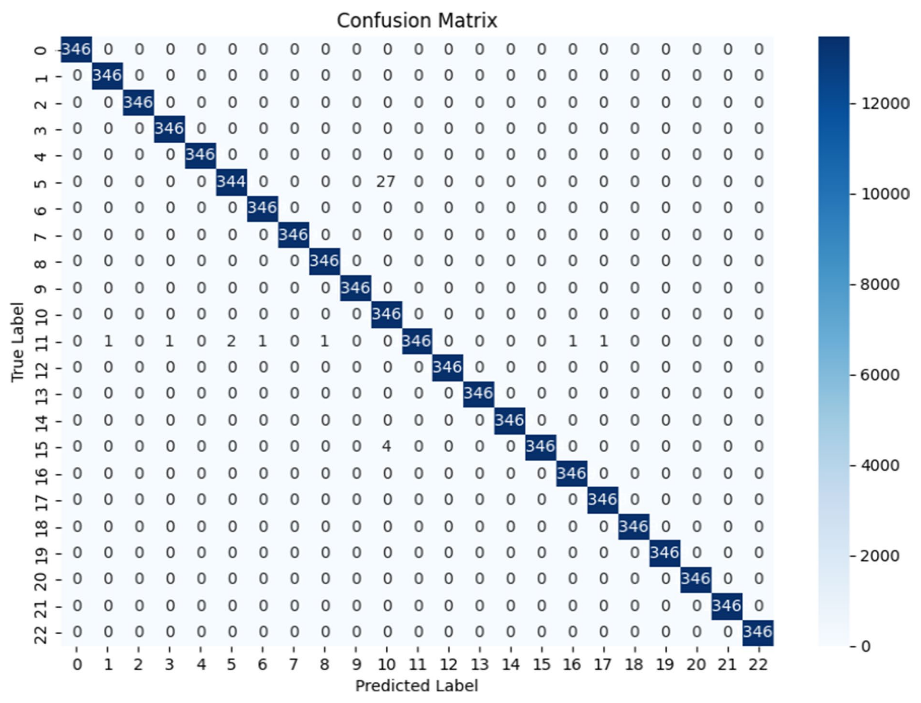

FIGURE 5
XGBossting.

TABLE 3 Support Vector Machine using full features.

	Precision	Recall	F1-score	Support
Accuracy	0.9925753760733420	0.9925753760733420	0.9925753760733420	0.9925753760733420
Macro avg	0.9926361652890440	0.9925724355169420	0.992555391610573	15489.0
Weighted avg	0.9926457833302430	0.9925753760733420	0.992562131963623	15489.0

TABLE 4 Performance evaluation of classifiers using accuracy, precision, recall, and F1-score using full features.

Classifiers	Accuracy	Precision	Recall	F1-score	Execution TIME
Random Forest	99.99%	1.00	1.00	1.00	332.34395813941956 s
Decision Tree	99.98%	1.00	1.00	1.00	95.22518181800842 s
XGBoost	99.99%	1.00	1.00	1.00	120.07075381278992 s
Naive Bayes	78%	0.81	0.78	0.76	13.097240209579468 s
Support Vector Machine	99.26%	0.99	0.99	0.99	128.66907286643982 s
Logistic Regression	97%	0.98	0.97	0.97	131.4768099784851 s
Gradient Boosting	99.82%	1.00	1.00	1.00	161.10451579093933 s

TABLE 5 Ensemble model using full features.

Metric	Value				
DT, RF, GB (voting hard)					
Accuracy	1.00	Precision	Recall	F1-score	Support
					15,489
Macro	Avg	1.00	1.00	1.00	15,489
Weighted	Avg	1.00	1.00	1.00	15,489
DT, RF, GB (stacking)					
Accuracy	1.00	Precision	Recall	F1-score	Support
					15,489
Macro	Avg	1.00	1.00	1.00	15,489
Weighted	Avg	1.00	1.00	1.00	15,489

precision, recall, and F1-score values of 1.00. NB has a much lower accuracy (78%), as well as worse precision, recall, and F1-score. SVM and LR both performed well, with accuracy rates of 99.26 and 97%, respectively, while retaining good precision, recall, and F1 scores. GB likewise produced outstanding results, with 99.82% accuracy and flawless precision, recall, and F1-score.

5.2 Ensemble model using full features

a- (DT+ RF+ GB) Voting hard

The ensemble model, which combines Decision Tree, Random Forest, and Gradient Boosting with hard voting, produced flawless classification results. It achieved an accuracy of 1.00, with precision, recall, and F1-score all at 1.00. This demonstrates strong predictive potential for the test results. As shown in Table 5 and Figure 6.

b- (DT+ RF+ GB) Stacking

The stacking ensemble model, which combines Decision Tree, Random Forest, and Gradient Boosting, likewise performed perfectly. It scored 1.00 for accuracy, precision, recall, and F1-score. This implies that stacking effectively harnessed the strengths of each base model. as shown in Table 5 and Figure 7.

Table 6 shows the performance of an ensemble model that combines DT, RF, and GB with the Voting Hard and stacking methods. The ensemble models performed flawlessly across all criteria, with accuracy, precision, recall, and an F1 score of 1.00.

5.3 Individual classifiers using gain information and manual information

The performance of Individual Classifiers by Gain Information and Manual Information, as shown in Table 1 is evaluated based on accuracy, precision, recall, and F1-score, the evaluation of different classifiers and ensemble models reveals the advantages and disadvantages of each model for certain classification tasks. In order to improve overall performance, this section covers both single classifiers and ensemble models, which integrate many classifiers.

a XGBoosting

The XGBoost model achieved impressive results with an accuracy of 99.97%, along with perfect precision, recall, and F1-score of 1.00, indicating flawless predictions and accurate detection of positive cases. This outstanding performance highlights XGBoost as a highly effective model, as shown in Table 7 and Figure 8.

b Decision Trees

The DT model achieved an accuracy of 99.97%, with perfect precision, recall, and F1-score values of 1.00, indicating flawless categorization of both positive and negative cases. It performed excellently in precision and recall, as shown in Table 7.

c Random forest

The RF model achieved an accuracy of 99.97%, with perfect precision, recall, and F1-score values of 1.00, indicating flawless performance in predicting and detecting positive cases. This makes it highly effective for the task, as shown in Table 7.

d Support Vector Machine

The SVM model achieved an accuracy of 88.09%, with precision and recall scores of 0.90 and 0.88, respectively.

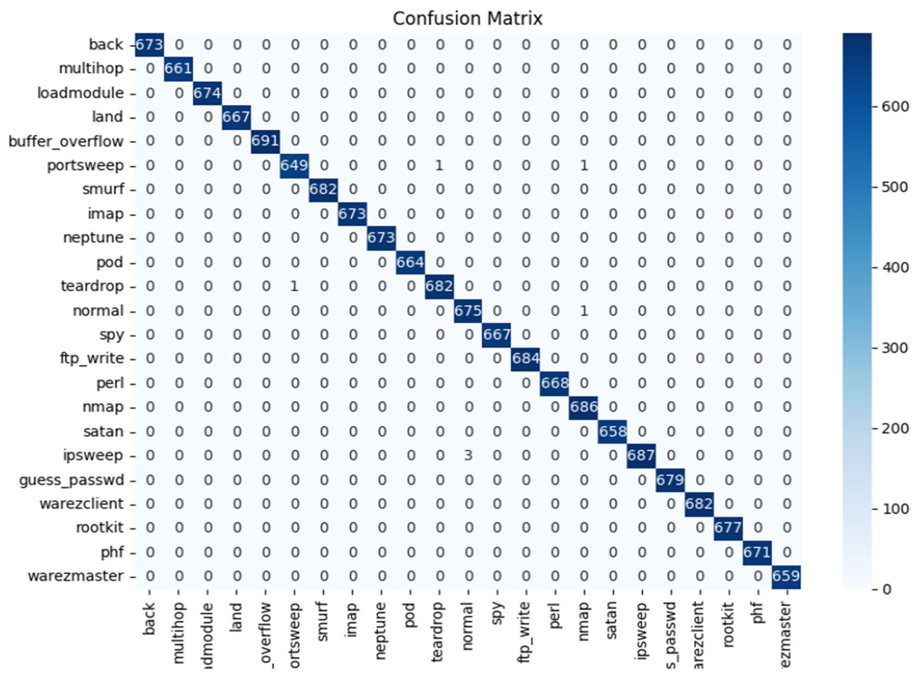

FIGURE 6
DT, RF, GB (voting hard).

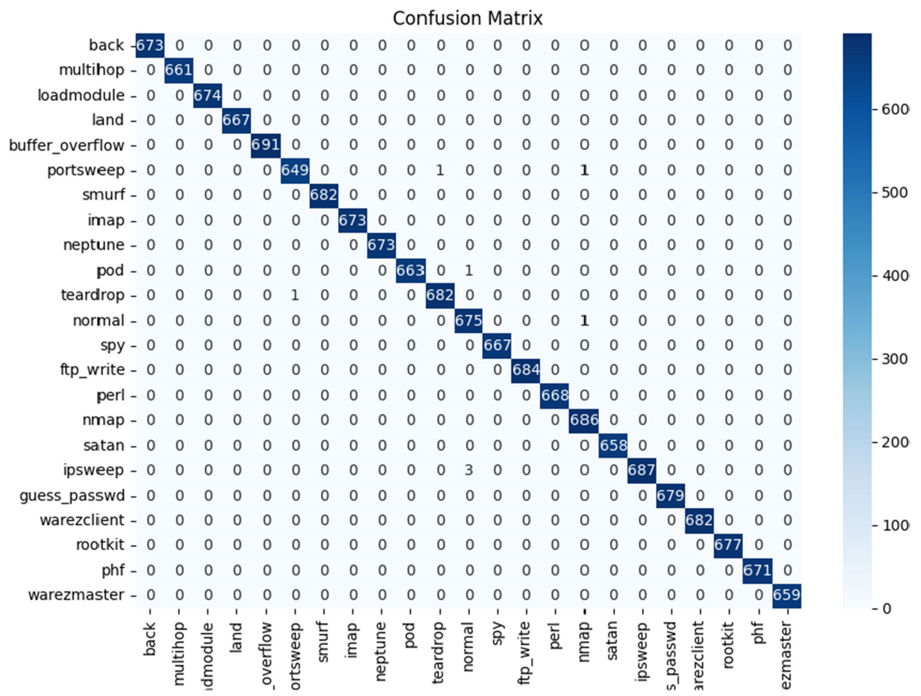

FIGURE 7
DT, RF, GB (stacking).

TABLE 6 Performance evaluation of ensemble model using accuracy, precision, recall, and F1-score using full features.

Classifier (ensemble model)	Accuracy	Precision	Recall	F1-score	Execution Time
DT + RF + GB (voting hard)	1.00	1.00	1.00	1.00	139.073 s
DT + RF + GB (stacking)	1.00	1.00	1.00	1.00	3571.27 s

TABLE 7 Individual classifiers using gain information and manual information.

Metric	Value				
XGBoosting					
Accuracy	1.00	Precision	Recall	F1-score	Support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Decision Trees					
Accuracy	1.00	Precision	Recall	F1-score	Support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Randomforest					
Accuracy	1.00	Precision	Recall	F1-score	Support
					309,778
Macro	Avg	1.00	1.00	1.00	309,778
Weighted	Avg	1.00	1.00	1.00	309,778
Support Vector Machine					
Accuracy	0.88	Precision	Recall	F1-score	Support
					9,294
Macro	Avg	0.90	0.88	0.88	9,294
Weighted	Avg	0.90	0.88	0.88	9,294
Naïve-Bayes					
Accuracy	0.63	Precision	Recall	F1-score	Support
					15,489
Macro	Avg	0.71	0.63	0.60	15,489
Weighted	Avg	0.71	0.63	0.60	15,489
Gradient Boosting					
Accuracy	1.00	Precision	Recall	F1-score	Support
					6,196
Macro	Avg	1.00	1.00	1.00	6,196
Weighted	Avg	1.00	1.00	1.00	6,196

While it demonstrated balanced performance with an F1-score of 0.88, its performance was notably lower compared to top models like RF and XGBoost, as shown in Table 7 and Figure 9.

e Naïve-Bayes

The NB model performed poorly, achieving an accuracy of 0.63. With a precision of 0.71 and recall of 0.63, it only correctly identified 63% of actual positive events. The F1-score of 0.60 indicates an imbalanced performance, highlighting the model's struggles in this classification task, as shown in Table 7 and Figure 10.

f Gradient Boosting

The GB model performed excellently with an accuracy of 99.89%. It achieved perfect values of 1.00 for accuracy, recall, and F1-score, indicating flawless classification of both positive and negative cases. This model outperformed all other classifiers, as shown in Table 7.

g Logistic Regression

The LR model has an accuracy of 72.0%, which means it accurately predicted 72% of the occurrences. Its accuracy of 0.74 and recall of 0.72 indicate that it made some mistakes in both forecasting positive events and detecting true positives. The F1-score of 0.71 indicates that the model's performance is not as great as that of some of the others, showing that there is still potential for development in terms of accuracy and recall, as shown in Table 8.

The performance of each classifier is displayed in Table 9, with particular attention paid to accuracy, precision, recall, and F1-score. Table 9 compares the performance of multiple classifiers, revealing that RF, DT, XGBoost, and GB all obtained near-perfect results, with 99.97% or 99.89% accuracy and flawless precision, recall, and F1 scores. In contrast, NB had an extremely low accuracy of 0.63%, as well as poor precision, recall, and F1 scores. SVM and LR performed moderately, with accuracy rates of 88.09 and 72.0%, respectively.

5.4 Ensemble model using gain information and manual information

a- (DT+ RF+ GB) Voting hard

The ensemble model, which integrates different classifiers, performed exceptionally well across the board. It achieved perfect scores of 1.00 in all metrics. This combination leverages the interpretability of DTs, the robustness of RFs, and the precision of GB, resulting in enhanced overall performance, as shown in Table 10 and Figure 11.

b- (DT+ RF+ GB) Stacking

The ensemble model incorporating Decision Tree, Random Forest, and Gradient Boosting with stacking likewise performed flawlessly, with accuracy, precision, recall, and F1-score all equal to 1.00. This demonstrates strong predictive potential for the test results. As shown in Table 10 and Figure 12.

In order to attain flawless detection performance across all measures, ensemble models incorporate multiple classifiers, as shown in Table 11.

Table 11 shows the performance of ensemble models that combine DT, RF, and GB with the Voting Hard and Stacking approaches. The ensemble models performed flawlessly across all measures, including accuracy, precision, recall, and an F1-score of 1.00.

5.5 Individual classifiers using RF-selected features

The performance of Individual Classifiers by RF-Selected Features, as shown in Table 1 evaluated based on accuracy,

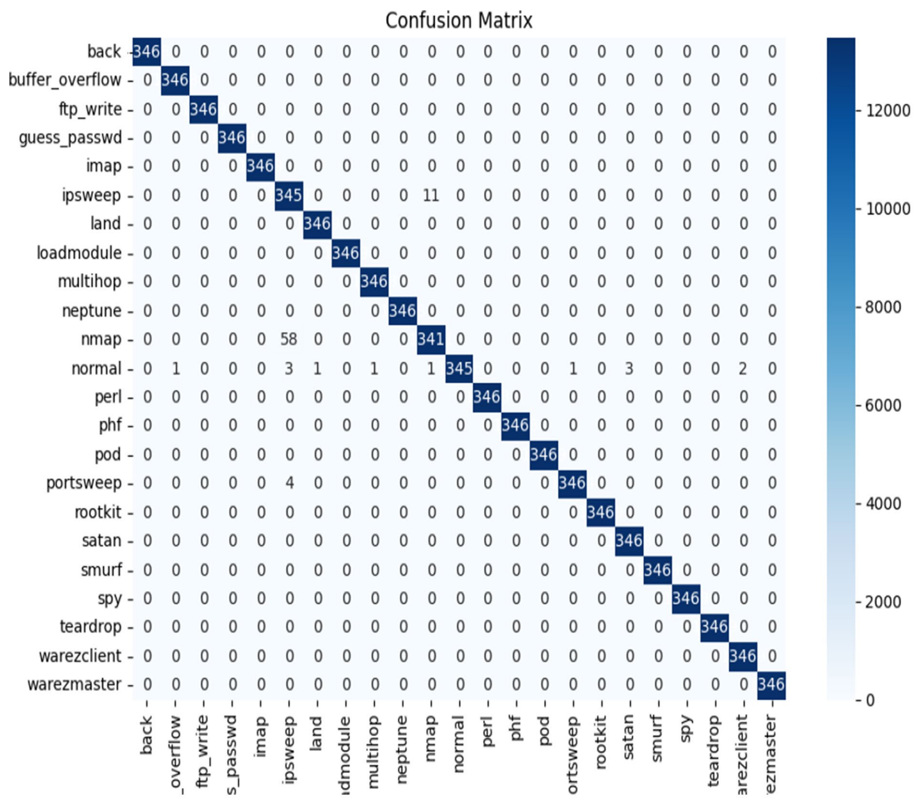

FIGURE 8
XGBoosting.

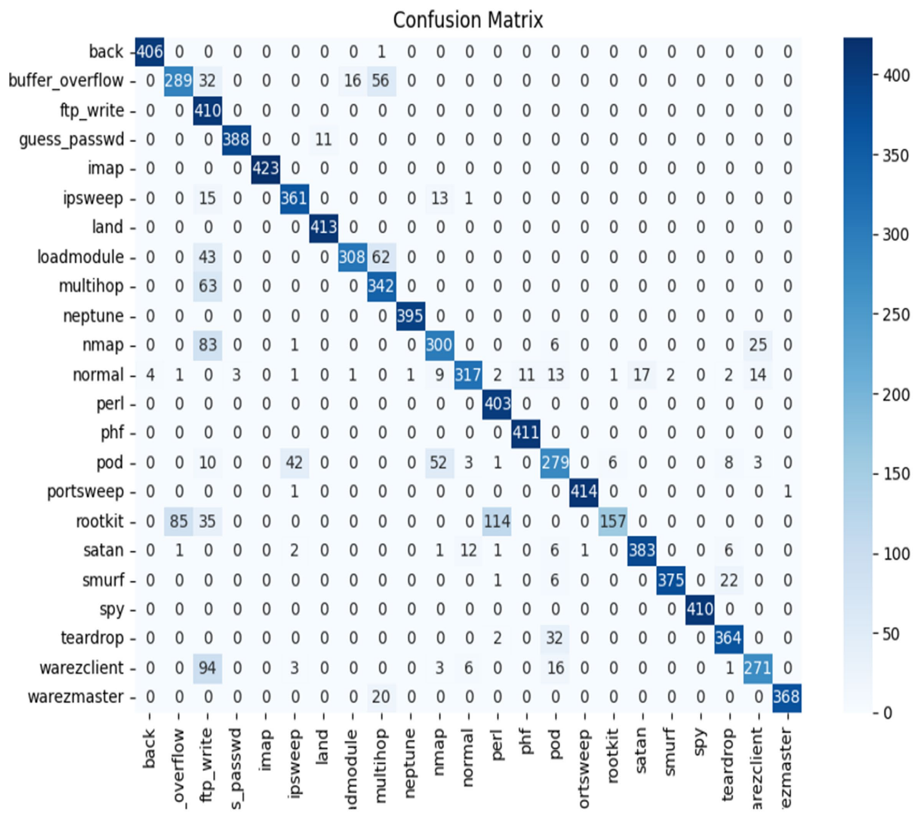

FIGURE 9
Support Vector Machine.

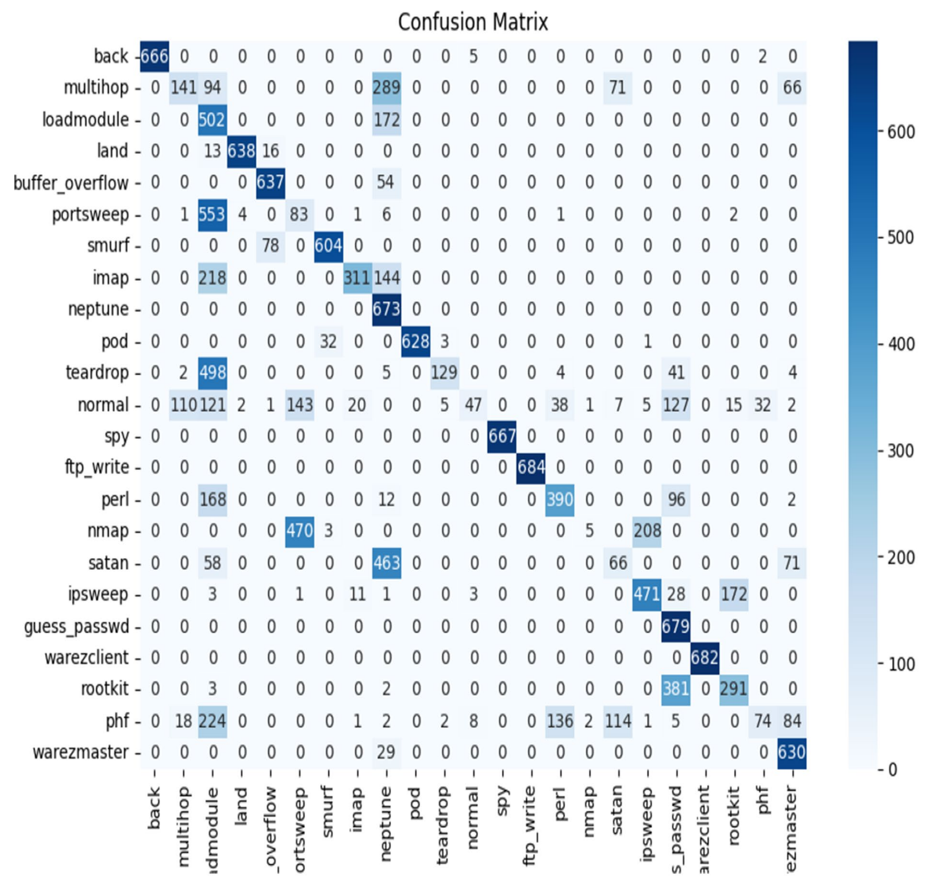

FIGURE 10
Naïve-Bayes.

TABLE 8 Logistic regression classifiers using gain information and manual information Figure 9: support vector machine.

Metric	Value				
Accuracy	0.72	Precision	Recall	F1-score	Support
					3,098
Macro	Avg	0.74	0.72	0.71	3,098
Weighted	Avg	0.74	0.72	0.71	3,098

precision, recall, and F1-score of separate classifiers and ensemble models are compared in this investigation. The objective is to evaluate the performance of each model separately and the classification skills of ensemble models, which combine many classifiers.

a Decision Tree

The DT model achieved an accuracy of 98.71%, nearly matching the RF model. However, its precision was 0.69, meaning only 69% of predicted positives were accurate. The recall was 0.65, indicating that the model correctly identified 65% of true positive events. The F1-score of 0.65 suggests that the DT model needs further improvement to balance precision and recall effectively, as shown in Table 12 and Figure 13.

b Naive Bayes

NB demonstrated poor performance, with an accuracy of just 33%. Its precision was 0.19, the recall was 0.37, and F1 score was 0.16. The model's poor results likely stem from the assumption of feature independence, which is not suitable for this dataset. The low accuracy and F1-score emphasize NB's limitations for this classification task, as shown in Table 12.

c Logistic Regression

LR achieved an accuracy of 90.56% but struggled with low precision (0.24), recall (0.20), and F1-score (0.21). As a linear model, it faced difficulty handling complex data correlations, which resulted in poor performance. The model's limited flexibility in adapting to the dataset's structure contributed to these lower metrics, as shown in Table 12 and Figure 14.

d Gradient Boosting

GB showed poor performance with an accuracy of 49.35%, precision of 0.48, recall of 0.29, and F1-score of 0.29. This underperformance may be due to overfitting or a mismatch with the dataset's characteristics, like data imbalance or feature correlation, as shown in Table 12.

TABLE 9 Performance evaluation of classifiers using accuracy, precision, recall, and F1-score using GI and MI.

Classifiers	Accuracy	Precision	Recall	F1-score	Execution time
Random Forest	99.97%	1.00	1.00	1.00	818.7610149383545 s
Decision Tree	99.97%	1.00	1.00	1.00	34.62117004394531 s
XGBoost	99.97%	1.00	1.00	1.00	91.20186018943787 s
Naive Bayes	0.63%	0.71	0.63	0.60	3.237001895904541 s
Support Vector Machine	88.09%	0.90	0.88	0.88	275.7197570800781 s
Logistic Regression	72.0%	0.74	0.72	0.71	1381.2660410404205 s
Gradient Boosting	99.89%	1.00	1.00	1.00	240.0945920944214 s

TABLE 10 Ensemble model using gain information and manual information.

Metric	Value				
DT, RF, GB (voting hard)					
Accuracy	1.00	Precision	Recall	F1-score	Support
					30,978
Macro	Avg	1.00	1.00	1.00	30,978
Weighted	Avg	1.00	1.00	1.00	30,978
DT, RF, GB (stacking)					
Accuracy	1.00	Precision	Recall	F1-score	Support
					30,978
Macro	Avg	1.00	1.00	1.00	30,978
Weighted	Avg	1.00	1.00	1.00	30,978

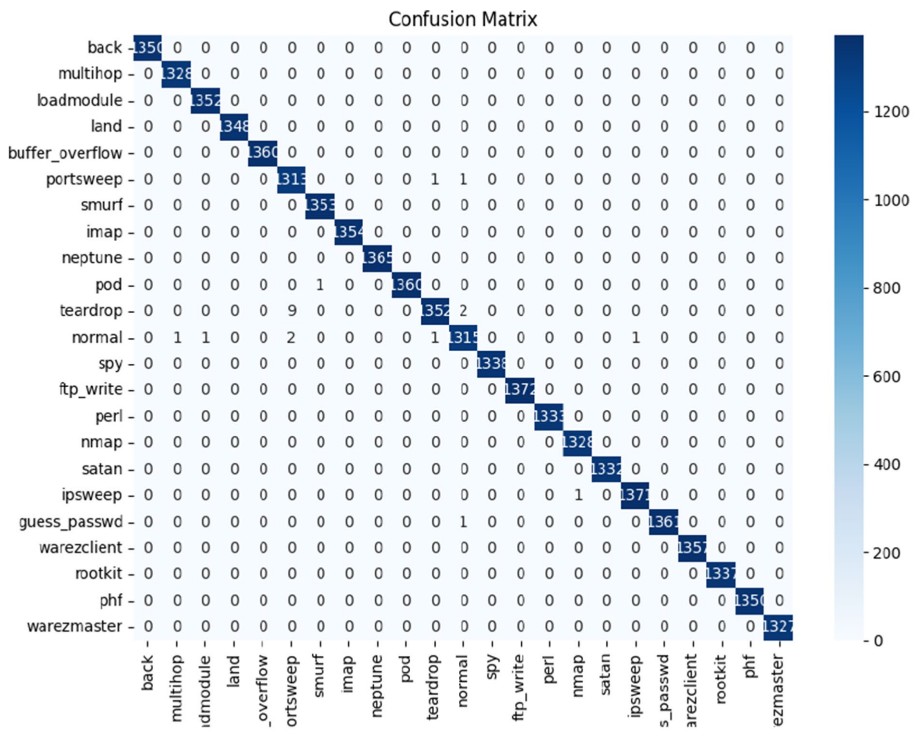

FIGURE 11
DT, RF, GB (voting hard).

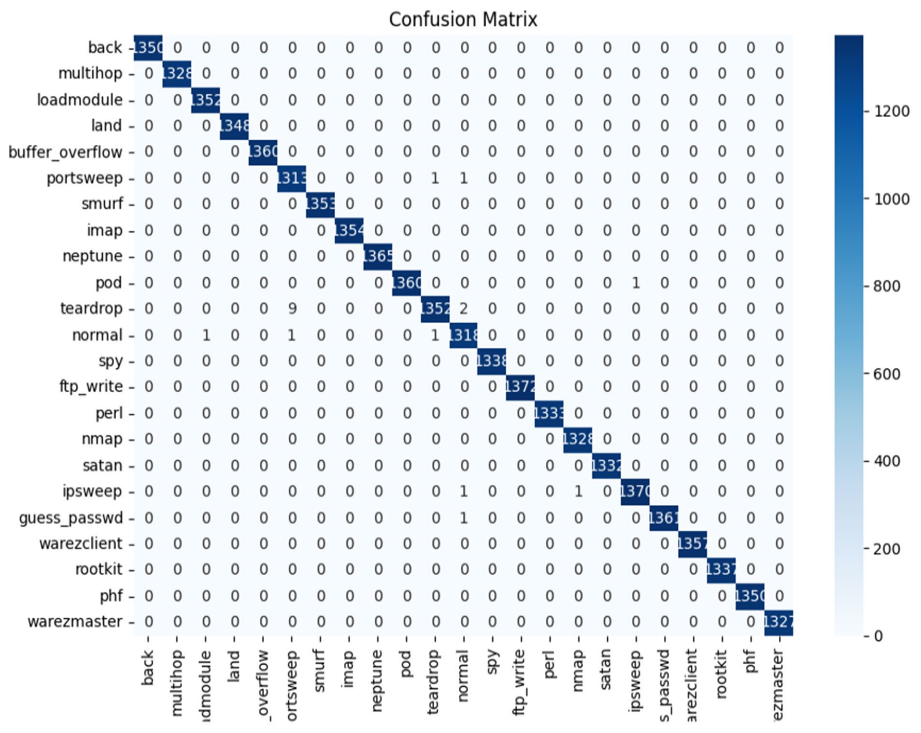

FIGURE 12
DT, RF, GB (stacking).

TABLE 11 Performance evaluation of ensemble model using accuracy, precision, recall, and F1-score using GI and MI.

Classifier (ensemble model)	Accuracy	Precision	Recall	F1-score	Execution time
DT + RF + GB (voting hard)	1.00	1.00	1.00	1.00	178.5962700843811 s
DT + RF + GB (stacking)	1.00	1.00	1.00	1.00	878 s

e Random Forest

RF achieved an accuracy of 98.83%, with a precision of 0.80, a recall of 0.75, and an F1-score of 0.76. While the accuracy is strong, the precision and recall indicate difficulties in managing false positives and false negatives. Overall, it performs well but could benefit from further adjustments, particularly to address class imbalances, as shown in Table 12 and Figure 15.

f XGBoost

XGBoost achieved 98.79% accuracy, with precision, recall, and F1-score of 0.76, 0.74, and 0.74, respectively. Although strong, its performance was slightly lower than RF, possibly due to hyperparameter settings or the dataset's structure, as shown in Table 12.

g Support Vector Machine

SVM achieved 92.40% accuracy but with poor precision (0.39), recall (0.27), and F1-score (0.29). Its low performance suggests it struggles with the dataset's complexity, possibly due to kernel selection or insufficient tuning, as shown in Table 12 and Figure 16.

Table 13 summarizes the performance of different classifiers, emphasizing accuracy, precision, recall, and F1 score.

Table 13 shows the performance evaluations of several classifiers. RF and DT produced reasonably high accuracy (98.83 and 98.71%, respectively), but with modest precision, recall, and F1 scores. XGBoost also fared well, with 98.79% accuracy, although it had lesser precision and recall than RF and DT. NB performed poorly, with an accuracy of just 33% and low precision, recall, and F1 scores. SVM and LR fared rather well, with accuracy values of 92.40 and 90.56%, respectively, but with low precision and recall. GB had the poorest result, with an accuracy of 49.35% and similarly low precision, recall, and F1 score.

5.6 Ensemble model using RF-selected features

a- (DT+ RF+ GB) Voting hard

TABLE 12 Individual classifiers using RF-selected features.

Metric	Value				
Decision Tree					
Accuracy	0.99	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.69	0.65	0.65	25,195
Weighted	Avg	0.99	0.99	0.99	25,195
Naive Bayes					
Accuracy	0.33	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.19	0.37	0.16	25,195
Weighted	Avg	0.88	0.33	0.35	25,195
Logistic Regression					
Accuracy	0.91	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.24	0.20	0.21	25,195
Weighted	Avg	0.86	0.91	0.88	25,195
Gradient Boosting					
Accuracy	0.49	Precision	Recall	F1-score	Support
					25,193
Macro	Avg	0.48	0.29	0.29	25,193
Weighted	Avg	0.96	0.49	0.57	25,193
Random Forest					
Accuracy	0.99	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.80	0.75	0.76	25,195
Weighted	Avg	0.99	0.99	0.99	25,195
XGBoosting					
Accuracy	0.99	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.76	0.74	0.74	25,195
Weighted	Avg	0.99	0.99	0.99	25,195
Support Vector Machine					
Accuracy	0.92	Precision	Recall	F1-score	Support
					25,195
Macro	Avg	0.39	0.27	0.29	25,195
Weighted	Avg	0.91	0.92	0.90	25,195

The Voting Hard ensemble model, consisting of DT, RF, and GR, achieved 0.99 accuracy. However, its precision (0.71), recall (0.68), and F1-score (0.68) were slightly lower than the stacking model, indicating room for improvement in precision and recall, as shown in Table 14 and Figure 17.

b- (DT+ RF+ GB) Stacking

The ensemble model, which used stacking to combine Decision Tree, Random Forest, and Gradient Boosting, obtained an accuracy of 0.99, a precision of 0.75, a recall of 0.68, and an F1 score of 0.70. This represents higher accuracy and F1-scores compared to the hard voting, stacking model, indicating room for improvement in precision and recall, as shown in Table 14 and Figure 18.

Table 15 evaluates the ensemble models, which integrate several classifiers to attain flawless detection performance across all criteria.

Table 15 shows the performance of the ensemble model that combines DT, RF, and GR with the Voting Hard and stacking approaches. The ensemble model with stacking outperforms hard voting in terms of precision (0.75 vs. 0.71), F1-score (0.70 vs. 0.68), and accuracy (0.99) while retaining recall (0.68). This shows that

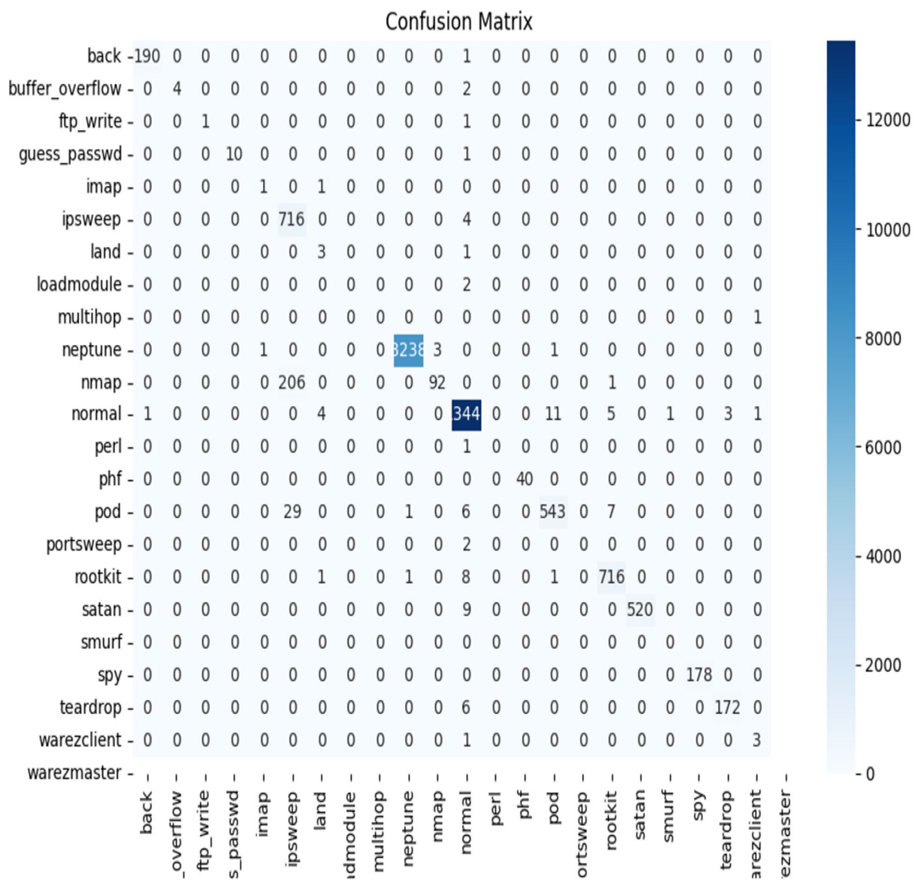

FIGURE 13
Decision Tree.

stacking gives a more balanced and effective mix of classifiers, increasing the model's capacity to properly detect positive situations while maintaining overall accuracy.

5.7 Comparing models and their statistical importance

- The study used five-fold cross-validation on individual models, such as Random Forest (RF) and Gradient Boosting (GB), to evaluate and compare classifier performance. This ensured that the findings showed strong generalization and did not overfit to a particular data split (20%test_size, 80%train_size).
- Performance differences were then statistically assessed using paired t-tests. A t-statistic of 42.7393 with a $p < 0.0001$ was obtained when comparing RF to GB, indicating a statistically significant advantage for RF in classification performance.

Furthermore, a hard Voting Classifier was assessed as compared to its base learners. While the differences between RF ($t = 0.3780$, $p = 0.7055$) and GB ($t = 0.7746$, $p = 0.4386$) were not statistically significant, the Voting Classifier performed much better than the Decision Tree ($t = 4.0835$, $p < 0.0001$), according to the results of the *t*-test. Given that it outperforms the strongest base models while enhancing overall resilience, this demonstrates the Voting ensemble's resilience.

- In addition, *t*-tests were used to choose features using Mutual Information and Information Gain in order to verify that the chosen features improved generalization by reducing dimensionality and greatly enhancing model performance.

5.8 Analysis of SHAP-driven feature importance in Random Forest and decision tree classifiers for intrusion detection

SHAP values for the Random Forest and Decision Tree classifiers were calculated to interpret model options. In order to visualize the effects of features on both individual predictions and the overall relevance of features, SHAP provides both local and global interpretability.

According to the SHAP study, several variables were consistently more influential in both models, which is consistent with behaviors associated with documented network intrusions. This consistency provides information about possible dimensionality reduction techniques in addition to confirming the models' validity.

While the Decision Tree model's simple structure makes it easier to grasp., the Random Forest model outperformed it while

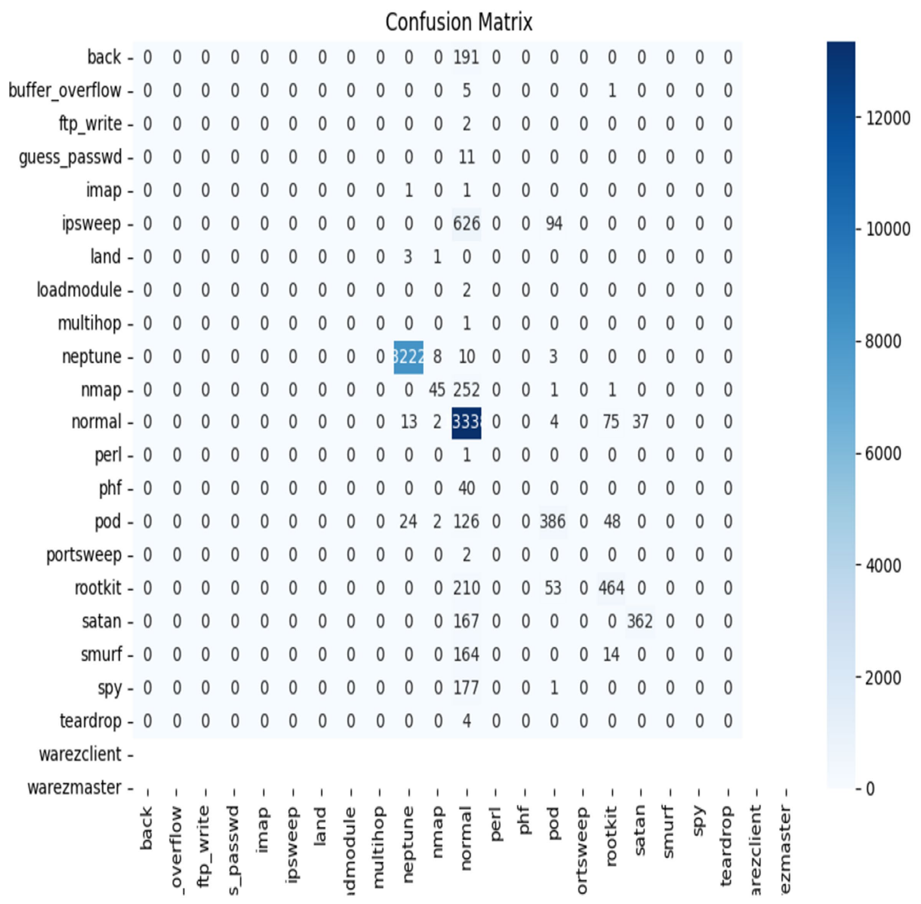

FIGURE 14
Logistic Regression.

allowing for insight through SHAP. SHAP's ability to clearly explain feature contributions reduced the complexity of the ensemble, proving that accuracy and interpretability do not have to be mutually incompatible.

Figure 19 demonstrates that while service_auth is of little significance, network packet sizes (src_bytes, dst_bytes) and connection flags are the most important features for attack detection in the model. This illustrates how the model successfully identifies malicious activity by concentrating on network traffic patterns.

As seen in Figure 20: The graphic shows that while service authentication data (service_auth) makes very little contribution to attack detection, network packet metrics (src_bytes, dst_bytes) and connection flags dominate feature relevance. These outcomes demonstrate how well the model uses fundamental network traffic features to identify malicious behavior across all threat categories.

5.9 Comparison of accuracy and feature selection methods in various studies

Table 16 compares several ML algorithms used to identify intrusions on the NSL-KDD dataset. Accuracy rates vary greatly based on the classifier and feature selection strategy employed.

5.9.1 In prior studies

- Attou et al. (2023) shown that utilizing RBFNN with all features resulted in an accuracy of 90.49%, which increased to 94.1% when only four features were picked by Random Forest (RF). This demonstrates that focused feature selection may improve speed while reducing complexity.
- Ogwara et al. (2022) produced exceptionally high performance with classic ML models, notably Random Forest (99.22%), Decision Tree (99.07%), and KNN (98.06%), utilizing the whole feature set. This implies that with adequate training, even classical models can perform well when all important attributes are kept.
- Rawat et al. (2022) investigated ensemble and deep learning approaches, such as LightGBM, DNN, and PCA + DNN, and found low accuracies ranging from 76.7 to 79.3%, indicating a potential mismatch between model complexity and dataset properties, or a lack of efficient feature reduction.
- Tauscher et al. (2021) presented results using five classifiers, with SVM outperforming the others at 80.47%, while Gradient Boosting (GB) and Decision Tree (DT) fared below 70% on the whole feature set.

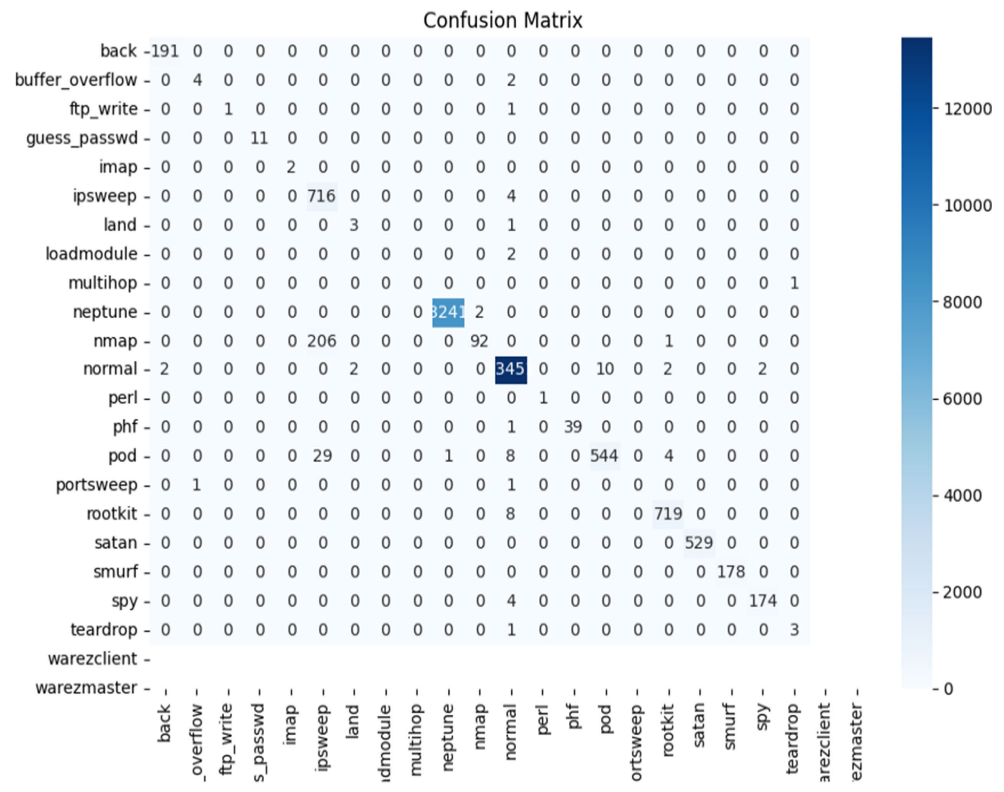

FIGURE 15
Random Forest.

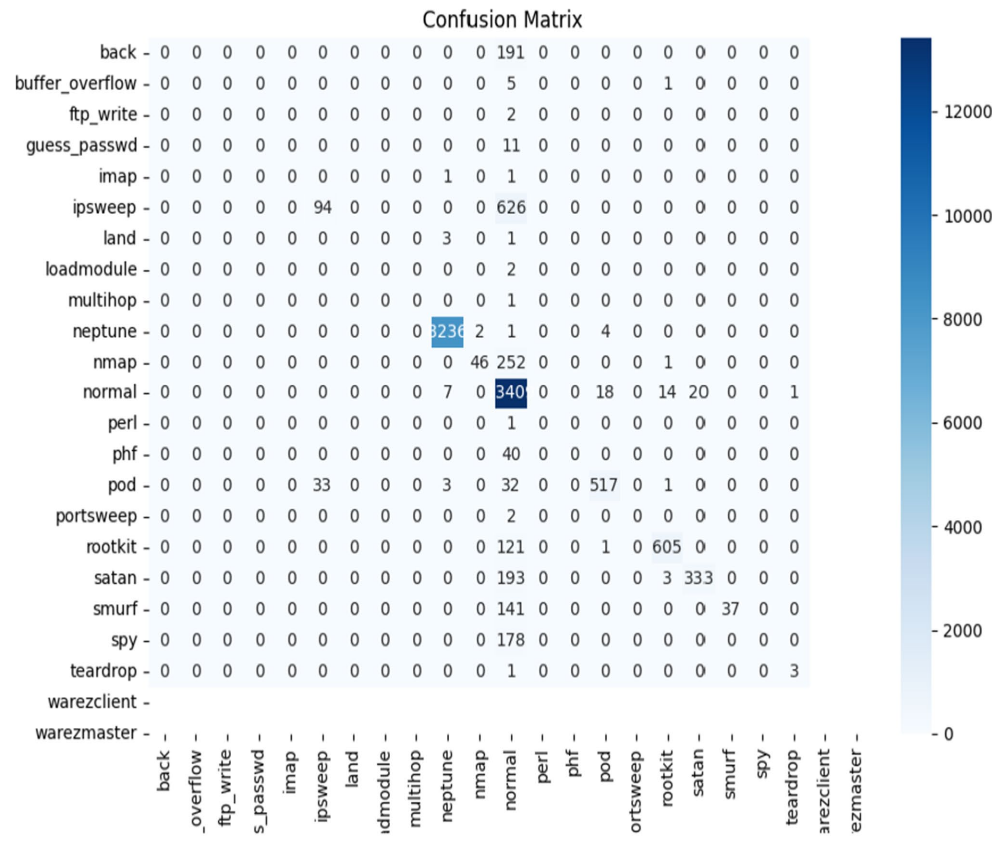

FIGURE 16
Support Vector Machine.

TABLE 13 Performance evaluation of classifiers using accuracy, precision, recall, and F1-score using RF-selected features.

Classifiers	Accuracy	Precision	Recall	F1-score	Execution time
Random Forest	98.83%	0.80	0.75	0.76	29.910604000091553 s
Decision Tree	98.71%	0.69	0.65	0.65	17.27623677253723 s
XGBoost	98.79%	0.76	0.74	0.74	29.366595029830933 s
Naive Bayes	33%	0.19	0.37	0.16	5.824549198150635 s
Support Vector Machine	92.40%	0.39	0.27	0.29	221.10965991020203 s
Logistic Regression	90.56%	0.24	0.2	0.2	35.62619614601135 s
Gradient Boosting	49.35%	0.48	0.29	0.29	379.5056371688843 s

TABLE 14 Ensemble model using RF-selected features.

Metric	Value				
DT, RF, GR (hard voting)					
Accuracy	0.99	Precision	Recall	F1-score	Support
					12,597
Macro	Avg	0.71	0.68	0.68	12,597
Weighted	Avg	0.99	0.99	0.99	12,597
DT, RF, GB (stacking)					
Accuracy	0.99	Precision	Recall	F1-score	Support
					12,597
Macro	Avg	0.75	0.68	0.70	12,597
Weighted	Avg	0.99	0.99	0.99	12,597

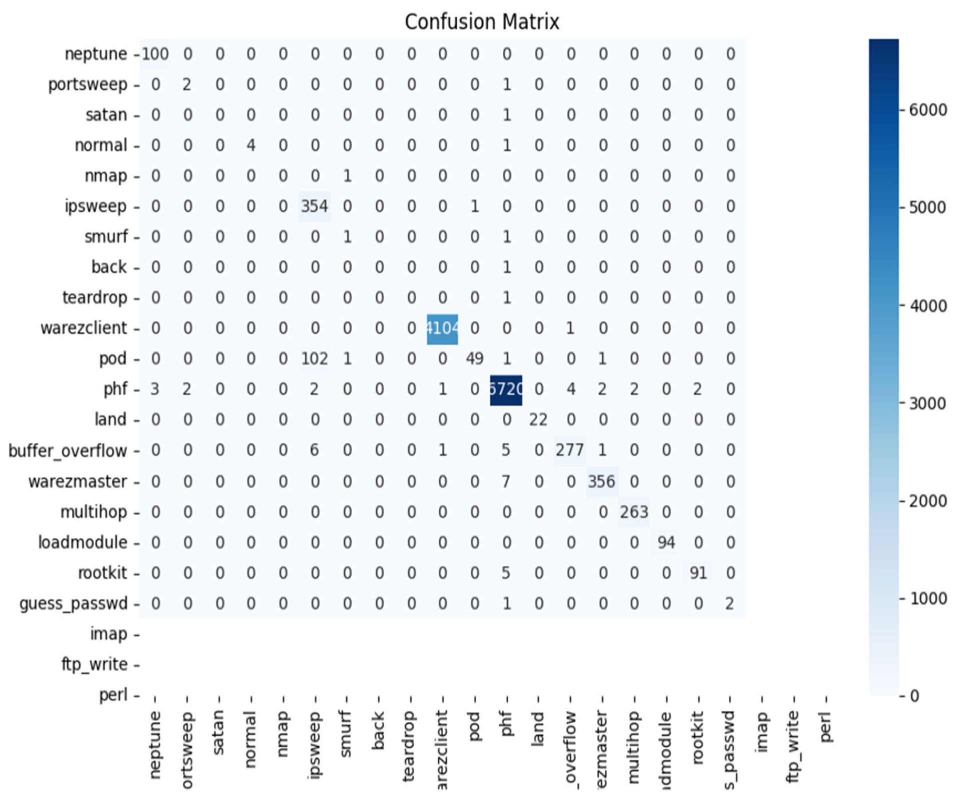

FIGURE 17
DT, RF, GR (hard voting).

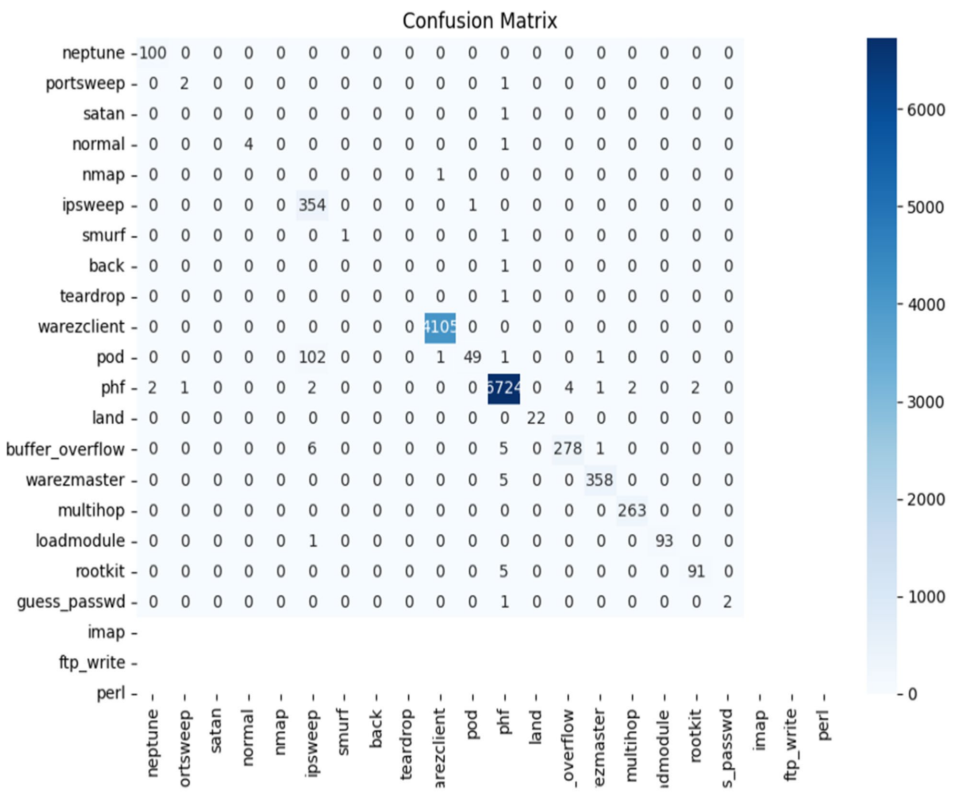

FIGURE 18
DT, RF, GB (stacking).

TABLE 15 Performance evaluation of ensemble model using accuracy, precision, recall, and F1-score using RF-selected features.

Classifiers	Accuracy	Precision	Recall	F1-score	Execution time
DT + RF + GB (voting hard)	0.99	0.71	0.68	0.68	409.4272561073303 s
DT + RF + GB (stacking)	0.99	0.75	0.68	0.70	2263.97 s

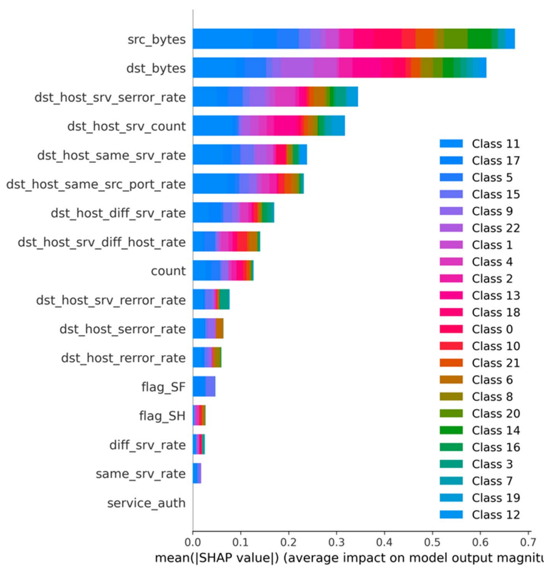

FIGURE 19
Global feature importance based on SHAP (DT).

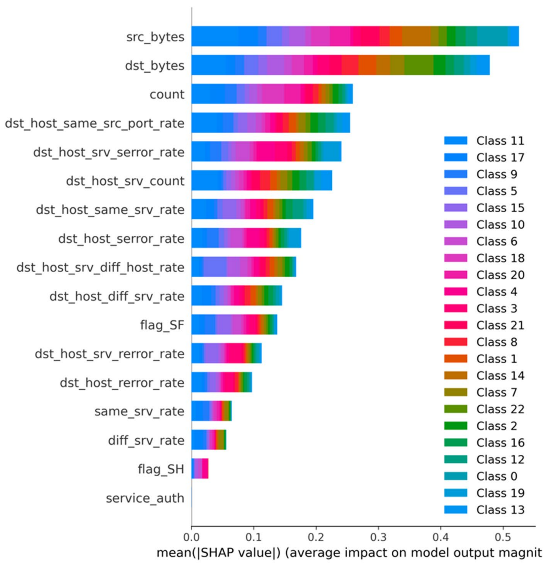

FIGURE 20
Global feature importance based on SHAP (RF).

TABLE 16 Comparison of accuracy and feature selection methods in various studies and the proposed model on the NSL-KDD dataset.

Research	Dataset	ML technique	Accuracy	Feature selection
Attou et al. (2023)	NSL-KDD	RBFNN classifier	90.49	Full dataset
			92.12	10 Features by RF
			94.16	4 Features by RF
Ogwara et al. (2022)	NSL KDD	DT	99.07	Full features
		RF	99.22	
		AdaBoost	94.05	
		Nıve Bayes	9.64	
		Stochastic dual coordinate ascent	16.91	
		Multilayer perceptron	84.79	
		K-Nearest Neighbors	98.06	
		Linear discriminant analysis	89.16	
		LR	79.69	
		SVM	79.62	
Rawat et al. (2022)	NSL-KDD	DT	0.778	Full features
		Extra Tree	0.767	
		Ensemble Extra Tree	0.769	
		Light GBM	0.776	
		Deep Neural Network	0.772	
		PCA + Deep Neural Network	0.793	
Tauscher et al. (2021)	NSL-KDD	RF	76.00%	Full features
		SVM	80.47%	
		NB	76.86%	
		DT	68.28%	
		GB	68.12%	
Our proposed approach	NSL-KDD	RF	99.99%	Full features
		DT	99.98%	
		XGBoost	99.99%	
		NB	78%	
		SVM	99.26%	
		LR	97%	
		GB	99.82%	
		DT + RF + GR (Voting hard)	1.00	
		DT + RF + GR (Stacking)	1.00	
		RF	99.97%	Selected features by manual information and gain information
		DT	99.97%	
		XGBoost	99.97%	
		NB	0.63%	
		SVM	88.09%	
		LR	72.0%	
		GB	99.89%	
		DT + RF + GR (Voting hard)	1.00	
		DT + RF + GR (Stacking)	1.00	
		RF	98.83%	Selected features by RF
		DT	98.71%	
		XGBoost	98.79%	
		NB	32.41%	
		SVM	92.40%	
		LR	90.56%	
		GB	49.35%	
		DT + RF + GR (Voting hard)	0.99%	
		DT + RF + GR (Stacking)	0.99%	

5.9.2 In contrast, the approach suggested in the study demonstrates consistently higher performance

- Voting Hard and Stacking DT, RF, and GB models resulted in a flawless 1.00 accuracy on both entire and chosen feature sets.
- XGBoost, RF, and GB obtained accuracies close to or greater than 99%, independent of the feature selection technique utilized.
- Models such as Naive Bayes (NB) and Logistic Regression (LR) fared worse when combined with specific characteristics, particularly those picked by RF alone (e.g., NB at 32.41%, GB at 49.35%).

5.9.3 Feature selection analysis

Three different feature selection approaches were tested:

- *Full Feature Set*
  Most models performed well with complete data, particularly ensemble approaches and tree-based classifiers.
  Voting and Stacking performed flawlessly (1.00).
- *Manual Selection (Information Gain & Domain Knowledge)*

This strategy used statistics and expert analysis to maintain critical qualities.

The bulk of classifiers still scored well, with Voting Hard and Stacking scoring 1.00 and RF, DT, and XGBoost approaching 99.97%.

NB and LR's accuracy was reduced, demonstrating their sensitivity to feature reduction.

- *RF-Based Feature Selection*

When Random Forest was used alone to choose features, performance declined marginally.

Ensemble techniques such as Voting and Stacking maintained excellent accuracy (0.99), while individual classifiers such as NB (32.41%) and GB (49.35%) had considerable decreases, indicating that critical features may have been over-reduced or eliminated.

6 Conclusion and future work

1) Cloud computing's rapid development has made it easier to use, but it has also made people more vulnerable to cyberattacks, necessitating the employment of sophisticated IDSs to combat sophisticated assaults. By offering an ensemble IDS architecture driven by ML, tailored for cloud systems, and evaluated on the NSL-KDD dataset, this paper addresses these problems.
2) When all features are used, the method achieves perfect scores (100%) in accuracy, precision, recall, and F1-score, combining ensemble classifiers Random Forest, Decision Tree, and Gradient Boosting through voting hard and Stacking strategies, proving that combining different models improves detection capabilities. By concentrating on significant characteristics like src_bytes and dst_host_serror_rate, feature selection strategies like Information Gain and Mutual Information improved detection. However, depending only on Random Forest-selected features occasionally resulted in decreased performance, suggesting the need for additional context-sensitive selection techniques.
3) SVM produced mediocre results, but ensemble-based models routinely outperformed standalone classifiers such as Naive Bayes and Logistic Regression, which were hampered by class imbalance and dependence problems. Despite slight decreases in precision and recall, the ensemble system demonstrated resilience across several feature sets, retaining almost perfect accuracy (99.97%) with manually selected features and 99% with Random Forest features. The superior accuracy and dependability of the suggested model were validated by comparisons with previous studies.
4) Application-wise, this study provides helpful recommendations for deploying scalable and effective IDS in cloud environments, emphasizing the necessity of striking a balance between resource requirements and accuracy. In the future, the model should be expanded with unsupervised approaches to identify zero-day threats, evaluated in real-time cloud settings, and made more visible using explainable AI methodologies. Overall, the study contributes a strong plan to defend contemporary cloud infrastructure against new cyber threats by bridging the gap between academic research and real-world application.
5) The study stresses the growing need for integrating AI-driven solutions into cloud security infrastructures. It highlights how combining a number of ML algorithms improves the flexibility and precision of threat identification, laying the groundwork for next-generation security systems that are both proactive and efficient. With cyberattacks becoming more complicated, the study promotes the development of intelligent, scalable, and robust cloud defense systems suited for both academic research and real-world applications.

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

KM: Writing – original draft, Methodology, Software, Visualization, Investigation, Data curation, Conceptualization, Funding acquisition, Formal analysis, Writing – review & editing. SA: Project administration, Writing – review & editing, Supervision, Formal analysis. FT: Validation, Writing – review & editing, Resources.

Funding

The author(s) declare that no financial support was received for the research and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The authors declare that no Gen AI was used in the creation of this manuscript.

Publisher's note

References

Adhikari, A., and Bal, B. K., Machine learning technique for intrusion detection in the field of the intrusion detection system. Conference Paper (2023). Available at: https://www.researchgate.net/publication/372312413

Akinbolaji, T. J. (2023). Advanced integration of artificial intelligence and machine learning for real-time threat detection in cloud computing environments. *Iconic Research and Engineering Journals*. vol. 6, 980–991. doi: 10.5281/zenodo.13963675

Alalhareth, M., and Hong, S., "An improved mutual information feature selection technique for intrusion detection systems in the Internet of Medical Things." *Sensors*. (2023) 23:4971. doi: 10.3390/s23104971

Aldallal, A., and Alisa, F. (2021). Effective intrusion detection system to secure data in the cloud using machine learning. *Symmetry* 13:2306. doi: 10.3390/sym13122306

Alhayali, R. A. I., Aljanabi, M., Ali, A. H., Mohammed, M. A., and Sutikno, T. (2021). Optimized machine learning algorithm for intrusion detection. *Indon. J. Electr. Eng. Comput. Sci.* 24, 590–599. doi: 10.11591/ijeecs.v24.i1.pp590-599

Ali, T. E., Chong, Y. W., and Manickam, S. (2023). Machine learning techniques to detect a DDoS Attack in SDN: a systematic review. *Appl. Sci.* 13:3183. doi: 10.3390/app13053183

Alkadi, S., Al-Ahmadi, S., and Ben Ismail, M. M. (2023). Toward improved machine learning-based intrusion detection for internet of things traffic. *Computers* 12. doi: 10.3390/computers12080148

Alotaibi, A., and Rassam, M. A., "Adversarial machine learning attacks against intrusion detection systems: a survey on strategies and defense," (2023), MDPI 15:62. doi: 10.3390/fi15020062

Al-Sharif, M., and Bushnag, A. (2024). Enhancing cloud security: a study on ensemble learning-based intrusion detection systems. *IET Commun.* 18, 950–965. doi: 10.1049/cmu2.12801

Attou, H., Mohy-eddine, M., Guezzaz, A., Benkirane, S., Azrour, M., Alabdultif, A., et al. (2023). Towards an intelligent intrusion detection system to detect malicious activities in cloud computing. *Appl. Sci.* 13, 1–19. doi: 10.3390/app13179588

Boldini, D., Grisoni, F., Kuhn, D., Friedrich, L., and Sieber, S. A. (2023). Practical guidelines for the use of gradient boosting for molecular property prediction. *J. Cheminform.* 15:73. doi: 10.1186/s13321-023-00743-7

Dattangire, R., Burle, R., Biradar, D., and Dewangan, L., "Machine learning-based security for cloud Proceedings of the 2024 IEEE North Karnataka Subsection Flagship International Conference (NKCon), computing challenges and implications," in Proceedings of the 2024 IEEE North Karnataka Subsection Flagship International Conference, (Bagalkote, India: NKCon) 2024, pp. 1–7. doi: 10.1109/NKCon62728.2024.10774633

Devi, T. A., and Jain, A., "Enhancing cloud security with deep learning-based intrusion detection in cloud computing environments," in 2024 2nd International Conference on Advancement in Computation & Computer Technologies (InCACCT), (2024), pp. 541–546.

Devidas, V. P., and Adesh, N. D. (2021). Comparative analysis of machine learning algorithms for intrusion detection. *IOP Conf. Ser. Mater. Sci. Eng.* 1013:012038, 1–8. doi: 10.1088/1757-899X/1013/1/012038

Eluri, R. K., Valicharla, K., Divya, M., and Anusha, K. B., "A scrutiny of machine learning methods for the detection and identification of cyber Intrusion," 2024 International Conference on Advances in Modern Age Technologies for Health and Engineering Science, Shivamogga, India: AMATHE, 11–14, (2024). doi: 10.1109/AMATHE61652.2024.10582241

Hidayat, I., Ali, M. Z., and Arshad, A. (2023). Machine learning-based intrusion detection system: an experimental comparison. *J. Comput. Cogn. Eng.* 2, 88–97. doi: 10.47852/bonviewJCCE2202270

John, A., Bin Isnin, I. F., Madni, S. H. H., and Muchtar, F. B. (2024). Enhanced intrusion detection model based on principal component analysis and variable ensemble machine learning algorithm. *Intell. Syst. Appl.* 24:200442. doi: 10.1016/j.iswa.2024.200442

Kurniabudi, D. S., Darmawijoyo, M. Y., Idris, B. B., Bamhdi, A. M., and Budiarto, R. (2020). CICIDS-2017 dataset feature analysis with information gain for anomaly detection. *IEEE Access* 8, 132911–132921. doi: 10.1109/ACCESS.2020.3009843

Liu, S., and Motani, M., "Improving mutual information based feature selection by boosting unique relevance," 1–13 (2022). Available online at: http://arxiv.org/abs/2212.06143

Megouache, L., Zitouni, A., Sadouni, S., and Djoudi, M. (2024). Machine Learning for Cloud Data Classification and Anomaly Intrusion Detection. *Ingen. Syst. Inform.* 29, 1809–1819. doi: 10.18280/isi.290514

Mehmood, M., Amin, R., Magboul, M., Muslam, A. L. I., Xie, J., and Aldabbas, H. (2023). Privilege escalation attack detection and mitigation in cloud using machine learning. *IEEE Access* 11, 46561–46576. doi: 10.1109/ACCESS.2023.3273895

Mghames, S. A. Z., and Ibrahim, A. A. (2023). Intrusion detection system for detecting distributed denial of service attacks using machine learning algorithms. *Indon. J. Electr. Eng. Comput. Sci.* 32, 304–311. doi: 10.11591/ijeecs.v32.i1.pp304-311

Nassif, A. B., Talib, M. A., Nasir, Q., Albadani, H., and Dakalbab, F. M. (2021). Machine learning for cloud security: a systematic review: Institute of Electrical and Electronics Engineers Inc. 9, 20717–20735. doi: 10.1109/ACCESS.2021.3054129

Ogwara, N. O., Petrova, K., and Yang, M. L. (2022). Towards the development of a cloud computing intrusion detection framework using an ensemble hybrid feature selection approach. *J. Comput. Netw. Commun.* 2022, 1–16. doi: 10.1155/2022/5988567

Parameswarappa, P., Shah, T., and Lanke, G. R. (2023). "A machine learning-based approach for anomaly detection for secure cloud computing environments," IDCIoT 2023 - International Conference on Intelligent Data Communication Technologies and Internet of Things, Proceedings, no. IDCIoT, Bengaluru, India: Institute of Electrical and Electronics Engineers (IEEE). 931–940.

Protić, D., and Stanković, M. (2023). Cybersecurity attacks: Which dataset should be used to evaluate an intrusion detection system? *Vojnotehnički* 71, 970–995. doi: 10.5937/vojtehg71-46524

Rawat, S., Srinivasan, A., Ravi, V., and Ghosh, U. (2022). "Intrusion detection systems using classical machine learning techniques versus integrated unsupervised feature learning and deep neural network". *Internet Technology Letters.* 5:e232. doi: 10.1002/itl2.232

Saran, M., Yadav, R. K., and Tripathi, U. N. (2022). Machine learning based security for cloud computing: a survey. *International Journal of Applied Engineering Research.* 17, 338–344. doi: 10.37622/IJAER/17.4.2022.338-344

Saranya, T., Sridevi, S., Deisy, C., Chung, T. D., and Khan, M. K. A. A. (2020). Performance analysis of machine learning algorithms in intrusion detection system: a review. *Proc. Comput. Sci.* 171, 1251–1260. doi: 10.1016/j.procs.2020.04.133

Somogyi, Z. (2021). Performance evaluation of machine learning models. *Appl. Artif. Intell.*, 87–112. doi: 10.1007/978-3-030-60032-7_3

Sundaramoorthy, K., Purushothaman, K. E., Jeba Sonia, J., and Kanthimathi, N. (2024). Enhancing cybersecurity in cloud computing and WSNs: a hybrid IDS approach. *Comput. Secur.* 147:104081. doi: 10.1016/j.cose.2024.104081

Tauscher, Z., Jiang, Y., Zhang, K., Wang, J., and Song, H., "Learning to detect: a data-driven approach for network intrusion detection," Conference Proceedings of the IEEE International Performance, Computing, and Communications Conference, Institute of Electrical and Electronics Engineers (IEEE). (2021).

Thapa, N., Liu, Z., Kc, D. B., Gokaraju, B., and Roy, K. (2020). Comparison of machine learning and deep learning models for network intrusion detection systems. *Future Internet* 12, 1–16. doi: 10.3390/fi12100167

Umar, M. A., Chen, Z., Shuaib, K., and Liu, Y. (2024). Effects of feature selection and normalization on network intrusion detection. *Data Sci. Manag.* 8, 23–39. doi: 10.1016/j.dsm.2024.08.001

Useni, D. E., Emmanuel, O. C., Job, G. K., and Ahmad, A. (2023). A review of machine learning-based algorithms for intrusion detection system, *International Journal of Engineering Research & Technology (IJERT)*, vol. 12, 251–256. Available at: https://www.ijert.org/research/a-review-of-machine-learning-based-algorithms-for-intrusion-detectionsystem-IJERTV12IS010082.pdf

Vibhute, A. D., Patil, C. H., Mane, A. V., and Kale, K. V. (2024). Towards detection of network anomalies using machine learning algorithms on the NSL-KDD benchmark datasets. *Proc. Comput. Sci.* 233, 960–969. doi: 10.1016/j.procs.2024.03.285

Zhang, B., Wang, Z., Li, H., Lei, Z., Cheng, J., and Gao, S. (2024). Information gain-based multi-objective evolutionary algorithm for feature selection. *Inf. Sci.* 677:120901. doi: 10.1016/j.ins.2024.120901

Zulifqar, I., Anayat, S., and Kharal, I. (2021). A Review of data security challenges and their solutions in cloud computing. *Int. J. Inform. Eng. Electro. Bus.* 13, 30–38. doi: 10.5815/ijieeb.2021.03.04