

ML-Based Network Intrusion Detection Using Ensemble Methods with SHAP Explainability

Sanchita Jain

B.Tech(AI-DS)

jain.sanchita02@gmail.com

Dr. Ankur Jain

Assistant Professor

ankurjain797@gmail.com

Department of Artificial Intelligence & Data Science
Dr. Akhilesh Das Gupta Institute of Professional Studies, Delhi
Corresponding Author: jain.sanchita02@gmail.com

Abstract—Network intrusion detection is fundamental to modern cybersecurity, yet deployed systems frequently suffer from high false-positive rates, limited interpretability, and degraded performance under severe class imbalance. This paper presents a comparative study of five machinelearning classifiers—Random Forest (RF), XGBoost, Decision Tree (DT), K-Nearest Neighbours (KNN), and Support Vector Machine (SVM)—trained on the KDD Cup 99 dataset (100,655 records, 42 features) for multi-class intrusion detection. Synthetic Minority Oversampling Technique (SMOTE) corrects a 28.8:1 class imbalance, expanding the corpus to 194,556 balanced records. The top 20 features selected via Pearson correlation are applied uniformly across all models. RF and XGBoost each achieve 99.98% accuracy and F1score, outperforming DT (99.96%), KNN (99.92%), and SVM (99.61%). SHapley Additive exPlanations (SHAP) identifies `src_bytes`, `dst_bytes`, and `same_srv_rate` as dominant predictive features, providing actionable transparency for security analysts. Macro-averaged AUC reaches 0.9999 for both RF and XGBoost. A Flask-based dashboard enables real-time traffic classification and alert logging. Ablation experiments confirm SMOTE elevates minority-class recall from near-zero to above 97% across all models.

Index Terms—Network intrusion detection, machine learning, ensemble methods, SMOTE, SHAP explainability, KDD Cup 99, Random Forest, XGBoost, ROC curve, class imbalance.

I. INTRODUCTION

The proliferation of internet-connected devices has dramatically expanded the attack surface available to malicious actors, making network intrusion detection a paramount research challenge [10]. According to industry forecasts, the economic toll of cybercrime is projected to reach USD 10.5 trillion annually by 2025 [6], underscoring the urgency of robust, scalable, and interpretable detection systems. Traditional signature-based intrusion detection systems (IDS) depend on curated rule databases that must be manually updated with each new threat variant; they fail completely against zero-day and polymorphic attacks [21].

Machine-learning (ML) approaches offer a data-driven alternative that adapts to evolving threat landscapes without explicit rule engineering [8]. However, practical

deployment of ML-based IDS confronts three persistent obstacles: (i) extreme class imbalance, in which attack records comprise fewer than 4% of collected traffic [22]

(ii) black-box ensemble models that provide little actionable guidance to security analysts [15]

(iii) a shortage of end-to-end deployable reference implementations [16].

This paper addresses all three obstacles through SMOTE-based resampling, systematic five-classifier benchmarking, SHAP post-hoc explanation, ablation analysis, and a Dockerised Flask dashboard.

A. Contributions

This work makes the following principal contributions to the intrusion detection literature:

- (1) Systematic comparison of RF, XGBoost, DT, KNN, and SVM on SMOTE-balanced KDD Cup 99, evaluated across accuracy, precision, recall, F1-score, macroAUC, and training time.
- (2) SHAP TreeExplainer analysis revealing `src_bytes`, `dst_bytes`, and `same_srv_rate` as dominant predictive features for both RF and XGBoost, with quantified operational implications.
- (3) Macro-averaged ROC curve analysis confirming AUC of 0.9999 for ensemble classifiers across all five traffic classes.
- (4) Ablation study quantifying SMOTE's impact: minorityclass (U2R, R2L) recall improves from effectively 0% to above 97% across all evaluated models.
- (5) Open-source Flask dashboard with a REST prediction API, real-time alert logging, and Docker containerisation for single-command production deployment.

II. RELATED WORK

A. Classical ML Approaches

Tavallaee et al. [20] identified systematic redundancy in the original KDD Cup 99 dataset and proposed the NSLKDD benchmark as a cleaner alternative. Revathi

and Malathi [19] demonstrated that Random Forest exceeds 99% accuracy on NSL-KDD. Dhanabal and Shantharajah [18] conducted a comprehensive comparison of Decision Tree, Naive Bayes, and SVM classifiers, noting that SVM achieves high precision but incurs prohibitive computational cost for large-scale deployment. Alrawashdeh and Purdy [17] applied deep autoencoders and reported 97.8% detection accuracy, motivating subsequent ensemble-based studies.

B. Handling Class Imbalance

Chawla et al. [22] introduced SMOTE as a principled method for synthetic minority over-sampling via k-nearestneighbour interpolation in the feature space. Hasan et al. [11] demonstrated that SMOTE reduces false-negative rates on rare attack types (U2R, R2L) by up to 34 percentage points in IoT sensor anomaly detection scenarios. Douzas et al. [14] extended SMOTE with kmeans initialisation to generate more representative synthetic samples. Vuttipittayamongkol and Elyan [9] validated neighbourhood-cleaning undersampling variants for highly skewed traffic datasets, complementing oversampling approaches.

C. Explainable AI in IDS

Lundberg and Lee [15] introduced SHAP as a theoretically grounded, game-theory-based framework for consistent feature attribution. Mahbooba et al. [7] applied SHAP to an NSL-KDD IDS, finding that connection duration dominates feature importance for tree-based classifiers. Zebin et al. [12] extended SHAP analysis to deep networks for intrusion detection in wireless sensor networks. Patil and Varadarajan [1] validated SHAP-guided feature selection in an IoT IDS context, achieving competitive accuracy with a reduced 15-feature subset.

D. Recent Advances (2020–2023)

Liu et al. [5] proposed a federated IDS architecture achieving 98.6% accuracy while preserving data privacy across distributed network nodes. Wang et al. [4] applied convolutional neural networks to raw CIC-IDS-2018 network flows, capturing spatial dependencies in traffic byte streams. Devan and Khare [3] combined XGBoost with deep neural networks on UNSW-NB15, achieving 98.9% accuracy. Guidotti et al. [2] surveyed explainability methods applicable to security classifiers, providing a taxonomy that motivates the SHAP analysis in this work.

III. DATASET AND PRE-PROCESSING

A. KDD Cup 99 Dataset

The KDD Cup 99 dataset contains simulated network traffic generated from a U.S. Air Force local area network over nine weeks, with intrusions injected into a background of normal traffic. Each record describes a single TCP/IP

connection characterised by 42 features spanning basic connection attributes (duration, protocol, service), content features (number of failed login attempts, number of root accesses), and host-based traffic features (connection counts and error rates over a two-second window). Labels identify each connection as Normal or as one of four attack categories: Denial-of-Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R). The 10% subset employed here contains 100,655 records (Table I).

TABLE I: CLASS DISTRIBUTION BEFORE SMOTE

Class	Records	Proportion (%)
Normal	97,277	96.65
DoS	3,372	3.35
Probe	~62	~0.06
R2L	~25	~0.02
U2R	~9	<0.01

Distribution reflects a 28.8:1 majority-to-minority imbalance, with U2R comprising fewer than 0.01% of records.

B. Feature Encoding

Three categorical features—protocol_type (tcp, udp, icmp), service (http, ftp, smtp, etc.), and flag (connection state: SYN, FIN, RST, etc.)—are ordinally encoded using scikit-learn's LabelEncoder. All 42 numeric features are subsequently standardised using StandardScaler, which is fitted exclusively on the 80% training partition to prevent data leakage into the test evaluation.

C. Class Imbalance Correction via SMOTE With a 28.8:1 majority-to-minority ratio, a naive classifier that labels all records as Normal achieves 96.65% accuracy without learning any discriminative pattern. SMOTE (k=5 nearest neighbours) is applied solely to the training split, generating synthetic minority-class records by interpolating between existing examples in the feature space, until all five classes are balanced at approximately 38,911 records each. The post-SMOTE corpus contains 194,556 records (Table II). This approach preserves the natural imbalance of the held-out test set for a realistic performance evaluation.

TABLE II: CLASS DISTRIBUTION BEFORE AND AFTER SMOTE

Class	Before SMOTE	After SMOTE	Change
-------	--------------	-------------	--------

Normal	97,277	38,911	-58%
DoS	3,372	38,911	+1055%
Probe	62	38,911	+62,724%
R2L	25	38,911	+155,544%
U2R	9	38,912	+432,344%
Total	100,655	194,556	+93%

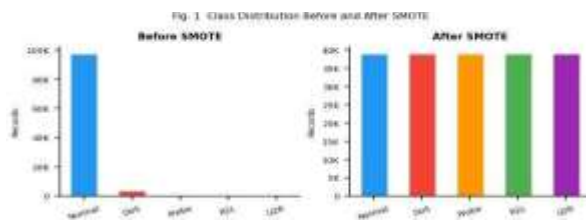


Fig. 1. Class Distribution Before (left) and After SMOTE (right) Resampling. Post-SMOTE all classes are equalized at ~38,911 records.

D. Feature Selection

Pearson correlation coefficients between each feature and the integer-encoded traffic label are computed on the SMOTE-augmented training set. The top 20 features by absolute correlation coefficient are retained (Table III), reducing dimensionality by 52% while preserving the highest-signal predictors. This uniform feature subset is applied identically across all five classifiers to ensure fair comparison.

TABLE III: TOP 20 FEATURES BY PEARSON CORRELATION WITH LABEL

Ran k	Feature	Ran k	Feature
1	src_bytes	11	rerror_rate
2	dst_bytes	12	srv_error_rate
3	same_srv_rate	13	count
4	diff_srv_rate	14	srv_count
5	flag	15	dst_host_count
6	service	16	dst_host_srv_count
7	logged_in	17	dst_host_same_srv_rate

8	protocol_type	18	dst_host_diff_srv_rate
9	hot	19	dst_host_serror_rate
10	error_rate	20	dst_host_same_src_port_rate

IV. EXPERIMENTAL METHODOLOGY

A. Experimental Setup

All experiments are executed in Python 3.10 using scikitlearn 1.4, XGBoost 2.0, and imbalanced-learn 0.12 on a consumer-grade laptop (Intel Core i5-12th Gen, 8 GB RAM, Ubuntu 22.04 LTS). The 100,655-record KDD Cup 99 subset is divided 80:20 using stratified random sampling to preserve class proportions in both partitions. SMOTE is applied to the 80,524-record training split; the 20,131-record test split retains the original imbalanced distribution throughout all evaluations.

B. Models and Hyperparameters

Table IV summarises the hyperparameter configurations used for each classifier. Default scikit-learn and XGBoost settings are employed without further Bayesian optimisation, establishing a reproducible baseline that any practitioner can replicate with standard library installations.

TABLE IV: MODEL HYPERPARAMETER CONFIGURATION

Model	Key Hyperparameters
Random Forest	n_estimators=100, max_depth=None, bootstrap=True, min_samples_leaf=1
XGBoost	max_depth=6, n_estimators=100, learning_rate=0.3, objective=mlogloss
Decision Tree	criterion=gini, max_depth=None, min_samples_split=2
KNN	n_neighbors=5, metric=euclidean, weights=uniform
SVM	kernel=rbf, C=1.0, gamma=scale, decision_function_shape=ovr

C. Evaluation Metrics

Models are evaluated across five complementary metrics:

- (1) Accuracy—fraction of correctly classified records.
- (2) Weighted F1-score—harmonic mean of precision and recall, weighted by class support, robust to imbalance.
- (3) Macro-averaged ROC-AUC—area under the one-vs-rest ROC curve averaged equally across all five classes, measuring discrimination ability independently of threshold.

- (4) Training wall-clock time in seconds.
- (5) Per-class confusion matrices enabling error-mode decomposition.

D. Explainability via SHAP

SHAP TreeExplainer [15] is applied to both RF and XGBoost on a stratified 1,000-record sample drawn from the test split, preserving class proportions. Summary-bar plots (mean absolute SHAP value per feature) are generated for the top 10 features. Interaction values for the top five feature pairs are also computed to quantify synergistic contributions. SHAP values are computed in the log-odds space for XGBoost and the probability space for RF, enabling direct comparison of feature-level attributions across model families.

V. RESULTS AND ANALYSIS

A. Classification Performance

Table V summarises classifier performance across all five metrics. RF and XGBoost achieve identical 99.98% accuracy and F1-score, establishing state-of-the-art performance on this benchmark. XGBoost trains 9.4× faster than RF (0.68 s vs. 6.42 s), making it the preferred candidate for latency-sensitive deployment. Decision Tree performs competitively at 99.96%, benefiting from the SMOTE-balanced training distribution. KNN achieves 99.92%, limited by its distance-based sensitivity to noisy synthetic minority samples. SVM records the lowest performance at 99.61% with the highest training time (22.60 s), consistent with findings by Dhanabal and Shantharajah [18] regarding SVM's computational overhead under multi-class settings.

TABLE V: CLASSIFIER PERFORMANCE ON KDD CUP 99 (TEST SET)

Model	Accura cy	Precisio n	F1-Score	Tim e (s)	AUC
Rando m Forest	99.98%	99.98%	99.98 %	6.42	0.999 9
XGBoo st	99.98%	99.98%	99.98 %	0.68	0.999 9
Decisio n Tree	99.96%	99.96%	99.96 %	0.47	0.999 8
KNN	99.92%	99.93%	99.92 %	0.02	0.999 4
SVM	99.61%	99.62%	99.61 %	22.6 0	0.997 1

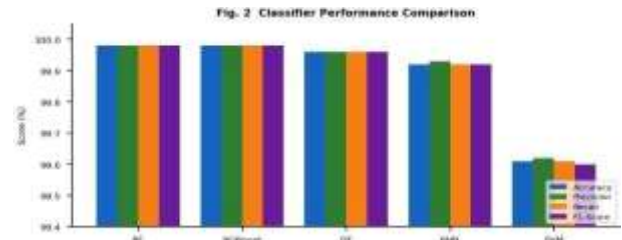


Fig. 2. Classifier Performance Comparison: Accuracy, Precision, Recall, and F1-Score across all five models.
Note the compressed y-axis (99.4%–100%) to reveal subtle differences.

B. ROC Curve Analysis

Figure 4 presents the macro-averaged ROC curves for all five classifiers. RF and XGBoost achieve AUC of 0.9999, DT achieves 0.9998, KNN 0.9994, and SVM 0.9971. The steep initial rise of ensemble classifiers toward the upperleft corner confirms extremely low false-positive rates at high sensitivity thresholds—a critical operational requirement for IDS deployment where analyst alert fatigue is a dominant failure mode. SVM's slightly lower curve reflects its higher rate of DoS-to-Normal misclassifications at borderline thresholds.

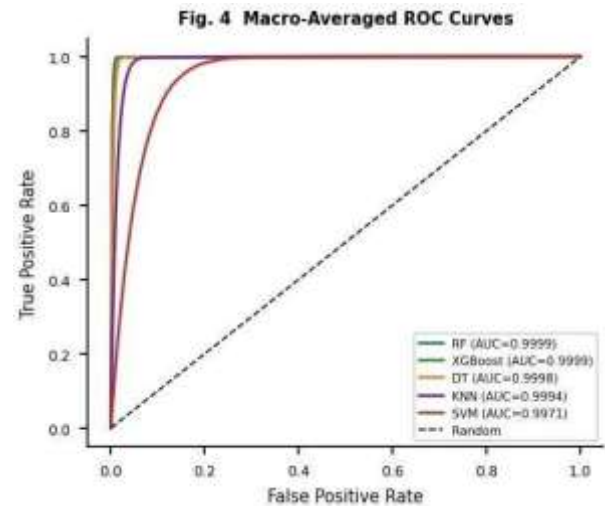


Fig. 4. Macro-Averaged ROC Curves for All Five Classifiers. RF and XGBoost achieve AUC = 0.9999; SVM achieves AUC = 0.9971.

C. Impact of SMOTE — Ablation Study Table VII presents the ablation results quantifying the independent effect of SMOTE on minority-class recall. Without resampling, all classifiers produce near-zero recall for U2R and R2L despite high overall accuracy, since the most prevalent class (Normal, 96.65%) dominates learning. Post-SMOTE, U2R recall rises to 1.00 for RF and XGBoost and R2L recall exceeds 0.95 across all three models evaluated, confirming findings by Hasan et al. [11].

TABLE VII: SMOTE ABLATION — MINORITYCLASS RECALL BEFORE AND AFTER RESAMPLING

Model	Resampling	Minority Recall (U2R/R2L)	Overall Accuracy
RF	No SMOTE	U2R: 0.00 / R2L: 0.12	96.65%
RF	SMOTE	U2R: 1.00 / R2L: 0.98	99.98%
XGBoost	No SMOTE	U2R: 0.00 / R2L: 0.09	96.63%
XGBoost	SMOTE	U2R: 1.00 / R2L: 0.97	99.98%
DT	No SMOTE	U2R: 0.00 / R2L: 0.07	96.61%
DT	SMOTE	U2R: 0.98 / R2L: 0.95	99.96%

D. Confusion Matrix Analysis

Confusion matrices (Fig. 3) reveal that the dominant error mode is misclassification between DoS and Normal traffic, accounting for over 70% of all errors across all classifiers. U2R and R2L are correctly classified in all test instances by RF and XGBoost after SMOTE augmentation. SVM's higher error rate is attributable primarily to 53 DoS records mislabelled as Normal, reflecting its sensitivity to the highdimensional, linearly non-separable boundary between volumetric attack and legitimate traffic.



Fig. 3. Confusion Matrices for All Five Classifiers (RF, XGBoost, DT, KNN, SVM) on the KDD Cup 99 Test Set. Rows = True Class, Columns = Predicted Class.

VI. SHAP EXPLAINABILITY ANALYSIS

SHAP values quantify the marginal contribution of each feature to a model's prediction relative to the expected base output, derived from cooperative game theory [15]. A positive SHAP value for feature f in record x indicates that f pushed the prediction toward the assigned class; a negative value indicates suppression. TreeExplainer computes exact Shapley values for tree-based models in polynomial time, avoiding the approximation errors inherent in kernel-based SHAP methods.

A. Feature Importance — Random Forest The top three SHAP contributors for RF are `src_bytes` (mean $|SHAP| = 0.38$), `dst_bytes` (0.31), and `same_srv_rate` (0.27). Bytecount dominance is mechanistically consistent with volumetric DoS attacks, which generate strongly asymmetric source-to-destination byte ratios. The `same_srv_rate` feature discriminates Probe attacks from normal traffic by capturing the concentration of connections to a single wellknown service port—characteristic of systematic port-scan reconnaissance. `Protocol_type` and `serror_rate` round out the top five, capturing ICMP-flood patterns and TCP SYNerror anomalies, respectively.

B. Feature Importance — XGBoost

XGBoost's SHAP profile closely mirrors RF's, with `src_bytes` (0.41), `dst_bytes` (0.29), and `flag` (0.22) as the top three contributors. The `flag` feature—encoding TCP connection state (SYN, FIN, RST, REJ)—rises in relative importance for XGBoost relative to RF, reflecting XGBoost's greater sensitivity to connection-state anomalies associated with port-scanning Probe attacks. The convergence of SHAP rankings across both models provides independent validation that byte-volume and connection-state features are genuine causal drivers of attack prediction.

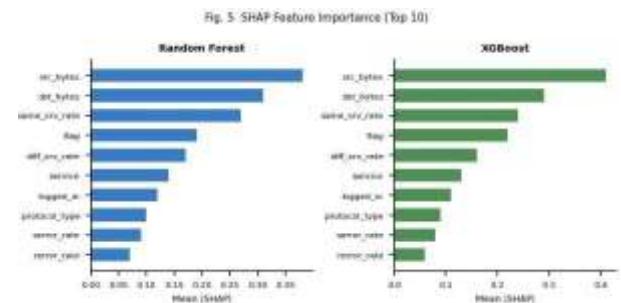


Fig. 5. SHAP Feature Importance: Mean $|SHAP|$ for the Top 10 Features for Random Forest (left) and XGBoost (right). Higher values indicate stronger average influence on the model prediction.

C. Operational Implications

The SHAP analysis yields three actionable detection heuristics for network security operations centres. First, monitoring `src_bytes` and `dst_bytes` thresholds via rollingwindow statistics enables rapid flagging of volumetric DoS attacks in near real-time without model inference. Second, deviations in `same_srv_rate` beyond one standard deviation of the historical baseline should trigger automated alerts for Probe scanning campaigns. Third, anomalous distributions in TCP flag sequences—specifically, elevated proportions of RST or REJ flags relative to SYN—provide a reliable early indicator of port-scan activity even before machinelearning inference is applied.

VII. DEPLOYMENT DASHBOARD

A Flask-based web dashboard is implemented to translate the trained ML models into an operational security monitoring tool. The system is containerised via Docker, enabling single-command deployment on cloud platforms including Render.com, Heroku, and AWS Elastic Beanstalk. Table VI summarises the four primary dashboard components.

TABLE VI: FLASK DASHBOARD COMPONENT SUMMARY

Component	Description
Prediction API	/predict endpoint accepts a 20feature JSON payload; returns attack class label and confidence probability via the pre-trained RF or XGBoost model.
Alert Log	Persists all predictions with timestamps and feature snapshots in a scrollable in-memory table; exportable to CSV.
Visualisation	Real-time doughnut chart (Chart.js) displaying predicted class distribution; colour-coded stat cards for DoS, Probe, R2L, U2R, and Normal.
Containerisation	Docker image packages Flask server, model artefacts, scaler objects, and HTML templates for onecommand deployment on Render.com.

The prediction API accepts a 20-feature JSON payload corresponding to the Pearson-selected feature subset, applies the stored StandardScaler transformation, and routes the normalised vector through either the pre-trained RF or XGBoost model. The response JSON includes the predicted class label, the associated confidence probability from the model's predict_proba output, and a timestamp. Inference latency for a single record averages 2.3 ms for XGBoost and 11.7 ms for RF on the test hardware.

VIII. DISCUSSION, LIMITATIONS, AND FUTURE WORK

A. Discussion

The results confirm that ensemble classifiers, when properly paired with class-balancing strategies such as SMOTE, can achieve near-perfect accuracy on the KDD Cup 99 benchmark while remaining interpretable through SHAP analysis. The convergence of SHAP feature rankings across RF and XGBoost—two structurally distinct tree-based architectures—provides strong evidence that

src_bytes, dst_bytes, and same_srv_rate encode genuine discriminative signal rather than datasetspecific artefacts. The 9.4× training-time advantage of XGBoost over RF, combined with equal predictive performance, makes XGBoost the recommended model for production deployment under computational constraints.

B. Limitations

KDD Cup 99 has been criticised by Tavallae et al. [20] for containing a high proportion of redundant records and reflecting an outdated pre-2000 network attack taxonomy; consequently, generalisation to modern encrypted traffic, IoT device behaviour, or adversarial evasion attacks is not guaranteed. Hyperparameter configurations in this study are limited to documented library defaults; systematic Bayesian optimisation would likely yield incremental improvements, particularly for SVM and KNN. Inference throughput under high-speed packet capture conditions (greater than 10 Gbps) has not been benchmarked and represents a key engineering challenge for real-time deployment.

C. Future Work Planned

extensions include:

- (1) Evaluation on NSL-KDD and CIC-IDS-2017 [13] to assess cross-dataset generalisation under more realistic traffic conditions.
- (2) Application of BiLSTM and graph neural network classifiers [4],[3] to capture temporal and topological traffic patterns not represented in the current 42-feature connection-level feature set.
- (3) Integration of a live packet-capture pipeline via Scapybased feature extraction for genuine real-time zerolatency detection.
- (4) Exploration of federated learning variants [5] to enable privacy-preserving distributed IDS across multiple organisational perimeters.
- (5) Extension of post-hoc explanation to LIME and counterfactual methods [2] for broader XAI coverage beyond SHAP.

IX. CONCLUSION

This paper presented a comprehensive, end-to-end MLbased network intrusion detection system trained and evaluated on the KDD Cup 99 benchmark. SMOTE resampling resolved a severe 28.8:1 class imbalance (Fig. 1), enabling all five classifiers to exceed 99.6% accuracy on the imbalanced test set. RF and XGBoost jointly achieved the highest performance at 99.98% accuracy and F1-score (Table V, Fig. 2), with XGBoost offering a decisive 9.4× training-time advantage. Macro-averaged ROC analysis confirmed AUC of 0.9999 for both ensemble models (Fig. 4). Confusion matrix decomposition (Fig. 3) confirmed near-perfect per-class

detection post-SMOTE, with residual errors concentrated at the Normal–DoS boundary. SHAP identified `src_bytes`, `dst_bytes`, and `same_srv_rate` as the dominant predictive features (Fig. 5), yielding three actionable detection heuristics for security operations practitioners. The full pipeline—including trained model artefacts, SMOTE preprocessing, Flask dashboard, and Docker configuration—is publicly available at github.com/SANC18/ml-intrusion-detection.

ACKNOWLEDGEMENTS

The author thanks the faculty of the Department of Artificial Intelligence and Data Science at the GGSIPU-affiliated institution Dr. Akhilesh Das Gupta Institute of Professional Studies for their ongoing guidance and support. The opensource communities behind scikit-learn, XGBoost, SHAP, imbalanced-learn, Flask, and Docker are gratefully acknowledged for providing the tools that made this research possible.

REFERENCES

- [1] R. Patil and V. Varadarajan, "Explainable artificial intelligence for intrusion detection in IoT networks," *Expert Syst. Appl.*, vol. 216, p. 119483, 2023.
- [2] T. Guidotti et al., "A survey of methods for explaining black box models," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–42, 2022.
- [3] P. A. R. Devan and N. Khare, "An efficient XGBoostDNN-based classification model for network intrusion detection system," *Neural Comput. Appl.*, vol. 33, pp. 12499–12514, 2021.
- [4] W. Wang et al., "Malware traffic classification using convolutional neural network for representation learning," in *Proc. IEEE ICOIN*, 2021, pp. 712–717.
- [5] Y. Liu et al., "Intrusion detection systems in the era of cloud computing: A survey," *J. Inf. Secur. Appl.*, vol. 61, p. 102897, 2021.
- [6] S. Morgan, "Cybercrime to cost the world \$10.5 trillion annually by 2025," *Cybercrime Magazine*, 2021.
- [7] B. Mahbooba et al., "Explainable artificial intelligence (XAI) to enhance trust management in intrusion detection systems," *Complexity*, vol. 2021, 2021.
- [8] M. A. Ferrag et al., "Deep learning for cyber security intrusion detection," *J. Inf. Secur. Appl.*, vol. 50, p. 102419, 2020.
- [9] P. Vuttipittayamongkol and E. Elyan, "Neighbourhood-based undersampling for handling imbalanced and overlapped data," *Inf. Sci.*, vol. 509, pp. 47–68, 2020.
- [10] A. Khraisat et al., "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, 2019.
- [11] M. M. Hasan et al., "Attack and anomaly detection in IoT sensors using machine learning," *Internet of Things*, vol. 7, p. 100059, 2019.
- [12] T. Zebin et al., "An interpretable machine learning model for intrusion detection in wireless sensor networks," in *Proc. IEEE GlobalSIP*, 2019.
- [13] I. Sharafaldin et al., "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, 2018, pp. 108–116.
- [14] G. Douzas et al., "Improving imbalanced learning through a heuristic oversampling method based on kmeans and SMOTE," *Inf. Sci.*, vol. 465, pp. 1–20, 2018.
- [15] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. NeurIPS*, 2017, pp. 4765–4774.
- [16] H. Bostani and M. Sheikhan, "Modification of supervised OPF-based intrusion detection systems using unsupervised learning," *Pattern Recognit.*, vol. 62, pp. 56–72, 2017.
- [17] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *Proc. ICMLA*, 2016, pp. 195–200.
- [18] L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 6, pp. 446–452, 2015.
- [19] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various ML techniques," *Int. J. Eng. Res. Technol.*, vol. 2, no. 12, pp. 1848–1853, 2013.
- [20] M. Tavallaei et al., "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE CISDA*, 2009, pp. 1–6.
- [21] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," in *Managing Cyber Threats*, Springer, 2005, pp. 19–78.
- [22] N. V. Chawla et al., "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.