INTELLIGENT SYSTEMS FOR ENHANCED ANOMALY DETECTION IN MODERN CYBERSECURITY

M RK CHAITANYA
Research Scholar, Department of Computer Science and Engineering (CSE), Sikkim Alpine University, Kamrang, Namchi, (Sikkim)
chaitanyasri2947@gmail.com

DR PRASADU PEDDI
Research Guide, Sikkim Alpine Univeristy, Kamrang, Namchi, (Sikkim)

DR. K SRINIVAS
Research Co-guide, Geethanjali College of Engineering and Technology

Abstract:

Machine learning (ML) models have become indispensable for enhancing the effectiveness of cybersecurity countermeasures, particularly in IDS deployments. However, recent studies have demonstrated that these models are highly susceptible to adversarial attacks. By introducing subtle perturbations into malicious network traffic features, attackers can successfully evade ML-based IDS mechanisms. Consequently, the development of robust defences against such adversarial manipulations has become an urgent priority. Achieving adversarial resilience in cybersecurity systems, however, involves multiple challenges. One of the most significant barriers is maintaining an appropriate balance between an ML model's robustness and its operational efficiency. Another challenge lies in designing defence techniques that generalise effectively across diverse adversarial attack strategies. Most existing defence mechanisms proposed in contemporary research are predominantly tailored for computer vision domains and are seldom validated using cybersecurity-specific datasets. Unlike images, audio, or video streams, network traffic data possess different temporal characteristics and structural properties.

The NSL-KDD dataset is employed to evaluate the proposed HyAD-F against two widely used gradient-based adversarial attack methods—Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). To assess model behaviour under both adversarial and non-adversarial conditions, three machine learning classifiers—Logistic Regression, Gradient Boosting Classifier, and Multi-Layer Perceptron—are utilised. The integrated defence mechanism yields a substantial improvement in adversarial accuracy for both PGD- and FGSM-generated perturbations, while maintaining minimal degradation in standard cyberattack detection performance. The findings underscore the necessity of conducting rigorous security evaluations on intrusion detection models prior to their deployment in operational environments.

1 INTRODUCTION:

Preventing unauthorized access to computer systems remains a central concern in the field of information security. To counter such unauthorized activities, effective detection and prevention mechanisms are essential. Users who exhibit suspicious or malicious behaviour are typically referred to as intruders. These individuals attempt to gain access to restricted areas of a computing environment. Intrusion detection processes aim to identify attempts to compromise a target system, determine whether these attempts were successful, and record the corresponding activity logs [1].

Even in cases where highly confidential communication is not involved, unauthorized individuals should not be permitted to read emails, misuse

computing resources to launch attacks on other systems, send deceptive messages, or gain access to personal information such as financial records or account details. Intruders—often referred to as attackers, crackers, or hackers—are generally not concerned with the identity of the system owner. Their primary objective is to obtain control over the compromised system and use it as a launchpad for attacks on other targets. Frequently, attackers focus on high-value systems, such as governmental or financial infrastructures, in order to conceal their true identity and location while facilitating further malicious activity[2].

Maintaining the confidentiality of tasks performed on a computer is equally important, whether users are managing documents or operating applications. Users must also ensure that stored information remains accurate and accessible. The potential for intentional misuse of a computer system by online intruders can lead to serious security breaches [3]. Additionally, even offline systems face risks such as hardware failures, theft, and power outages. While such incidents may be unpredictable, several preventive measures can mitigate the likelihood of both intentional and accidental threats. Before exploring methods to protect a computer system or home network, it is essential to understand the types of threats that commonly arise.[4]

Table 1 categorizes the common types of network-based attacks. Intrusions may occur when attackers gain access to a system through the Internet, a local network, the operating system of a compromised machine, or vulnerabilities in third-party applications. Such attackers may attempt to block legitimate users from accessing resources, exploit system privileges, or abuse security mechanisms for malicious gain [5,6,7, 8].

Table 1.1: Attack Kinds with Description [6]

Attacks Category	Description	TCP/IP Layer
DoS	Denial-of-service (fake address generate)	Application Layer
DoS	Denial-of-service (fake address generate)	Transport Layer
U2R	Unauthorized admittance to local super user (root) privileges	Application Layer
R2L	Unauthorized admittance from a remote machine	Application Layer
R2L	Unauthorized admittance from a remote machine	Transport Layer
Probe	Surveillance and other probing	Application Layer

Probe	Surveillance and other probing	Transport Layer

The design of IDSs generally follows two overarching approaches aimed at protecting networks from malicious activities [9]:

1.The proactive security-centric approach, which relies on extensive use of encryption mechanisms and authorization techniques to construct a highly secure network environment. However, in practical scenarios, achieving a completely secure system is infeasible because different users and applications introduce varying levels of vulnerability across diverse operational contexts[10].

2.The reactive IDS-based approach, which is employed when the proactive strategy proves insufficient or impractical. Rather than attempting to eliminate all vulnerabilities in advance, this approach focuses on detecting and responding to malicious activities as they occur.

In this approach, attacks are monitored in real time, and appropriate countermeasures must be executed immediately upon detection[11].

2 LITREATURE SURVEY

As internet usage continues to expand, the frequency, diversity, and sophistication of cyberattacks have increased exponentially. IDSs (IDSs) have thus become a critical component of cybersecurity infrastructures. Their primary function is to differentiate malicious activities from legitimate network traffic and assist systems in identifying, analyzing, and responding to attacks. This chapter provides a comprehensive review of existing research in intrusion detection, including dominant detection techniques, clustering methods, classification approaches, and feature-selection strategies[12].

A variety of studies emphasize the need for hybrid methodologies that integrate clustering with classification to improve IDS performance, particularly when addressing challenges associated with imbalanced datasets and high-dimensional feature spaces. The chapter concludes with an analysis of the strengths and limitations of various IDS approaches. Li's study introduced a wrapper-based feature-selection technique named Modified Random Mutation Hill Climbing (RMHC) using SVM as the evaluation classifier. Their modified approach improved feature-selection speed by nearly 50% without compromising performance; however, classification accuracy comparisons were not provided[13].

The authors enhanced an existing AUCBoost algorithm by incorporating class-imbalance considerations, resulting in AUCBoostFS. The method outperformed AdaBoostFS across all attack categories, though broader benchmarking on public datasets would further validate its generalizability. Li proposed a wrapper-based Gradually Feature Removal (GFR) method utilizing SVM to reduce KDD99 features from 41 to 19, and further down to 10 using hybrid and filter methods. Although GFR provided the highest accuracy, it required the longest training time. Even with a 0.05% accuracy reduction, GFR improved classification speed by 41%[14].

This work introduced a wrapper-based differential-evolution feature-selection technique paired with the Extreme Learning Machine (ELM). Using the NSL-

KDD dataset, the model obtained 80.15% accuracy with only nine features, outperforming ANN and SVM baselines. The authors applied ranking-based methods (Information Gain, Correlation, Relief, and Symmetrical Uncertainty) to a C4.5 classifier. Their results showed that C4.5 with Information Gain achieved the highest accuracy (99.68%) using 17 features[15].

Using a GA-based wrapper with Logistic Regression, the authors produced an optimal feature subset for KDD99 and UNSW-NB15. With only 18 features, they achieved 99.90% accuracy and exceptionally low FAR (0.105%). A four-branch ensemble model combining J48, C5.0, Naïve Bayes, and PART was evaluated on NSL-KDD. Applying KNN post-classification improved tie-breaking accuracy, demonstrating that ensemble approaches enhance detection precision. The authors developed an Emerging Neutrosophic Logic Classifier (ENLCRID) enhanced through an Improvised Genetic Algorithm (IGA). Working with seven features chosen via Best First Search, the model achieved 99.02% detection rate with 3.19% FAR[16].

A three-phase model utilizing K-Means clustering, Naïve Bayes ranking, Kruskal–Wallis testing, and C4.5 classification was introduced. Using only 13 selected features, the system demonstrated strong intrusion detection performance. A Random Effects Logistic Regression (RELR) model incorporating uncertainty factors was presented. With only five selected features, the approach achieved 98.74% accuracy. By applying SVM, MARS, and Linear Genetic Programming, the authors reduced KDD99 from 41 attributes to 6 and improved classification performance by 1%[17].

3 METHODOLOGY

Adversarial attacks are deliberate perturbations crafted to mislead machine learning (ML) models, causing them to generate incorrect predictions or classifications. Mitigating these threats requires the integration of security mechanisms at every stage of the ML lifecycle. A comprehensive security assessment must therefore evaluate ML models against both evasion and poisoning attacks. Secure machine learning—an emerging subfield of ML—focuses on embedding defensive strategies throughout the lifecycle to safeguard models and training data from adversarial manipulation.

As outlined in the preceding chapter, existing defense mechanisms typically target only specific categories of adversarial attacks or protect isolated phases of the ML pipeline. No single defensive technique currently offers comprehensive protection across the diverse spectrum of adversarial strategies. Consequently, there is a need for an attack-agnostic defense approach capable of securing the entire ML lifecycle.

In this research, a state-of-the-art secure ML–driven, attack-agnostic Hybrid Adversarial Defense (HyAD) framework is introduced for IDSs (IDS). The principal aim of HyAD is to overcome adversarial-example vulnerabilities within the cybersecurity domain and to enhance the robustness and resilience of IDS models. The HyAD architecture is grounded in the principles of the secure ML lifecycle, ensuring end-to-end protection. The

framework is evaluated under a gray-box threat model during both training and testing phases. Its effectiveness is measured using additional robustness metrics, including adversarial accuracy and evasion rate, to assess performance under evasion-based attacks.

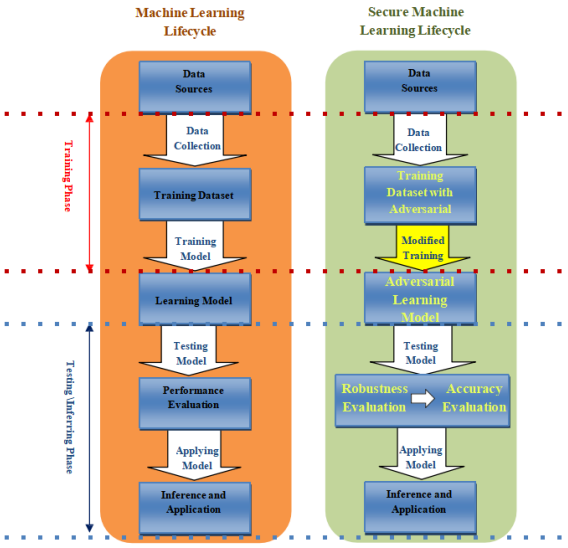

Figure 3.1: Machine learning lifecycle vs secure Machine Learning lifecycle

3.1 SECURE MACHINE LEARNING LIFECYCLE

The lifecycle of a machine learning (ML) model typically comprises data collection and preprocessing, feature selection, model training, performance evaluation, and deployment. Securing each of these stages is essential to reducing exposure to adversarial threats. Data poisoning attacks generally compromise the pre-training and training stages, while evasion attacks primarily target the inference or testing phase of the ML pipeline. Consequently, protecting ML-based systems throughout their entire lifecycle is imperative.

The proposed framework adopts the secure ML lifecycle paradigm, which integrates security considerations into every phase of the ML process. Because adversarial attacks can affect both the training and testing stages, the framework incorporates multiple defensive mechanisms. During data preprocessing and model training, pre-processing–based defense and data filtration defense techniques are employed to safeguard the training data and learning processes against poisoning-based manipulations. Furthermore, an adversarial training defense is embedded within the model training stage, wherein adversarial samples are deliberately injected into the training data to strengthen the model's resilience against evasion attempts during testing and deployment. The robustness of the adversarially trained model is subsequently assessed using additional evaluation metrics—adversarial accuracy and evasion rate—which quantify the model's effectiveness and resilience under adversarial threat conditions[18,19].

3.2 HYBRID ADVERSARIAL DEFENSE (HYAD) FRAMEWORK

The Hybrid Adversarial Defense (HyAD) framework is designed to strengthen ML models against adversarial attacks during both training and inference by integrating multiple complementary defense strategies. The framework comprises three layers of adversarial defenses.

The first layer is a pre-processing defense, for which a robust feature selection technique, termed **PerturbSense**, is introduced. PerturbSense fuses feature perturbation sensitivity with feature importance to identify features that are inherently robust against manipulation, thereby providing a proactive defense mechanism within the HyAD architecture.

The second layer functions as a data sanitization defense and is implemented

using an unsupervised anomaly detection method—**Isolation Forest**. This defensive layer aims to detect and filter adversarially manipulated inputs prior to training, mitigating the influence of data poisoning attacks and preserving data integrity.

The third layer introduces a proactive adversarial training defense to enhance robustness against evasion attacks. For this purpose, an adversarial example generation method, **CoS-GAN**, is proposed. CoS-GAN is a Generative Adversarial Network enhanced with a cosine similarity loss function that facilitates the creation of high-quality adversarial examples. These synthesized examples are used during training to fortify the model against evasion attempts during deployment.

The overall architecture of HyAD is depicted in Figure 3.1. Each layer contributes distinct and complementary defenses: PerturbSense reduces the attack surface by selecting adversarially resilient features; the Isolation Forest–based data sanitization layer filters adversarial samples to counter poisoning attempts; and the CoS-GAN–based adversarial training mechanism improves the model's ability to withstand evasion attacks. Collectively, these layers form a comprehensive, multi-stage defense strategy that protects ML models across their entire lifecycle, establishing HyAD as a robust and attack-agnostic defense framework.

NSL-KDD Dataset

The KDD Cup dataset received considerable criticism for its extensive number of redundant and duplicate instances, which skewed model evaluation results by enabling learning biases and inflating accuracy.

The NSL-KDD dataset addresses these issues through several enhancements:

- **Comprehensive attack taxonomy:** The dataset includes a broad set of cyber-attacks grouped into four primary categories—Denial of Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R).
- **Benchmark consistency:** Its widespread adoption allows consistent comparison across multiple research studies.

Table 3.1: Feature description of NSL-KDD dataset

Feature Number	Feature Name	Description	Feature Class
	Basic Features		
1.	Duration	Connection Duration	Continuous
2.	Protocol_type	Network Protocol Type	Discrete
3.	Service	Destination Network Application	Discrete
4.	Flag	Connection Status	Discrete
5.	Src_bytes	Bytes Sent	Continuous
6.	Dst_bytes	Bytes Received	Continuous
7.	Land	Indicator for Same Host/Port Connection (1 = Yes)	Discrete

8.	Wrong_fragment	Count of Malformed/Wrong Fragments	Continuous
9.	Urgent	Urgent Packet Count	Continuous
	Content Features		
10.	Hot – Count of "hot" indicators	Hot – Count of "hot" indicators	Continuous
11.	Num_failed_logins – Number of failed login attempts	Num_failed_logins – Number of failed login attempts	Continuous
12.	Logged_in – Indicator of successful login (1 = logged in)	Logged_in – Indicator of successful login (1 = logged in)	Discrete
13.	Num_compromised – Number of compromised conditions detected	Num_compromised – Number of compromised conditions detected	Continuous
14.	Root_shell – Indicator that a root shell was obtained (1 = yes)	Root_shell – Indicator that a root shell was obtained (1 = yes)	Discrete
15.	Su_attempted – Indicator that the su_root command was attempted (1 = yes)	Su_attempted – Indicator that the su_root command was attempted (1 = yes)	Discrete
16.	Num_root – Number of root-level accesses	Num_root – Number of root-level accesses	Continuous
17.	Num_file_creations – Number of file-creation operations	Num_file_creations – Number of file-creation operations	Continuous
18.	Num_shells – Number of shell prompts invoked	Num_shells – Number of shell prompts invoked	Continuous
19.	Num_access_files – Number of operations on access-control files	Num_access_files – Number of operations on access-control files	Continuous
20.	Num_outbound_cmds – Number of	Num_outbound_cmds – Number of	Continuous

	outbound commands in an FTP session	outbound commands in an FTP session	
21.	Is_host_login – Indicator of "host" login (1 = yes)	Is_host_login – Indicator of "host" login (1 = yes)	Discrete
22.	Is_guest_login – Indicator of "guest" login (1 = yes	Is_guest_login – Indicator of "guest" login (1 = yes	Discrete
	Traffic Features		
23.	Count	Connection counts to the same host	Continuous

Evaluation Metrics

Rigorous evaluation metrics are essential for assessing IDS performance. In binary classification for intrusion detection, "positive" denotes attack traffic and "negative" denotes benign traffic.

Metrics include[20]:

Accuracy

Measures the proportion of correctly classified benign and attack samples.

Recall (Detection Rate)

Proportion of actual attack samples correctly classified:

$$\text{Recall}=\frac{TP}{TP+FN}$$

A high recall is essential to minimize undetected intrusions.

Precision

Measures the reliability of attack predictions:

$$\text{Precision}=\frac{TP}{TP+FP}$$

F1-Score

Harmonic mean of precision and recall:

$$F1=2\cdot\frac{PR\cdot RR}{PR+RR} \qquad (3.15)$$

4 EXPEREIMENT & RESULTS:

The ranking based solely on perturbation sensitivity (Rank1) is generated by sorting features from the lowest to the highest f-PSI values. The *urgent* feature exhibits the lowest susceptibility to adversarial perturbation and thus receives the highest robustness rank, whereas *srv_count* shows the highest sensitivity and is assigned the lowest f-PSI rank.

Rank2 is computed by ordering features according to their importance scores from highest to lowest. Among all features, *dst_host_serror_rate* demonstrates the greatest contribution to model decisions and therefore attains the highest feature-importance rank, while *dst_host_srv_count* is ranked lowest due to minimal contribution.

The final aggregated ranking is derived by combining Rank1 and Rank2 using the weighted ranking mechanism defined in the PerturbSense methodology. Feature-wise comparisons are visualized in Figures 4.1 and 4.2, and a combined comparative analysis is presented in Figure 4.3.

Table 4.1: Continuous Feature Importance and f-PSI Metrics in NSL-KDD Dataset

eature Number	Feature Name	f-PSI	Importance	Rank 1	Rank 2	Rank
1	Duration	0.9119	0.6188	27	7	16
5	src_bytes	0.8950	0.5378	15	14	13
	dst_bytes	0.9072	0.5388	22	13	17
8	wrong_fragment	0.8958	0.7518	16	3	6
9	Urgent	0.0000	0.5378	1	15	5
10	Hot	0.9114	0.4828	25	25	28
11	num_failed_logins	0.9231	0.5193	29	22	29
13	num_compromised	0.7260	0.5343	7	19	12
16	num_root	0.9403	0.5354	30	18	25
17	num_file_creations	0.9873	0.5205	31	21	30
18	num_shells	0.4543	0.5297	3	20	9
19	num_access_files	0.9034	0.5374	19	16	18
20	num_outbound_cmds	0.9119	0.5395	26	12	22
23	Count	0.9120	0.5938	28	9	20
24	srv_count	1.0000	0.6759	32	5	19
25	serror_rate	0.9101	0.2100	23	30	31
26	srv_serror_rate	0.9004	0.0106	18	31	27
27	rerror_rate	0.8570	0.6887	11	4	3
28	srv_rerror_rate	0.8276	0.6205	10	6	4
29	same_srv_rate	0.8946	0.5774	14	10	10
30	diff_srv_rate	0.7970	0.4292	9	28	21

Feature 38 (*dst_host_serror_rate*) does not exhibit strong robustness against adversarial perturbations; however, it provides substantial contribution to the model's decision-making process. Conversely, Feature 33 (*dst_host_srv_count*) demonstrates high sensitivity to even minimal perturbations and offers minimal influence in the classification process.

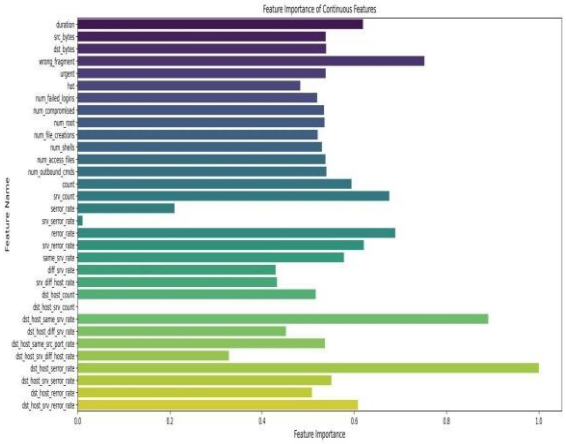

Figure 4.1: Importance of Various Continuous Features in NSL-KDD

45.7% adversarial accuracy under PGD when trained solely on continuous features. The clean accuracy of 96.6% was adopted as the selection threshold for identifying robust and influential features from the full feature set F. The model satisfied this threshold using **14 continuous features**, resulting in the following selected feature subset:

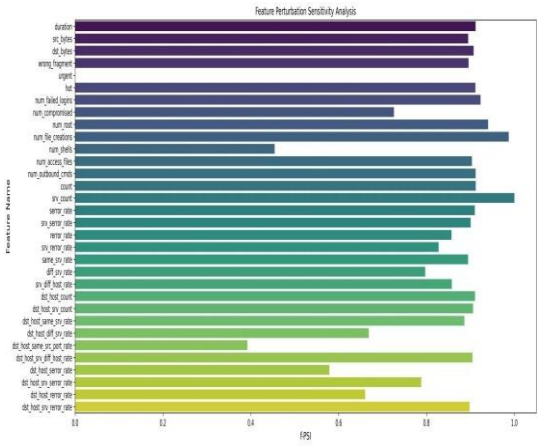

Figure 4.2: f-PSI score of various continuous features of NSL-KDD

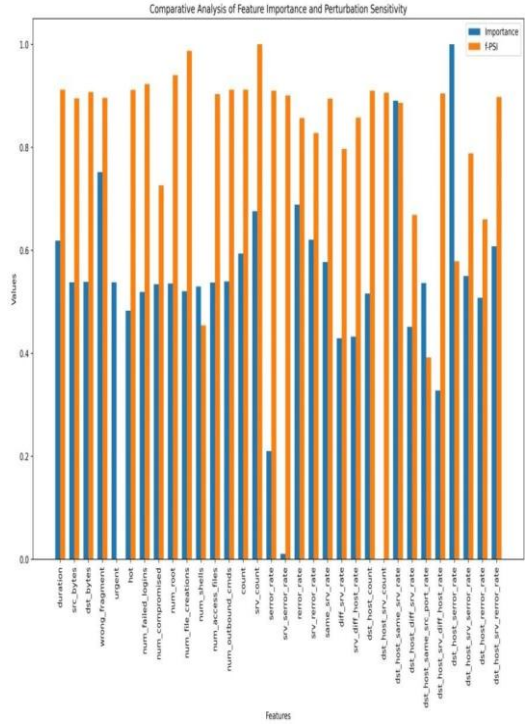

Figure 4.3: Feature Importance and f-PSI: A Comparative Study

96.8% accuracy using the features, along with substantially improved adversarial robustness—**72.3% adversarial accuracy** under FGSM and **62.5% adversarial accuracy** under PGD.

Data Sanitizer

A significantly degrade the integrity and reliability of ML models. A robust IDS must therefore remain effective even when adversaries attempt to influence its training distribution.

To counter these threats, the **data sanitizer** in the HyAD-F applies anomaly detection to identify and eliminate adversarially manipulated samples before model training. In this work, **Isolation Forest**, an unsupervised anomaly detection algorithm, is employed to implement this second-layer defensive mechanism.

The sanitization process is performed in two stages:

1. **Training Phase**
   A clean dataset—assumed to be free of poisoned instances—is used to train the Isolation Forest model. This step enables the algorithm to learn the normal patterns and distributions of benign and attack traffic.
2. **Detection Phase**
   The trained Isolation Forest is then applied to the training dataset potentially contaminated with poisoned samples. Samples with high anomaly scores are flagged as adversarial and removed, producing a sanitized dataset suitable for secure and robust ML model training.

Table 4.2: Performance Assessment of ML models over clean datasets

Class	Models	Acc(%)	P (%)	R(%)	F1(%)
	LR	98.8	97.2	96.8	96.2
Benign	Gradient Boosting Classifi	99.4	98.0	98.7	97.9

	er				
	MLP	98.8	98.9	99.0	99.0
	Logistic Regression	95.5	95.8	96.1	95.0
Attack	GBClassifier	97.2	98.6	97.2	98.3
	Multi-Layer Perceptron	98.9	98.9	98.6	97.7

Table 4.3: Evaluating ML Models over a 30% Poisoned Dataset

Class	Models	Acc(%)	P(%)	R(%)	F1(%)
	LR	94.8	66.4	95.5	78.1
Benign	Gradient Boosting Classifier	98.7	56.2	98.7	71.8
	MLP	80.6	85.3	81.3	83.2
	Logistic Regression	40.9	90.2	40.7	56.9
Attack	GBClassifier	10.6	88.1	10.9	18.8
	Multi-Layer Perceptron	95.7	78.8	84.7	81.6

Table 4.3: ML Model Performance Assessment Using 30% Poisoned Data

Class	Models	Acc(%)	P(%)	R(%)	F1(%)
	LR	93.2	92.1	92.0	95.9
Benign	GradientBoosting Classifier	98.8	98.3	98.6	98.4
	MLP	97.8	97.3	97.8	97.5
	Logistic Regression	96.5	94.8	97.1	95.7
Attack	GBClassifier	98.9	98.4	98.8	98.6
	Multi-Layer Perceptron	96.8	98.1	97.2	97.5

The performance evaluation of the machine learning classifiers after applying data sanitization using the proposed Isolation Forest–based anomaly detection method is summarized in Table 4.3. The assessment is conducted using standard performance metrics—accuracy, precision, recall, and F1-score—computed on the sanitized training dataset. These results reflect the effectiveness of the data sanitization process in restoring model integrity and improving classification reliability in the presence of poisoning attempts.

5 CONCLUSION:

The Data Sanitizer, implemented using the Isolation Forest algorithm, demonstrated strong effectiveness with detection rates of 93% for FGSM-based adversarial samples and 91% for PGD-based samples. These results highlight the combined efficacy of PerturbSense and the Data Sanitizer in reinforcing the security of the machine learning lifecycle and reducing the impact of adversarial poisoning on model training.

6 REFERENCES

[1] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, "A Survey on Security Threats and Defensive Techniques of Machine Learning: A Data Driven View," IEEE Access, vol. 6, pp.12103–12117, 2018, doi: 10.1109/ACCESS.2018.2805680.

[2] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," Mach Learn, vol. 81, no. 2, pp. 121–148, Nov. 2010, doi: 10.1007/s10994-010-5188-5.

[3] V. Kyatham, D. Mishra, and P. AP, "Variational Inference with Latent Space Quantization for Adversarial Resilience," in 2020 25th International Conference on Pattern

Recognition (ICPR), Jan. 2021, pp. 9593–9600. doi: 10.1109/ICPR48806.2021.9412896.

[4] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, "Hybrid Intelligent Intrusion Detection Scheme," in Advances in Intelligent and Soft Computing, 2011, pp. 293–303. doi: 10.1007/978-3-642-20505-7_26.

[5] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A Novel Ensemble of Hybrid Intrusion Detection System for Detecting Internet of Things Attacks," Electronics, vol. 8, no. 11, Art. no. 11, Nov. 2019, doi: 10.3390/electronics8111210.

[6] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in 2018 20th International Conference on Advanced Communication Technology (ICACT), Chuncheon-si Gangwon-do, Korea (South): IEEE, Feb. 2018, pp. 178–183. doi:10.23919/ICACT.2018.8323688.

[7] Prasadu Peddi (2015) "EXPLORING THE IMPACT OF DATA MINING AND MACHINE LEARNING ON STUDENT PERFORMANCE", International Journal of Emerging Technologies and Innovative Research, ISSN:2349-5162, Vol.1, Issue 6, page no. pp314-318, November-2014, Available at : http://www.jetir.org/papers/JETIR1701B47.pdf

[8] B. Biggio, B. Nelson, and P. Laskov, "Poisoning Attacks against Support Vector Machines." arXiv, Mar. 25, 2013. doi: 10.48550/arXiv.1206.6389.

[9] B. Biggio et al., "Evasion Attacks against Machine Learning at Test Time," in Lecture Notes in Computer Science, 2013, pp. 387–402. doi: 10.1007/978-3-642-40994-3_25.

[10] B. Biggio, G. Fumera, and F. Roli, "Security Evaluation of Pattern Classifiers under Attack," IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 4, pp. 984–996, Apr.2014, doi: 10.1109/TKDE.2013.57.

[11] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial Examples for Malware Detection," in Lecture Notes in Computer Science, 2017, pp. 62–79. doi:10.1007/978-3-319-66399-9_4.

[12] W. Hu and Y. Tan, "Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN." arXiv, Feb. 20, 2017.Available: http://arxiv.org/abs/1702.05983

[13] Prasadu peddi, "The Adoption of a Big Data and Extensive Multi-Labled Gradient Boosting System for Student Activity Analysis" International Journal of All Research Education and Scientific Methods (IJARESM) ISSN: 2455-6211, Volume 3, Issue 7, July- 2015, Impact Factor 2.287

[14] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks," in 2016 IEEE Symposium on Security and Privacy (SP), San Jose, CA: IEEE, May 2016, pp. 582–597. doi: 10.1109/SP.2016.41.

[15] R. Laishram and V. V. Phoha, "Curie: A method for protecting SVM Classifier from Poisoning Attack." arXiv, Jun. 06, 2016. Available: http://arxiv.org/abs/1606.01584

[16] T. M. Mitchell, "Machine Learning". in McGraw-Hill Series in Computer Science. New York: McGraw-Hill, 1997.

[17] A. A. Alurkar, S. B. Ranade, S. V. Joshi, S. S. Ranade, G. R. Shinde, P. A. Sonewar, and P. N. Mahalle, "A Comparative Analysis and Discussion of Email Spam Classification Methods Using Machine Learning Techniques," in Applied Machine Learning for Smart Data Analysis, CRC Press, 2019, pp. 185–206. doi: 10.1201/9780429440953-10.

[18] E. G. Dada, J. S. Bassi, H. Chiroma, S. M. Abdulhamid, A. O. Adetunmbi, and O. E. Ajibuwa, "Machine learning for email spam filtering: review, approaches and open research problems," Heliyon, vol. 5, no. 6, p. e01802, Jun. 2019, doi: 10.1016/j.heliyon.2019.e01802.

[19] A. K. Jain, D. Goel, S. Agarwal, Y. Singh, and G. Bajaj, "Predicting Spam Messages Using Back Propagation Neural Network," Wireless Personal Communications, vol. 110, no. 1, pp.403–422, Jan. 2020, doi: 10.1007/s11277-019-06734-y.

[20] E. Ileberi, Y. Sun, and Z. Wang, "A machine learning based credit card fraud detection using the GA algorithm for feature selection," Journal of Big Data, vol. 9, no. 1, Feb. 2022, doi:10.1186/s40537-022-00573-8.