AutoML and explainable AI-based approach to enhance the efficiency and interpretability of IDS*

Dmytro Tymoshchuk[1,*,†], Nataliya Zagorodna[1,†], Yurii Klots[2,†], Vasyl Yatskiv[3,†] and Nataliia Petliak[2,†]

[1] *Ternopil Ivan Puluj National Technical University, Ruska str. 56, Ternopil, 46001, Ukraine*
[2] *Khmelnytskyi National University, 11, Instytuts'ka str., Khmelnytskyi, 29016, Ukraine*
[3] *West Ukrainian National University, 11 Lvivska str., 46009 Ternopil, Ukraine*

Abstract

This study presents an approach to develop an intelligent Intrusion Detection System based on Automated Machine Learning (AutoML) integrated with the Explainable Artificial Intelligence (XAI) methods. The experiments were conducted using a dataset derived from UNSW-NB15, containing examples of both normal and malicious network traffic. The AutoML workflow was implemented using the PyCaret library, which enabled automated preprocessing, selection of the most effective algorithms, and hyperparameter optimization with no manual manipulations. The best performance was achieved by the Random Forest Classifier, which, at the optimal decision threshold determined by Youden's J index (J = 0.617), reached accuracy = 0.9972 and AUC = 0.9999, indicating an almost perfect discriminative capability. The application of the SHAP method allowed to interpret the contribution of individual features to the classification process and showed the transparency of the model's decisions. The developed model was integrated into an IDS system deployed in a KVM-based virtualized laboratory environment, allowing real-time evaluation under realistic network load conditions. The obtained results demonstrate that combination of AutoML and XAI provides an effective approach to building accurate, robust, and interpretable next-generation cybersecurity systems.

Keywords

IDS, machine learning, Explainable AI (XAI), SHAP, AutoML, cybersecurity, hypervisor, operating systems. [1]

1. Introduction

The intensive digitization of all spheres of human activity has necessitated a high level of cybersecurity, on which the integrity and continuity of computer networks, industrial systems, and state information resources depend. At the same time, the growing number of connected devices and the increasing volume of data exchange have led to a rise in vulnerabilities that can be exploited by attackers for unauthorized access, data theft, or disruption of system operations. Over the past decade, there has been a dramatic increase in the number of cyberattacks, while their complexity and level of concealment have significantly complicated timely detection and response. Malicious actors increasingly employ traffic encryption, code obfuscation, and multi-stage attack techniques, rendering traditional detection approaches progressively less effective. This decline in effectiveness underscores the need for innovative, intelligent methods to strengthen cybersecurity and enhance the detection and prevention of emerging threats.

Most modern Intrusion Detection and Prevention Systems (IDS/IPS) are based on two main approaches — signature-based and behavior-based detection [1,2]. Signature-based methods rely on the use of previously known indicators of cyberattacks, such as characteristic patterns of network

* *AdvAIT-2025: 2nd International Workshop on Advanced Applied Information Technologies,: AI & DSS December 05, 2025, Khmelnytskyi, Ukraine, Zilina, Slovakia*
[1,*] Corresponding author.
[†] These authors contributed equally.
dmytro.tymoshchuk@gmail.com (D. Tymoshchuk); Zagorodna.n@gmail.com (N. Zagorodna); klots@khmnu.edu.ua (Y. Klots); jazkiv@ukr.net (V. Yatskiv); npetlyak@khmnu.edu.ua (N. Petliak)
0000-0003-0246-2236 (D. Tymoshchuk); 0000-0002-1808-835X (N. Zagorodna); 0000-0002-3914-0989 (Y. Klots); 0000-0001-9778-6625 (V. Yatskiv); 0000-0001-5971-4428 (N. Petliak)

packets, sequences of system commands, or malware code signatures. This approach ensures high accuracy in detecting known types of threats through precise matching against a signature database. However, it proves to be ineffective against new, modified, or previously unseen (zero-day) attacks that lack corresponding entries in the system's database. In contrast, behavior-based (anomaly-based) methods analyze network traffic, user activities, or system logs to identify anomalies from a predefined "normal" behavior. Such systems are capable of detecting new, previously unknown types of attacks that have no signature counterparts. Nevertheless, their performance largely depends on the quality of the constructed behavioral profiles, and excessive sensitivity to environmental dynamics often results in a high rate of false positives, necessitating careful calibration of system parameters.

Given the limitations of traditional methods, the use of Machine Learning (ML) technologies in cybersecurity has become increasingly widespread [3–5]. ML algorithms are capable to identify complex patterns in large volumes of data automatically, adapting to new types of threats, and improving the accuracy of attack detection [6]. Their application enables the integration of the advantages of both traditional approaches – detecting both known and unknown attacks while minimizing false alarms. The development of intelligent IDS/IPS systems based on machine learning opens new opportunities for proactive network protection and for enhancing the cyber resilience of modern information systems. State-of-the-art research in cybersecurity focuses on developing optimal models to detect threats accurately and efficiently while minimize false alarms and maintain a high level of decision explainability. One of the most promising directions in this regard is automated machine learning (AutoML), which enables the automatic selection of optimal algorithms, hyperparameters, and model architectures without the need for extensive expert involvement. Through AutoML, the process of building models for intrusion detection and prevention systems becomes more flexible, reproducible, and scalable. Such approaches allow rapid adaptation of models to new types of network traffic and emerging threats while preserving high accuracy and performance.

At the same time, an important direction in improving intelligent security systems is the application of Explainable Artificial Intelligence (XAI) methods, which ensure transparency in the decision-making processes of ML models. XAI technologies make it possible to interpret which features influenced the classification of events as malicious or legitimate, thereby increasing the transparency and trustworthiness of automated threat detection systems. The integration of AutoML with XAI methods gives the foundation for the development of intelligent, self-adaptive, and interpretable next-generation cybersecurity systems capable of continuous learning and autonomous improvement during operation.

2. Related Work

Machine learning methods have proven to be effective tools for analyzing network traffic and identifying various types of attacks, providing high classification accuracy. In recent years, considerable research attention has been devoted to automating the development and optimization of machine learning models in the field of cybersecurity. Traditionally, building ML-based intrusion detection models requires a high level of expert knowledge in algorithm selection, hyperparameter tuning, and feature engineering — factors that limit the adaptability and deployment speed of such systems against new attack types. With the emergence of AutoML technologies, this process has become significantly more efficient and scalable. AutoML enables the automatic selection of optimal models, parameters, and feature combinations, allowing the creation of highly accurate threat detection systems. Recent studies demonstrate the successful application of AutoML in tasks such as network traffic classification, anomaly detection, malware identification, and phishing detection. These solutions substantially reduce the time required to develop effective models and enhance their adaptability to the dynamically changing conditions of modern network environments.

The authors of [7] investigated and compared the performance of AutoML models with contemporary state-of-the-art approaches for wireless signal classification, as well as their robustness against white-box and black-box attacks. They proposed several AutoML-based architectures ResNet, CLDNN, CNN, and RNN which demonstrated high classification accuracy while significantly reducing the time required for hyperparameter tuning and model training. The authors of [8] proposed a network-oriented AutoML architecture for detecting DDoS attacks in software-defined sensor networks (SDSNs). The proposed solution automatically selects the most suitable machine learning algorithm by taking into account network load, traffic heterogeneity, and detection latency, thereby ensuring efficient operation under attack conditions. The architecture was implemented using open-source networking tools and multiple ML models. In [9], the authors evaluated the effectiveness of several AutoML frameworks for network intrusion detection and compared them with traditional machine learning methods using the NSL-KDD dataset. The automated algorithms eliminated the need for manual feature selection, reduced false positives, and improved detection accuracy. Experimental results showed that H2O AutoML and MLJar achieved 90% accuracy, outperforming FLAML (79%) and classical models, which confirms the potential of AutoML for developing scalable and adaptive cybersecurity systems. The authors of [10] developed ICS-Defender, an automated protection mechanism for Industrial Control Systems (ICS) leveraging AutoML technologies. The proposed approach combines intelligent feature processing with automated model selection, training, and optimization, thereby reducing dependence on domain experts. Experimental results demonstrated that ICS-Defender outperformed existing AutoML-based solutions, achieving up to 94% accuracy and enhancing the resilience of ICS environments to cyberattacks. The authors of [11] introduced a method for automating feature construction and selection from raw datasets, which represents a logical extension of the ExploreKit algorithm. The method integrates a tree-structured representation of the AutoFE (Automated Feature Engineering) search space with evolutionary optimization. Experiments conducted on the UNSW-NB15, CICDDoS2019, and APA-DDoS datasets demonstrated that the evolutionary feature generation algorithm achieved accuracy comparable to or higher than existing methods while maintaining high computational efficiency. The authors of [12] described the Network Traffic Analyzer, a key component of the CTI2SA architecture developed within the Cyber-pi project, aimed at improving cybersecurity efficiency and ensuring compliance with GDPR requirements. The system is built upon the Lambda (λ) architecture, which combines batch and stream processing for large-scale network data analysis. Its core module incorporates an automatic machine learning model selection mechanism that dynamically identifies the optimal model to ensure continuous and accurate detection of cyber threats.

In parallel with the development of AutoML, research is actively being conducted in the field of Explainable AI, which aims to enhance the transparency and trustworthiness of automated systems. In cybersecurity, XAI is applied to interpret the decisions of machine learning models, particularly by identifying the contribution of individual features to the classification of events as malicious or normal. The use of XAI techniques such as SHAP [13], LIME [14], and Permutation Feature Importance [15] enables security analysts to better understand the internal mechanisms of models, detect training errors, and improve the overall quality of analysis. Integrating XAI into IDS/IPS systems not only increases their interpretability but also supports the development of new response strategies focused on the causal relationships between attack features.

In [16], the authors addressed the issue of trust in deep learning systems and identified the sources of mistrust related to model selection and interpretability. They outlined two key directions for enhancing AI reliability: AutoML, which automates the design and optimization of neural networks, and interpretability methods, which explain the reasoning behind model decisions and improve robustness against adversarial attacks. The study bridges theoretical concepts with industrial applications. In [17], the growing importance of XAI in cybersecurity was analyzed, emphasizing that trust in machine learning models is a critical requirement. The authors proposed a taxonomy of XAI methods that considers security-related properties and threats specific to the domain, and they developed a novel black-box attack designed to assess the robustness of gradient-

based explanation methods. Experiments conducted on three datasets confirmed that the proposed attack can distort model explanations without altering the model's outputs, thus paving the way for the development of more secure XAI approaches. The work presented in [18] examined the risks associated with the use of XAI methods, particularly counterfactual explanations, in cybersecurity. The authors demonstrated that, while such explanations enhance model transparency and trust, they can also introduce new attack vectors, including membership inference, model extraction, data poisoning, and backdoor attacks. A novel black-box attack leveraging XAI to compromise model confidentiality was proposed, and experiments confirmed its effectiveness on cybersecurity datasets. In [19], the authors explored the application of XAI methods for interpreting attack classification results in IoT/IIoT networks. Using the TON IoT dataset, they compared the performance of Decision Tree, Random Forest, AdaBoost, XGBoost, ANN, and MLP algorithms, all achieving over 96% accuracy in binary classification. To explain the decisions of complex models, the authors employed LIME, SHAP, and ELI5, which improved transparency, trust, and interpretability in IoT attack detection. The study in [20] proposed a two-stage pipeline aimed at enhancing the reliability of network intrusion detection systems. In the first stage, an XGBoost model was used for supervised attack detection, and its results were interpreted using the SHAP method. In the second stage, the obtained explanations were used to train an autoencoder, enabling the detection of previously unseen attacks. Experiments on the NSL-KDD dataset confirmed high accuracy and competitiveness of the proposed approach compared with existing cybersecurity methods.

The aim of this study is to develop and evaluate AutoML-based models integrated with XAI methods for network traffic analysis in order to enhance threat detection accuracy, reduce false positive rates, and ensure interpretability of results within cybersecurity systems.

3. Materials and Methods

In this study, we use the experimental UNSW-NB15 dataset [21–23] for training and evaluating machine learning models in the task of network intrusion detection. The dataset was designed to reproduce real network operating conditions and ensure representativeness for next-generation intrusion detection systems.

Overall, the our dataset, constructed based on UNSW-NB15 dataset, contains 257,673 records, of which 93,000 observations correspond to normal (benign) network traffic, while the remaining entries represent anomalous samples associated with different categories of malicious traffic. It contains 38 input features describing various characteristics of network activity. Each record in the dataset corresponds to an individual network session, enabling analysis at the session level of interactions between network nodes. The dataset includes both legitimate network traffic samples and examples of various types of attacks (Figure 1). Within this study, the problem is formulated as a binary classification task, where the target variable indicates whether the network traffic belongs to one of two classes: 0 – Normal and 1 – Attack. The aggregation of all attack types into a single Attack class simplified the model training process and enabled a more accurate assessment of the models' effectiveness in distinguishing between normal and malicious network traffic.

To build and evaluate the performance of machine learning models, the prepared dataset was divided into training and testing subsets in a 70/30 ratio, while maintaining the proportional distribution of the target variable. This approach helped to prevent class imbalance and ensured a reliable assessment of the models' generalization capability. For each category of malicious traffic, the corresponding proportion of samples was preserved within both subsets, guaranteeing data representativeness and validity of testing results. To automate the process of model construction and optimization, this study employed AutoML PyCaret [24], a framework built on the scikit-learn library. This tool provides a comprehensive end-to-end approach to model development, encompassing all stages – from data preprocessing to model evaluation. The PyCaret AutoML environment automatically performs essential data preparation steps, including missing value imputation, feature scaling, categorical encoding, class balancing, and selection of the most

informative features. After preprocessing, the system compares a wide range of algorithms including Random Forest [25], Extreme Gradient Boosting (XGBoost) [26], Extra Trees [27], Light Gradient Boosting Machine (LightGBM) [28], Gradient Boosting [29], Ridge Classifier [30], Logistic Regression [31], Multilayer Perceptron (MLP) [32], and Support Vector Machine (SVM, linear kernel) [33]. Their performance is evaluated using key metrics such as Accuracy, Precision, Recall, F1 score, AUC, Matthews Correlation Coefficient (MCC), and Cohen's Kappa [34].

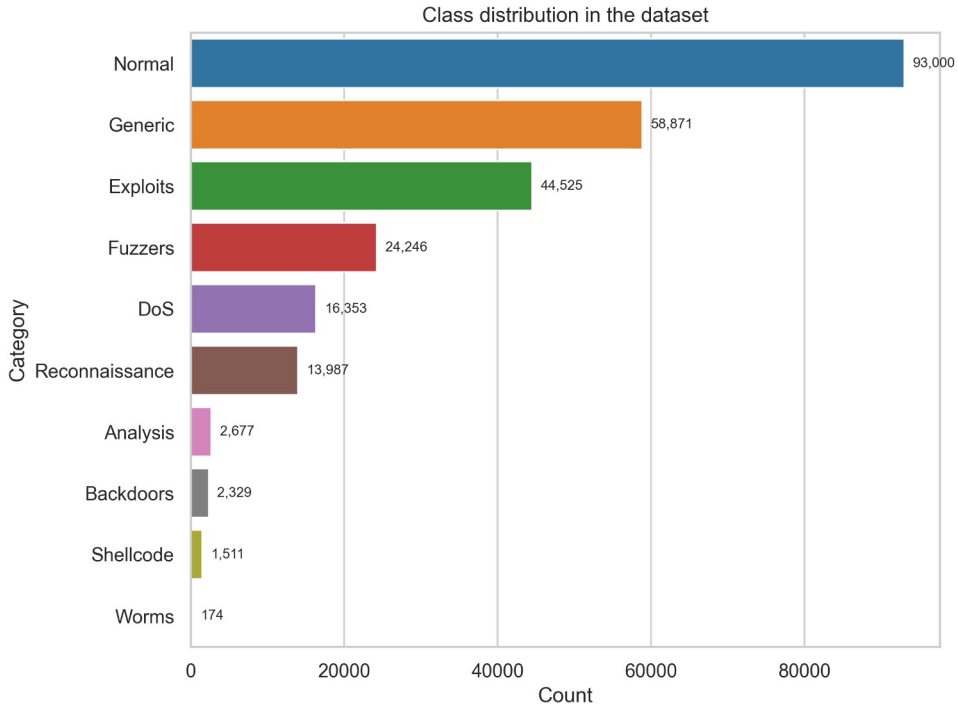


Figure 1: Class distribution in the dataset.

Accuracy represents the proportion of correctly classified samples among all observations and reflects the overall effectiveness of the model. However, it may be insufficiently informative in cases of class imbalance. Precision indicates the proportion of instances predicted as "attack" that are truly attacks, and its high value is crucial for reducing the number of false alarms in an intrusion detection system. Recall measures the model's ability to identify all actual attack instances and is essential in cybersecurity, where missing even a small fraction of threats can have critical consequences. F1 score combines Precision and Recall into a single balanced indicator, and a high F1 value demonstrates an optimal trade-off between detecting attacks and minimizing false positives. Area Under the ROC Curve (AUC) describes the model's ability to distinguish between classes across different decision thresholds; values close to 1 indicate excellent discriminative performance, whereas 0.5 corresponds to random guessing. Matthews Correlation Coefficient (MCC) assesses the correlation between predicted and actual classes, accounting for all combinations of true and false outcomes, and is considered one of the most reliable measures of binary classification performance, especially when class distributions are uneven. Cohen's Kappa evaluates the level of agreement between the model's predictions and the true labels while considering random coincidence, and high Kappa values indicate that the model's predictive accuracy substantially exceeds random chance.

After identifying the most promising models, automated hyperparameter tuning is performed to improve generalization quality without manual intervention from the researcher. The tuning results are validated using k-fold cross-validation, which ensures the objectivity of the evaluation. The selected models are then calibrated to enhance the accuracy of probabilistic predictions, after

which a final pipeline is constructed that integrates all stages of data transformation and prediction.

To interpret the results of the machine learning models, this study employed the SHAP method, which belongs to the family of XAI approaches. SHAP provides a quantitative assessment of the contribution of each input feature to the model's prediction, enabling an understanding of the decision-making logic even in complex "black-box" models. The method is based on cooperative game theory, where each feature is treated as a player that contributes to the overall outcome, which is the model prediction. For each traffic record, SHAP values are calculated to show how much a particular feature increases or decreases the predicted value compared with the baseline or average prediction. Positive values indicate a higher likelihood that the sample belongs to the Attack class, while negative values indicate a lower likelihood. One of the main advantages of SHAP is its consistency and additivity, meaning that the sum of all SHAP values for the features equals the difference between the model prediction and its baseline value. This property ensures a transparent and theoretically sound interpretation. For global feature importance analysis, the mean absolute SHAP values are used to represent the overall impact of each feature on the model across the dataset. For local interpretation of individual observations, force plots or waterfall diagrams are created to illustrate the specific factors that lead to a particular prediction. In this study, the SHAP method was applied to analyze models developed using the AutoML PyCaret framework. This approach made it possible not only to achieve high classification accuracy but also to obtain interpretable and reliable results, which is important for increasing trust in intrusion detection systems and for developing effective cybersecurity policies.

4. Results and Discussion

Within this study, the AutoML PyCaret framework was employed to automatically build and optimize the data processing pipeline and to identify the most effective model for network traffic classification. The Random Forest Classifier demonstrated the highest performance during the AutoML search (Figure 2).

The average accuracy for this configuration was approximately 0.9475, indicating high precision in recognizing different types of attacks and stable model performance on the test data. The constructed PyCaret pipeline consisted of three main stages: filling missing values (when necessary), calibrating predicted probabilities, and performing classification using an ensemble of decision trees.

At the first stage, the dataset was checked for missing values. The main classifier used was the Random Forest Classifier, which represents an ensemble of independent decision trees that combine their voting results to produce the final prediction. The model parameters included n_estimators = 325 (the number of trees in the forest), criterion = 'gini' (the splitting criterion for assessing node purity), max_features = 'sqrt' (used to reduce correlation between trees), and max_depth = None, which allows the trees to grow until full data separation, providing high model flexibility. The parameter n_jobs = -1 enabled the use of all processor cores for parallel computation, while random_state = 22 ensured experiment reproducibility. Since reliable probability estimates are crucial in cybersecurity tasks, not only class labels were obtained but also calibrated probabilities. To achieve this, the classifier was additionally calibrated using CalibratedClassifierCV with parameters method = 'sigmoid', cv = 5, and ensemble = True. Calibration based on the sigmoid function (Platt's method) made it possible to correctly estimate prediction uncertainty and produce accurate probabilistic assessments of class membership [35]. Averaging the results of five folds during cross validation provided model generalization and reduced random fluctuations in the probability distributions. The use of PyCaret AutoML provides built-in mechanisms for mitigating overfitting, including k-fold cross-validation, automated hyperparameter optimization, and a separate test set, which reduces the risk of excessive model fitting to the training data.

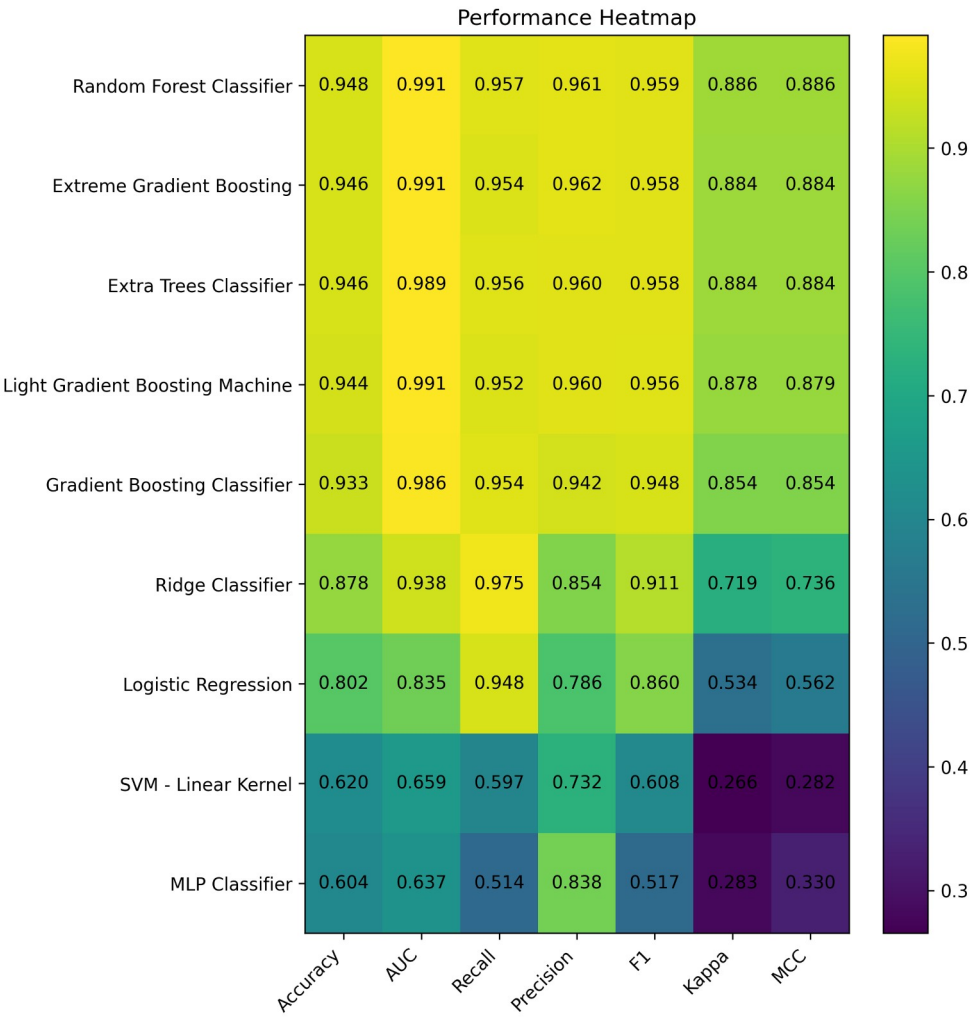
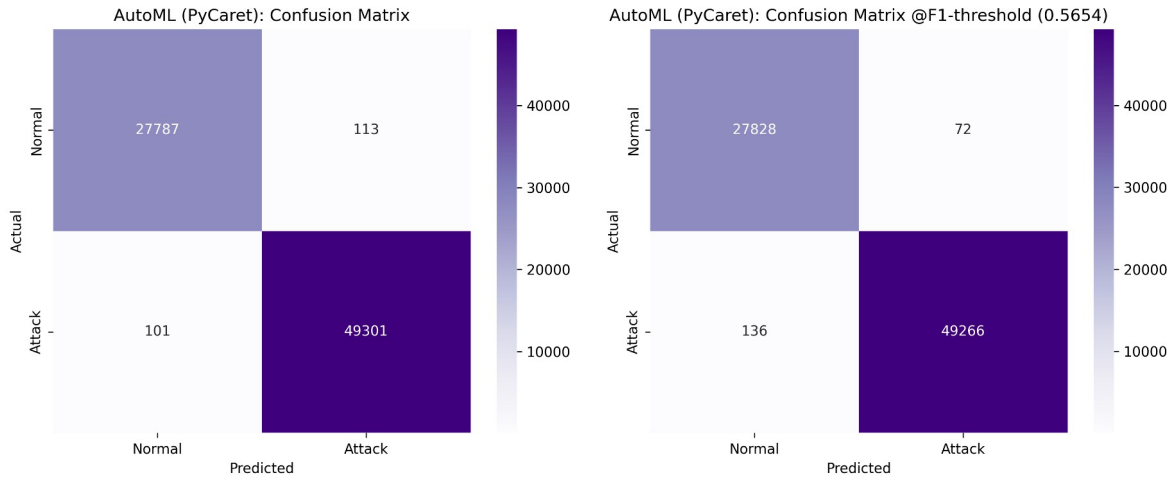
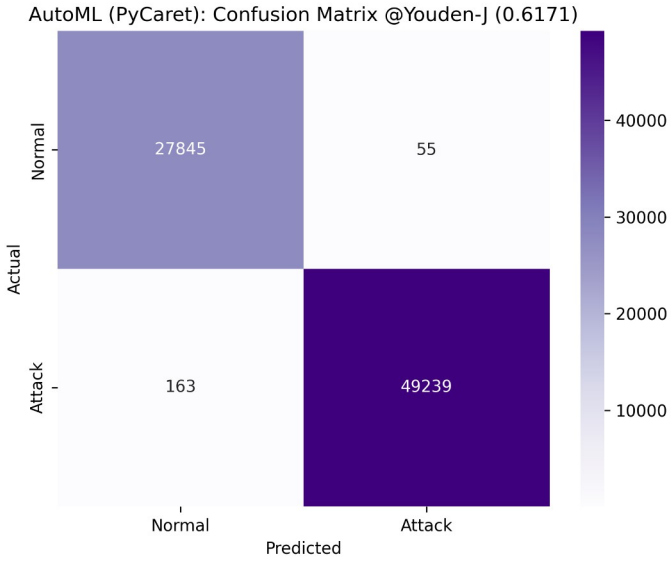


Figure 2: Performance heatmap comparing the efficiency of AutoML (PyCaret) models.

To evaluate the classification performance, three versions of the confusion matrix were constructed, showing the distribution of correct and incorrect model predictions under different decision thresholds (Figure 3).



a) b)



c)

Figure 3: Confusion matrices of the model at different decision thresholds: (a) standard (0.5), (b) F1-based (0.565), and (c) Youden's J-based (0.617).

The base confusion matrix (Figure 3a) was constructed for the standard decision threshold of 0.5, while the other two correspond to optimized thresholds determined using the maximum F1 score (Figure 3b) and Youden's J index (Figure 3c). The F1 threshold approach provides the best compromise between Precision and Recall, whereas the Youden J threshold method aims to maximize the overall discriminative capability of the model by simultaneously improving Recall and Specificity. The obtained results show that at the default threshold of 0.5, the model achieved high classification accuracy, correctly identifying 27,787 true negatives (TN) and 49,301 true positives (TP), with 113 false positives (FP) and 101 false negatives (FN). Optimization according to the F1 criterion at a threshold of 0.565 reduced the number of false positives to 72, while maintaining a high number of true positives (TP = 49,266). In turn, Youden's J index defined a slightly higher optimal threshold of 0.617, which provided the best balance between Recall and Specificity: the number of false positives decreased to 55, and the number of true positives remained at 49,239. Thus, adjusting the decision threshold improved classification reliability by reducing the proportion of false alarms without a significant loss in detection completeness. The most balanced result was achieved at Youden's J = 0.617, which can be considered the optimal threshold for practical implementation in IDS.

Figure 4 presents a comparison of the Precision, Recall, and the combined F1 score metrics for each class at the optimal decision threshold determined using Youden's J index (Youden J threshold = 0.617).

The Normal class is characterized by Precision = 0.9942, Recall = 0.9980, and F1 = 0.9961, indicating a very small number of false positive predictions. For the Attack class, Precision is slightly higher (Precision = 0.9989, F1 = 0.9978), although Recall = 0.9967 suggests a few missed attack instances. The overall Accuracy represents the proportion of all correctly classified samples (both normal traffic and attacks) relative to the total number of records. This metric summarizes the model's general performance, showing how well the classifier reproduces the structure of all classes without focusing on individual distributions. The total classification accuracy reached 0.9972, confirming the stable performance of the model across both classes.

Figure 5 shows the average metric values in two forms: macro average and weighted average.

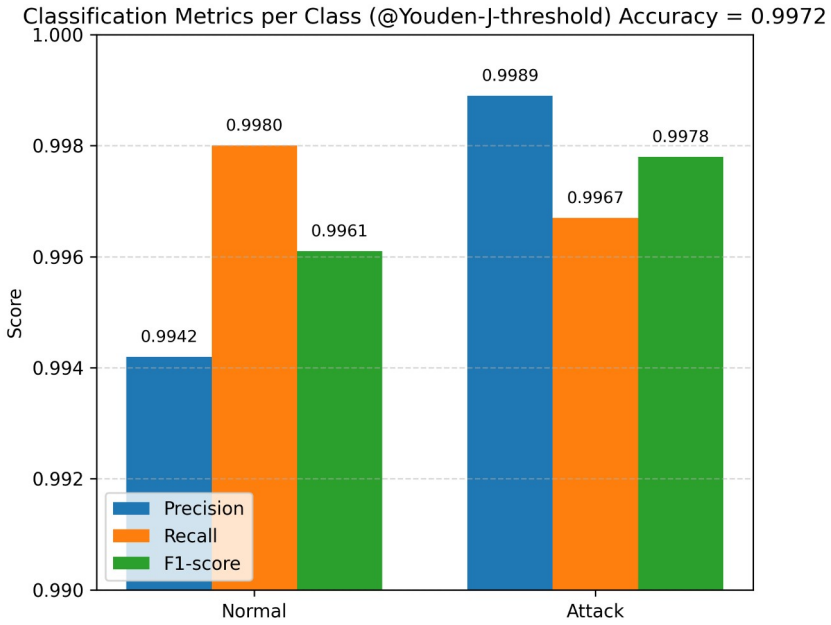


Figure 4: Classification performance metrics per class and averaged values at the Youden's J threshold.

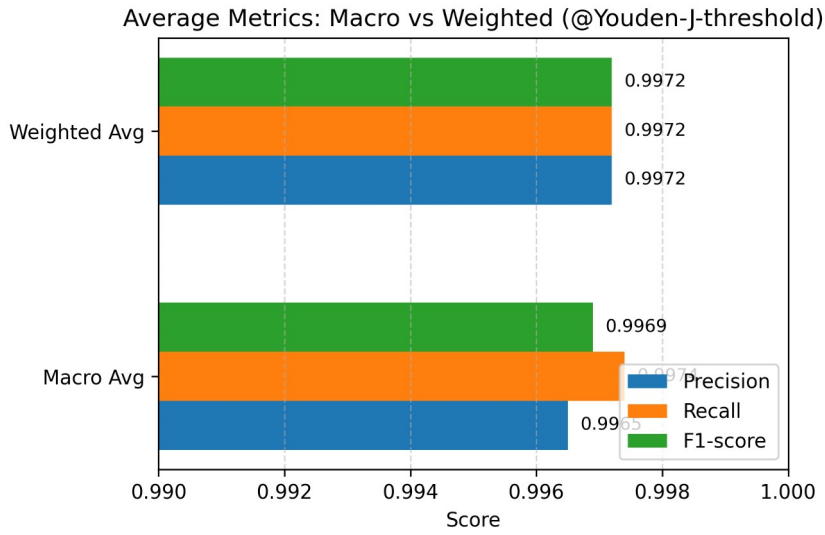


Figure 5: Comparison of macro and weighted average classification metrics at the Youden's J threshold.

The Macro Average metric represents the arithmetic mean of the metric values calculated across all classes, without considering their frequency in the dataset:

$$Macro\ Avg(M) = \frac{1}{K}\sum_{i=1}^{K} M_i \tag{1}$$

where K is the number of classes and M_i is the value of the metric (Precision, Recall, or F1) for the i-th class.

In contrast, the Weighted Average takes into account the proportion of each class in the dataset, which allows a more accurate evaluation of the overall model performance in the presence of class imbalance.:

$$\textit{Weighted Avg}(M) = \frac{\sum_{i=1}^{K} (n_i \cdot M_i)}{\sum_{i=1}^{K} n_i} \tag{2}$$

where n_i is the number of samples (support) in the i-th class.

Based on the calculations, the results show Macro Avg (Precision) = 0.9965, Macro Avg (Recall) = 0.9974, and Macro Avg (F1) = 0.9969, while the Weighted Average displays consistently high values of 0.9972 for all three metrics. This indicates the balanced performance of the classifier. The model effectively recognizes both legitimate and malicious traffic without favoring one class over the other.

Figure 6 illustrates the relationship between the True Positive Rate (TPR), True Negative Rate (TNR), and the Youden's J index (J = TPR − FPR) as a function of the decision threshold. As the threshold increases, a predictable rise in TNR (Specificity) and a slight decrease in TPR (Recall) can be observed, allowing the determination of the optimal equilibrium point between these indicators. The maximum value of Youden's J index is achieved at the threshold 0.6171, which provides the best balance between effective attack detection and the minimization of the False Positive Rate (FPR).

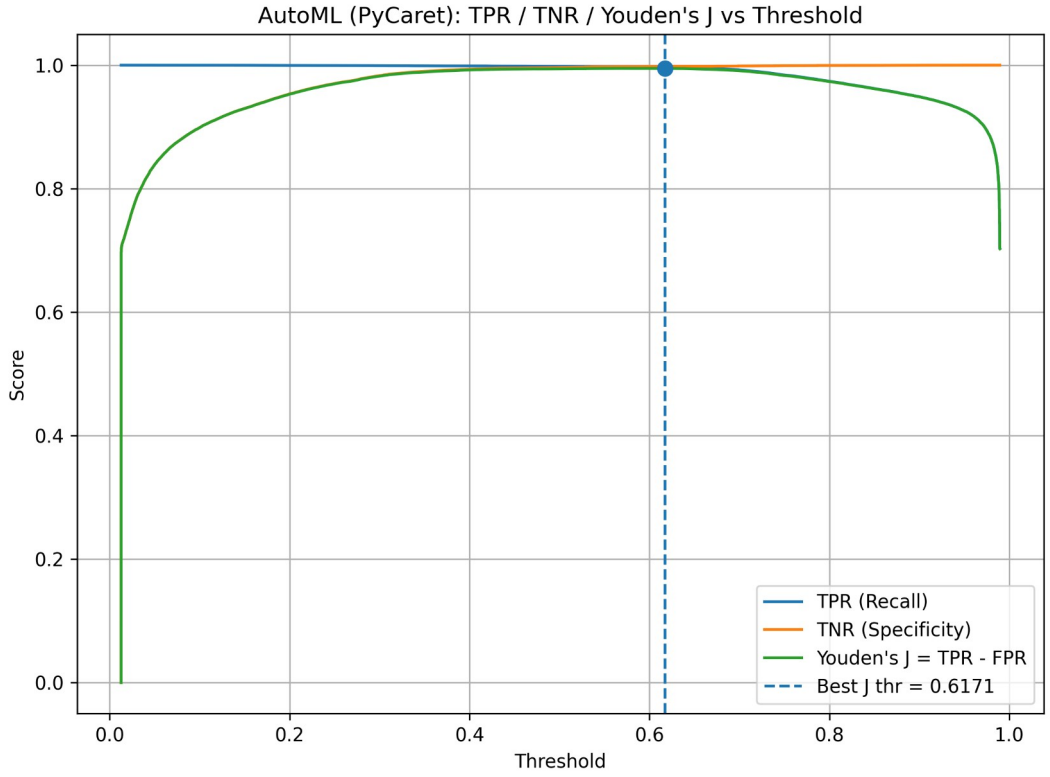


Figure 6: TPR, TNR, and Youden's J versus decision threshold.

Figure 7 presents the Receiver Operating Characteristic (ROC) curve of the model, with the point corresponding to the optimal threshold determined by Youden's J index marked on the graph.

The Area Under the Curve (AUC = 0.9999) indicates an exceptionally high discriminative ability of the model, showing that it almost perfectly distinguishes between normal and attack traffic. This

result confirms the consistency of the classifier with the optimal decision criterion and demonstrates its effectiveness for practical implementation in intrusion detection systems.

Figure 8 presents the results of the global SHAP analysis for the Attack class (class 1), illustrating the relative contribution of the most important features to the decision-making process of the Random Forest model.

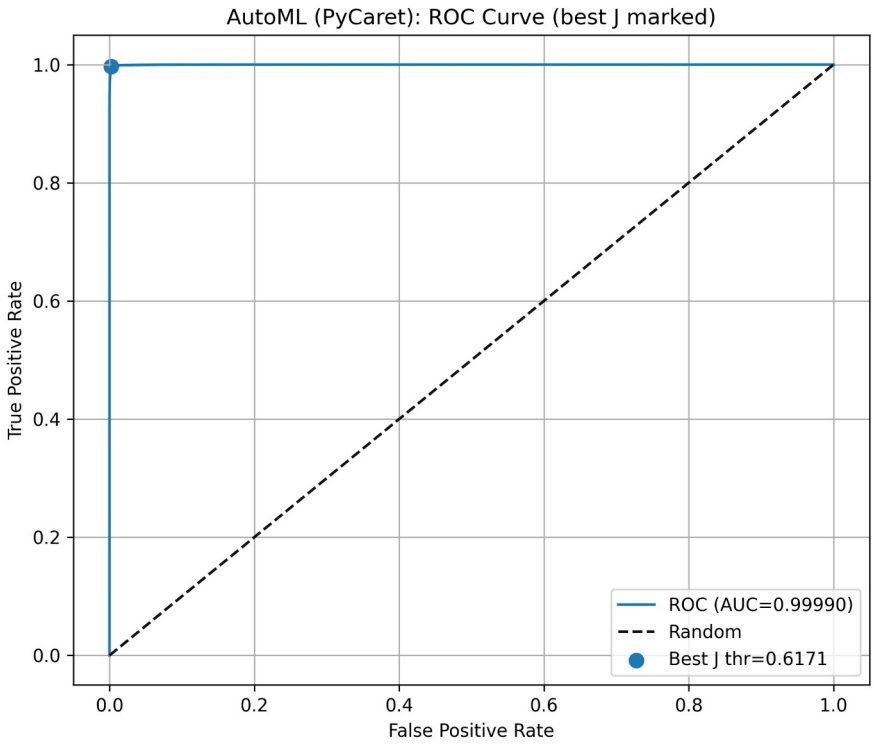


Figure 7: ROC curve with the optimal Youden's J threshold marked.

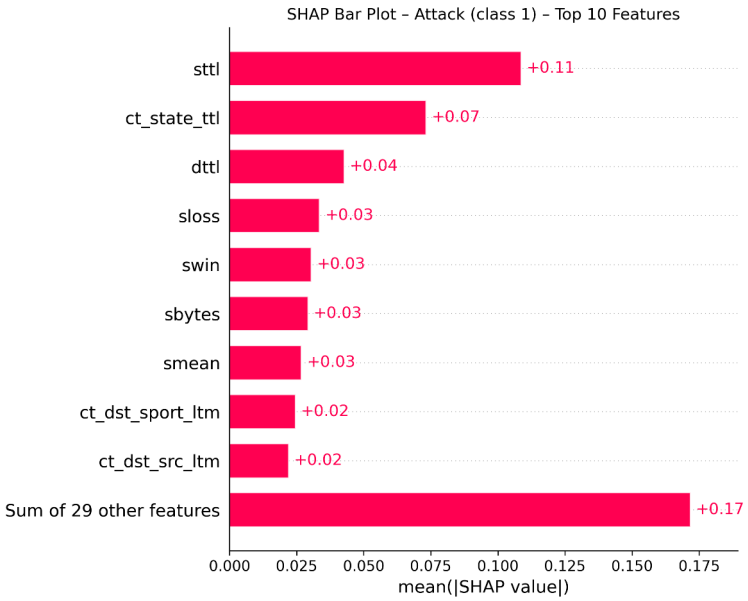


Figure 8: Global SHAP feature importance for the Attack class (Random Forest model).

The horizontal axis shows the mean absolute SHAP value, which quantitatively reflects the importance of each feature in predicting the probability of an attack. The parameters sttl (≈ 0.11) and ct_state_ttl (≈ 0.07) have the strongest influence on the predictions. These features describe the temporal characteristics of packets and the connection states at the network level. Their high importance indicates that variations in Time to Live (TTL) values and the frequency of specific TTL states are key indicators distinguishing legitimate from malicious traffic. The next most influential features are dttl, sloss, swin, sbytes, and smean (within the range of 0.03–0.04), which represent behavioral and transport properties of network flows such as packet loss rate, TCP window parameters, total transmitted bytes, and average packet size. These indicators allow the model to effectively detect channel overloads or deviations from normal session profiles, which are typical for DoS and scanning attacks. The features ct_dst_sport_ltm and ct_dst_src_ltm (≈ 0.02) show a slightly lower yet still significant contribution, as they represent the recurrence of connections between the same IP addresses and ports within the last 100 sessions — a common pattern for brute force or port scanning attacks. The combined contribution of the remaining 29 features is approximately 0.17.

Figure 9 shows the SHAP summary plot for the Attack class (class 1), which illustrates the distribution of feature impacts (SHAP values) on the Random Forest model's decision to classify network traffic as malicious.

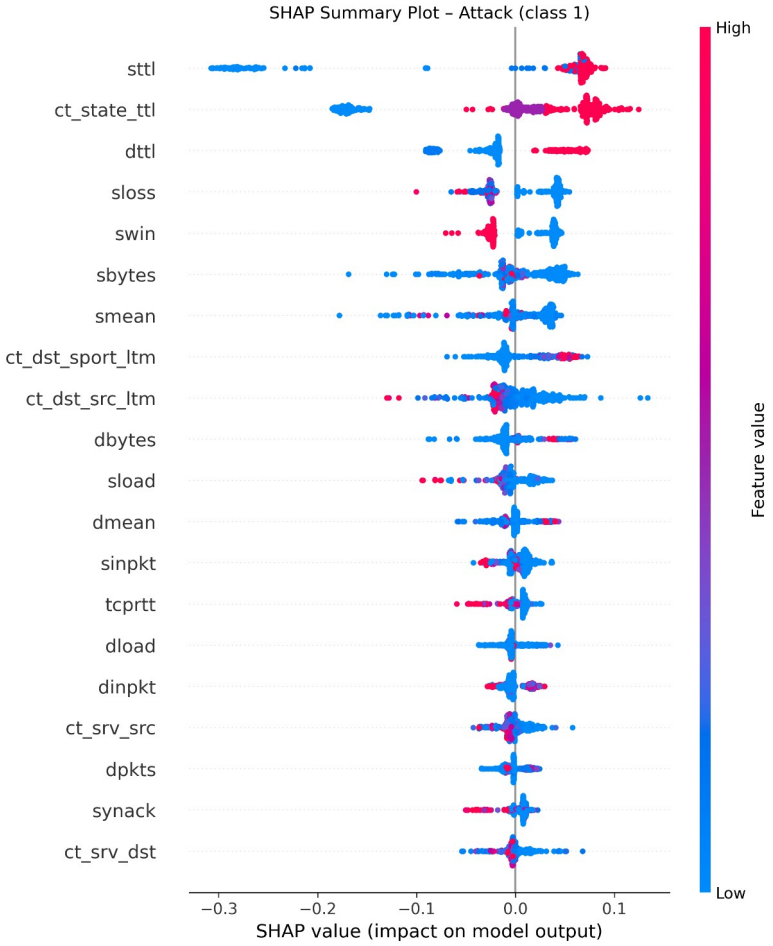


Figure 9: SHAP summary plot for the Attack class showing feature impact on the Random Forest model's decisions.

On the horizontal axis, the plot displays the extent and direction of each feature's contribution to the prediction. Higher SHAP values push the model toward the Attack class, whereas lower values reduce this likelihood. The color scale reflects the normalized feature values, where red

corresponds to high feature values and blue to low ones. The most influential features in predicting attacks are sttl, ct_state_ttl, and dttl, which describe the temporal parameters and connection states at the network level. High feature values substantially increase the probability of classification as Attack, whereas low values reduce it. This behavior highlights the model's sensitivity to abnormal TTL characteristics.

Figure 10 presents a local SHAP explanation in the form of a waterfall plot for an individual traffic sample (Sample 73), which was classified by the Random Forest model as Attack (class 1) with a predicted probability of $f(x) = 0.85$.

The plot illustrates how individual features contribute to shifting the model output from the baseline value $E[f(X)] = 0.675$ (the average predicted probability across the dataset) toward the final result for this particular observation. Positive contributions (shown in red) increase the probability of classification as Attack, while negative contributions (shown in blue) decrease it. The strongest positive effects come from the features sttl (+0.06) and dttl (+0.04). Slightly smaller but still noticeable contributions are made by ct_dst_src_ltm (+0.02), sload (+0.02), dur (+0.02), and dinpkt (+0.02), which describe source activity and flow duration, typical indicators of high connection intensity that often occur during attacks. The results of the SHAP analysis confirm that the model is based on logically consistent relationships aligned with the behavioral characteristics of network attacks. This provides not only high classification accuracy but also interpretability, which is essential for practical use in intrusion detection systems.

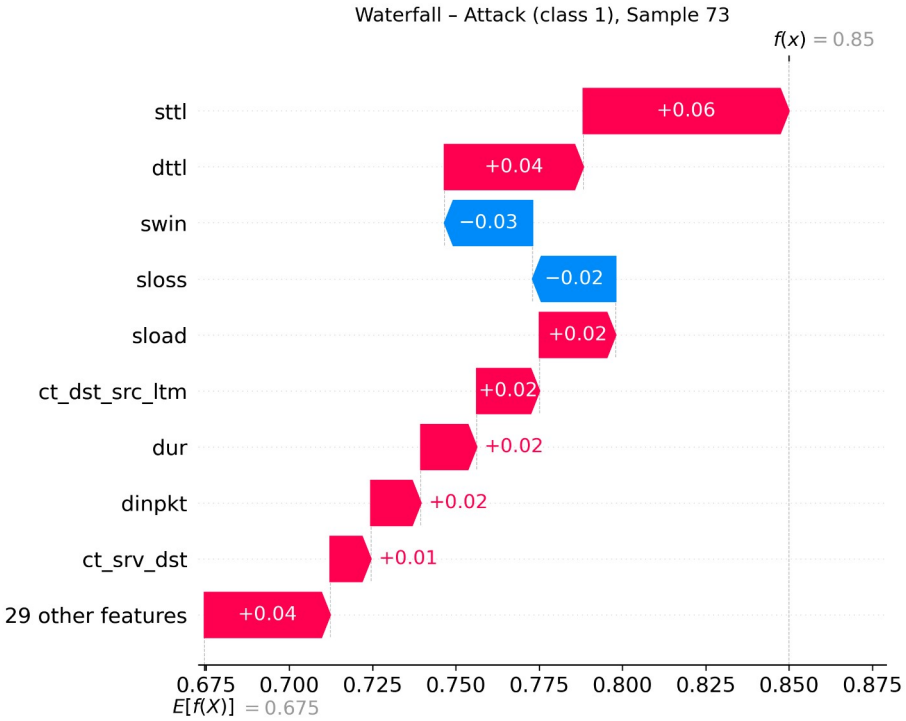


Figure 10: Local SHAP explanation for Sample 73 classified as Attack.

The developed machine learning model was integrated into the IDS as an analytical module for real time network traffic classification. The integration was carried out within an isolated laboratory environment deployed on a Kernel based Virtual Machine (KVM) hypervisor, which made it possible to simulate a complex multilayer infrastructure and create controlled conditions for testing. This approach allowed the reproduction of realistic interaction scenarios between legitimate users and potential attackers and enabled the evaluation of model performance under conditions similar to real corporate networks. The virtual environment included several operating

systems with different functional roles. Parrot Security OS and Kali Linux were used to initiate attacks and perform penetration tests, generating traffic that contained signs of malicious activity. Ubuntu Linux operated as a server providing common network services such as HTTP, DNS, IMAP, SMTP, and SSH. Metasploitable VM served as a vulnerable machine for testing exploits, while Windows Server simulated a corporate environment, running Active Directory, DNS, DHCP, File Services, and IIS. During testing, network traffic from all virtual machines was redirected through the IDS sensor, which performed data collection, preliminary processing, and feature extraction [36, 37] for further analysis. The processed data were then sent to the machine learning model, which classified the network flows as Normal or Attack. The classification results were stored for statistical evaluation, comparison with reference data, and performance metric calculation. This setup enabled a comprehensive assessment of the model under realistic network load conditions. The laboratory integration confirmed that the IDS equipped with the embedded machine learning model was able to detect attacks with high accuracy and a low false positive rate.

5. Conclusion

This study demonstrated that combining the AutoML approach with Explainable AI methods enables the development of a highly accurate and interpretable model for intrusion detection systems. A reproducible processing pipeline was designed, covering all stages from preprocessing to probability calibration. It made possible to automate algorithm selection, hyperparameter optimization, and improve the stability of the results. The best configuration was achieved using the Random Forest model, which at the optimal threshold determined by Youden's J index reached Accuracy = 0.9972 and an almost perfect ability to distinguish between classes (AUC = 0.9999). Optimization of the decision threshold according to the Youden's J criterion reduced the number of false positives without a noticeable loss of Recall, while interpretability analysis with the SHAP method confirmed the logical and well-founded nature of the model's decisions. The integration of the developed model as an analytical module into the IDS within a KVM-based laboratory environment with a multi component infrastructure confirmed its practical effectiveness for real time operation. The combination of AutoML methods and XAI explanations improved accuracy, transparency and trust in the results, which are key prerequisites for implementing such solutions in modern cybersecurity systems.

Declaration on Generative AI

During the preparation of this work, the authors used Grammarly in order to grammar and spell check, and improve the text readability. After using the tool, the authors reviewed and edited the content as needed to take full responsibility for the publication's content.

References

[1] What is an intrusion detection system?, Palo Alto Networks (online). URL: https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-detection-system-ids.

[2] What is an intrusion prevention system?, Palo Alto Networks (online). URL: https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-prevention-system-ips.

[3] D. Tymoshchuk, O. Yasniy, M. Mytnyk, N. Zagorodna, V. Tymoshchuk, Detection and classification of DDoS flooding attacks by machine learning method, CEUR Workshop Proceedings 3842 (2024) 184–195.

[4] N. Petliak, Y. Klots, M. Karpinski, V. Titova, D. Tymoshchuk, Hybrid system for detecting abnormal traffic in IoT, CEUR Workshop Proceedings 4057 (2025) 21–36.

[5] Y. Klots, V. Titova, N. Petliak, D. Tymoshchuk, N. Zagorodna, Intelligent data monitoring anomaly detection system based on statistical and machine learning approaches, CEUR Workshop Proceedings 4042 (2025) 80–89.

[6] M. Chornobuk, V. Dubrovin, L. Deineha, Cybersecurity: research on methods for detecting DDoS attacks, Comput. Syst. Inf. Technol. 4 (2023) 6–9. doi:10.31891/csit-2023-4-1.

[7] K. S. Durbha, S. Amuru, AutoML models for wireless signals classification and their effectiveness against adversarial attacks, in: Proc. 14th Int. Conf. on Communication Systems & Networks (COMSNETS), IEEE (2022). doi:10.1109/comsnets53615.2022.9668448.

[8] E. Horsanali, Y. Yigit, G. Secinti, A. Karameseoglu, B. Canberk, Network-aware AutoML framework for software-defined sensor networks, in: Proc. 17th Int. Conf. on Distributed Computing in Sensor Systems (DCOSS), IEEE (2021). doi:10.1109/dcoss52077.2021.00076.

[9] N. K. Gyimah, R. Akinie, J. Mwakalonge, B. Izison, A. Mukwaya, D. Ruganuza, M. Sulle, An AutoML-based approach for network intrusion detection, in: Proc. IEEE SoutheastCon (2025) 1177–1183. doi:10.1109/southeastcon56624.2025.10971461.

[10] D. Vasan, E. J. S. Alqahtani, M. Hammoudeh, A. F. Ahmed, An AutoML-based security defender for industrial control systems, Int. J. Crit. Infrastruct. Prot. (2024) 100718. doi:10.1016/j.ijcip.2024.100718.

[11] L. Yang, A. Shami, Towards autonomous cybersecurity: an intelligent AutoML framework for autonomous intrusion detection, in: Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS) (2023) 68–78. doi:10.1145/3689933.3690833.

[12] A. Papanikolaou, A. Alevizopoulos, C. Ilioudis, K. Demertzis, K. Rantos, An AutoML network traffic analyzer for cyber threat detection, Int. J. Inf. Secur. (2023). doi:10.1007/s10207-023-00703-0.

[13] SHAP: A game theoretic approach to explain the output of any machine learning model, GitHub repository. URL: https://github.com/shap/shap.

[14] Local interpretable model-agnostic explanations (LIME), InterpretML documentation. URL: https://interpret.ml/docs/lime.html.

[15] Permutation feature importance, scikit-learn documentation. URL: https://scikit-learn.org/0.24/modules/permutation_importance.html.

[16] M. Amirian, L. Tuggener, R. Chavarriaga, Y. P. Satyawan, F.-P. Schilling, F. Schwenker, T. Stadelmann, Two to trust: AutoML for safe modelling and interpretable deep learning for robustness, in: Trustworthy AI – Integrating Learning, Optimization and Reasoning, Springer, Cham (2021) 268–275. doi:10.1007/978-3-030-73959-1_23.

[17] A. Kuppa, N.-A. Le-Khac, Black box attacks on explainable artificial intelligence (XAI) methods in cyber security, in: Proc. Int. Joint Conf. on Neural Networks (IJCNN), IEEE (2020). doi:10.1109/ijcnn48605.2020.9206780.

[18] A. Kuppa, N.-A. Le-Khac, Adversarial XAI methods in cybersecurity, IEEE Trans. Inf. Forensics Secur. 16 (2021) 4924–4938. doi:10.1109/tifs.2021.3117075.

[19] S. Tabassum, N. Parvin, N. Hossain, A. Tasnim, R. Rahman, M. I. Hossain, IoT network attack detection using XAI and reliability analysis, in: Proc. 25th Int. Conf. on Computer and Information Technology (ICCIT), IEEE (2022). doi:10.1109/iccit57492.2022.10055236.

[20] P. Barnard, N. Marchetti, L. A. D. Silva, Robust network intrusion detection through explainable artificial intelligence (XAI), IEEE Netw. Lett. (2022) 1. doi:10.1109/lnet.2022.3186589.

[21] N. Moustafa, J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: Proc. Military Communications and Information Systems Conf. (MilCIS), IEEE (2015). doi:10.1109/milcis.2015.7348942.

[22] D. Wells, UNSW-NB15 dataset, Kaggle (2019). URL: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15/data.

[23] N. Moustafa, G. Creech, J. Slay, Big data analytics for intrusion detection system: statistical decision-making using finite Dirichlet mixture models, in: Data Analytics and Decision Support for Cybersecurity, Springer, Cham (2017) 127–156. doi:10.1007/978-3-319-59439-2_5.

[24] AutoML PyCaret, GitHub repository. URL: https://github.com/dasarpai/automl-pycaret.

[25] RandomForestClassifier, scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[26] E. Kavlakoglu, E. Russi, What is XGBoost?, IBM (online). URL: https://www.ibm.com/think/topics/xgboost.
[27] ExtraTreesClassifier, scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html.
[28] LightGBM documentation. URL: https://lightgbm.readthedocs.io/en/stable/.
[29] B. Clark, F. Lee, What is gradient boosting?, IBM (online). URL: https://www.ibm.com/think/topics/gradient-boosting.
[30] RidgeClassifier, scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RidgeClassifier.html.
[31] F. Lee, What is logistic regression?, IBM (online). URL: https://www.ibm.com/think/topics/logistic-regression.
[32] Neural network models (supervised), scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/neural_networks_supervised.html.
[33] Support vector machines, scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/svm.html.
[34] Classification performance metrics and indices, Online resource. URL: https://adriancorrendo.github.io/metrica/articles/available_metrics_classification.html.
[35] CalibratedClassifierCV, scikit-learn documentation. URL: https://scikit-learn.org/stable/modules/generated/sklearn.calibration.CalibratedClassifierCV.html
[36] B. Lypa, I. Horyn, N. Zagorodna, D. Tymoshchuk, T. Lechachenko, Comparison of feature extraction tools for network traffic data, CEUR Workshop Proceedings 3896 (2024) 1–11.
[37] O. Savenko, S. Lysenko, A. Kryshchuk, Y. Klots, Botnet detection technique for corporate area network, in Proceedings of the the IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS) IEEE, 2013, pp. 363-368.