Systematic Evaluation for Machine Learning-Based Intrusion Detection on NSL-KDD: Feature Selection and Class-Imbalance Strategies

Mohammed Bekkouche[1*], Sidi Mohammed Benslimane[1]

[1]LabRI-SBA Lab., Ecole Superieure en Informatique, Sidi Bel Abbes, Algeria

*Corresponding author: m.bekkouche@esi-sba.dz

Abstract

Machine learning-based intrusion detection has received extensive attention using the NSL-KDD dataset; however, reported performance can vary substantially with the classifier, feature representation, class-distribution strategy, and classification setting. This study presents a systematic experimental evaluation covering binary and multiclass intrusion detection with nine machine learning models. The experimental framework considers seven feature-selection methods and twelve class-imbalance strategies, including oversampling, undersampling, hybrid, and cost-sensitive approaches, under a common evaluation protocol. Performance is evaluated using complementary metrics, including accuracy, balanced accuracy, precision, recall, F1-score, specificity, Kappa, ROC-AUC, PR-AUC, log-loss, and Brier score when applicable. A weighted multi-metric ranking procedure is used to compare complete experimental configurations.

The results reveal clear differences between binary and multiclass classification. Binary detection achieves stronger performance, with the best configuration reaching an F1-score of 0.9583 and a balanced accuracy of 0.9531, whereas the best multiclass configuration reaches an F1-score of 0.8548 and a balanced accuracy of 0.7265. Feature selection improves F1-score for all evaluated binary models, while its contribution varies across multiclass classifiers. Class-imbalance processing provides particularly strong improvements in the multiclass setting, where its average improvement exceeds that obtained through feature selection. The highest-ranked binary configuration combines Gradient Boosting, ANOVA selection, and SVM-SMOTE, achieving an overall score of 0.9931. In multiclass classification, the highest-ranked configuration combines Gradient Boosting, Random Forest selection, and SVM-SMOTE, achieving an overall score of 0.9841. The findings indicate that the most suitable configuration depends on the classifier and classification setting, supporting systematic evaluation across models, feature-selection methods, class-imbalance strategies, and complementary performance metrics.

Keywords

Intrusion Detection System, NSL-KDD, Machine Learning, Feature Selection, Class Imbalance.

1. Introduction

Intrusion Detection Systems (IDSs) are an important component of network security, as they support the identification of malicious activities and unauthorized behavior in computer networks [5]. Machine learning has been extensively investigated for IDS because learning models can identify patterns associated with normal and malicious network traffic [13]. Recent studies have explored classical machine learning, ensemble learning, and deep learning approaches for intrusion detection, including experiments conducted on the NSL-KDD dataset [4, 8, 9].

The effectiveness of a machine learning-based IDS depends on several experimental factors. In particular, the classification model, feature representation, feature-selection strategy, and treatment of class imbalance can substantially influence the resulting performance. Feature selection approaches have been investigated to reduce the dimensionality of NSL-KDD and retain informative attributes, using statistical, information-based, and model-based strategies [3, 11, 14, 15]. Similarly, resampling and ensemble-based approaches have been investigated to address the highly uneven distribution of attack categories [1, 6, 7]. These studies demonstrate that experimental choices beyond the classifier itself can play an important role in IDS performance.

The NSL-KDD dataset remains a widely used benchmark for intrusion detection research. It provides a common basis for evaluating models under both binary and multiclass settings. Previous studies have considered these settings using different machine learning and deep learning approaches [4, 15]. However, results reported across studies are difficult to compare directly because the selected models, feature-selection procedures, class-imbalance strategies, classification settings, and evaluation criteria may differ.

This issue motivates a systematic experimental investigation in which these factors are evaluated under a common framework. Rather than focusing on the development of a single new classifier, this study examines how different combinations of machine learning models, feature-selection methods, and class-imbalance strategies affect intrusion-detection performance on NSL-KDD. Both binary and multiclass classification are considered to provide a broader evaluation of the investigated configurations.

The experimental results show clear differences between the two classification settings. Binary classification achieves stronger performance than multiclass classification, with the best configuration reaching an F1-score of 0.9583 and a Balanced Accuracy of 0.9531, whereas the best multiclass configuration reaches an F1-score of 0.8548 and a Balanced Accuracy of 0.7265. Feature selection provides substantial improvements across binary classifiers, while class- imbalance strategies show a stronger average contribution in the multiclass setting. The highest-ranked binary configuration combines Gradient Boosting, ANOVA feature selection, and SVM-SMOTE, whereas the highest-ranked multiclass configuration combines Gradient Boosting, Random Forest feature selection, and SVM-SMOTE. These results indicate that the preferred configuration depends on both the classifier and the classification setting.

The contributions presented in this study are:

- A systematic evaluation of nine machine learning models for binary and multiclass intrusion detection using the NSL-KDD dataset.
- A comparative evaluation of seven feature-selection methods, covering filter and embedded approaches, with configurations using reduced feature representations.
- A systematic evaluation of twelve class-imbalance strategies, covering oversampling, undersampling, hybrid, and cost-sensitive approaches.
- A comprehensive evaluation using complementary metrics, including accuracy, balanced accuracy, precision, recall, F1-score, specificity, Kappa, ROC-AUC, PR-AUC, log-loss, and Brier score when applicable.
- A weighted multi-metric ranking procedure for comparing complete configurations and identifying high-performing combinations of classifiers, feature-selection methods, and class-imbalance strategies.
- A comparative analysis between binary and multiclass classification, showing differences in model performance and in the contribution of feature selection and class-imbalance strategies.
- An empirical analysis showing that the preferred configuration depends on the classifier and classification setting, with Gradient Boosting combined with ANOVA and SVM-SMOTE providing the strongest binary configuration and Gradient Boosting combined with Random Forest selection and SVM-SMOTE providing the strongest multiclass configuration.

The remainder of this paper is organized as follows. Section 2 reviews recent research on NSL-KDD-based intrusion detection and identifies the important limitations addressed by this study. Section 3 describes the experimental methodology. Section 4 presents the experimental results, with separate analyses for binary classification in Subsection 4.1 and multiclass classification in Subsection 4.2. Supplementary results are provided in Subsection 4.3. Section 5 discusses the important findings, while Section 6 concludes the paper and presents directions for future research.

2. Related Work

Machine learning-based intrusion detection has been extensively investigated using the NSL-KDD dataset, which remains a widely used benchmark for evaluating intrusion detection models. Recent research has considered classical machine learning, ensemble learning, deep learning, feature selection, and class-imbalance strategies, with substantial variation in experimental protocols and evaluation criteria.

Several recent studies have investigated machine learning and deep learning models for NSL-KDD intrusion detection. A hybrid Self-Organizing Map model combined dimensionality reduction with XGBoost for multiclass intrusion detection and reported improvements in precision, recall, and F1-score for several attack categories [4]. Another recent study proposed a sparse autoencoder with attention modules for feature representation and dimensionality reduction, followed by a multilayer perceptron classifier. The approach was evaluated on NSL-KDD together with CICIDS2017 and UNSW-NB15, demonstrating the potential of attention-based representation learning for intrusion detection [8]. Ensemble learning has also been investigated as a strategy for improving threat detection performance, highlighting the potential benefits of combining multiple classifiers [9].

Feature selection represents another important research direction in NSL-KDD-based intrusion detection. Statistical, information-based, correlation-based, and tree-based approaches have been investigated to reduce the feature space and improve classification performance [3, 11]. Other studies have combined feature selection with deep learning and ensemble models to obtain compact feature representations [8]. A hybrid intrusion detection architecture employed an autoencoder incorporating CNN and GRU components for dimensionality reduction and feature selection, followed by an enhanced LSTM-CNN residual network [14]. Although these approaches demonstrate the potential benefits of feature reduction, comparisons generally involve a limited number of feature-selection techniques. Systematic comparisons among several filter and embedded methods under a common experimental protocol remain limited.

Class imbalance represents another important challenge in NSL-KDD, particularly for the R2L and U2R categories. Several studies have investigated resampling and learning strategies to improve the recognition of minority attack classes. Oversampling techniques, including ADASYN and SMOTE-based approaches, as well as hybrid resampling methods, have been applied to address uneven class distributions [1, 7]. Other approaches, such as [6], have addressed both feature redundancy and class imbalance by combining feature selection with loss-based balancing strategies. Other research has explored ensemble-based and cost-sensitive approaches for imbalanced intrusion detection [7, 11]. These studies demonstrate the importance of class distribution in IDS evaluation. Nevertheless, comparative experiments covering a broader range of oversampling, undersampling, hybrid, and cost-sensitive strategies within a common framework remain limited.

The classification setting also represents an important experimental factor. Binary classification distinguishes normal traffic from attack traffic, whereas multiclass classification involves discrimination among Normal, DoS, Probe, R2L, and U2R categories. Several studies have evaluated one or both settings [1]. However, systematic comparisons between binary and multiclass settings while maintaining comparable models, feature-selection procedures, and class-imbalance strategies remain less explored.

Regarding evaluation criteria, recent NSL-KDD studies commonly report accuracy, precision, recall, and F1-score [4, 8, 11]. These measures provide complementary information about classifier performance. Nevertheless, additional criteria such as balanced accuracy, specificity, Kappa, ROC-AUC, and PR-AUC can provide further information, particularly under class imbalance. A multi-metric evaluation can therefore provide a broader characterization of IDS performance.

Another limitation concerns the joint analysis of experimental factors. Feature selection, class-imbalance strategy, and model choice are frequently investigated as separate components [1, 7, 11]. Consequently, fewer studies systematically evaluate their combined configurations under a common experimental protocol. Such an analysis is important because the performance associated with a feature-selection method may depend on both the learning model and the selected class-imbalance strategy.

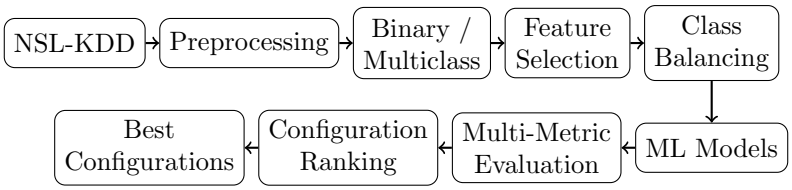

Figure 1. Experimental procedure for the systematic evaluation of machine learning-based intrusion detection configurations on the NSL-KDD dataset.

Overall, recent research confirms the potential of machine learning and deep learning for NSL-KDD intrusion detection. However, existing studies remain largely focused on individual models, specific feature-selection approaches, particular class-imbalance strategies, or individual experimental settings. The present study addresses this gap through a systematic evaluation combining multiple machine learning models, feature-selection methods, and class-imbalance strategies. The evaluation considers complementary performance metrics and supports both binary and multiclass classification. Furthermore, a multi-metric ranking procedure is employed to compare complete experimental configurations and identify configurations providing strong overall performance.

3. Experimental Methodology

3.1. Experimental Procedure

The experimental study follows a common procedure for evaluating machine learning-based intrusion detection configurations using the NSL-KDD dataset. The main experimental stages are presented in Fig. 1.

First, the NSL-KDD training and testing sets are loaded and processed using the preprocessing operations described in Section 3.3. Feature selection is subsequently applied to the training data, and the selected feature subset is used to represent both the training and testing data.

The selected class-balancing strategy is then applied to the training data. The resulting training set is used to train the selected machine learning model, while the original testing set is retained for independent evaluation.

The trained model is evaluated using a set of complementary performance metrics, including accuracy, balanced accuracy, precision, recall, F1-score, specificity, Kappa, ROC-AUC, PR-AUC, log-loss, and Brier score. The availability of these metrics depends on the outputs provided by each model. Metrics that cannot be computed for a given model are recorded as missing and are excluded from its score during the subsequent multi-metric ranking, while the corresponding metric remains available for other models when it can be computed.

Finally, each experimental configuration is characterized by its model, feature-selection method, class-balancing strategy, and number of selected features. A normalized multi-metric scoring procedure is then used to rank the configurations.

The same experimental procedure is applied to both binary and multiclass classification. The classification target differs between the two settings, while the main data processing, feature selection, class balancing, model training, evaluation, and ranking procedures remain consistent.

3.2. NSL-KDD Dataset

The experiments are conducted using the NSL-KDD dataset [10], a widely used benchmark for intrusion-detection research. The dataset provides network connection records described by 41 original attributes and an associated attack label. In the present study, the standard training and testing partitions, `KDDTrain+` and `KDDTest+`, are used without combining the two partitions. The training set contains 125,973 instances, while the testing set contains 22,544 instances.

The original feature space contains 41 attributes, including three categorical attributes (`protocol_type`, `service`, and `flag`) and 38 non-categorical attributes. The non-categorical attributes include binary and numerical variables describing connection-level, content-based, and traffic-related characteristics. The categorical attributes are transformed using one-hot encoding during preprocessing. Consequently, the resulting representation contains 123 features, as recorded by the experimental framework before feature-selection experiments.

For multiclass classification, the original attack labels are mapped into five classes: Normal, DoS, Probe, R2L, and U2R. The DoS category includes attacks such as `neptune`, `smurf`, and `teardrop`; Probe includes attacks such as `nmap`, `ipsweep`, and `portsweep`; R2L includes attacks such as `guess_passwd`, `ftp_write`, and `warezclient`; and U2R includes attacks such as `buffer_overflow`, `rootkit`, and `sqlattack`. For binary classification, the labels are reduced to two categories: Normal and Attack, where all attack categories are grouped into the Attack class.

The class distribution shows a substantial imbalance, particularly for the minority attack categories. In the training set, the five multiclass categories contain 67,343 Normal instances, 45,927 DoS instances, 11,656 Probe instances, 995 R2L instances, and only 52 U2R instances. The testing set contains 9,711 Normal instances, 7,167 DoS instances, 2,421 Probe instances, 3,178 R2L instances, and 67 U2R instances. This distribution provides an important motivation for investigating different class-imbalance strategies as part of the experimental evaluation.

Table 1 summarizes the principal characteristics of the dataset. The distinction between the original 41 attributes and the 123 features obtained after preprocessing is maintained to clearly separate the characteristics of the source dataset from the representation used by the machine-learning models.

Table 1. Characteristics of the NSL-KDD dataset used in the experiments.

Characteristic	Value
Training instances	125,973
Testing instances	22,544
Original attributes	41
Categorical attributes	3
Non-categorical attributes	38
Features after preprocessing	123
Binary classes	2
Multiclass classes	5

For the multiclass setting, the class distributions are reported in Table 2. The substantial difference between the majority and minority classes, particularly for U2R and R2L traffic, makes class imbalance an important experimental factor in the subsequent evaluation.

Table 2. Multiclass distribution in the NSL-KDD training and testing sets.

Class	Training	Testing
Normal	67,343	9,711
DoS	45,927	7,167
Probe	11,656	2,421
R2L	995	3,178
U2R	52	67
Total	125,973	22,544

3.3. Data Preprocessing

A preprocessing pipeline is applied to the NSL-KDD training and testing data before model training. The same preprocessing configuration is used throughout the experiments to ensure consistency

between the different machine-learning models, feature-selection methods, and class-imbalance strategies.

The preprocessing procedure consists of dataset validation, missing value treatment, categorical encoding, feature transformation, and feature scaling. Dataset validation is enabled to verify the expected schema and data consistency. Although mean imputation is configured, no missing values are detected in either the training or testing partition; therefore, the imputation step does not modify the available observations. Duplicate removal is disabled in the experimental configuration.

The three categorical attributes, namely `protocol_type`, `service`, and `flag`, are encoded using one-hot encoding. The remaining attributes are retained as numerical variables. No additional feature transformation is applied. Finally, standard scaling is used to normalize the resulting feature representation.

The preprocessing parameters used in the experiments are summarized in Table 3. After categorical encoding, the original 41 attributes result in a representation containing 123 features. This representation is subsequently provided to the feature-selection procedures. Thus, the number of features considered before feature selection is 123 rather than the original 41 attributes.

To prevent information from the testing partition from influencing the model-development process, preprocessing parameters are fitted using the training data and subsequently applied to the testing data. The processed training and testing sets therefore preserve the predefined NSL-KDD train/test separation.

Table 3. Preprocessing configuration used in the experiments.

Preprocessing step	Configuration
Dataset validation	Enabled
Duplicate removal	Disabled
Missing-value strategy	Mean
Categorical encoding	One-hot encoding
Feature transformation	None
Feature scaling	Standard scaling
Categorical attributes	3
Initial processed features	123

3.4. Classification Settings

Two classification settings are considered in the experimental study: binary classification and multiclass classification. Both settings use the same NSL-KDD training and testing partitions, while the target labels are defined differently according to the classification task.

Binary Classification. The binary setting reduces the original attack labels to two classes: Normal and Attack. Normal traffic is assigned to class 0, whereas all intrusion categories are grouped into class 1:

$$y = \begin{cases} 0, & \text{Normal}, \\ 1, & \text{Attack}. \end{cases} \tag{1}$$

The objective is to determine whether a network connection represents legitimate traffic or an intrusion, independently of the specific type of attack. This setting therefore evaluates the ability of the investigated models to detect malicious activity at a general level.

Multiclass Classification. The multiclass setting preserves the main attack categories in the NSL-KDD dataset. Five classes are considered: Normal, DoS, Probe, R2L, and U2R. The corresponding numerical encoding is defined as:

$$y = \begin{cases} 0, & \text{Normal}, \\ 1, & \text{DoS}, \\ 2, & \text{Probe}, \\ 3, & \text{R2L}, \\ 4, & \text{U2R}. \end{cases} \tag{2}$$

The objective is to distinguish legitimate traffic from malicious traffic while also identifying the category associated with each intrusion. Compared with the binary setting, this task is more challenging because the classifier must discriminate among attack categories with substantially different frequencies.

The two settings are evaluated using the same general experimental protocol, including preprocessing, feature selection, class-imbalance strategies, model training, and multi-metric evaluation. This configuration makes it possible to compare the performance of the investigated approaches under both general intrusion detection and attack-category identification.

3.5. Machine Learning Models

A diverse set of machine learning models is considered to evaluate intrusion-detection performance under different learning paradigms. The selected models include linear, tree-based, instance-based, probabilistic, and ensemble approaches. This diversity allows the impact of the classifier choice to be examined under the same preprocessing, feature-selection, and class-imbalance configurations.

The models evaluated in the experiments are listed in Table 4. Logistic Regression and SGD represent linear classifiers, while Decision Tree provides a tree-based baseline. Random Forest, Extra Trees, Gradient Boosting, and AdaBoost represent ensemble learning approaches. k-Nearest Neighbors provides an instance-based approach, whereas Naive Bayes represents a probabilistic classifier [2].

Table 4. Machine learning models evaluated in the study.

Model	Category
Logistic Regression	Linear
SGD	Linear
Decision Tree	Tree-based
Random Forest	Ensemble
Extra Trees	Ensemble
Gradient Boosting	Ensemble
AdaBoost	Ensemble
k-Nearest Neighbors	Instance-based
Naive Bayes	Probabilistic

The same set of classifiers is evaluated across the investigated feature-selection and class-imbalance configurations, allowing model performance to be compared under consistent experimental conditions.

3.6. Feature Selection

Feature selection is investigated to assess whether reducing the preprocessed feature space can maintain or improve intrusion-detection performance. Seven methods are considered and grouped into filter and embedded approaches [5, 12].

Filter Methods. Four filter methods are evaluated: Variance, Chi-square (Chi2), ANOVA, and Mutual Information. These methods select features according to statistical or information-based criteria without directly depending on a specific classification model. Variance removes features with low variability, while Chi-square and ANOVA measure the association between features and class labels. Mutual Information evaluates the dependency between each feature and the target variable.

Embedded Methods. Three embedded methods are considered: Random Forest, Extra Trees, and Logistic L1. These methods perform feature selection as part of the model-learning process. Tree-based methods use feature importance, whereas L1-regularized Logistic Regression favors a sparse feature representation by reducing the contribution of less informative features.

For each method, a fixed number of features is selected and the resulting feature subset is used in the subsequent classification experiments. The selected feature count is recorded for every experimental configuration to enable comparison between feature selection strategies.

3.7. Class-Imbalance Handling

The strong class imbalance present in NSL-KDD, particularly for the R2L and U2R categories, is considered as an experimental factor. Several strategies are evaluated to examine their influence on intrusion-detection performance. The investigated methods are grouped into four categories [5].

Oversampling. The evaluated oversampling methods are Random Over-Sampling, SMOTE, Borderline-SMOTE, SVM-SMOTE, and ADASYN. These methods increase the representation of minority classes through replication or synthetic sample generation.

Undersampling. Random Under-Sampling, NearMiss, Tomek Links, and Edited Nearest Neighbours are investigated to modify the class distribution by reducing or selecting samples from the majority classes.

Hybrid Methods. SMOTE-Tomek and SMOTE-ENN combine minority-class oversampling with sample-cleaning or majority-class reduction strategies.

Cost-Sensitive Learning. Class Weight is considered as a cost-sensitive strategy that assigns different importance to classes during model training rather than modifying the training samples.

These strategies are evaluated under the same experimental settings to compare their influence on classification performance across the investigated models and feature-selection methods.

3.8. Evaluation Metrics

The performance of the evaluated intrusion-detection models is assessed using complementary metrics that capture different aspects of classification performance. The metrics considered are accuracy, balanced accuracy, precision, recall, F1-score, specificity, Kappa, ROC-AUC, PR-AUC, log-loss, and Brier score. Their availability depends on the classification setting and the prediction outputs provided by each model.

For binary classification, the evaluation includes accuracy, balanced accuracy, precision, recall, F1-score, specificity, Kappa, ROC-AUC, log-loss, and Brier score. PR-AUC is not computed for the binary experiments. For multiclass classification, accuracy, balanced accuracy, precision, recall, F1-score, Kappa, ROC-AUC, and PR-AUC are considered, while specificity, log-loss, and Brier score are not computed.

For multiclass classification, precision, recall, F1-score, ROC-AUC, and PR-AUC are aggregated using weighted averaging across the five classes. This accounts for the different class frequencies while retaining the contribution of each attack category. For binary classification, these metrics are computed with respect to the attack class.

Accuracy

Accuracy measures the proportion of correctly classified instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3}$$

Balanced Accuracy

Balanced accuracy gives equal consideration to the classes by averaging their recall values. For binary classification:

$$BalancedAccuracy = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right). \tag{4}$$

For multiclass classification, it corresponds to the mean recall across the classes.

Precision, Recall, and F1-score

Precision measures the proportion of correctly identified positive instances among instances predicted as positive:

$$Precision = \frac{TP}{TP + FP}. \tag{5}$$

Recall measures the proportion of positive instances correctly identified:

$$Recall = \frac{TP}{TP + FN}. \tag{6}$$

The F1-score is the harmonic mean of precision and recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{7}$$

For binary classification, these measures are calculated with respect to the attack class. For multiclass classification, weighted averages across the five classes are used.

Specificity

Specificity measures the proportion of negative instances correctly classified:

$$Specificity = \frac{TN}{TN + FP}. \tag{8}$$

It is computed only for the binary classification setting and reflects the ability to correctly identify normal traffic.

Kappa

Kappa measures the agreement between predicted and reference classes while accounting for agreement expected by chance:

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \tag{9}$$

where p_o is the observed agreement and p_e is the expected agreement by chance.

ROC-AUC

ROC-AUC evaluates the discrimination ability of a classifier across different decision thresholds. It is computed when suitable probability estimates or decision scores are available. For multiclass classification, the corresponding multiclass formulation is used.

PR-AUC

PR-AUC summarizes the relationship between precision and recall across different decision thresholds. It is computed for multiclass experiments when the needed prediction scores are available. It is not included in the binary experiments.

Log-Loss and Brier Score

Log-loss evaluates the quality of predicted class probabilities, with lower values indicating better probabilistic predictions:

$$LogLoss = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right]. \tag{10}$$

The Brier score measures the squared difference between predicted probabilities and the corresponding binary outcomes:

$$Brier = \frac{1}{N} \sum_{i=1}^{N} (p_i - y_i)^2. \tag{11}$$

Both metrics are computed only for binary classification and when the needed probability estimates are available.

Metric Availability

Metric availability also depends on the evaluated model. In particular, the SGD-based classifier does not provide ROC-AUC, PR-AUC, log-loss, or Brier score within the experimental implementation. These metrics are therefore recorded as missing for SGD rather than estimated or replaced.

The resulting metric availability is preserved in the experimental results. Missing values are subsequently excluded when calculating the multi-metric configuration score, with the corresponding weights renormalized over the metrics available for each configuration.

3.9. Multi-Metric Configuration Ranking

Because no single performance metric completely characterizes the quality of an intrusion-detection system, a multi-metric ranking procedure is employed to compare the experimental configurations. Each configuration is defined by the combination of a classification model, a feature-selection method, a class-imbalance strategy, and a classification setting. The ranking procedure provides a single overall score while preserving the information provided by the individual evaluation metrics.

Let $\mathcal{M}$ denote the set of evaluation metrics considered in the experiment. Since the metrics have different scales and interpretations, their values are first normalized to the interval $[0, 1]$. For a higher-is-better metric, such as accuracy, F1-score, balanced accuracy, specificity, Kappa, ROC-AUC, and PR-AUC, the normalized value for configuration i and metric j is calculated as:

$$z_{ij} = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j)}. \tag{12}$$

For lower-is-better metrics, such as log-loss and Brier score, the normalization is reversed:

$$z_{ij} = \frac{\max(x_j) - x_{ij}}{\max(x_j) - \min(x_j)}. \tag{13}$$

This transformation ensures that a larger normalized value always corresponds to better performance, regardless of the original direction of the metric.

The minimum and maximum values used for normalization are calculated from the valid values available for each metric. If all valid configurations have the same value for a metric, the normalized value is assigned to the same constant value for all configurations for which that metric is available. This avoids division by zero during normalization.

The overall score of configuration i is then calculated using a weighted average of its available normalized metrics:

$$S_i = \frac{\sum_{j \in \mathcal{M}_i} w_j z_{ij}}{\sum_{j \in \mathcal{M}_i} w_j}, \tag{14}$$

where w_j denotes the weight assigned to metric j, and $\mathcal{M}_i$ represents the subset of metrics that is available for configuration i.

A metric is considered available at the experiment level when it is computed for at least one configuration. Consequently, a metric is not removed from the ranking simply because it cannot be computed for some models. For a particular configuration, however, a missing metric is excluded from its score. The weights associated with the remaining metrics are consequently renormalized through the denominator of the above equation.

Different weights are assigned to the evaluation metrics according to their relevance to intrusion-detection performance. The weights used in the ranking procedure are:

$$
\begin{aligned}
&w_{\text{accuracy}} = 0.03, \quad w_{\text{balanced\_accuracy}} = 0.14, \quad w_{\text{precision}} = 0.07,\\
&w_{\text{recall}} = 0.18, \quad w_{\text{F1}} = 0.20, \quad w_{\text{specificity}} = 0.03,\\
&w_{\text{Kappa}} = 0.10, \quad w_{\text{ROC-AUC}} = 0.06, \quad w_{\text{PR-AUC}} = 0.12,\\
&w_{\text{log-loss}} = 0.04, \quad w_{\text{Brier}} = 0.03.
\end{aligned}
\tag{15}
$$

The weights sum to one and give greater importance to F1-score, recall, balanced accuracy, PR-AUC, and Kappa, which provide useful information for intrusion detection under class-imbalanced conditions. Accuracy and specificity receive lower weights because they can provide less discriminative information when the class distribution is uneven. Probability-based metrics, including ROC-AUC, PR-AUC, log-loss, and Brier score, are assigned complementary weights when they are available.

For each configuration, only the metrics that can be computed for that configuration contribute to its overall score. The corresponding weights are automatically renormalized through the denominator of the weighted-score equation. Thus, configurations with unavailable metrics are not assigned artificial values for those metrics.

The number of metrics contributing to each configuration is also recorded. This value provides additional information about the comparability of the resulting scores, particularly when some models do not provide probability or decision-score outputs needed for metrics such as ROC-AUC, PR-AUC, log-loss, or Brier score.

Finally, configurations are sorted in descending order according to their overall score S_i. The configuration with the highest score receives rank 1. This procedure is applied to the experimental results to identify the strongest configurations across models, feature-selection methods, class-imbalance strategies, and classification settings.

4. Experimental Results

The experiments evaluate nine machine learning models under three configuration levels: a baseline using the complete preprocessed feature representation without feature selection or class-imbalance processing, configurations using feature-selection methods, and configurations using class-imbalance strategies. The same evaluation criteria and multi-metric ranking procedure are applied throughout the experiments. The results are reported separately for binary and multiclass intrusion detection.

4.1. Binary Classification Results

The binary experiments distinguish normal traffic from attack traffic. Starting from the baseline configuration with 123 input features, the analysis examines the performance changes produced by feature selection, class-imbalance strategies, and their combined use. The results first compare the machine learning models under the baseline setting, followed by feature-selection and class-imbalance configurations. A summary comparison and the highest-ranked configurations are then presented.

Comparison of Machine Learning Models

Table 5 presents the baseline results obtained without feature selection or class-imbalance processing. Gradient Boosting achieves the highest overall score (0.9375), followed by Decision Tree (0.9185), Random Forest (0.8386), and Extra Trees (0.8295). Decision Tree achieves the highest accuracy (0.8642), balanced accuracy (0.8768), recall (0.7856), and F1-score (0.8682), while Random Forest records the highest ROC-AUC (0.9778). These results provide the reference for the subsequent configurations.

Feature Selection Comparison

Table 6 reports the best feature-selection configuration for each model without class-imbalance processing. Feature selection improves the F1-score for all nine models relative to the corresponding baseline. The strongest result is obtained by Gradient Boosting with ANOVA, for which the F1-score increases from 0.8515 to 0.9220 and balanced accuracy increases from 0.8634 to 0.9201. Similar improvements are obtained for Extra Trees, KNN, SGD, Logistic Regression, and AdaBoost. These

Table 5. Binary classification baseline results without feature selection or class-imbalance strategy.

Model	Acc.	Bal. Acc.	Prec.	Rec.	F1	Spec.	Kappa	ROC-AUC	Log-Loss	Brier	Score	Rank
Gradient Boosting	0.8491	0.8634	0.9678	0.7602	0.8515	0.9665	0.7025	0.9730	0.3851	0.1064	0.9375	1
Decision Tree	0.8642	0.8768	0.9702	0.7856	0.8682	0.9681	0.7313	0.8768	4.8956	0.1358	0.9185	2
Random Forest	0.8032	0.8239	0.9705	0.6748	0.7961	0.9729	0.6175	0.9778	0.3959	0.1220	0.8386	3
Extra Trees	0.8021	0.8227	0.9690	0.6739	0.7949	0.9715	0.6152	0.9707	0.4088	0.1309	0.8295	4
SGD	0.8291	0.8400	0.9254	0.7612	0.8353	0.9189	0.6612	–	–	–	0.7933	5
Logistic Regression	0.8272	0.8382	0.9242	0.7587	0.8333	0.9177	0.6576	0.9122	0.8237	0.1503	0.7844	6
AdaBoost	0.8068	0.8206	0.9230	0.7207	0.8094	0.9205	0.6194	0.9521	0.5251	0.1690	0.7526	7
KNN	0.7948	0.8113	0.9288	0.6927	0.7936	0.9298	0.5981	0.8511	5.6134	0.1871	0.6762	8
Naive Bayes	0.5649	0.6170	0.9782	0.2410	0.3867	0.9929	0.2086	0.8148	15.6480	0.4350	0.1136	9

Note: "–" indicates that the corresponding metric was not available in the reported results. The overall score is calculated using the multi-metric ranking procedure and only the metrics available for each model.

improvements are obtained while reducing the representation from 123 features to 20 features for all methods except variance-based selection.

Table 6. Best feature-selection configuration for each machine learning model.

Model	Feature Selection	Features	Acc.	Bal. Acc.	Prec.	Rec.	F1	Spec.	Kappa	ROC-AUC	Log-Loss	Brier	Score	Rank
Gradient Boosting	ANOVA	20	0.9150	0.9201	0.9647	0.8829	0.9220	0.9574	0.8289	0.9768	0.2257	0.0655	0.9649	1
Random Forest	ANOVA	20	0.8765	0.8880	0.9736	0.8048	0.8812	0.9712	0.7549	0.9817	0.2834	0.0832	0.9028	4
Extra Trees	Logistic L1	20	0.8897	0.8983	0.9652	0.8364	0.8962	0.9601	0.7799	0.9512	1.5164	0.0879	0.9018	5
KNN	Logistic L1	20	0.8966	0.9040	0.9631	0.8510	0.9036	0.9570	0.7931	0.9277	2.6454	0.0944	0.8989	7
SGD	Logistic L1	20	0.9031	0.9036	0.9273	0.9004	0.9137	0.9067	0.8034	–	–	–	0.8834	8
AdaBoost	Logistic L1	20	0.9033	0.9042	0.9299	0.8978	0.9136	0.9106	0.8040	0.9560	0.5285	0.1702	0.8829	9
Decision Tree	ANOVA	20	0.8877	0.8979	0.9746	0.8243	0.8931	0.9716	0.7766	0.8997	3.9573	0.1109	0.8786	10
Logistic Regression	Logistic L1	20	0.8834	0.8867	0.9272	0.8629	0.8939	0.9105	0.7648	0.9322	0.4460	0.0935	0.8402	16
Naive Bayes	Logistic L1	20	0.8440	0.8529	0.9266	0.7884	0.8519	0.9174	0.6893	0.8833	3.0503	0.1521	0.7327	41

The complete results are provided in the supplementary document `S1-Binary-Model-FeatureSelection-CompleteResults.csv`.

Class-Imbalance Strategy Comparison

Table 7 presents the best balancing configuration for each model without feature selection. Class-imbalance strategies improve F1-score and balanced accuracy for all nine models relative to their corresponding baseline configurations. Gradient Boosting with SVM-SMOTE reaches an F1-score of 0.9073, while Random Forest with ADASYN reaches 0.8388. AdaBoost with Borderline-SMOTE and SGD with Borderline-SMOTE also show substantial improvements. These results indicate that different balancing strategies provide different levels of improvement depending on the classifier.

Table 7. Best class-imbalance configuration for each machine learning model.

Model	Balancing Method	Acc.	Bal. Acc.	Prec.	Rec.	F1	Spec.	Kappa	ROC-AUC	Log-Loss	Brier	Score	Rank
Gradient Boosting	SVM-SMOTE	0.9007	0.9083	0.9683	0.8535	0.9073	0.9631	0.8014	0.9691	0.2839	0.0724	0.9626	1
AdaBoost	Borderline-SMOTE	0.8652	0.8772	0.9664	0.7908	0.8698	0.9636	0.7330	0.9586	0.5297	0.1706	0.8752	9
Decision Tree	ENN	0.8745	0.8860	0.9719	0.8027	0.8792	0.9693	0.7510	0.8860	4.5246	0.1255	0.8656	12
Random Forest	ADASYN	0.8385	0.8547	0.9718	0.7378	0.8388	0.9717	0.6830	0.9796	0.3217	0.1060	0.8447	17
Extra Trees	ADASYN	0.8318	0.8488	0.9716	0.7256	0.8308	0.9720	0.6703	0.9755	0.3712	0.1141	0.8282	21
SGD	Borderline-SMOTE	0.8728	0.8776	0.9274	0.8425	0.8829	0.9128	0.7443	–	–	–	0.8223	23
Logistic Regression	SVM-SMOTE	0.8561	0.8632	0.9262	0.8119	0.8653	0.9145	0.7123	0.9347	0.6483	0.1216	0.7899	33
KNN	ADASYN	0.8365	0.8477	0.9340	0.7669	0.8423	0.9284	0.6759	0.8608	5.2255	0.1555	0.7173	78
Naive Bayes	NearMiss	0.6014	0.6488	0.9774	0.3069	0.4672	0.9906	0.2680	0.8212	14.1401	0.3983	0.2070	97

The complete results are provided in the supplementary document `S2-Binary-Model-ClassImbalance-CompleteResults.csv`.

Comparison Summary: Feature Selection and Class-Imbalance Strategies

Table 8 compares the best feature-selection and class-imbalance configurations with their corresponding baselines using Balanced Accuracy and F1-score. Feature selection provides larger improvements than class-imbalance processing alone for all nine models. The largest F1-score improvement is obtained for Naive Bayes, increasing from 0.3867 to 0.8519 ($\Delta F1 = +0.4652$). KNN, Extra Trees, and AdaBoost also show substantial F1-score improvements exceeding 0.10.

Table 8. Comparison of binary configurations using Balanced Accuracy and F1-score.

Model	Base Bal. Acc.	Base F1	Feature Selection	FS Bal. Acc.	FS F1	Δ Bal. Acc.	Δ F1	Balancing Method	Bal. Acc.	F1	Δ Bal. Acc.	Δ F1
Gradient Boosting	0.8634	0.8515	ANOVA	0.9201	0.9220	+0.0567	+0.0705	SVM-SMOTE	0.9083	0.9073	+0.0449	+0.0558
Decision Tree	0.8768	0.8682	ANOVA	0.8979	0.8931	+0.0211	+0.0249	ENN	0.8860	0.8792	+0.0092	+0.0110
Random Forest	0.8239	0.7961	ANOVA	0.8880	0.8812	+0.0641	+0.0851	ADASYN	0.8547	0.8388	+0.0308	+0.0427
Extra Trees	0.8227	0.7949	Logistic L1	0.8983	0.8962	+0.0756	+0.1013	ADASYN	0.8488	0.8308	+0.0261	+0.0359
SGD	0.8400	0.8353	Logistic L1	0.9036	0.9137	+0.0636	+0.0784	Borderline-SMOTE	0.8776	0.8829	+0.0376	+0.0476
Logistic Regression	0.8382	0.8333	Logistic L1	0.8867	0.8939	+0.0485	+0.0606	SVM-SMOTE	0.8632	0.8653	+0.0250	+0.0320
AdaBoost	0.8206	0.8094	Logistic L1	0.9042	0.9136	+0.0836	+0.1042	Borderline-SMOTE	0.8772	0.8698	+0.0566	+0.0604
KNN	0.8113	0.7936	Logistic L1	0.9040	0.9036	+0.0927	+0.1100	ADASYN	0.8477	0.8423	+0.0364	+0.0487
Naive Bayes	0.6170	0.3867	Logistic L1	0.8529	0.8519	+0.2359	+0.4652	NearMiss	0.6488	0.4672	+0.0318	+0.0805

Class-imbalance processing also improves both Balanced Accuracy and F1-score across all models, although the magnitude is generally smaller than that observed with feature selection. The strongest improvements are observed for Gradient Boosting, AdaBoost, and Naive Bayes. Gradient Boosting increases its F1-score from 0.8515 to 0.9073 with SVM-SMOTE, while AdaBoost increases from 0.8094 to 0.8698 with Borderline-SMOTE. These results indicate that feature selection provides a stronger individual contribution for the binary task, while class-imbalance processing provides additional improvements that vary according to the classifier.

The comparison also shows that the preferred strategy is model-dependent. ANOVA provides the strongest feature-selection configuration for Gradient Boosting, Random Forest, and Decision Tree, whereas Logistic L1 is preferred for Extra Trees, KNN, SGD, AdaBoost, Logistic Regression, and Naive Bayes. This variation indicates that no single preprocessing strategy consistently provides the highest performance across all models.

Best-Performing Setup for Each Machine Learning Model

Table 9 presents the best configuration obtained for each model when feature selection and class-imbalance strategies are considered jointly. Three F1-score differences are reported: the improvement relative to the baseline configuration, the best feature-selection-only configuration, and the best class-imbalance-only configuration.

Table 9. Best-performing configuration for each machine learning model with F1-score improvements relative to feature-selection, class-imbalance, and baseline configurations.

Model	Feature Selection	Balancing Method	F1	Δ F1 (FS)	Δ F1 (Balancing)	Δ F1 (Baseline)	ROC-AUC	Rank
Gradient Boosting	ANOVA	SVM-SMOTE	0.9583	+0.0363	+0.0510	+0.1068	0.9796	1
SGD	ANOVA	SVM-SMOTE	0.9327	+0.0190	+0.0498	+0.0974	-	9
Random Forest	ANOVA	NearMiss	0.9145	+0.0333	+0.0757	+0.1184	0.9825	21
AdaBoost	Logistic L1	SVM-SMOTE	0.9265	+0.0129	+0.0567	+0.1171	0.9473	33
Extra Trees	Logistic L1	ENN	0.9205	+0.0243	+0.0897	+0.1256	0.9457	39
Logistic Regression	ANOVA	SVM-SMOTE	0.9252	+0.0313	+0.0599	+0.0919	0.9250	42
Decision Tree	ANOVA	ENN	0.9206	+0.0275	+0.0414	+0.0524	0.9212	61
KNN	Logistic L1	SVM-SMOTE	0.9146	+0.0110	+0.0723	+0.1210	0.9290	83
Naive Bayes	Logistic L1	ENN	0.8559	+0.0040	+0.3887	+0.4692	0.8834	429

Every combined configuration has a positive $\Delta F1$ relative to both the best feature-selection-only configuration and the best balancing-only configuration. Thus, the two strategies provide complementary benefits rather than one consistently replacing the other.

Gradient Boosting provides the strongest combined configuration, with ANOVA and SVM-SMOTE producing an F1-score of 0.9583 and a ROC-AUC of 0.9796. Its F1-score increases by 0.0363 relative to the best feature-selection configuration and by 0.0510 relative to the best class-imbalance configuration. Random Forest, Extra Trees, and Logistic Regression also obtain substantial additional improvements, with F1-score increases over their balancing-only configurations of 0.0757, 0.0897, and 0.0599, respectively.

The contribution of the combined configuration is more limited for KNN, AdaBoost, and Decision Tree. Naive Bayes presents a distinct pattern: its combined configuration reaches an F1-score of 0.8559, but the additional improvement over its feature-selection-only configuration is only 0.0040. Most of its improvement is therefore associated with feature selection rather than the addition of the balancing strategy.

The results also show that the preferred balancing strategy can change when feature selection is introduced. For example, Random Forest uses ADASYN in the balancing-only experiment but NearMiss in its best combined configuration, while Extra Trees changes from ADASYN to ENN. These findings indicate that the two experimental factors interact and that the best configuration cannot always be identified by optimizing each factor independently.

Top-10 Binary Configurations

Table 10 presents the ten highest-ranked configurations according to the complete multi-metric ranking. The reported improvements are measured relative to the corresponding baseline models without feature selection or class-imbalance processing.

The strongest configuration is Gradient Boosting with ANOVA selection and SVM-SMOTE, which increases Balanced Accuracy from 0.8634 to 0.9531 and F1-score from 0.8515 to 0.9583. This configuration also achieves the highest ROC-AUC (0.9796) and overall score (0.9931).

Table 10. Top-10 binary configurations according to the multi-metric ranking.

Rank	Model	Feature Selection	Balancing Method	Features	Bal. Acc.	Δ Bal. Acc.	F1	Δ F1	ROC-AUC	Overall Score
1	Gradient Boosting	ANOVA	SVM-SMOTE	20	0.9531	+0.0897	0.9583	+0.1068	0.9796	0.9931
2	Gradient Boosting	ANOVA	Borderline-SMOTE	20	0.9404	+0.0770	0.9451	+0.0936	0.9784	0.9692
3	Gradient Boosting	Extra Trees	SVM-SMOTE	20	0.9371	+0.0737	0.9404	+0.0889	0.9745	0.9595
4	Gradient Boosting	ANOVA	ADASYN	20	0.9319	+0.0685	0.9410	+0.0895	0.9775	0.9594
5	Gradient Boosting	Extra Trees	ADASYN	20	0.9302	+0.0668	0.9332	+0.0817	0.9714	0.9455
6	Gradient Boosting	Logistic L1	SVM-SMOTE	20	0.9239	+0.0605	0.9300	+0.0785	0.9796	0.9425
7	Gradient Boosting	ANOVA	Random Over	20	0.9268	+0.0634	0.9294	+0.0779	0.9782	0.9423
8	Gradient Boosting	Logistic L1	ADASYN	20	0.9237	+0.0603	0.9296	+0.0781	0.9777	0.9409
9	SGD	ANOVA	SVM-SMOTE	20	0.9193	+0.0793	0.9327	+0.0974	–	0.9370
10	Gradient Boosting	ANOVA	Tomek	20	0.9228	+0.0594	0.9250	+0.0735	0.9771	0.9343

Gradient Boosting dominates the Top-10, appearing in nine configurations, while SGD appears once. ANOVA is frequently selected among the highest-ranked configurations, particularly with Gradient Boosting, and SVM-SMOTE is also repeatedly present. The largest improvements in Balanced Accuracy and F1-score are obtained by the first-ranked configuration, followed by the SGD configuration using ANOVA and SVM-SMOTE.

The results indicate that the strongest binary configurations generally combine feature reduction with class-imbalance processing. These findings support evaluating feature-selection and class-imbalance strategies together rather than considering either factor independently.

The complete results are provided in the supplementary document `S3-Binary-Model-FeatureSelection-ClassImbalance-CompleteResults.csv`.

4.2. Multiclass Classification Results

The multiclass experiments extend the analysis by distinguishing the different intrusion categories in NSL-KDD. Using the same experimental framework, the results examine the baseline models, feature-selection configurations, class-imbalance configurations, and their combinations.

Comparison of Machine Learning Models

Table 11 presents the baseline multiclass results. SGD and Logistic Regression provide the strongest overall rankings, with scores of 0.9752 and 0.9743, respectively. Logistic Regression achieves the highest accuracy (0.7877) and balanced accuracy (0.6473), whereas SGD records the highest F1-score (0.7559). Among the models with available ROC-AUC values, Random Forest obtains the highest value (0.9636).

The gap between accuracy and balanced accuracy across the models highlights differences in performance across the intrusion classes. Naive Bayes provides the weakest baseline results, with accuracy of 0.4423, balanced accuracy of 0.4066, F1-score of 0.4608, and Kappa of 0.1888. These baseline results provide the reference for the subsequent configurations.

Feature Selection Comparison

Table 12 summarizes the best feature-selection configuration for each model without class-imbalance processing. Feature selection improves balanced accuracy and F1-score for most models. The strongest

Table 11. Multiclass classification baseline results without feature selection or class-imbalance strategy.

Model	Acc.	Bal. Acc.	Prec.	Rec.	F1	Kappa	ROC-AUC	PR-AUC	Score	Rank
SGD	0.7845	0.6266	0.8163	0.7845	0.7559	0.6704	–	–	0.9752	1
Logistic Regression	0.7877	0.6473	0.8116	0.7877	0.7555	0.6740	0.9544	0.8782	0.9743	2
Extra Trees	0.7721	0.6190	0.8258	0.7721	0.7269	0.6403	0.9564	0.9187	0.9355	3
Gradient Boosting	0.7743	0.5963	0.8185	0.7743	0.7374	0.6465	0.9166	0.9053	0.9138	4
Random Forest	0.7573	0.5569	0.8099	0.7573	0.7069	0.6134	0.9636	0.9308	0.8645	5
AdaBoost	0.7808	0.5311	0.7300	0.7808	0.7240	0.6573	0.9272	0.8595	0.8235	6
KNN	0.7696	0.5939	0.7951	0.7696	0.7306	0.6412	0.8193	0.7013	0.7990	7
Decision Tree	0.7567	0.5604	0.7913	0.7567	0.7164	0.6209	0.7410	0.6359	0.7088	8
Naive Bayes	0.4423	0.4066	0.6123	0.4423	0.4608	0.1888	0.7139	0.5384	0.0000	9

improvement is observed for Logistic Regression with Extra Trees selection, increasing balanced accuracy from 0.6473 to 0.7629 and F1-score from 0.7555 to 0.8463. Gradient Boosting also shows a substantial increase, with balanced accuracy rising from 0.5963 to 0.7409 and F1-score from 0.7374 to 0.8274.

Table 12. Best feature-selection configuration for each machine learning model in multiclass intrusion detection.

Model	Feature Selection	Features	Acc.	Bal. Acc.	Prec.	Rec.	F1	Kappa	ROC-AUC	PR-AUC	Score	Rank
Logistic Regression	Extra Trees	20	0.8448	0.7629	0.8573	0.8448	0.8463	0.7707	0.9548	0.9916	0.9926	1
Gradient Boosting	Variance	122	0.8382	0.7409	0.8578	0.8382	0.8274	0.7556	0.9542	0.9041	0.9660	2
AdaBoost	Random Forest	20	0.8187	0.7418	0.8420	0.8187	0.8187	0.7304	0.9092	0.8501	0.9160	8
SGD	Variance	122	0.8012	0.7206	0.8282	0.8012	0.7912	0.7010	–	–	0.8989	12
Extra Trees	Variance	122	0.7853	0.6720	0.8333	0.7853	0.7461	0.6642	0.9645	0.9255	0.8646	16
Random Forest	Logistic L1	20	0.7835	0.6076	0.8202	0.7835	0.7594	0.6639	0.9336	0.8967	0.8197	21
KNN	Variance	122	0.7887	0.6905	0.7751	0.7887	0.7625	0.6781	0.8286	0.7128	0.7664	33
Naive Bayes	Random Forest	20	0.7017	0.6780	0.7914	0.7017	0.7419	0.5863	0.8735	0.7695	0.7265	37
Decision Tree	Logistic L1	20	0.7928	0.6188	0.8184	0.7928	0.7748	0.6814	0.7773	0.6841	0.7251	38

AdaBoost and SGD also show clear improvements, while Naive Bayes improves considerably despite its weak baseline performance. Several of the best configurations use only 20 features instead of the original 123. However, the results remain model-dependent, and feature selection does not improve every metric for every classifier.

The complete results are provided in the supplementary document

`S4-Multiclass-Model-FeatureSelection-CompleteResults.csv`.

Class-Imbalance Strategy Comparison

Table 13 presents the best class-imbalance configuration for each model without feature selection. Class-imbalance strategies improve balanced accuracy and F1-score for all evaluated models compared with their corresponding baselines. Random Forest with Random Under-Sampling achieves the highest balanced accuracy (0.7862), while Gradient Boosting with Random Over-Sampling reaches an F1-score of 0.8395. SGD, Extra Trees, and Logistic Regression also obtain clear improves in balanced performance.

Table 13. Comparison of multiclass classification models using their best class-imbalance strategy without feature selection.

Model	Balancing Method	Acc.	Bal. Acc.	Prec.	Rec.	F1	Kappa	ROC-AUC	PR-AUC	Score	Rank
Random Forest	Random Under	0.8388	0.7862	0.8722	0.8388	0.8360	0.7591	0.9700	0.9280	0.9946	1
Gradient Boosting	Random Over	0.8457	0.7539	0.8620	0.8457	0.8395	0.7686	0.9610	0.9150	0.9850	2
SGD	SVM-SMOTE	0.8207	0.7490	0.8478	0.8207	0.8177	0.7352	–	–	0.9631	9
Extra Trees	Random Under	0.8078	0.7404	0.8371	0.8078	0.7987	0.7095	0.9547	0.9011	0.9459	13
Logistic Regression	SVM-SMOTE	0.8220	0.7271	0.8369	0.8220	0.8077	0.7332	0.9506	0.8724	0.9450	14
AdaBoost	Random Under	0.7789	0.7270	0.8483	0.7789	0.8010	0.6834	0.9098	0.8178	0.9082	28
Decision Tree	SVM-SMOTE	0.8275	0.6641	0.8401	0.8275	0.8144	0.7409	0.8065	0.7332	0.8825	43
KNN	Borderline-SMOTE	0.7882	0.6900	0.8210	0.7882	0.7591	0.6741	0.8288	0.7163	0.8527	56
Naive Bayes	Random Under	0.7056	0.6700	0.7781	0.7056	0.7178	0.5864	0.8439	0.7140	0.7993	73

The preferred balancing method varies across classifiers. Random Under-Sampling is selected

for Random Forest, Extra Trees, AdaBoost, and Naive Bayes, while SVM-SMOTE is selected for SGD, Logistic Regression, and Decision Tree. Gradient Boosting performs best with Random Over-Sampling, and KNN with Borderline-SMOTE. This variation indicates that the strongest balancing strategy depends on the classifier.

The complete results are provided in the supplementary document `S5-Multiclass-Model-ClassImbalance-CompleteResults.csv`.

Comparison Summary: Feature Selection and Class-Imbalance Strategies

Table 14 provides a compact comparison between the baseline, the best feature-selection configuration, and the best class-imbalance configuration for each model using Balanced Accuracy and F1-score. Both approaches improve these metrics across all nine classifiers, although their relative contribution varies considerably.

Table 14. Comparison of multiclass classification configurations using Balanced Accuracy and F1-score.

Model	Base Bal. Acc.	Base F1	Feature Selection	FS Bal. Acc.	FS F1	Δ Bal. Acc.	Δ F1	Balancing Method	Bal. Acc.	F1	Δ Bal. Acc.	Δ F1
SGD	0.6266	0.7559	Variance	0.7206	0.7912	+0.0940	+0.0353	SVM-SMOTE	0.7490	0.8177	+0.1224	+0.0618
Logistic Regression	0.6473	0.7555	Extra Trees	0.7629	0.8463	+0.1156	+0.0908	SVM-SMOTE	0.7271	0.8077	+0.0798	+0.0522
Extra Trees	0.6190	0.7269	Variance	0.6720	0.7461	+0.0530	+0.0192	Random Under	0.7404	0.7987	+0.1214	+0.0718
Gradient Boosting	0.5963	0.7374	Variance	0.7409	0.8274	+0.1446	+0.0900	Random Over	0.7539	0.8395	+0.1576	+0.1021
Random Forest	0.5569	0.7069	Logistic L1	0.6076	0.7594	+0.0507	+0.0525	Random Under	0.7862	0.8360	+0.2293	+0.1291
AdaBoost	0.5311	0.7240	Random Forest	0.7418	0.8187	+0.2107	+0.0947	Random Under	0.7270	0.8010	+0.1959	+0.0770
KNN	0.5939	0.7306	Variance	0.6905	0.7625	+0.0966	+0.0319	Borderline-SMOTE	0.6900	0.7591	+0.0961	+0.0285
Decision Tree	0.5604	0.7164	Logistic L1	0.6188	0.7748	+0.0584	+0.0584	SVM-SMOTE	0.6641	0.8144	+0.1037	+0.0980
Naive Bayes	0.4066	0.4608	Random Forest	0.6780	0.7419	+0.2714	+0.2811	Random Under	0.6700	0.7178	+0.2634	+0.2570

Feature selection provides the largest Balanced Accuracy improvement for Naive Bayes (+0.2714), followed by AdaBoost (+0.2107) and Gradient Boosting (+0.1446). Naive Bayes also records the largest F1-score improvement (+0.2811). Class-imbalance processing produces its largest Balanced Accuracy improvement for Random Forest (+0.2293) and its largest F1-score improvement for Naive Bayes (+0.2570).

Across the nine models, class-imbalance processing provides a larger average improvement than feature selection for both Balanced Accuracy and F1-score. The average improvements are approximately +0.1522 and +0.0975, respectively, for class-imbalance processing, compared with +0.1217 and +0.0838 for feature selection. However, this pattern is not uniform across classifiers. Feature selection provides larger improvements for Logistic Regression and AdaBoost, whereas class-imbalance processing provides larger improvements for Random Forest, Extra Trees, Gradient Boosting, SGD, and Decision Tree. KNN shows very similar improvements from both approaches.

These results indicate that the relative contribution of feature selection and class-imbalance processing depends on the classifier. In particular, the stronger role played by class-imbalance processing in the multiclass setting suggests that uneven class distributions become more influential when the model must distinguish among several intrusion categories. The results therefore support evaluating both factors jointly rather than selecting one strategy independently.

Best-Performing Setup for Each Machine Learning Model

Table 15 presents the best complete configuration identified for each model after jointly considering feature-selection and class-imbalance strategies. The F1-score differences indicate the additional improvement relative to the best feature-selection configuration, the best class-imbalance configuration, and the baseline

The combined configurations consistently achieve higher F1-scores than the corresponding configurations using feature selection alone or class-imbalance processing alone, indicating that the two strategies can provide complementary improvements when applied together.

Random Forest exhibits the largest additional improvement relative to its best feature-selection-only configuration (+0.0844), followed by Extra Trees (+0.0580). For these models, introducing class balancing after feature selection provides a clear additional improvement. Gradient Boosting achieves the highest F1-score among the reported combined configurations (0.8548), although its additional improvements relative to feature selection (+0.0274) and balancing (+0.0153) are comparatively moderate because both individual configurations already provide strong performance.

Table 15. Best-performing setup for each machine learning model in multiclass intrusion detection, with F1-score improvements relative to feature-selection, class-imbalance, and baseline configurations.

Model	Feature Selection	Balancing Method	F1	Δ F1 (FS)	Δ F1 (Balancing)	Δ F1 (Baseline)	ROC-AUC	Rank
Gradient Boosting	Random Forest	SVM-SMOTE	0.8548	+0.0274	+0.0153	+0.1174	0.9599	1
Logistic Regression	Extra Trees	SMOTE-Tomek	0.8487	+0.0024	+0.0410	+0.0932	0.9543	2
Random Forest	Random Forest	Random Under	0.8438	+0.0844	+0.0078	+0.1369	0.9664	7
SGD	Variance	SVM-SMOTE	0.8188	+0.0276	+0.0011	+0.0629	–	26
Extra Trees	Variance	Random Under	0.8041	+0.0580	+0.0054	+0.0772	0.9606	41
AdaBoost	Random Forest	SMOTE	0.8187	-0.0000	+0.0177	+0.0947	0.9091	58
KNN	Extra Trees	Random Under	0.7727	+0.0102	+0.0136	+0.0421	0.8949	222
Decision Tree	Variance	SVM-SMOTE	0.8023	+0.0275	-0.0121	+0.0859	0.7966	234
Naive Bayes	Mutual Information	SVM-SMOTE	0.7490	+0.0071	+0.0312	+0.2882	0.8736	332

Logistic Regression shows a different pattern, with only a small additional improvement relative to feature selection (+0.0024) but a larger improvement relative to balancing (+0.0410). Extra Trees, SGD, and Random Forest similarly show larger additional improvements relative to their feature-selection-only configurations than relative to their balancing-only configurations. In contrast, KNN exhibits only small additional changes from the combined configuration.

Decision Tree is the only model for which the combined configuration provides a lower F1-score than the best balancing-only configuration, as indicated by $\Delta F1_{\text{Balancing}} = -0.0121$. AdaBoost also shows almost no additional improvement relative to its feature-selection-only configuration. These cases indicate that combining feature selection and class balancing does not necessarily produce an F1-score higher than both individual configurations.

Naive Bayes presents the largest improvement relative to the baseline (+0.2882), but only limited additional improvements relative to its feature-selection-only (+0.0071) and class-imbalance-only (+0.0312) configurations. Overall, the results indicate that the contribution from combining feature selection and class balancing is classifier-dependent, with some models benefiting substantially from the combination and others obtaining most of their improvement from one individual strategy.

Top-10 Multiclass Configurations

Table 16 presents the ten highest-ranked configurations from the complete experimental search space. The reported improvements are measured relative to the corresponding baseline models without feature selection or class-imbalance processing. The Top-10 contains configurations from Gradient Boosting, Logistic Regression, and Random Forest. All ten configurations outperform their corresponding baselines in both balanced accuracy and F1-score.

Table 16. Top-10 multiclass configurations and their improvements over the corresponding baseline models.

Rank	Model	Feature Selection	Balancing Method	Features	Bal. Acc.	Δ Bal. Acc.	F1	Δ F1	ROC-AUC	Overall Score
1	Gradient Boosting	Random Forest	SVM-SMOTE	20	0.7265	+0.1302	0.8548	+0.1174	0.9599	0.9841
2	Logistic Regression	Extra Trees	SMOTE-Tomek	20	0.7661	+0.1188	0.8487	+0.0932	0.9543	0.9841
3	Logistic Regression	Extra Trees	SMOTE-ENN	20	0.7639	+0.1166	0.8469	+0.0914	0.9540	0.9827
4	Logistic Regression	Extra Trees	SMOTE	20	0.7629	+0.1156	0.8463	+0.0908	0.9548	0.9819
5	Logistic Regression	Extra Trees	Random Over	20	0.7638	+0.1164	0.8457	+0.0902	0.9532	0.9805
6	Gradient Boosting	Variance	Random Over	122	0.7538	+0.1575	0.8395	+0.1021	0.9610	0.9799
7	Random Forest	Random Forest	Random Under	20	0.7381	+0.1812	0.8438	+0.1369	0.9664	0.9792
8	Logistic Regression	Extra Trees	SVM-SMOTE	20	0.7573	+0.1100	0.8476	+0.0921	0.9479	0.9781
9	Gradient Boosting	Mutual Information	SVM-SMOTE	20	0.6965	+0.1002	0.8443	+0.1069	0.9625	0.9744
10	Random Forest	Mutual Information	Random Under	20	0.7300	+0.1731	0.8389	+0.1320	0.9664	0.9729

The first-ranked configuration combines Gradient Boosting, Random Forest feature selection, and SVM-SMOTE. It achieves balanced accuracy of 0.7265, F1-score of 0.8548, ROC-AUC of 0.9599, and an overall score of 0.9841. Logistic Regression appears three times among the Top-10, with Extra Trees selection combined with SMOTE-Tomek, SMOTE-ENN, SMOTE, and other balancing strategies producing F1-scores between 0.8457 and 0.8487.

Random Forest with Random Forest selection and Random Under-Sampling achieves the largest balanced-accuracy improvement within the Top-10, increasing by 0.1812 relative to its baseline, while its F1-score increases by 0.1369. Gradient Boosting with Random Forest selection and SVM-SMOTE

achieves the highest F1-score among the Top-10 configurations.

The complete results are provided in the supplementary document `S6-Multiclass-Model-FeatureSelection-ClassImbalance-CompleteResults.csv`.

The multiclass results show that the strongest configurations combine feature selection with an appropriate class-imbalance strategy. Gradient Boosting and Logistic Regression dominate the highest-ranked configurations, while Random Forest provides a particularly strong improvement in balanced accuracy. The results also confirm that the preferred combination depends on the classifier, supporting a systematic comparison across models, feature-selection methods, and class-imbalance strategies.

4.3. Supplementary Results

Complete experimental results are provided in six supplementary CSV files. Each file contains performance metrics, the overall multi-metric score, and rankings.

`S1-Binary-Model-FeatureSelection-CompleteResults.csv`:
model–feature selection results.

`S2-Binary-Model-ClassImbalance-CompleteResults.csv`:
model–class-imbalance results.

`S3-Binary-Model-FeatureSelection-ClassImbalance-CompleteResults.csv`:
complete results for models, feature selection methods, and class-imbalance strategies.

`S4-Multiclass-Model-FeatureSelection-CompleteResults.csv`:
model–feature selection results.

`S5-Multiclass-Model-ClassImbalance-CompleteResults.csv`:
model–class-imbalance results.

`S6-Multiclass-Model-FeatureSelection-ClassImbalance-CompleteResults.csv`:
complete results for models, feature selection methods, and class-imbalance strategies.

Results are ranked using the overall multi-metric score.

5. Discussion

The experimental results show that intrusion-detection performance is strongly influenced by the classification setting, the feature representation, the class distribution, and the learning model. The results also indicate that the relative contribution of feature selection and class-imbalance processing changes between binary and multiclass classification. Consequently, model performance should be interpreted together with the experimental configuration rather than as an isolated property of the classifier.

5.1. Comparison Between Binary and Multiclass Classification

Binary classification provides substantially stronger performance than multiclass classification. In the binary setting, the best configuration reaches an F1-score of 0.9583 and a Balanced Accuracy of 0.9531, whereas the highest F1-score among the selected multiclass configurations is 0.8548, with a Balanced Accuracy of 0.7265. The difference is consistent with the greater discrimination needed in the multiclass task, where the model must distinguish Normal, DoS, Probe, R2L, and U2R traffic rather than a single Attack class.

The baseline results also show different model behavior across the two settings. Gradient Boosting and Decision Tree are among the strongest binary baseline models, while SGD and Logistic Regression obtain the highest baseline ranks in multiclass classification. This difference indicates that a classifier that performs strongly for Normal–Attack separation does not necessarily retain the same position when several attack categories must be distinguished.

The stronger gap between accuracy and Balanced Accuracy in the multiclass results further indicates uneven predictive performance across classes. This is particularly relevant for NSL-KDD because the minority categories are much less represented than the dominant classes. The multiclass results

therefore provide a more demanding test for both the classifier and the preprocessing configuration. The Top-10 multiclass configurations nevertheless improve Balanced Accuracy and F1-score relative to their corresponding baselines, with the strongest configuration reaching an F1-score of 0.8548 and a ROC-AUC of 0.9599.

5.2. Feature Selection

Feature selection is particularly effective in the binary classification setting. All nine models obtain higher F1-scores with their best feature-selection configuration than with the corresponding baseline. Gradient Boosting with ANOVA increases its F1-score from 0.8515 to 0.9220, while KNN, Extra Trees, and AdaBoost also obtain increases exceeding 0.10. Naive Bayes exhibits the largest change, with F1 increasing from 0.3867 to 0.8519. These improvements are obtained while reducing the representation to 20 features for the corresponding configurations.

The multiclass results show a more classifier-dependent pattern. Feature selection produces substantial improvements for several models, with the largest Balanced Accuracy and F1-score changes observed for Naive Bayes. AdaBoost and Gradient Boosting also show substantial improvements, while Logistic Regression achieves the highest-ranked feature-selection configuration using Extra Trees selection.

The preferred feature-selection method is therefore not consistent across classifiers or classification settings. ANOVA is particularly successful for Gradient Boosting and several binary configurations, while Logistic L1, Extra Trees, Random Forest, and Variance-based selection become more competitive in other settings. This suggests that feature relevance depends on the decision function learned by the classifier and on the target structure. Importantly, dimensionality reduction does not systematically degrade performance; in several cases, a 20-feature representation provides better results than the complete representation.

5.3. Class-Imbalance Strategies

Class-imbalance processing provides improvements for all nine models in the binary setting, although the magnitude differs across classifiers. Gradient Boosting with SVM-SMOTE, AdaBoost with Borderline-SMOTE, and SGD with Borderline-SMOTE show particularly strong improvements. The results therefore indicate that modifying the class distribution can improve the balance between detecting attacks and maintaining performance on the normal class.

The role of class imbalance becomes more pronounced in the multiclass setting. Across the nine classifiers, class-imbalance processing provides larger average improvements in both Balanced Accuracy and F1-score than feature selection. The mean improvements are approximately 0.1522 and 0.0975, respectively, compared with 0.1217 and 0.0838 for feature selection. Random Forest shows the largest Balanced Accuracy improvement with Random Under-Sampling, while Naive Bayes shows the largest F1-score improvement.

The selected balancing strategy varies substantially across models. Random Under-Sampling is preferred for several classifiers, whereas SVM-SMOTE, Random Over-Sampling, and Borderline-SMOTE provide the strongest configurations for other models. This variation indicates that the suitability of a balancing strategy depends on the classifier rather than being a property of the dataset alone.

The difference between the binary and multiclass results is important. Feature selection has the larger average contribution in the binary experiments, whereas class-imbalance processing has the larger average contribution in the multiclass experiments. This suggests that class distribution becomes particularly important when the classifier must distinguish multiple intrusion categories with substantially different class frequencies.

5.4. Contribution from Combining Both Strategies

The combined configurations provide additional F1-score improvements for most models, but their contribution is clearly classifier-dependent. In binary classification, every selected combined configura-

tion improves on both the best feature-selection-only and the best class-imbalance-only configuration. Gradient Boosting provides the strongest combined result, while Random Forest, Extra Trees, and Logistic Regression also obtain substantial additional improvements. The best binary configuration combines Gradient Boosting, ANOVA, and SVM-SMOTE, reaching an F1-score of 0.9583.

The multiclass results are more heterogeneous. Random Forest obtains the largest additional improvement relative to its feature-selection configuration, whereas Logistic Regression obtains a larger additional improvement relative to its balancing-only configuration. Extra Trees also benefits more from adding balancing to its selected feature representation. Conversely, the combined configuration for Decision Tree has a lower F1-score than its best balancing-only configuration, with $\Delta F1 = -0.0121$. AdaBoost shows almost no additional improvement relative to its feature-selection-only configuration. These cases demonstrate that combining the two strategies does not guarantee an improvement beyond every individual configuration.

The results also show that the best strategy cannot necessarily be selected independently. For example, the balancing method associated with the best combined configuration can differ from the balancing method that performs best without feature selection. This supports evaluating complete configurations rather than selecting the best method for each experimental factor separately.

5.5. Model Robustness Across Settings

Gradient Boosting is the most consistently strong classifier across the binary experiments, dominating the Top-10 binary configurations. It appears in nine of the ten highest-ranked binary configurations, with ANOVA and SVM-SMOTE forming the first-ranked configuration. The leading binary configuration improves Balanced Accuracy from 0.8634 to 0.9531 and F1-score from 0.8515 to 0.9583.

The multiclass ranking is more diverse. Gradient Boosting provides the highest F1-score among the model-specific combined configurations, followed by Logistic Regression and Random Forest. The Top-10 also contains repeated Logistic Regression configurations based on Extra Trees selection and SMOTE-based strategies. Random Forest provides a particularly strong improvement in Balanced Accuracy.

Thus, Gradient Boosting shows the strongest robustness across the two tasks, but the multiclass results demonstrate that other models become competitive when the feature representation and class distribution are adapted to the task. The ranking of classifiers should therefore not be generalized from binary detection to multiclass detection.

5.6. Implications of Multi-Metric Evaluation

The results also illustrate why evaluating a classifier using a single metric can lead to different conclusions. In the binary baseline, Decision Tree obtains higher Accuracy, Balanced Accuracy, Recall, and F1-score than Gradient Boosting, while Gradient Boosting provides stronger ROC-AUC and substantially better probabilistic metrics. The multi-metric ranking consequently places Gradient Boosting first and Decision Tree second.

This distinction is particularly important for intrusion detection. Accuracy alone can favor models that perform well on dominant classes, while F1-score emphasizes the precision–recall trade-off and Balanced Accuracy gives greater importance to the individual classes. ROC-AUC, log-loss, and Brier score provide additional information when the corresponding model outputs are available. The ranking procedure therefore provides a broader basis for configuration comparison than any single metric.

However, the ranking should be interpreted together with metric availability. Some models do not provide all probability- or decision-score-based metrics, so their scores are based only on the metrics that can be computed. The resulting rank represents the multi-metric configuration used in this study and should not be interpreted as a universal ordering of the classifiers.

5.7. Binary and Multiclass Configuration Sensitivity

A central finding is that the preferred configuration changes with the classification setting. The binary Top-10 is strongly concentrated around Gradient Boosting, ANOVA, and SVM-SMOTE, whereas the multiclass Top-10 includes Gradient Boosting, Logistic Regression, and Random Forest with a wider range of feature-selection and class-imbalance strategies.

This difference suggests that configuration selection should be performed separately for binary and multiclass intrusion detection. The feature subset and class distribution that improve the separation between Normal and Attack traffic may not retain the same discriminative information when the task requires separating DoS, Probe, R2L, and U2R. Accordingly, results obtained in one classification setting should not be directly transferred to the other without independent evaluation.

6. Conclusion

This study presented a systematic evaluation of machine learning-based intrusion detection using the NSL-KDD dataset, considering nine classifiers, seven feature-selection methods, several class-imbalance strategies, and both binary and multiclass classification. The results demonstrate that classifier performance depends substantially on the experimental configuration and that conclusions based on a single model or a single preprocessing strategy can be misleading.

Binary classification achieves considerably stronger performance than multiclass classification. The best binary configuration combines Gradient Boosting, ANOVA feature selection, and SVM-SMOTE, reaching an F1-score of 0.9583, a Balanced Accuracy of 0.9531, and a ranking score of 0.9931. In multiclass classification, the highest-ranked configuration combines Gradient Boosting, Random Forest feature selection, and SVM-SMOTE, reaching an F1-score of 0.8548 and a Balanced Accuracy of 0.7265. These results demonstrate that the classification setting substantially influences the suitability of a model and preprocessing configuration.

Feature selection consistently improves several binary configurations and provides substantial improvements for selected multiclass models. Class-imbalance processing has a particularly strong contribution in the multiclass setting, where its average improvement exceeds that obtained with feature selection. Neither factor has a universally optimal method: the preferred feature-selection and class-imbalance strategies vary according to the classifier and classification setting.

The experiments also show that combining feature selection with class-imbalance processing can provide additional improvements, but the magnitude of this additional contribution is classifier-dependent. In binary classification, the strongest configurations consistently benefit from the combination, whereas the multiclass results include cases where the combined configuration provides little additional improvement or performs below the best single-strategy configuration. This finding emphasizes the value of evaluating complete configurations rather than selecting preprocessing methods independently.

The use of complementary evaluation metrics is particularly important for NSL-KDD because of its uneven class distribution and the different types of performance captured by Accuracy, Balanced Accuracy, precision, recall, F1-score, Kappa, ROC-AUC, and probabilistic metrics when available. The multi-metric ranking provides a consistent framework for comparing the large number of configurations evaluated in this study while accounting for metric availability.

Future work will extend this evaluation toward model reliability and interpretability. Selective prediction will be investigated to allow models to abstain from uncertain predictions, while explainable artificial intelligence methods will be used to identify the factors behind model decisions and assess the consistency of feature contributions. These extensions will provide a further evaluation of whether the highest-performing intrusion-detection configurations are also reliable and interpretable in practical detection scenarios.

References

[1] Ahmed Abdelkhalek and Maggie Mashaly. Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning: A. abdelkhalek, m. mashaly. *The journal of Supercomputing*, 79(10):10611–10644, 2023.

[2] Ali Hussein Alshammari, Gergely Bencsik, and Almashhadani Hasnain Ali. A survey of six classical classifiers, including algorithms, methodological characteristics, foundational variants, and recent advances. *Algorithms*, 19(1), 2026.

[3] XingYu Gong, Yi Yang, Yi Zhang, Na Li, Yu Guan, and Rongkun Jiang. Feature selection method for network intrusion based on hybrid meta-heuristic dynamic optimization algorithm. *Computers & Security*, 156:104512, 2025.

[4] Noveela Iftikhar, Mujeeb Ur Rehman, Mumtaz Ali Shah, Mohammed JF Alenazi, and Jehad Ali. Intrusion detection in nsl-kdd dataset using hybrid self-organizing map model. *CMES-Computer Modeling in Engineering and Sciences*, 143(1):639–671, 2025.

[5] Yang Li, Zhengming Li, and Mengyao Li. A comprehensive survey on intrusion detection algorithms. *Computers and Electrical Engineering*, 121:109863, 2025.

[6] Taotao Liu, Yu Fu, Kun Wang, Xueyuan Duan, and Qiuhan Wu. A multiscale approach for network intrusion detection based on variance–covariance subspace distance and eql v2. *Computers & Security*, 148:104173, 2025.

[7] Ghalia Nassreddine, Mohamad Nassereddine, and Obada Al-Khatib. Ensemble learning for network intrusion detection based on correlation and embedded feature selection techniques. *Computers*, 14(3):82, 2025.

[8] Akanksha Pamena and Manohar Naik Sugali. Sae-am: Enhancing network intrusion detection using sparse autoencoders with attention modules. *Applied Cybersecurity & Internet Governance*, 2025.

[9] Ahmad Sanmorino, Rendra Gustriansyah, Shinta Puspasari, and Fauziah Afriyani. Improving threat detection in information security with ensemble learning. *Applied Cybersecurity & Internet Governance*, 4:art–no, 2025.

[10] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. Ieee, 2009.

[11] Ankit Thakkar and Ritika Lohiya. Fusion of statistical importance for feature selection in deep neural network-based intrusion detection system. *Information Fusion*, 90:353–363, 2023.

[12] Dipti Theng and Kishor K Bhoyar. Feature selection techniques for machine learning: a survey of more than two decades of research. *Knowledge and Information Systems*, 66(3):1575–1637, 2024.

[13] Zhendong Wang, Yaodi Liu, Daojing He, and Sammy Chan. Intrusion detection methods based on integrated deep learning model. *computers & security*, 103:102177, 2021.

[14] Yankun Xue, Chunying Kang, and Hongchen Yu. Hae-hrl: A network intrusion detection system utilizing a novel autoencoder and a hybrid enhanced lstm-cnn-based residual network. *Computers & Security*, 151:104328, 2025.

[15] Yujie Zhang and Zebin Wang. Feature engineering and model optimization based classification method for network intrusion detection. *Applied Sciences*, 13(16):9363, 2023.