Analyzing Ensemble Stacking Technologies for Supervised Anomaly Detection

Arcangelo Castiglione[1], Gianni D' Angelo[1(✉)], Francesco Palmieri[1], and Antonio Scarfó[2]

[1] Department of Computer Science, University of Salerno, Fisciano, Salerno, Italy
{arcastiglione,giadangelo,fpalmieri}@unisa.it
[2] Fortinet, Rome, Italy
ascarfo@fortinet.com

Abstract. Network Anomaly Detection (NAD) has emerged as one of the most effective approaches for timely identifying network-based cyber-attacks, offering clear advantages over traditional signature-based intrusion detection systems. The integration of advanced machine learning (ML) techniques can further enhance the adaptability and automation of detection mechanisms, improving their capacity to identify evolving and polymorphic threats. This study evaluates several supervised detection schemes employing ensemble-based ML models, specifically, bagging, boosting, and stacking, to assess their relative strengths and limitations. The analysis aims to develop novel architectures that exploit ensemble learning's capacity to combine diverse models, thereby improving detection performance even under highly imbalanced data conditions. Experimental results indicate that ensemble methods based on decision tree weak learners, such as Random Forest and XGBoost, achieve superior performance compared to both conventional classifiers and stacked ensembles, effectively reducing variance through bagging, mitigating bias via boosting, and enhancing overall prediction accuracy.

Keywords: Ensemble Learning · Network Anomaly Detection · Stacking · XGBoost · Random Forest · Imbalanced Datasets

1 Introduction

In modern cybersecurity, network intrusion and attack detection have become central to the protection and management of organizational infrastructures. Effective detection requires processing large-scale data, making big data analytics and machine learning (ML) essential for improving event classification and detection accuracy. Since no single ML model universally optimizes key performance indicators (KPIs) such as accuracy and robustness, ensemble learning offers a viable strategy by integrating multiple models to enhance performance [23] [28] [14]. However, existing ML-based detection methods still face

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026 Workshops, LNCS 16759, pp. 607–623, 2027.
https://doi.org/10.1007/978-3-032-30518-3_38

critical limitations: they often rely on hand-crafted feature selection, struggle with highly imbalanced traffic distributions, and lack robustness against evolving attack patterns. Moreover, many conventional classifiers suffer from high variance or bias when trained on noisy or incomplete network flow data, leading to degraded generalization in real-world deployments.

In supervised intrusion detection, ensemble methods mitigate overfitting and bias while increasing sensitivity and reducing false alarms. This study evaluates detection performance, measured through standard KPIs, of several homogeneous and heterogeneous weak learners combined via bootstrap aggregation, gradient boosting, or stacking. The analysis prioritizes integrated algorithmic frameworks but also explores hybrid stacked architectures using base-learners (level-0) and meta-learners (level-1). The proposed architectures are benchmarked on widely adopted labeled datasets to establish reliable performance references and model-selection guidelines for supervised network anomaly detection.

The core classifiers used in stacked models include Gaussian Naive Bayes, Decision Trees, Support Vector Machines, and Logistic Regression. Meanwhile, Random Forest and eXtreme Gradient Boosting (XGBoost) represent state-of-the-art ensemble approaches that leverage bagging and boosting, respectively.

Although stacking multiple heterogeneous learners improves performance, the integrated tree-based ensembles, Random Forest and XGBoost, consistently outperform traditional classifiers and stacked combinations. Bagging and boosting reduce variance and bias, respectively, resulting in superior predictive stability. Moreover, gradient boosting further minimizes error rates through regularization, pruning, and cross-validation, thereby combining the advantages of its base learners while mitigating their drawbacks. Both Random Forest and XGBoost exhibit scalability across diverse attack scenarios and varying class balance conditions.

The remainder of this paper is organized as follows: Sect. 2 reviews related work; Sect. 3 introduces the ensemble and baseline learning methods; Sect. 4 details the experimental architectures and datasets; Sect. 5 discusses results; and Sect. 6 presents the final remarks.

2 Related Work

Ensemble Learning is a machine learning methodology that combines several baseline models to create a model that outperforms them. Ensemble learning within a meta-layered architecture [34] has demonstrated significant improvements in base model performance across several domains [18], particularly in medical image and sentiment analysis. However, its adoption in network traffic and Network Anomaly Detection (NAD) remains limited, despite its suitability for supervised binary classification problems. According to [22], ensemble techniques rank among the top ten methods employed, with hybrid combinations frequently used for anomaly detection. Nonetheless, [35] notes that ensemble-based NAD studies represent only a small fraction compared to conventional machine learning approaches.

Within the meta-learning class, stacking is relatively uncommon yet predominant [18]. For instance, [13] proposes a stacking framework combining KNN, AdaBoost, and Random Forest, optimized via Logistic Regression. Similarly, [4] contrasts an SVM meta-classifier with nine base classifiers on the NSL-KDD dataset [8], showing that only some multi-classifier models outperform SVM across key performance metrics. In [9], XGBoost surpasses various supervised and semi-supervised methods for in-vehicle CAN Bus NAD using the Avante CN7 dataset.

Other works, e.g., [1], employ Naïve Bayes, ANN, J48, and REPTree in bagging, boosting, and stacking schemes on UNSW-NB15 [19], showing stacking achieves the best accuracy. According to [21], a collective prediction is achieved through majority voting among ANN, SVM, and MARS models. Likewise, [36] demonstrates that combining one-class classifiers (LGP, ANFIS, RF) on KDD-Cup 1999 [33] enhances detection accuracy by leveraging model complementarity. Using CICDDoS 2019 [16,30] shows that stacking-based ensembles outperform both traditional and other ensemble methods in detection accuracy and false alarm rate.

In web anomaly detection, [31] compares strong ensemble base learners (GBM, RF, XGB) on multiple datasets (CSIC-2010v2, CICIDS-2017, NSL-KDD, UNSW-NB15), revealing superior performance of stacked ensembles. Similarly, [26] confirms that meta-learning improves classification accuracy and reduces false positives using real traffic data. Finally, [11] highlights that ensemble models better detect complex attacks and minimize false alarms through multi-source correlation.

Overall, the cited works confirm that ensemble learning enhances the performance of both weak and strong base learners across various application domains.

3 Background

Algorithms and techniques adopted in our analysis are concisely introduced to provide the essential methodological background.

3.1 Ensemble Learning

Ensemble learning [34] combines multiple base models to enhance prediction accuracy. Two main strategies exist: **sequential** (models trained in series, each correcting previous errors) and **parallel** (models trained independently, aggregated via voting or averaging). Ensembles can be *homogeneous* or *heterogeneous* depending on whether the base learners are of the same type.
The main techniques relevant to this work are bagging, boosting, and stacking, defined as follows:

– **Bagging (Bootstrap Aggregation)**: a parallel approach that trains multiple models on random bootstrap samples of the dataset and aggregates their predictions. It effectively reduces variance and is well-suited for unstable learners such as Random Forests or Extra Trees.

– **Boosting**: a sequential technique that iteratively combines weak learners to minimize bias and residual errors. Misclassified instances receive higher weights in subsequent iterations. Variants include AdaBoost, Gradient Boosting Machine (GBM), XGBoost, LightGBM, and CatBoost. In gradient boosting [5], the model is updated iteratively to reduce prediction error:

$$F_m = F_{m-1} - \mu_m \cdot \nabla \varepsilon_m \tag{1}$$

where μ_m is the learning rate and ε_m denotes the prediction error at step m.
– **Stacking (Stacked Generalization)**: an ensemble method that integrates the predictions of several base-level (level-0) models using a meta-model (level-1). The meta-model, often a simple regressor or classifier (e.g., Linear or Logistic Regression), learns from the outputs of heterogeneous base learners to improve generalization and reduce bias or variance.

3.2 Logistic Regression

Logistic Regression [15] is a statistically interpretable model for binary classification. It employs the logistic (sigmoid) function to estimate the probability of the positive class, with parameters optimized via Maximum Likelihood Estimation to maximize the likelihood of the observed outcomes.

3.3 Support Vector Machines

Support Vector Machines (SVMs) [6] are supervised learning models designed to find the optimal separating hyperplane that maximizes the margin between classes in a high-dimensional feature space. Kernel functions (e.g., linear, polynomial, radial-basis, sigmoid) enable nonlinear mapping. The optimization process minimizes the norm of the weight vector subject to margin constraints, relying on *support vectors*, i.e., samples near the decision boundary that define the classifier.

3.4 Decision Tree and Random Forest

A Decision Tree [12] partitions the feature space through hierarchical decisions, where internal nodes represent feature tests and leaves denote class labels. Features appearing at upper tree levels typically have greater discriminative importance. Pruning reduces overfitting by eliminating weak branches.
Random Forest [2] extends this principle through a bagging-based ensemble of multiple decision trees, each trained on a random subset of data/features. Aggregating these predictors via majority voting (classification) or averaging (regression) lowers variance, enhances generalization, and mitigates overfitting. This was selected as the most effective bagging-based solution in our analysis.

3.5 Gaussian Naive Bayes

The Gaussian Naive Bayes (GNB) classifier assumes conditional independence among features and Gaussian-distributed variables. Based on Bayes' theorem, the predicted class is determined by:

$$pred = \arg\max_{class} p(class|features) = \frac{p(features|class) \cdot p(class)}{p(features)} \quad (2)$$

Despite its simplicity, GNB is computationally efficient and performs effectively on high-dimensional data.

3.6 eXtreme Gradient Boosting

eXtreme Gradient Boosting (XGBoost) [5] extends gradient boosting by incorporating second-order optimization (Newton-Raphson method) and advanced regularization to enhance scalability and convergence. It constructs an ensemble of regularized decision trees that sequentially correct previous residuals, thereby improving stability and preventing overfitting. XGBoost's support for parallel and distributed computing makes it suitable for large-scale datasets, motivating its adoption as the reference gradient-boosting solution in this study.

4 The Analysis Framework

Our analysis framework aims to assess the capabilities of different ensemble-based ML architectures to inspect network traffic flows in multiple flow-based anomaly detection scenarios and identify attack-related flows that occur among normal ones. It essentially consists of a supervised binary classification framework in which training is typically performed on a highly unbalanced baseline, as in a typical traffic observation/capture scenario, where the number of normal traffic flows (negatives in the supervisory signal) significantly exceeds that of anomalous ones (positives).

4.1 Architectural Layout

The implemented architectures are designed to address distinct analysis tasks through tailored compositions of Stacking, Bootstrap Aggregation, and Boosting techniques. These configurations aim to evaluate both individual learners and their combinations across four detection scenarios, each tested on different datasets. The overall structure is illustrated in the four schemes of Fig. 1, each corresponding to a specific evaluation task.

In particular, Fig. 1a presents the comparison among the three primary ensemble approaches-bagging, boosting, and stacking. For this purpose, Random Forest and XGBoost were selected as representative models for bootstrap aggregation and optimized boosting, respectively. Their integration within a unified algorithmic framework ensures a balance between optimization efficiency

and architectural simplicity, yielding stable and effective solutions. The stacking ensemble adopts a two-level configuration: Logistic Regression, Gaussian Naive Bayes, Decision Tree, and SVM serve as level-0 learners, whose predictions are combined by a Logistic Regression meta-learner at level 1. This evaluation not only benchmarks stacking against bagging and boosting but also investigates the benefits of aggregating heterogeneous learners with varying complexity and predictive accuracy. Given the difficulty of identifying the optimal single model for a dataset, stacking offers a straightforward alternative that integrates multiple learners, often achieving superior overall performance.

The second configuration (Fig. 1b) compares standalone bagging (Random Forest) and boosting (XGBoost) models with their stacked combination, where a Logistic Regression meta-learner aggregates their outputs to assess whether this hybridization enhances predictive performance.

The third scheme (Fig. 1c) focuses exclusively on stacking, comparing (i) pure stacking, where level-0 learners are trained independently and the meta-classifier learns from their predictions, and (ii) passthrough stacking, where the meta-model is trained on both base model predictions and original features, providing contextual information for improved combination. Both stacking variants are evaluated for the configurations in Fig. 1a and Fig. 1b.

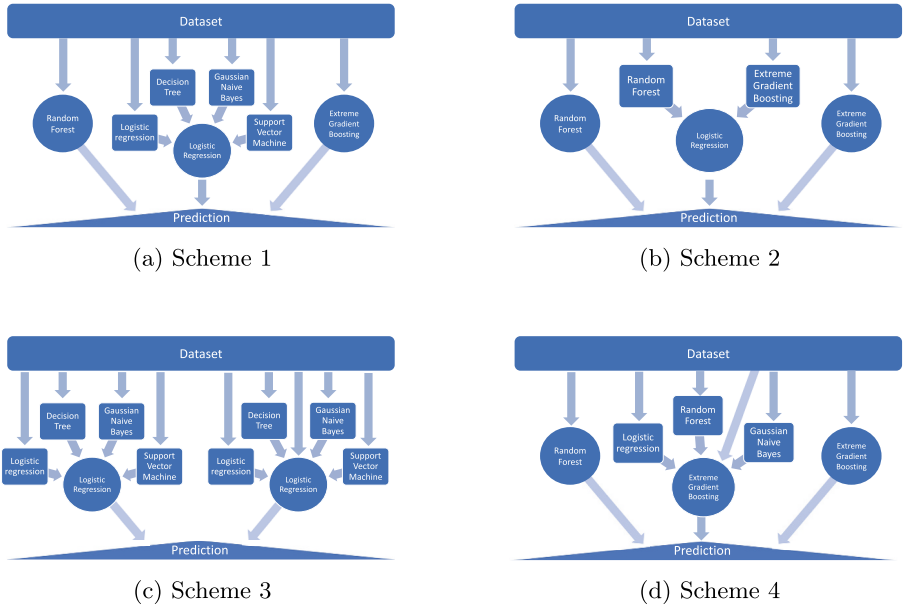
Finally, the fourth scheme (Fig. 1d) explores the use of a boosted meta-classifier (XGBoost) within a passthrough stack comprising heterogeneous learners (e.g., Logistic Regression, Naive Bayes) and ensemble methods (e.g., Random Forest), to assess its effectiveness relative to pure bagging or boosting architectures.

4.2 Datasets

In all conducted experiments, the objective was to ensure comparability with existing state-of-the-art studies; therefore, we employed the most widely adopted benchmark datasets in the field [27]. Although numerous datasets provide labeled flows and raw packet data, only a few are recognized as standard references in the field of anomaly and intrusion detection research. This is primarily due to limitations in traffic generation methods, testing environments, and data realism, which must align with the evolving complexity of modern networks and emerging cybersecurity threats.

The first dataset used was NSL-KDD [32] [8] [33], developed by the Canadian Institute for Cybersecurity as an enhanced and de-duplicated version of the DARPA Lincoln Labs KDD'99 dataset [7]. It contains labeled flows extracted from several weeks of simulated U.S. Air Force LAN traffic using the Bro IDS [24], including 24 attack types and 41 features per flow, divided into training and testing subsets. Despite criticism regarding its artificial nature [3], KDD99 remains among the most frequently used datasets in intrusion detection research [22].

The second dataset, UNSW-NB15 [19], was created at UNSW Canberra to address major deficiencies in KDD99 and NSL-KDD [20], such as outdated traffic patterns and unbalanced data distributions. Generated in a controlled environment in 2015, it comprises over two million labeled flows, each with 49 features,



(a) Scheme 1 (b) Scheme 2

(c) Scheme 3 (d) Scheme 4

Fig. 1. Layout of the architectures used in the analysis.

extracted using the Bro IDS. Approximately 87.3% of the records correspond to normal traffic and 12.7% to anomalous activity.

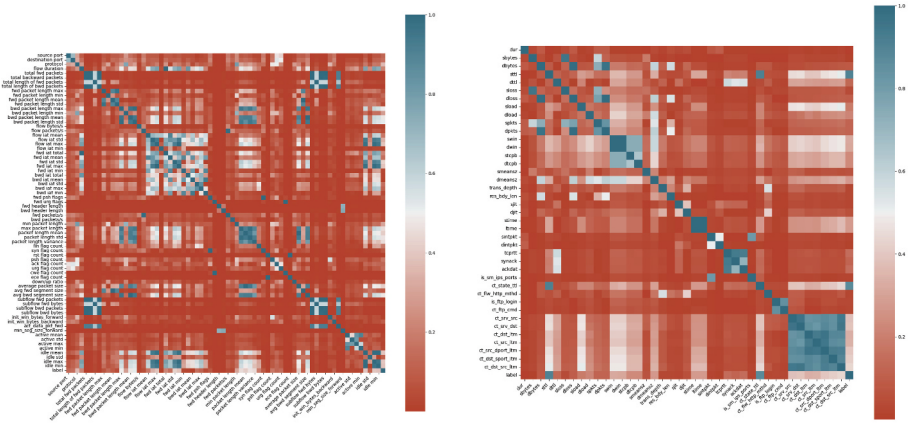
Lastly, the CIC-IDS-2017 dataset [29], also from the Canadian Institute for Cybersecurity, encompasses five days of realistic network traffic across a full-scale topology including firewalls, switches, routers, and heterogeneous hosts (Windows, Ubuntu, macOS). Using CICFlowMeter [10] [17], more than 80 flow features were extracted from both intra-LAN and Internet communications, covering multiple attack scenarios mixed with regular network behavior. The dataset comprises nearly three million flows, with 80.3% labeled as normal and 19.7% labeled as anomalous.

In our experiments, for NSL-KDD, we used the already provided split. For both UNSW-NB15 and CIC-IDS-2017, 70% of the entire dataset was used for training, with the remaining part serving as the testing reference.

4.3 Features

All the features that are obviously unnecessary (source and destination IP addresses of flows, timestamps, and attack labels) for classification were not considered by preliminarily removing them from the feature set. In addition, to optimize the effectiveness and responsiveness of the overall ML architecture, it is necessary to reduce the problem dimensions by minimizing the number of features used for both training and testing, according to a strong diversification

principle that prioritizes the choice of highly discriminative and mutually independent features. This implies the determination of a more compact knowledge representation, also improving predictiveness through the reduction of redundancy in feature space, since a set of fully independent features, when considered together, is more effective in performing binary classification. To provide an example at a glance of feature correlation, we report in Fig. 2 the correlation heatmaps related to the CIC-IDS-2017 and UNSW-NB15 datasets, respectively.



(a) CIC-IDS-2017 (b) UNSW-NB15

Fig. 2. Correlation heatmaps.

According to the previous considerations, some feature engineering has been performed on the original datasets. More precisely, in the presence of two or more highly correlated features (characterized by a mutual correlation greater than 0.95), we only kept one of them and removed all the others. The features removed by the datasets of interest are summarized in Table 1.

Finally, all numeric feature values have been normalized to the range $[0, 1]$ through standard scaling, and categorical features have been remapped into ordered numeric values, preserving the original order information.

5 Performance Evaluation and Results Analysis

In this section, we describe the results of the experiments performed in our analysis to evaluate the effectiveness of the available ensemble-based detection approaches by also comparing them to determine the best options in terms of configuration as well as the more interesting properties characterizing their behavior, together with their weaknesses, depending on the specific usage scenario.

Table 1. Feature Engineering Details

Dataset	Removed Features
NSL-KDD	'su attempted' (15), 'serror rate' (25), 'rerror rate' (27), 'dst host srv diff host rate' (37), 'dst host serror rate' (38), 'dst host rerror rate' (40), 'dst host srv rerror rate' (41)
CIC-IDS-2017	'flow id', 'source ip', 'source port', 'destination ip', 'destination port', 'timestamp', 'attack cat', 'total backward packets', 'total length of bwd packets', 'fwd packet length std', 'bwd packet length mean', 'bwd packet length std', 'fwd iat total', 'fwd iat max', 'fwd packets/s', 'packet length std', 'syn flag count', 'cwe flag count', 'ece flag count', 'average packet size', 'avg fwd segment size', 'avg bwd segment size', 'subflow fwd packets', 'subflow fwd bytes', 'subflow bwd packets', 'subflow bwd bytes', 'idle mean', 'idle max', 'idle min'
UNSW-NB15	'srcip', 'sport', 'dstip', 'dsport', 'attack cat', 'sloss', 'dloss', 'dpkts', 'dwin', 'ltime', 'ct srv dst', 'ct src dport ltm', 'ct dst src ltm'

A simple testbed has been implemented in `Python` by using the `scikit-learn` machine learning framework on an Intel Xeon-based server equipped with double 2.2Ghz CPUs and 160 GB RAM. The most significant hyperparameters of all the learners used within the assessed ensemble have been tuned by using grid search (i.e., exhaustive exploration of all the combinations in specific ranges or sets of values) 5-fold cross-validation for selecting the best values to be used in all the runs.

To ensure reproducibility, we report the optimal hyperparameters obtained via grid search with 5-fold cross-validation:

- **Random Forest**: `n_estimators=200`, `max_depth=20`, `min_samples_split=5`.
- **XGBoost**: `learning_rate=0.1`, `n_estimators=150`, `max_depth=6`, `gamma=0`, `reg_alpha=0.1`, `reg_lambda=1`.
- **Logistic Regression**: `C=1.0`, `solver='lbfgs'`.
- **SVM**: `C=1.0`, `kernel='rbf'`, `gamma='scale'`.
- **Gaussian Naive Bayes** and **Decision Tree** (`max_depth=10`): default scikit-learn parameters were used unless otherwise tuned.

All other ensemble-specific parameters were kept at their default values as provided by the adopted machine learning frameworks [25].

Common performance metrics used to evaluate machine learning-based classification models have been considered in all experiments performed within the

analysis framework. In particular, Precision, Recall, and F1-score, determined from the confusion matrix, have been examined for both the negative and positive classes.

5.1 Scheme 1 Experiments

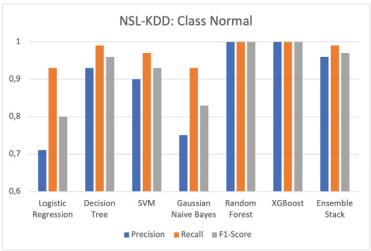
In the analysis scenario depicted in Fig. 1a, the results presented in Figs. 3, 4, and 5 demonstrate that ensemble-based approaches consistently outperform conventional classifiers (SVM, Gaussian Naive Bayes, Logistic Regression, Decision Tree). Among them, Random Forest and XGBoost achieved the highest accuracy across all evaluated datasets, with XGBoost showing a slight advantage on larger and more imbalanced datasets (UNSW-NB15 and CIC-IDS-2017). The superior performance of these ensemble methods arises from their inherent algorithmic characteristics. Random Forest mitigates variance and overfitting by averaging multiple decorrelated decision trees generated via bootstrap aggregation, while feature randomization enhances robustness to missing data and improves generalization. Conversely, XGBoost constructs successive shallow trees in a boosting framework, iteratively correcting previous misclassifications. Its pruning strategy and the exploitation of feature distribution statistics optimize split selection and reduce computational complexity. Regularization terms-γ (minimum loss reduction for split creation), α (L1), and λ (L2)-further balance bias and variance by penalizing model complexity, which were tuned in this study to enhance predictive performance.

The stacking ensemble, although marginally inferior to Random Forest and XGBoost, produced results comparable to those of the strongest base learner (typically the Decision Tree) while surpassing the performance of individual models. This improvement stems from the complementary learning of level-0 models trained on distinct data partitions, which effectively combines diverse prediction patterns and reduces correlated errors.

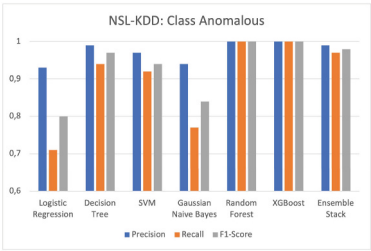
Finally, all models exhibited higher accuracy on the negative class, except for NSL-KDD. This behavior is attributed to the pronounced class imbalance in the other two datasets, where negative samples comprise approximately 80% of the training data.

5.2 Scheme 2 Experiments

Regarding the analysis scenario reported in Fig. 1b, from the results provided in Figs. 6, 7, and 8, we can immediately appreciate that no performance gain is introduced by stacking the XGBoost and Random Forest models on all three reference datasets. The reason is that both algorithms are already highly optimized and directly achieve a performance that is extremely difficult to improve. Hence, the benefits introduced by stacking are either imperceptible or negligible. Indeed, the main aim of stacking is to complement the strengths of one learner with the weaknesses of another and vice versa, in a logic of mutual support between the involved individual learners. However, in our case, both the

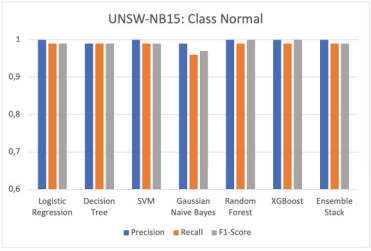


(a) Negative Class.

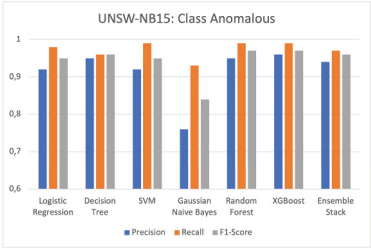


(b) Positive Class.

Fig. 3. NSL-KDD dataset Per-class precision, recall, and F1-score for Scheme 1 (bagging: Random Forest; boosting: XGBoost; stacking vs. conventional classifiers: SVM, GNB, LR, DT).

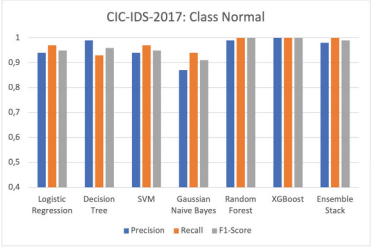


(a) Negative Class.

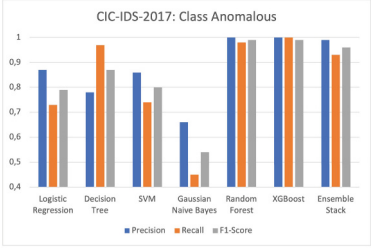


(b) Positive Class.

Fig. 4. UNSW-NB15 dataset Per-class performance comparison for Scheme 1. Ensemble methods (RF, XGB) outperform individual baselines across both classes.



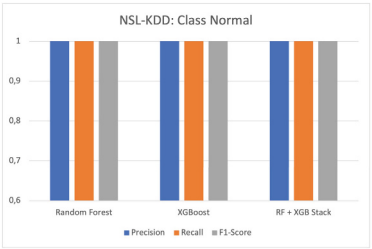
(a) Negative Class.



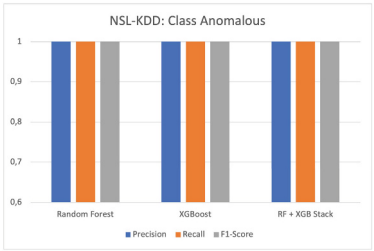
(b) Positive Class.

Fig. 5. CIC-IDS-2017 dataset Scheme 1 results. XGBoost shows a slight advantage over RF on the imbalanced positive class.

XGBoost and Random Forest algorithms achieve nearly perfect accuracy on all the datasets, leaving no room for mutual support.

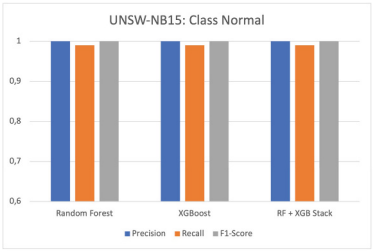


(a) Negative Class.

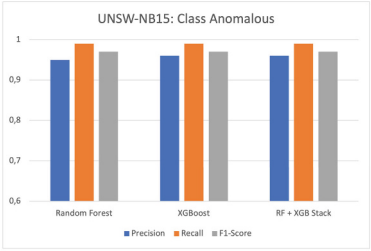


(b) Positive Class.

Fig. 6. NSL-KDD: Per-Class Scheme 2 performance.

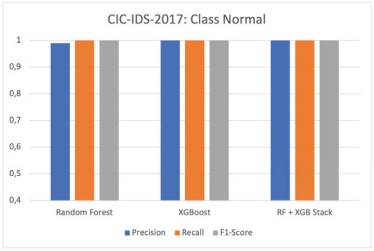


(a) Negative Class.

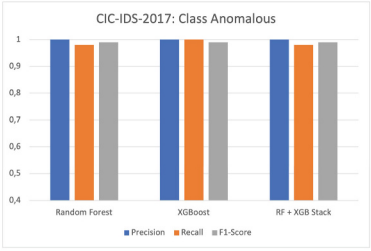


(b) Positive Class.

Fig. 7. UNSW-NB15: Per-Class Scheme 2 performance.



(a) Negative Class.

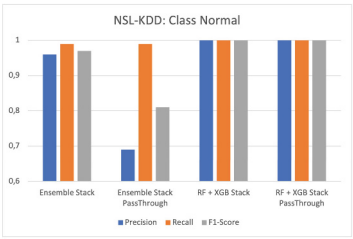


(b) Positive Class.

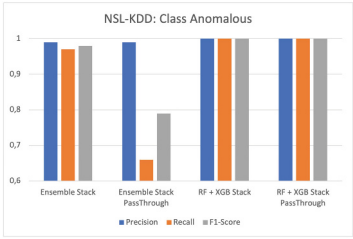
Fig. 8. CIC-IDS-2017: Per-Class Scheme 2.

5.3 Scheme 3 Experiments

Regarding the analysis scenario reported in Fig. 1c, from the results provided in Fig. 9, Fig. 10 and Fig. 11 we can see that training the metamodel in the stacking architectures on the predictions made by the base models merged with the training original data, leads in several cases to worse results, mainly when the weaker learners are used in the level-0 (this effect is less evident, but also present, when the XGBoost and Random Forest models are stacked. This may

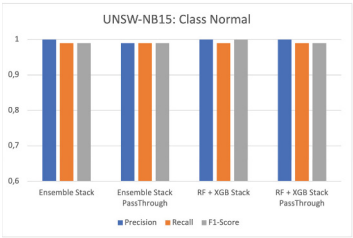


(a) Negative Class.

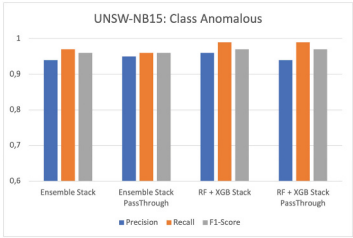


(b) Positive Class.

Fig. 9. NSL-KDD: Per-Class Scheme 3 performance.

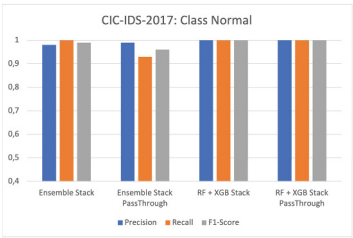


(a) Negative Class.

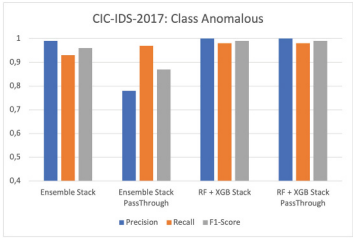


(b) Positive Class.

Fig. 10. UNSW-NB15: Per-Class Scheme 3 performance.



(a) Negative Class.



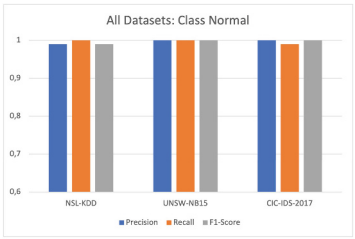
(b) Positive Class.

Fig. 11. CIC-IDS-2017: Per-Class Scheme 3 performance.

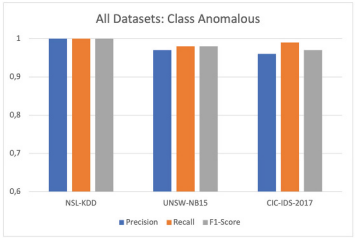
be due to the heterogeneity of data in the original datasets compared to those from the individual learners' predictions.

5.4 Scheme 4 Experiments

Finally, regarding the analysis scenario reported in Fig. 1d, the same considerations made in the two previous cases can be extended to the results provided in Fig. 12, where, for simplicity, the data relative to the passthrough stacking ensemble have been aggregated on a single chart for all three datasets. Here, it is immediate to see how the use of XGBoost as a meta-classifier in the stacked



(a) Negative Class.



(b) Positive Class.

Fig. 12. All Datasets: Per-Class Scheme 4 performance.

configuration, taking as its input the predictions of the level-0 classifiers (Logistic Regression, Random Forest, Gaussian Naive Bayes) merged with the original dataset, does not introduce any advantage and, in some cases, its effectiveness is also slightly perturbated by the effect of the weaker level-0 learners (Gaussian Naive Bayes).

6 Conclusions and Future Work

This study conducts a comparative evaluation of various ensemble-based machine learning (ML) architectures for analyzing network traffic flows across multiple flow-based anomaly detection scenarios, to identify attack-related flows concealed among legitimate ones. The analysis primarily focused on selecting optimal ensemble configurations, favoring approaches integrated within single algorithmic frameworks, while also exploring their use as base (level-0) or meta (level-1) learners in stacked architectures. The resulting reference detection models were assessed using widely recognized labeled datasets in the field, i.e., NSL-KDD, UNSW-NB15, and CIC-IDS-2017, to provide practical guidance for selecting suitable supervised learning models for network anomaly detection.

The performance evaluation confirmed the superior effectiveness of ensemble methods, even under conditions of significant class imbalance. Most ensemble-based classifiers enhanced the predictive performance of their constituent models, particularly when the latter exhibited limited standalone accuracy. Despite the capacity of stacked ensembles to integrate complementary learning strategies, the results demonstrated that well-designed single-algorithm ensembles, specifically, Random Forest (bagging-based) and XGBoost (boosting-based), outperformed both traditional classifiers and their stacked counterparts. These models achieved optimal detection performance by reducing variance through bagging, mitigating bias via boosting, and thereby improving overall prediction quality. However, it is worth noting that stacking may still prove beneficial in contexts where the level-0 models are weak learners (e.g., simple classifiers with high bias or variance), rather than already-optimized ensembles like XGBoost or Random Forest.

Our evaluation has several limitations. First, experiments were conducted on simulated, though widely used, datasets (NSL-KDD, UNSW-NB15, CIC-IDS-

2017); real-world network traffic may exhibit different distributions and concept drift. Second, we only considered supervised binary classification; multi-class attack identification and semi-supervised scenarios were not tested. Third, computational efficiency and inference latency, that is critical for real-time deployment, were not benchmarked. Finally, hyperparameter values were optimized per dataset and may not generalize to other environments.

Addressing these aspects is part of our future work. Also, future research could explore semi- or self-supervised ensemble frameworks to reduce dependence on large labeled datasets. Incorporating automated feature engineering and deep representation learning may enable the extraction of more discriminative patterns from raw network flows. Efforts should also focus on improving computational efficiency via model distillation or online learning for real-time applications. Strengthening robustness against adversarial evasion and concept drift, alongside integrating explainable AI for interpretability, remains essential. Extending these approaches to cross-domain and multimodal detection tasks would further enhance generalizability and practical utility.

Acknowledgments. This work was partially supported by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.

References

1. Belouch, M., El Hadaj, S.: Comparison of ensemble learning methods applied to network intrusion detection. In: Proceedings of the 2nd International Conference on Internet of Things, Data and Cloud Computing (ICC 2017), pp. 1–4. ACM, New York, NY, USA (2017)
2. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
3. Brugger, S.T., Chow, J.: An assessment of the DARPA ids evaluation dataset using snort, UC Davis, Department of Computer Science (2007). Tech. rep
4. Chand, N., Mishra, P., Krishna, C.R., Pilli, E.S., Govil, M.C.: A comparative analysis of SVM and its stacking with other classification algorithm for intrusion detection. In: 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA), pp. 1–6. Dehradun, India (2016)
5. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016), pp. 785–794. ACM, New York, NY, USA (2016)
6. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. DARPA: KDD cup 1999 data set (1999). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
8. Mohi-ud din, G.: NSL-KDD. IEEE Dataport (2018)
9. Dong, Y., Chen, K., Peng, Y., Ma, Z.: Comparative study on supervised versus semi-supervised machine learning for anomaly detection of in-vehicle can network. In: IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), pp. 2914–2919. Macau, China (2022)

10. Drapper Gil, G., Lashkari, A.H., Mamun, M., Ghorbani, A.A.: Characterization of encrypted and VPN traffic using time-related features. In: 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016), pp. 407–414 (2016)
11. Folino, G., Sabatino, P.: Ensemble based collaborative and distributed intrusion detection systems: a survey. J. Netw. Comput. Appl. **66**, 1–16 (2016)
12. Furnkranz, J.: Decision tree. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning. Springer, Boston, MA (2011). https://doi.org/10.1007/978-0-387-30164-8
13. Hai, T., Hoang, L., Huh, E.: Network anomaly detection based on late fusion of several machine learning algorithms. Int. J. Comput. Netw. Commun. **12**, 117–131 (2020)
14. Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE Trans. Pattern Anal. Mach. Intell. **12**, 993–1001 (1990)
15. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer, New York (2001). https://doi.org/10.1007/978-0-387-84858-7
16. Khoei, T.T., Aissou, G., Hu, W.C., Kaabouch, N.: Ensemble learning methods for anomaly intrusion detection system in smart grid. In: 2021 IEEE International Conference on Electro Information Technology (EIT), pp. 129–135. Mt. Pleasant, MI, USA (2021)
17. Lashkari, A.H., Draper-Gil, G., Mamun, M.S.I., Ghorbani, A.A.: Characterization of tor traffic using time based features. In: 3rd International Conference on Information System Security and Privacy, Porto, Portugal (2017)
18. Mohammed, A., Kora, R.: A comprehensive review on ensemble deep learning: opportunities and challenges. J. King Saud Univ. Comput. Inf. Sci. **35**(2), 757–774 (2023)
19. Moustafa, N., Slay, J.: UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS), pp. 1–6 (2015)
20. Moustafa, N., Slay, J.: The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Inf. Secur. J. A Global Perspect. **25**(1–3), 18–31 (2016)
21. Mukkamala, S., Sung, A.H., Abraham, A.: Intrusion detection using an ensemble of intelligent paradigms. J. Netw. Comput. Appl. **28**(2), 167–182 (2005)
22. Nassif, A.B., Talib, M.A., Nasir, Q., Dakalbab, F.M.: Machine learning for anomaly detection: a systematic review. IEEE Access **9**, 78658–78700 (2021)
23. Oza, N.C., Tumer, K.: Classifier ensembles: select real-world applications. Inf. Fus. **9**(1), 4–20 (2008)
24. Paxson, V.: Bro: a system for detecting network intruders in real-time. Comput. Netw. **31**(23), 2435–2463 (1999)
25. Pedregosa, F., Varoquaux, G., Gramfort, A., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
26. Possebon, I.P., Silva, A.S., Granville, L.Z., Schaeffer-Filho, A., Marnerides, A.: Improved network traffic classification using ensemble learning. In: IEEE Symposium on Computers and Communications (ISCC), pp. 1–6. , Barcelona, Spain (2019)
27. Ring, M., Wunderlich, S., Scheuring, D., Landes, D., Hotho, A.: A survey of network-based intrusion detection data sets. Comput. Secur. **86**, 147–167 (2019)
28. Schapire, R.E.: The strength of weak learnability. Mach. Learn. **5**(2), 197–227 (1990)

29. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: 4th International Conference on Information Systems Security and Privacy (ICISSP) (2018)
30. Talukder, M.A., Uddin, M.A.: CIC-ddos2019 dataset. Mendeley Data, V1 (2023)
31. Tama, B.A., Nkenyereye, L., Islam, S.M.R., Kwak, K.S.: An enhanced anomaly detection in web traffic using a stack of classifier ensemble. IEEE Access **8**, 24120–24134 (2020)
32. Tavallaee, M.: NSL-KDD dataset (2012). http://www.iscx.ca/NSL-KDD
33. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD cup 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, pp. 1–6. ON, Canada, Ottawa (2009)
34. Valiant, L.G.: A theory of the learnable. Commun. ACM **27**, 1134–1142 (1984)
35. Vanerio, J., Casas, P.: Ensemble-learning approaches for network security and anomaly detection. In: Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (Big-DAMA 2017), pp. 1–6. ACM, New York, NY, USA (2017)
36. Zainal, A., Maarof, M.A., Shamsuddin, S.M., Abraham, A.: Ensemble of one-class classifiers for network intrusion detection system. In: 4th International Conference on Information Assurance and Security, pp. 180–185. Naples, Italy (2008)