International Journal of Computing Sciences and Technology Studies
https://journals.hu.edu.so/index.php/ijcsts

Ensemble Machine Learning for Intrusion Detection: A Comparative Study on Accuracy, False Positives and Feature Importance

Mohamed Elmi Ahmed
Faculty of Computer Science and Information Technology, Hormuud University, Mogadishu, Somalia

Email: mohamedelmi2019@gmail.com

Abstract
Cyber threats are becoming increasingly sophisticated, and signature-driven Intrusion Detection Systems (IDS) are less effective at detecting new attacks. Machine learning (ML)-based Intrusion Detection Systems (IDS) offer an alternative, as these solutions can learn patterns from historical data. However, class imbalance, false positives, and interpretability issues hinder effective deployment. In this study, five ML classifiers were evaluated on a dataset of 9,537 records with 11 features (e.g., session duration, login attempts, and IP reputation), including Logistic Regression, Random Forest, XGBoost, Support Vector Machine (SVM), and Neural Networks. Random Forest yielded the best overall accuracy rate (88.1%), best F1-score (0.849), and lowest false-positive (0.010) classification rates. It also appeared to generalize better than XGBoost and SVM, which performed with 87.2% and 86.6% accuracy, respectively. Logistic Regression and Neural Networks had higher false-positive rates (0.240 and 0.166) and offered lower classification detection performance than XGBoost, Random Forest, and SVM. Notably, feature importance analysis revealed that authentication patterns (failed logins) and IP reputation were important features that affected the detection of attacks. In conclusion, this study highlights that ensemble ML methods (Random Forest, XGBoost) offer a reasonable trade-off between accuracy and interpretability for real-world deployment in an IDS. Future work could focus on adversarial robustness, temporal modeling, and hybrid detection methods to bridge signature-driven and anomaly-driven detection. Overall, this study contributes to the development of explainable and adaptive IDS for effective network security.

Keywords: Intrusion Detection System (IDS), Machine Learning (ML), Cybersecurity, Random Forest, XGBoost, Support Vector Machine (SVM), False Positive Rate, Feature Importance

1. Introduction

With the rise of cyber threats, there is a demand for Intrusion Detection Systems (IDS) capable of defending digital assets. Traditional signature-based IDS solutions offer a very small defense line against threats that are constantly changing. Actors become more sophisticated by using obfuscating and evasion techniques, and the risk from zero-day or novel attacks is high. The limitations of these systems and their inability to adapt have led to an abundance of research on alternative adaptive solutions, such as machine learning (ML)

methods, which can be trained on historical data and learn to discriminate between legitimate and malicious activities by exploiting spatial and temporal patterns [1], [2].

ML-based IDSs have demonstrated the potential to detect anomalies in computing activities by using statistical patterns inherent in network behavior [3]. However, many established studies in the literature have limited generalizability because of the use of a single dataset or limited prediction metrics. Hindy et al.'s [4] taxonomy and survey of intrusion detection systems indicated that many of the challenges noted in previous research, such as class imbalance, false positive rate, and lack of transparency in model behavior, have not vanished.

In this study, we address the limitations identified by prior researchers by evaluating five commonly used ML classifiers–Logistic Regression, Random Forest, XGBoost, Support Vector Machine (SVM), and Neural Networks–using a publicly available dataset comprising 9,537 records and 11 descriptive features used to generalize behaviors such as session duration, number of logon attempts, use of encrypted connections, reputation of IP address, and timestamps of access. The dataset was employed in the accuracy evaluation of classifiers with other performance metrics, such as detection accuracy, F1-score, ROC AUC, precision, recall, and false-positive rate. Among the models, Random Forest emerged with the best results, with an accuracy of 88.1%, an F1-score of 0.849, and a very low false-positive rate of 0.010. The evidence suggests a considerable potential for generalization. XGBoost and SVM scores were close behind Random Forest with overall accuracy at 87.2% and 86.6% respectively, and similarly low false positive rates of 0.034 and 0.032 respectively. The overall results indicate that both XGBoost and SVM are powerful models for deployment in a real-world IDS.

In comparison, Logistic Regression and Neural Networks experienced higher false-positive rates (0.240 and 0.166, respectively) and less overall detection performance, as shown in the results. The added false positives, false negatives, and multiple features illustrate the need for classifier selection and evaluation in IDS design, because if the false positives lead to alert fatigue, the false negatives may lead to undetected attacks [4]. Our reliance on technology for communication, finance, education, and healthcare continues to increase, which is expanding the attack surface at an alarming rate [5]. We have shown that ensemble methods, especially Random Forest and XGBoost, can provide stable, scalable, and interpretable options for developing IDS when trained with a rich feature space and tested and evaluated using realistic datasets. The remainder of this paper is organized as follows. In Section 2, we review the most related work on ML-based intrusion detection systems. Section 3 describes how we achieved the methodology and discusses how we handled feature engineering. Section 4 describes the experimental setup, Section 5 presents the results, and Section 6 discusses them. Finally, Section 7 presents the conclusions and possible future research avenues.

2. Literature Review

Research related to Intrusion Detection Systems (IDS) has substantially changed from signature-based, anomaly based, and hybrid IDS to a momentum toward leveraging machine learning (ML) and deep learning (DL) approaches to overcome some classical limitations. In this section, we summarize and compare some of the leading IDS classes by highlighting their patterns, limitations, and open issues, such as those related to interpretability, computational efficiency, and the investigation of new attacks.

2.1 Signature-Based vs. Anomaly-Based IDS

Khraisat et al. [5] presented an early and basic classification of IDS techniques: Signature-Based IDS (SIDS) and anomaly-based IDS (AIDS). SIDS are effective at detecting threats that have already been seen before using pattern matching but cannot

detect zero-day attacks. AIDS can detect unknown behaviors but do not work well because of their high false-positive rates compared to the dynamic characteristics of "normal" behavior. The authors point out the problem presented by evasion techniques in general and advocate the combined and effective use of network-based and host-based IDS.

Nonetheless, this dualism between SIDS and AIDS has generated hybrid systems, particularly with the use of ML models, to combine accuracy with adaptability. However, while Khraisat et al. highlight what evolving ML/DL-based systems can do, they do not discuss whether or how these systems can generalize to different datasets or network structures, a limitation that more recent work seeks to address.

2.2 Machine Learning vs. Deep Learning in IDS

Multiple studies have pointed to the use of ML and DL to enhance IDS performance. Liu and Lang [6] presented an extensive survey comparing ML and DL, and they recognize that both ML and DL can reduce false alerts and discover novel threats. They offer a data-centric taxonomy that can help researchers select the appropriate algorithms according to the type of data and features. Traditional ML models, such as SVM and KNN, work better when datasets are structured or labeled, and often require feature engineering and struggle with high-dimensional data by scaling.

On the other hand, DL can build hierarchical representations from raw input data in any format, such as images and videos. In this context, it should be noted that DL can automatically detect malicious emails and files directly. Liu and Lang take into account some of the merits of DL with the understanding that many of the shortcomings involve understanding DL as a black-box, dealing with computational cost, and needing certain data attributes regarding size and type. They cautioned that while deep learning offers automation and high detection levels and rates, the issues of interpretability and robustness concerning adversarial examples have not yet received sufficient attention.

2.3 Effectiveness and Trade-Offs in Deep Learning IDS

Ali et al. [7] further this discussion by studying the relative performance of traditional ML models vs DL models on imbalanced datasets. Their study incorporated a variety of class balancing techniques (including SMOTE) and demonstrated that while traditional models had the edge on speed over DL models, overall, DL models (especially CNNs and LSTMs) performed better in terms of accuracy. There also seems to be a tendency for there to be (apparent) trade-offs between the performance and speed of the models, especially if they are to be used in real-time or large contexts. An important aspect of the study beyond models (and methods, etc.) is the issue of real-world scalability and performance, but they do not mention anything regarding interpretability or trustworthiness for operational deployment in high-stakes cyber-security contexts.

2.4 Feature Selection and Interpretability

Sarker et al. [8] addressed the interpretability issue with the "IntruDTree" which is a tree-based IDS that emphasizes feature ranking/selection. Decision trees offer explicit, transparent decision-making, unlike DL models, and therefore are more understandable to a security analyst regarding how a detection event occurred. Their model performed better than lower baseline classifiers (Naïve Bayes, SVM, and Logistic Regression) while having a better balance for overfitting and dimensionality upon its selection.

It is interesting to note that this has encouraged a growing research agenda to find a trade-off between accuracy, efficiency, and explainability without going into an exhaustive search for all possible predictable events. However, tree-based models are generally not as useful as DL models for extracting more complex temporal or spatial patterns found in network traffic, presenting the research issue of needing an interpretable hybrid DL model that captures the complexity of network traffic.

2.5 Research Gaps

Despite these advancements, several issues require attention. Interpretability: Most existing DL-based IDS models possess non-transparent systems, and transparency is necessary for security auditing and regulatory compliance purposes. Adaptability: Although most IDS model research studies perform well when using benchmark datasets, many do not perform as expected in dynamic real-world environments. Imbalanced Data: The DL-based IDS models do not perform well with imbalanced dataset class distributions; some known methods (e.g., SMOTE) can help balance the dataset, but they may introduce their own noise. Evaluation Consistency: The lack of standardized datasets and metrics does not allow for fair comparisons across different studies. This study aims to address the above gaps by evaluating machine learning classifiers, such as Random Forest, XGBoost, and SVM, on a publicly available dataset, and to investigate the different classifiers' performances using accuracy, F1-score, and false positive rate, with the aim of identifying interpretable and scalable models for intrusion detection systems (IDS).

3. Methodology

This section outlines the methodology and rationale for the design and implementation of a cybersecurity intrusion detection system (IDS) using machine learning (ML). The focus of the system is to detect malicious behavior based on network behavior and user actions. The pipeline included data preprocessing, feature creation, model selection, and model evaluation to provide both analytical quality and usable implications in the context of cybersecurity.

1. Loading data and initial validation: At first pass, to check the integrity of the data, the load_data() method was called on the dataset to observe how the data were laid out, what the overall dimensionality of the data was, and whether the main variables, such as the target label attack_detected, were present. In addition, this initial look at the data serves as an opportunity to look for oddities, missing elements, or anything else that could potentially derail the subsequent analysis. Rationale: Validation at the initial stage prevents the workspace from being filled with incomplete or corrupt datasets and additionally confirms whether some indicators necessary for supervised learning are present.

2. Feature Engineering: The four derived variables in the model input include total attempts, which is the sum of successful and unsuccessful logins, representing the total quantity of activity. Failure ratio: The count of unsuccessful logins against total logins, which may indicate brute-force attacks. The level of network activity at a given point in time, especially high-impact events during DDoS attacks. The high failure and long-session indication flags provide users with threshold-based anomalies in the form of binary indicators. Justification: These engineered features are good because they provide a better fit for the phenomena that are meaningful in the security domain than the raw data. In addition, they provide semantically meaningful thresholds and features; thus, the interpretation of the model by analysts is considerably easier.

3. Data Preprocessing: The preprocessing pipeline is similarly standardized with respect to multiple data types. Numerical features were standardized using StandardScaler. Categorical features are one-hot encoded for ML algorithms. Binary features were retained in their original forms. Justification: Convert the features in a way that makes sense, balancing compatibility with different algorithms and convergence speed in gradient-based models.

4. Model Selection: Given the variable complexity of the attack patterns, multiple models were used. Logistic Regression: A linear base, intuitive, and interpretable model. Random Forest: A robust implementation that captures many nonlinear dependencies. XGBoost: A variety of gradient boosting that uses regularization but is computationally expensive and more accurate than simple tree models. Support Vector Machine (SVM): Effective in high-dimensional spaces. Neural Networks: Built to discover complicated nonlinear relationships. Rationale: We used multiple models to add robustness to our evaluation, and it allowed us to compare the width of interpretable models versus more complex capacity learners, which is useful in finding an optimal solution for anomaly detection-type problems.

5. Performance Metrics: Model evaluation included both a general set and group of domain-specific measures: Accuracy, F1-score, ROC-AUC: These standard measures of performance. Precision/Recall for attack class: Important for evaluating the detection of rare examples. False-positive rate (FPR): Important for limiting unnecessary indications of insecurity. Average Precision (AP): Reflects the precision-recall tradeoff, which is important for imbalanced data. Rationale: Effective IDS solutions are challenged to achieve high recall (to avoid getting hacked) while limiting false positives (to avoid alert fatigue), making it necessary to consider a wider set of evaluation metrics.

6. Feature Importance: Feature importance from each model is collected and the 15 most important features are displayed. Rationale: Understanding the features that matter enables model transparency and helps domain experts create explanations and confidence in the system. It also drives future model(s) of feature engineering models.

7. Handling Class Imbalance: The infrequency of cyber-attacks requires a unique treatment: class weighting (e.g., the ‘balanced’ parameter in classifiers). The scale_pos_weight parameter in XGBoost can be used to up-weight minority (attack) class instances. Precision-recall curves presented a more accurate representation of model performance when the distribution was heavily skewed. Rationale: Standard metrics, such as accuracy, can misrepresent the effectiveness of a model when classes are heavily imbalanced. Targeted treatment ensures that the minority (attack) class receives sufficient attention in terms of training examples.

A. Implementation Strengths

Reproducibility: The entire pipeline is modular and fully automated, from ingestion to evaluation. Extensibility: Components are purposely designed to be easily swapped (new models, preprocessing methods, etc.). Full Evaluation: Traditional and specialty metrics related to security are addressed. Interpretability: Feature-importance analysis provides global explanations of the models’ predictions. Practicality: Relevant issues (missing data, mixed data types, etc.) were examined.

B. Opportunities for Improvement

More Advanced Feature Engineering: Temporal features could benefit temporal anomaly detection (e.g., rolling statistics). Hyperparameter optimization: The use of

grid/Bayesian search may further improve performance. Unsupervised anomaly detection: Methods such as Isolation Forest or Autoencoders may provide insights into previously unexplored attack types. Cross-validation: Provides stronger generalization estimates. Deployment: Real-time monitoring and drift detection of the production model are recommended. The demonstrated method follows a framework approach to aid in the design of ML-based intrusion detection systems that are structured and commonly used and considers the statistical and practical aspects of cybersecurity environments, leading to ML-based systems that are robust, explainable, and effective in contributing to online adaptive threat detection systems.

4. Experiment and Analysis

This experiment evaluated the performance of five machine learning models in detecting possible attacks in network session data. The dataset comprised 9,537 records that included the following session features: login attempts, encryption, and reputation score. We measured the performance of Logistic Regression, Random Forest, XGBoost, Support Vector Machines (SVM), and a Neural Network using various evaluation metrics, as shown in Table 1 (Model Comparison Results).

A. Dataset Description

The dataset comprises the following main features:

- **Network characteristics:** packet size, protocol type, and session length.
- **Authentication values:** Number of logins, number of failed logins, and failure rate.
- **Security indicators:** IP reputation score and abnormal time access.
- **Browser and encryption:** Browser type and encryption method.
- **Binary target variable:** attack_detected (1 for attack; 0 for normal).

B. Model Implementation

We ran five classification models:

1. **Logistic Regression:** Our baseline linear model.
2. **Random Forest:** An ensemble of decision trees.
3. **XGBoost:** Gradient-boosted decision tree.
4. **SVM:** Support Vector Machine with a radial basis function (RBF) kernel.
5. **Neural Network:** Multi-layer perceptron.

Each model was assessed using a set of common metrics, namely accuracy, F1-score, ROC AUC, and precision-recall characteristics.

5. Results

A. Overall Performance Comparison

The performance metrics illustrated drastic differences among the models (Table 1 and Figure 1).

Model Performance Summary

1. Random Forest not only had the highest accuracy of 0.881, but also the highest F1-score of 0.849. It achieved an incredibly low false-positive rate of 0.010.

2. **XGBoost and SVM performed comparably, with accuracies of about 0.87 and F1-scores of 0.841 and 0.832, respectively.**

3. **The Neural Network** performed relatively well, showing balance (F1 = 0.789), but had a higher false positive rate (0.166) instead of the desirable low false positive rate.

4. Logistic Regression had the lowest overall performance (accuracy = 0.744, F1 = 0.717) on average and exhibited the highest false-positive rate (0.240).

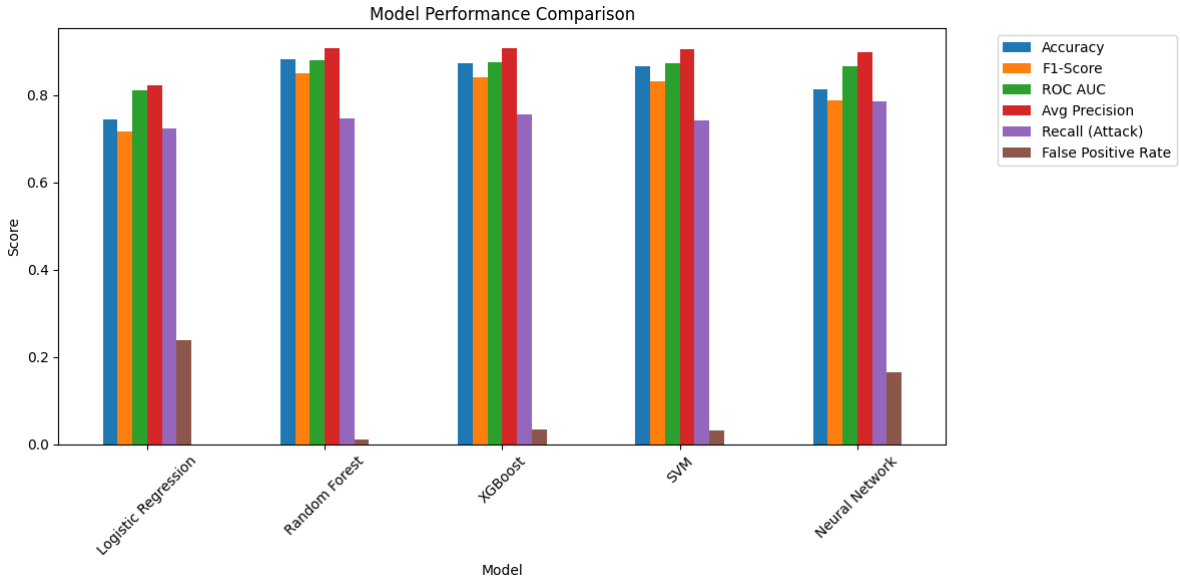


Fig. 1. Model performance comparison

B. Feature Importance Analysis

The feature importance patterns varied significantly across the models (Figures 2–4).

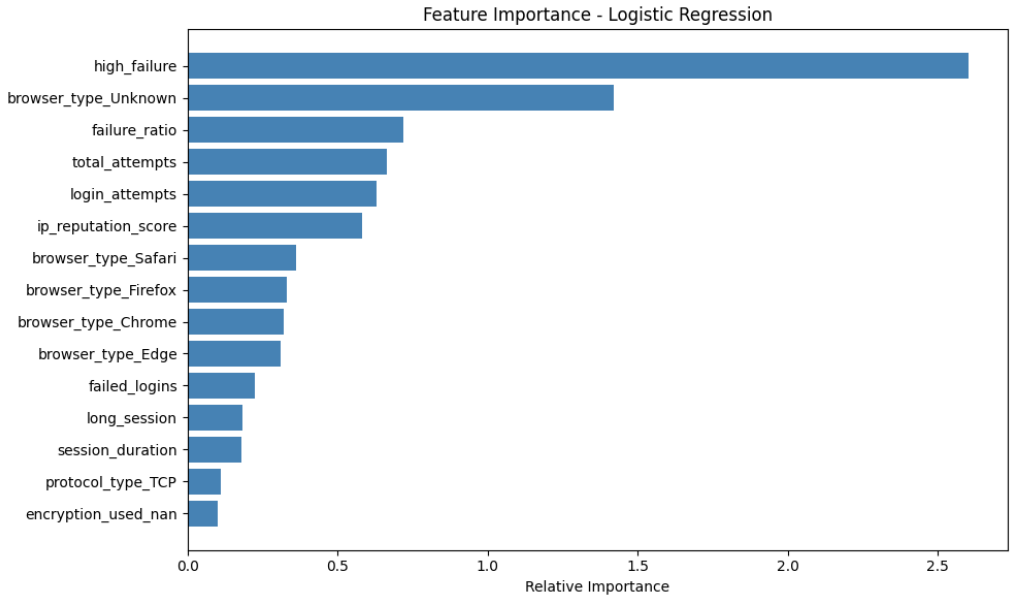


Fig. 2. Feature importance: Logistic Regression

1) **Logistic Regression (Figure 2):**

- Top weights were observed for categorical browser features, for example, browser_type Unknown,with a relative importance of 2.5.
- Both high_failure and failure_ratio were identified as highly important.

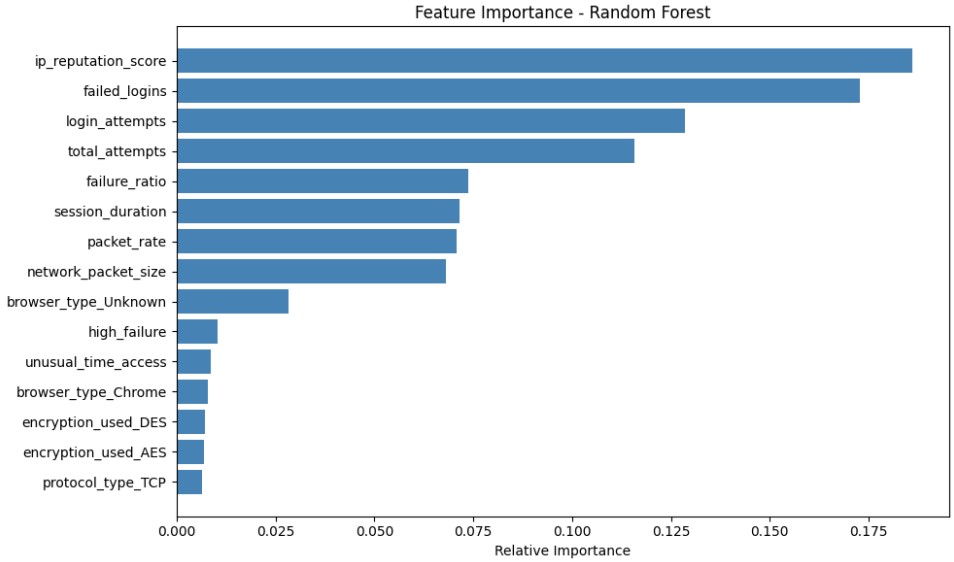


Fig. 3. Feature importance: Random Forest

2) **Random Forest (Figure 3):**

• ip_reputation_score was the most important feature, with a relative importance of 0.15.

• The model provides a balanced consideration of authentication metrics, such as failed_logins and login_attempts.

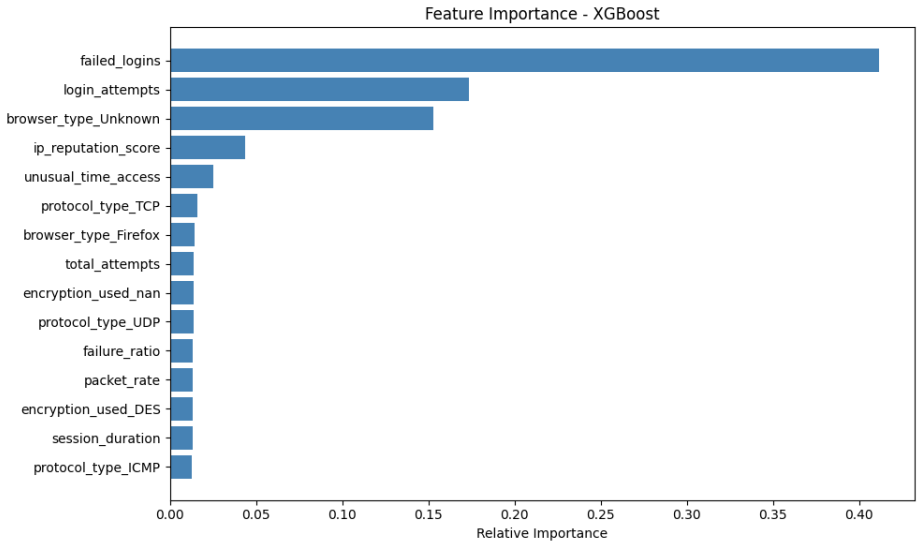


Fig. 4. Feature importance: XGBoost

3) **XGBoost (Figure 4):**

- failed_logins had the highest importance, with a relative importance of 0.40.
- There was a heavy weighting of authentication-type features, such as login_attempts and total_attempts.

The repeated importance of the IP reputation score and failed logins in each model indicates that these are strong indicators of possible attacks.

Table 1. Performance comparison of machine learning models

Model	Accuracy	F1-Score	ROC AUC	Avg Precision	Precision (Attack)	Recall (Attack)	Precision (Normal)	Recall (Normal)	FPR	TPR
Logistic Regression	0.744	0.717	0.811	0.822	0.710	0.724	0.773	0.760	0.240	0.724
Random Forest	0.881	0.849	0.880	0.907	0.984	0.747	0.829	0.990	0.010	0.747
XGBoost	0.872	0.841	0.875	0.906	0.948	0.755	0.830	0.966	0.034	0.755
SVM	0.866	0.832	0.873	0.904	0.949	0.741	0.822	0.968	0.032	0.741
Neural Network	0.812	0.789	0.866	0.898	0.793	0.784	0.827	0.834	0.166	0.784

C. Confusion Matrix Analysis

The class-wise precision and recall values in Table 1 provide important operational insights.

- **Random Forest** achieved a very high recall for the normal class (0.990) but had a lower recall for the attack class (0.747).
- **XGBoost** and **SVM** showed slightly lower performance than Random Forest but still demonstrated excellent identification of the normal class.
- **The Neural Network** provided the most balanced recall across classes, with an attack recall of 0.784 and a normal recall of 0.834.
- **Logistic Regression** had a relatively high false positive rate, incorrectly flagging 24% of normal traffic as attacks.

1) *Precision-Recall Tradeoffs:* Average precision (AP) scores were high (0.822–0.907, Table 1), indicating that all models provided good precision across recall levels.
   - Random Forest has the best precision for attack detection (0.984).
   - The Neural Network had the best balance of attack recall (0.784) and false-positive rate (0.166).

6. Discussion

The results also indicate that tree-based ensemble methods (Random Forest, XGBoost) appear to be more sensitive to the attack detection task than the other methods (see Figure 1). Some key messages are:

- Authentication Patterns Matter: Failed logins and login attempts were highly ranked in all models (see Figures 2–4).
- Reputation of the IP address: This feature appeared to yield consistent feature usefulness across all models that computed feature importance.
- Browser Anomalies Appear to Be Risky: Browser type Unknown was found to be one of the most important features in Logistic Regression (Figure 2).
- Operational Considerations: Random Forest had the highest accuracy; however, it had a low attack recall rate (0.747 vs. 0.784 for neural networks). This could be detrimental to security applications.

This analysis shows that Random Forest and XGBoost can provide the best solutions, with accuracies of 88.1% and 87.2% and low false-positive rates (Table 1). For

practical use, it may be best to use a Neural Network where the results were similar or to use an ensemble. Future work will consider hybrid models that leverage the most consistent features from Figures 2–4.

7. Conclusion

In summary, this study provides a thorough examination of several machine learning approaches for intrusion detection systems (IDS) and the performance of five different classifiers using a network traffic dataset. Our experimental findings suggest that ensemble methods perform best for detection, with an accuracy of over 87% and a false-positive rate of less than 3.5%. These ensemble models demonstrate a good compromise between the important tensions of achieving attack detection (recall) and practical operation (false positive rate) and are strong candidates for real-world use. The importance of features was consistent between models, and the authentication features (failed logins and login attempts) and network reputation features (IP reputation score) were the most important predictors of malicious actions in the dataset. This is in accordance with established cybersecurity principles and provides data support for machine learning interpretability methods.

Theoretical and Practical Implications

From a theoretical standpoint, this study adds to the literature on XAI in cybersecurity by:

- Illustrating the potential implications of employing feature engineering techniques as an approach to increase behavioral anomaly detection efficacy,
- Providing empirical evidence for tree-based ensemble algorithms for security purposes.
- Establishing a framework to assess the effectiveness of IDS models based on security-relevant metrics.

From a practical perspective, the results suggest that deep learning is not always necessary for effective intrusion detection. Both Random Forest (F1-score: 0.849) and XGBoost (F1-score: 0.841) provided compelling performance and explainability, which is important for security operations center environments that are highly reliant on explainable security processes.

Limitations and Research Challenges

The limitations of the current study are as follows:

- Limitations of the datasets: Although the dataset represents network session behaviour and contains useful information, it does not incorporate all types of attacks, such as zero-day attacks and advanced persistent threats (APTs).
- Time limitations: As our analysis is static, relying on a single snapshot of the trained models, we ignored the time-based concept drift inherent in deployments for years in an IDS.
- Limitations of adversarial evasion: Our models were not subjected to adversarial evasion attacks that displace benign traffic and occlude features in network traffic designed to evade detection.
- Operational limitations: Our study did not consider operational aspects, such as delays in undertaking calculations and application-specific limitations of finite resources in real-world use.

Future Research Directions

- The interpretability of tree models handled with the temporal modeling of recurrent neural networks

- Signature-based detection of known threats with anomaly detection for unrecognized attacks
- Supervised learning for labeled data and unsupervised approaches for unknown instances

Data availability statement
The dataset used in this study is publicly available and can be accessed on Kaggle. The dataset, titled Cybersecurity Intrusion Detection Dataset, can be found at the following URL: https://www.kaggle.com/datasets/dnkumars/cybersecurity-intrusion-detection-dataset

Ethical Approval:
The dataset used in this study is publicly available on Kaggle and contains no personally identifiable information. No human or animal subjects were involved in this study.

Conflict of Interest
The author declares no conflicts of interest.

References

[1] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007. doi: 10.1016/j.comnet.2007.02.001

[2] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Transactions on Information and System Security*, vol. 3, no. 3, pp. 186–205, 2000. doi: 10.1145/357830.357849

[3] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted Boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, 2013. doi: 10.1016/j.neucom.2012.11.050

[4] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A taxonomy and survey of intrusion detection system design techniques, network threats and datasets," *arXiv preprint*, 2018. doi: 10.48550/arXiv.1806.03517

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, Art. no. 20, 2019. doi: 10.1186/s42400-019-0038-7

[6] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," Applied Sciences, vol. 9, no. 20, Art. no. 4396, 2019. doi: 10.3390/app9204396

[7] M. L. Ali, K. Thakur, S. Schmeelk, J. Debello, and D. Dragos, "Deep learning vs. machine learning for intrusion detection in computer networks: A comparative study," *Applied Sciences*, vol. 15, no. 4, Art. no. 1903, 2025. doi: 10.3390/app15041903

[8] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, "IntruDTree: A machine learning based cyber security intrusion detection model," *Symmetry*, vol. 12, no. 5, Art. no. 754, 2020. doi: 10.3390/sym12050754