Machine Learning-based Spyware Detection Systems: An Undersampling Performance Analysis

Diogo Santos de Oliveira M. de Souza
São Paulo State Univ. of Tech. (FATEC)
Ourinhos, Brazil
diogo.souza@fatecourinhos.edu.br

Maria Clara Souza Ramos
São Paulo State Univ. of Tech. (FATEC)
Ourinhos, Brazil
maria.ramos@fatecourinhos.edu.br

Wilson Cabral de Oliveira Junior
São Paulo State Univ. of Tech. (FATEC)
Ourinhos, Brazil
wilson.junior@fatecourinhos.edu.br

Thiago José Lucas
São Paulo State Univ. of Tech. (FATEC)
Ourinhos, Brazil
thiago@fatecourinhos.edu.br

Tiago Martins Ferreira
São Paulo State University (UNESP)
Bauru, Brazil
tiago.ferreira@unesp.br

Kelton Augusto Pontara da Costa
São Paulo State University (UNESP)
Bauru, Brazil
kelton.costa@unesp.br

***Abstract*—This study examined Machine Learning (ML) techniques with and without the application of Undersampling to improve Spyware identification in Intrusion Detection Systems (IDS). The problem of data imbalance and how Undersampling can help select more relevant and diverse data subsets was discussed. The research adopts a methodical approach, which includes a systematic literature review, selection of relevant data, appropriate preprocessing, and evaluation of the performance of ML classifiers. The findings suggest that the use of Undersampling techniques can significantly influence the effectiveness of IDS, with certain classifiers showing notable improvements after being trained with these methods.**

***Keywords*—Spyware; Intrusion Detecion; Machine Learning.**

I. Introduction

Spyware is a malicious software that collects user data without their consent, potentially compromising their privacy and security, and can also be used to prepare a ransomware attack. This type of attack poses a challenge for Intrusion Detection Systems (IDS), as they can hide among legitimate files or alter their characteristics to avoid identification.

One approach to detecting spyware is through machine learning (ML) techniques, which allow network traffic to be classified as benign or non-benign based on patterns and anomalies. However, using ML for spyware detection requires a reliable and representative database, which is not always available or easily accessible.

Furthermore, many databases are imbalanced, meaning they contain a much larger number of examples of one class than another, which can affect classifier performance. One way to address the imbalance problem is to use undersampling techniques, which consist of reducing the size of the majority class by selecting a more relevant and diverse subset.

The objective of this work is to evaluate the effects of different undersampling techniques on the performance of ML-based intrusion detection systems for spyware detection. The steps taken to achieve the general objective are described below:

- Conduct a systematic literature review to understand related work.
- Synthesize ideas from related work related to our article.
- Select a relevant data set for model training.
- Perform preprocessing to ensure it is suitable for training.
- Verify the model to create the intrusion detection system.
- Subject the model to tests to extract results such as accuracy and recall to assess its effectiveness.
- Conclusion, observations, and suggestions for future research.

This article is divided into five sections: the introduction (I), which contextualizes the topic and presents the problem under study; the systematic literature review (II), presenting relevant works and research methods through a flowchart; the methodology (III), describing the procedures adopted; the results (IV), showing the outcomes of each technique used; and, finally, the conclusion (V), discussing the results and suggesting approaches for future research.

II. Theoretical Background

A. Related Work

In [1] the authors researched a new partial undersampling technique to deal with imbalanced data in cyberthreat detection, using an incremental clustering algorithm and cluster-based classification rules. The method combines unsupervised and supervised learning techniques to improve the accuracy of the minority class without significantly sacrificing the majority

class. Three imbalanced datasets (us_crime, ecoli, and libras move) and four conventional classifiers (KNN, Random Forest, SVM, and Adaboost) are used to evaluate the performance of the proposed method, showing that it achieves relatively higher accuracy in the minority classes.

In [2], a spyware injection system on Android devices using a fake gaming application was presented. Spyware aims to collect personal and confidential information from victims, such as contacts, messages, call logs, web history, location, accounts, and installed applications. Spyware can also send remote commands to the victim's device, such as vibrating the phone or sending fake notifications. The system consists of three parts: an Android application, a spy control panel, and a web service. The paper presents the design, implementation, and test results of the system, as well as some ways to avoid spyware applications and ethical discussions about its use.

As show in [3], a technique to detect spyware using multi-objective particle swarm optimization and Jordan canonical form was proposed. The technique reduces data complexity and improves the accuracy of the artificial neural network classifier. The paper presents experimental results showing that the technique achieves 99% accuracy in identifying spyware and discusses the advantages and disadvantages of different feature selection and classifier approaches.

In [4] the authors present an overview of the use of five ML algorithms (LDA, NNET, CART, SVM, and random forest) to accurately predict the outcomes of legal cases related to online privacy violations in the US and help legal practitioners better understand the patterns and trends of online privacy cases. Machine learning algorithms are applied to classify cases into two categories: guilty or not guilty. The study analyzes the factors that influence judicial decisions, such as the type of violation, the harm caused, the defendant's intent, the plaintiff's consent, the jurisdiction, and the applicable law. Of the five models, the best was CART, with a prediction rate of 83.16%, and the worst was the neural network (NNET), with 81.79%.

The paper from [5] proposes a new intrusion detection method based on machine learning and undersampling. The method uses a density-based clustering subsampling algorithm called Density-peak, which selects a set of representative samples from the majority class and combines them with the minority class to form a balanced training set. The Density-peak algorithm is capable of detecting non-spherical clusters and automatically determining the optimal number of clusters. The method also uses a density- and distance-based selection measure, called P, which identifies points with high density and distance from the minority class. The method is evaluated using the CICIDS2017 dataset and different classifiers based on decision trees and random forests. The results show that the proposed method outperforms existing methods in terms of balanced accuracy and G-mean, especially at high imbalance rates.

In [6], it was presented a research on network intrusion detection using machine learning methods. Different algorithms, such as decision trees, gradient boosting, and deep neural networks, were compared in terms of accuracy, sensitivity, and specificity. The impact of class imbalance and cross-validation on model evaluation was also analyzed. The authors concluded that LightGBM is the most effective and robust method for network intrusion detection.

The work from [7] presents a new network intrusion detection method using machine learning. The method combines a focal loss function, which gives more weight to minority class examples, with an information gain-based feature selection algorithm, which reduces data dimensionality. The method is evaluated using the Bot-IoT dataset, which contains real and simulated cyberattacks. The results show that the proposed method outperforms other existing methods in terms of accuracy, sensitivity, and specificity.

The research by [8] presents a novel dataset and a spyware detection model for Android devices, based on network traffic analysis. Data were collected from five commercial spyware systems using a unified activity list and a packet capture tool. They divided the data into three classes: normal traffic, spyware installation traffic, and spyware operation traffic. They used the random forest algorithm to classify the data into binary and multiclass traffic, achieving average accuracies of 79% and 77%, respectively. They also performed validation on the dataset using a confusion matrix and compared their model with other related works. They concluded that their dataset is useful for spyware detection research and that their model is effective and robust.

As show in [9], a research on privacy and machine learning in the Internet of Things (IoT), addressing the challenges, solutions, and practical applications in this context. The text presents the three-layer IoT architecture, the importance of GDPR as a regulation, the vulnerabilities, threats, and attacks on privacy in IoT, the privacy-preserving mechanisms in IoT based on perturbation, restriction, encryption, obfuscation, and decentralized and multi-level machine learning, the experiments and evaluation using the MalMemAnalysis dataset to detect obfuscated malware in IoT with machine learning algorithms, the comparative analysis with the state-of-the-art that shows the superiority of the proposed methodology with an accuracy of 99.50%, and the conclusions and future work that highlight the contributions, limitations, and challenges of the work.

In [10], the authors address the importance of security in mobile devices, with an emphasis on detecting spyware and attacks perpetrated through malicious messages and emails. Using machine learning techniques, a comparative analysis

was performed between three classifiers: Convolutional Neural Network (CNN), Decision Tree (DT), and K-Nearest Neighbors (KNN), using the Canadian Institute for Cybersecurity (CICIDS2017) dataset. The conclusion is that the Convolutional Neural Network stands out as the most efficient method for early detection of spyware attacks on mobile devices.

B. Taxonomy of related works

Table I compiles the main variables obtained from related works:

III. Methodology

This study was conducted entirely with free software: all experiments were run on a Debian 12 Linux server using the scikit-learn library in Python to assess the performance of Intrusion Detection Systems (IDS) in identifying spyware. The primary objective is to analyze the impact of undersampling techniques on the effectiveness of these systems.

A. Dataset

The dataset CIC-MalMem-2022[1] from Canadian Institute for Cybersecurity[2] was used in this research simulates real-world scenarios by incorporating a mix of benign and malicious memory dumps, aiming to evaluate the detection of obfuscated malware.

It comprises a balanced distribution, with 50% benign and 50% malicious memory dumps, totaling 58,596 instances. The dataset includes samples from various malware families, such as spyware, ransomware, and Trojan horses.

To enhance the realism of the dataset, debug mode was enabled during the memory dump process. This ensured that the dumping procedure itself was excluded from the captured data, more accurately simulating the behavior and artifacts typically observed in real user environments during malware execution.

B. Undersampling Techniques

Two undersampling strategies were applied: NearMiss and Random Undersampling. Each method was executed under four different settings, resulting in four distinct datasets per technique. The undersampling ratios used were 0.35, 0.40, 0.45, and “auto”.

C. NearMiss

The NearMiss technique was applied to the original dataset using the following configurations:

1) 0.35: The dataset was reduced to 35% of its original size.
2) 0.40: The dataset was reduced to 40% of its original size.
3) 0.45: The dataset was reduced to 45% of its original size.
4) auto: The algorithm automatically determined the optimal undersampling ratio.

Varied undersampling rate proportions were conducted in the range of 5% to 95%, varying by 5%. It was decided to keep only the range of results 35-45% in the work because empirically this was where the best performance was observed.

D. Random Undersampling

The Random Undersampling technique was also applied to the original dataset using the same four configurations adopted for the NearMiss approach.

In summary, the following steps were performed:

1) Data Preparation: The dataset provided by the instructor was prepared for the ML process. This dataset includes various instances of spyware and legitimate network traffic.
2) Application of Undersampling Techniques: The NearMiss and RandomUndersampling techniques were applied to the dataset. Each technique was applied four times, with parameters set to 0.35, 0.4, 0.45, and ‘auto’, resulting in four distinct datasets for each technique.
3) Classifier Testing: All available two-class classifiers on the Microsoft Azure Machine Learning Studio platform were tested with each of the generated datasets.
4) Comparison with the Original Dataset: The classifiers were also tested using the original dataset, without applying any undersampling techniques, for comparison purposes.

E. Overview of the Result Analysis

After applying the undersampling techniques and testing the classifiers, the results will be analyzed in detail in the following section of this paper. This analysis aims to determine the impact of the undersampling techniques on the performance of Intrusion Detection Systems. The comparison between the results obtained with the original dataset and those obtained with the undersampled datasets will enable a comprehensive evaluation of the impact of these techniques.

F. Confusion Matrix

The confusion matrix is a table designed to facilitate visualization of a classification algorithm’s performance. It presents in more detail the classification models’ results and compares the model predictions with the real data values. Figure 1 ilustrates a confusion matrix.

- TP - True Positive: corresponds to the number of attack instances classified as an attack.
- FP - False Positive: is the number of benign instances incorrectly classified as attack.

[1]CIC-MalMem-2022 - https://www.unb.ca/cic/datasets/malmem-2022.html
[2]Canadian Institute for Cybersecurity - https://www.unb.ca/cic/about

Table I
MAIN VARIABLES FROM RELATED WORKS. SOURCE: PREPARED BY THE AUTHOR.

Work	Classifier	Dataset	Undersampling Technique
[1]	KNN, Random Forest, SVM, and Adaboost	us_crime, ecoli, and libras move	Supervised and Unsupervised Partial Undersampling
[3]	Decision Tree, KNN, ANN, and SVM	Unspecified Dataset	Interdisciplinary Approach for Spyware Detection
[11]	Not used. Uses honeytoken	Blackshades, JRat and QuasarRAT	Malware Analysis technique
[4]	LDA, NNET, CART, SVM, Random Forest	CICIDS2017	Density-Peak Undersampling
[6]	Random Forest, LightGBM and XGBoost	CSE-CIC-IDS2018	Random Undersampling
[7]	Light GBM, CatBoost, XGBoost, Random Forest, Logistic Regression, Naive Bayes, Decision Tree, and Multilayer Perceptron	Bot-IoT	Random Undersampling and Ensemble Feature Selection Techniques
[5]	Decision Tree, Random Forest	CICIDS2017	Density-Peak Clustering, Random Under-Sampling and RUSBoost
[8]	Decision Forest, Gradient Boosting and DNN	CSE-CIC-IDS2018	Random Undersampling
[9]	Decision Tree, SVM and Logistic Regression	MalMemAnalysis	Not Used
[10]	Decision Tree, KNN, CNN	CICIDS2017	Near Miss Undersampling

Table II
PRESENTATION OF THE TECHNIQUES EMPLOYED. SOURCE: PREPARED BY THE AUTHOR.

Undersampling technique	Parameter	Description
NearMiss	0.35	The dataset was reduced to 35% of its original size.
NearMiss	0.4	The dataset was reduced to 40% of its original size.
NearMiss	0.45	The dataset was reduced to 45% of its original size.
NearMiss	auto	The algorithm automatically determined the optimal undersampling ratio.
RandomUndersampling	0.35	The dataset was reduced to 35% of its original size.
RandomUndersampling	0.4	The dataset was reduced to 40% of its original size.
RandomUndersampling	0.45	The dataset was reduced to 45% of its original size.
RandomUndersampling	auto	The algorithm automatically determined the optimal undersampling ratio.

		Predicted	
		Positive	Negative
Real	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

Fig. 1. Confusion Matrix.

- TN - True Negative: corresponds to the number of benign instances correctly classified as benign.
- FN - False Negative: refers to the number of attack instances incorrectly classified as benign.

G. Evaluation Metrics

Accuracy (ACC): This metric determines the proportion of correct predictions made by the model relative to the total number of predictions. The formula for accuracy show in Eq. 1:

$$\text{ACC} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision (PR): This metric assesses the proportion of correct positive predictions. Precision is calculated using the Eq. 2:

$$\text{PR} = \frac{TP}{TP + FP} \tag{2}$$

Recall (REC): This metric calculates the proportion of true positives correctly identified by the model. Recall is given by Eq. 3:

$$\text{REC} = \frac{TP}{TP + FN} \tag{3}$$

IV. RESULTS

This section presents a detailed analysis of the effectiveness of undersampling techniques in Intrusion Detection Systems. Through tables and graphs, the performance of various classification models is compared. A qualitative discussion follows on the most effective classifiers, highlighting those with the highest accuracy and efficiency in identifying spyware. The results indicate that undersampling techniques such as NearMiss and RandomUndersampling can significantly improve classifier performance.

The following abbreviations for the classification models are considered in the results tables:

- **AP** – Averaged Perceptron
- **BPM** – Bayes Point Machine
- **BDT** – Boosted Decision Tree
- **DF** – Decision Forest
- **DJ** – Decision Jungle
- **LDS** – Locally Deep Support Vector Machine
- **LR** – Logistic Regression
- **NN** – Neural Network
- **SVM** – Support Vector Machine

A. Dataset Without Any Technique Applied

Table III
PRESENTATION OF RESULTS WITHOUT THE USE OF UNDERSAMPLING. SOURCE: PREPARED BY THE AUTHOR.

Model	TP	TN	FP	FN	ACC	PR	REC
AP	9967	29250	48	53	0.99743	0.99521	0.99471
BPM	9921	29082	216	99	0.99199	0.97869	0.99012
BDT	10020	29082	2	0	0.99995	0.99980	1.00000
DF	10020	29296	2	0	0.99995	0.99980	1.00000
DJ	10020	29295	3	0	0.99992	0.99970	1.00000
LDP	9965	29231	67	55	0.99690	0.99332	0.99451
LR	9984	29274	24	36	0.99847	0.99760	0.99641
NN	10006	29291	7	14	0.99947	0.99930	0.99860
SVM	9979	29273	25	41	0.99832	0.99750	0.99591

In Figures 2 and 3, the values are presented graphically for a clearer visualization of each classifier. It is observed that the classifiers Boosted Decision Tree, Decision Forest, and Decision Jungle stood out, achieving accuracy close to 99.98%. Using NearMiss auto, the results are comparable to the others, with a slight advantage over NearMiss 0.45.

In Figures 2 and 3, the classification results using the NearMiss undersampling technique are presented in graphical format, allowing for a clearer comparison among the classifiers. It is evident that the classifiers Boosted Decision Tree, Decision Forest, and Decision Jungle consistently outperformed the others, reaching accuracy values close to 99.98%, regardless of the undersampling ratio applied. This suggests a high degree of robustness in their learning capacity under data reduction. Notably, the use of the "NearMiss auto" configuration yielded results that were not only comparable to those obtained with predefined ratios but also slightly superior to the configuration with a 0.45 ratio. This indicates that the automatic configuration was able to effectively balance the class distribution while preserving relevant information for the learning process, resulting in a marginal but consistent improvement in accuracy.

In Figures 4 and 5, the performance of classifiers under the RandomUndersampling technique is depicted. Once again, the classifiers BDT, DF, and DJ emerge as the top performers, maintaining their high levels of accuracy even with varying degrees of data reduction. However, a more nuanced behavior can be observed among the classifiers Logistic Regression,

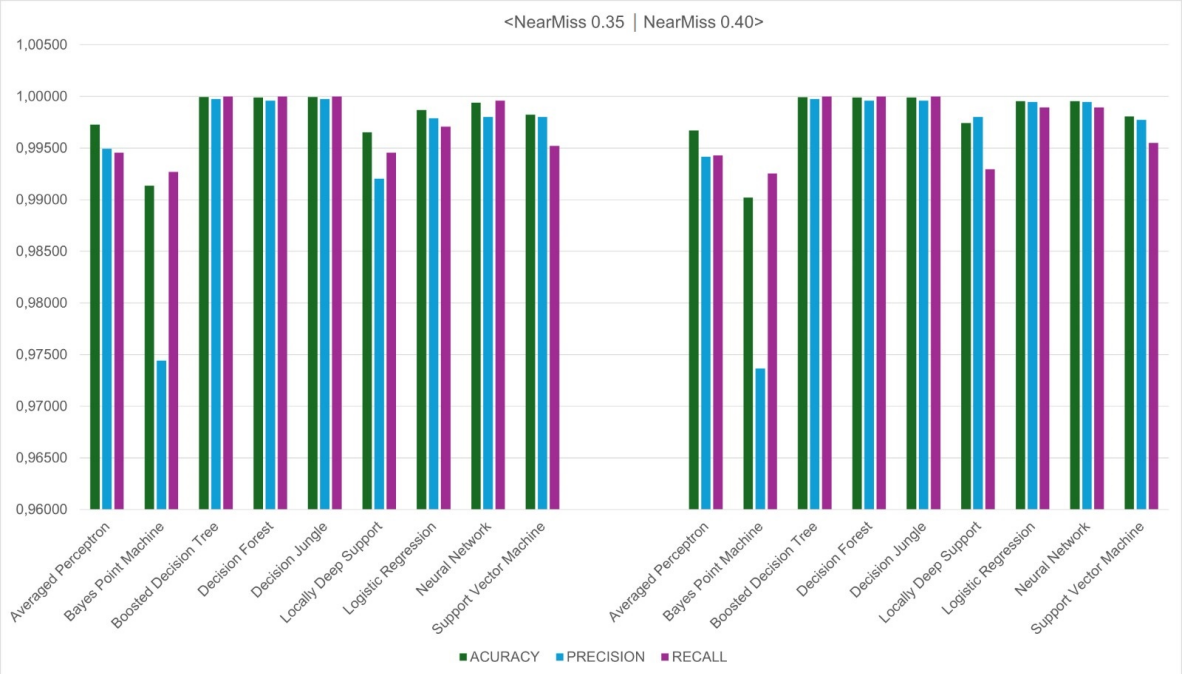

Fig. 2. Graphical Representation of NearMiss at 0.35 and 0.4. Source: Prepared by the author.

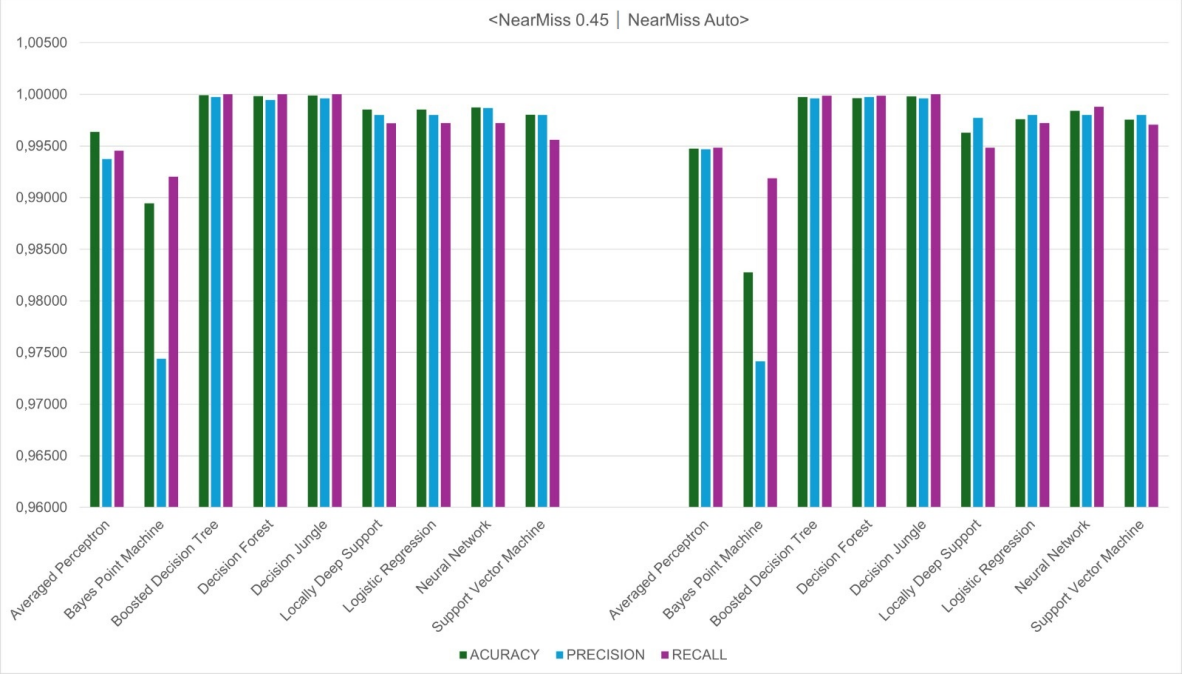

Fig. 3. Graphical Representation of NearMiss at 0.45 and Auto. Source: Prepared by the author.

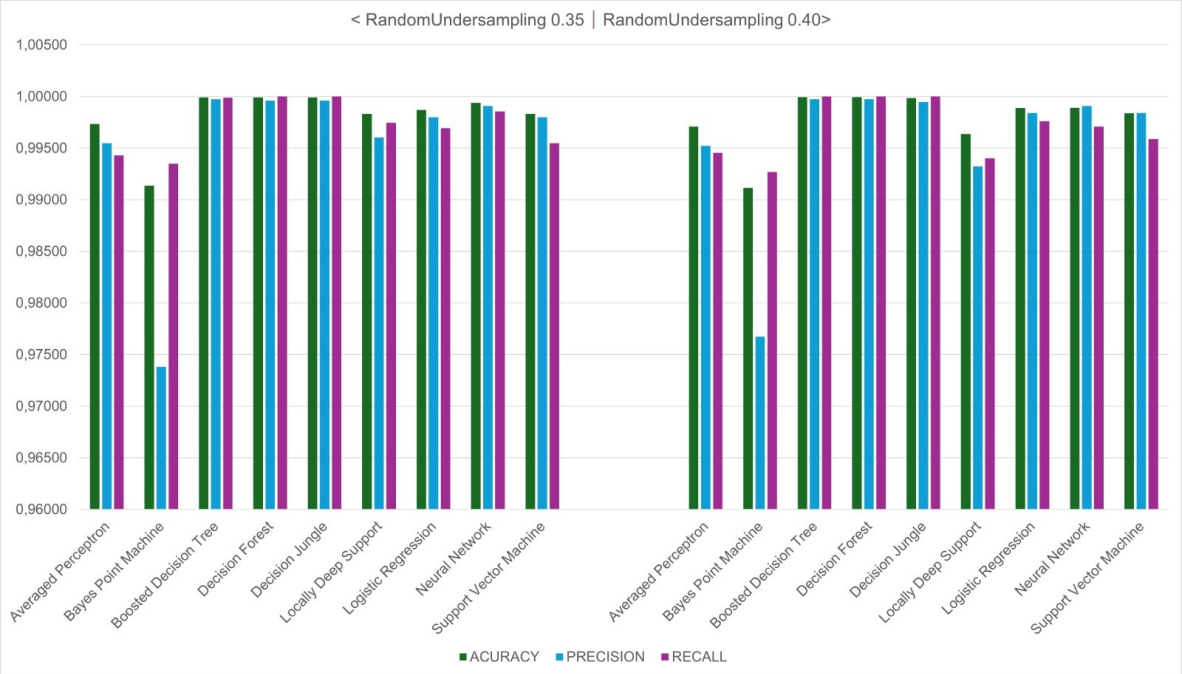

Fig. 4. Graphical Representation of RandomUndersampling at 0.35 and 0.4. Source: Prepared by the author.

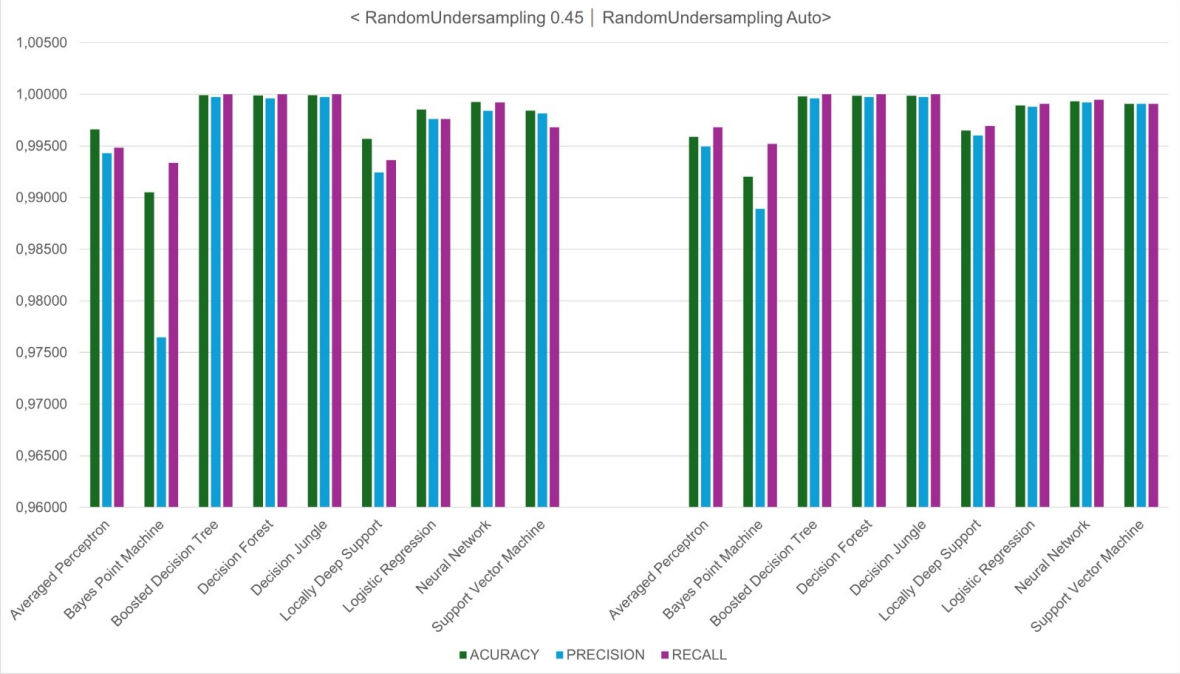

Fig. 5. Graphical Representation of RandomUndersampling at 0.45 and Auto. Source: Prepared by the author.

Neural Network, and Support Vector Machine. These models exhibited slightly lower accuracy rates compared to the top-performing tree-based classifiers but demonstrated a notable pattern of internal consistency, particularly when using the "RandomUndersampling auto" configuration. Unlike fixed-ratio settings (0.35, 0.4, or 0.45), which produced more volatile results among these classifiers, the automatic configuration led to results that were not only more stable but also more aligned across the three models. This consistency suggests that the automatic undersampling method may provide a more adaptive and balanced subset of the data, especially beneficial for models that are more sensitive to class imbalance or data distribution variations.

The results shown in Figures 2 to 5 demonstrate that tree-based classifiers—Boosted Decision Tree, Decision Forest, and Decision Jungle — consistently achieve top accuracy near 99.98% across both NearMiss and RandomUndersampling techniques. Automatic configurations ("NearMiss auto" and "RandomUndersampling auto") perform as well or better than fixed ratios by effectively balancing classes and preserving data quality. While Logistic Regression, Neural Network, and Support Vector Machine have slightly lower accuracy, they show greater stability with the automatic RandomUndersampling setting, indicating its advantage for sensitive models. These findings highlight the effectiveness of automated undersampling in improving classifier performance on imbalanced datasets.

V. Conclusion

As demonstrated by both the numerical tables and graphical analyses, the application of undersampling techniques leads to a marked improvement in model training for intrusion detection. For example, comparing the baseline results shown in Table III with those obtained using the NearMiss technique reveals that the automatic configuration of NearMiss delivers notably consistent and reliable performance. Similarly, the RandomUndersampling method with an automatic setting also exhibits stable and robust results, as evidenced by the graphical representation in Figure 5.

Across all applied techniques, the results remain consistently strong, with the Boosted Decision Tree, Decision Forest, and Decision Jungle classifiers emerging as the most effective, each achieving accuracy rates near 99.98%. The Neural Network classifier follows closely behind, attaining an accuracy of approximately 99.92%. These findings indicate a clear superiority of ensemble tree-based models in this context, while also highlighting the competitive performance of neural networks.

Such outcomes provide a solid foundation for directing future research towards further refinement of classification methods, with the goal of approaching an ideal accuracy of 100%. Implementing these advanced undersampling strategies

in training Intrusion Detection Systems has the potential to significantly enhance their detection capabilities, surpassing the performance of conventional undersampling approaches. However, it is important to recognize that the effectiveness of these methods may vary according to dataset characteristics and experimental settings. Therefore, extending the evaluation of these techniques to additional datasets is highly recommended, to better understand their impact on training variability and the operational performance of deployed IDS solutions.

ACKNOWLEDGMENTS

The authors are grateful to State Center for Technological Education "Paula Souza" (CEETEPS) for research conditions at the Defensive Cybersecurity and Artificial Intelligence Laboratory (Detect.AI), and to Research Support Foundation of São Paulo State (FAPESP), Brazil grants #2023/12830-0.

REFERENCES

[1] M. Moniruzzaman, A. Bagirov, and I. Gondal, "Partial undersampling of imbalanced data for cyber threats detection," in *Proceedings of the Australasian Computer Science Week Multiconference*, ser. ACSW '20. New York, NY, USA: Association for Computing Machinery, 2020. [Online]. Available: https://doi.org/10.1145/3373017.3373026

[2] H. M. Salih and M. S. Mohammed, "Spyware injection in android using fake application," in *2020 International Conference on Computer Science and Software Engineering (CSASE)*, 2020, pp. 100–105.

[3] V. Mahesh and S. Devi K.A., "Detection and prediction of spyware for user applications by interdisciplinary approach," in *2020 International Conference on Computational Intelligence for Smart Power System and Sustainable Energy (CISPSSE)*, 2020, pp. 1–6.

[4] M. Park and S. Chai, "Ai model for predicting legal judgments to improve accuracy and explainability of online privacy invasion cases," *Applied Sciences*, vol. 11, no. 23, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/23/11080

[5] M. Mohseni and J. Tanha, "A density-based undersampling approach to intrusion detection," in *2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA)*, 2021, pp. 1–7.

[6] R. Zuech, J. Hancock, and T. M. Khoshgoftaar, "Detecting web attacks in severely imbalanced network traffic data," in *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)*, 2021, pp. 267–273.

[7] J. L. Leevy, J. Hancock, T. M. Khoshgoftaar, and N. Seliya, "Iot reconnaissance attack classification with random undersampling and ensemble feature selection," in *2021 IEEE 7th International Conference on Collaboration and Internet Computing (CIC)*, 2021, pp. 41–49.

[8] M. K. Qabalin, M. Naser, and M. Alkasassbeh, "Android spyware detection using machine learning: A novel dataset," *Sensors*, vol. 22, no. 15, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/15/5765

[9] S. El-Gendy, M. S. Elsayed, A. Jurcut, and M. A. Azer, "Privacy preservation using machine learning in the internet of things," *Mathematics*, vol. 11, no. 16, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/16/3477

[10] A. Kumari and I. Sharma, "Towards securing mobile communication from spyware attacks with artificial intelligence techniques," in *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)*, 2023, pp. 1–5.

[11] H. Badih, B. Bond, and J. Rrushi, "On second-order detection of webcam spyware," in *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, 2020, pp. 424–431.