Bulletin of Electrical Engineering and Informatics
Vol. 13, No. 5, October 2024, pp. 3653~3662
ISSN: 2302-9285, DOI: 10.11591/eei.v13i5.7939

Enhancing spyware detection by utilizing decision trees with hyperparameter optimization

Mosleh M. Abualhaj[1], Ahmad Sami Al-Shamayleh[2], Alhamza Munther[3], Sumaya Nabil Alkhatib[1], Mohammad O. Hiari[1], Mohammed Anbar[4]
[1]Department of Networks and Cybersecurity, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan
[2]Department of Data Science and Artificial Intelligence, Faculty of Information Technology, Al-Ahliyya Amman University, Amman, Jordan
[3]Department of Information Technology, College of Computing and Information Sciences, University Technology and Applied Sciences, Sur, Sultanate of Oman
[4]National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia, Penang, Malaysia

Article Info

Article history:

Received Nov 27, 2023
Revised Feb 23, 2024
Accepted Mar 6, 2024

Keywords:

Cybersecuirty
Decision trees
Machine-learning
Malware
Obfuscated-MalMem2022
Spyware

ABSTRACT

In the realm of cybersecurity, spyware has emerged as a formidable adversary due to its persistent and stealthy nature. This study delves deeply into the multifaceted impact of spyware, meticulously examining its implications for individuals and organizations. This work introduces a systematic approach to spyware detection, leveraging decision trees (DT), a machine-learning classifier renowned for its analytical prowess. A pivotal aspect of this research involves the meticulous optimization of DT's hyperparameters, a critical operation for enhancing the precision of spyware threat identification. To evaluate the efficacy of the proposed methodology, the study employs the Obfuscated-MalMem2022 dataset, well-regarded for its comprehensive and detailed spyware-related data. The model is implemented using the Python programming language. Significantly, the findings of this study consistently demonstrate the superiority of the DT classifier over other methods. With an accuracy rate of 99.97%, the DT proves its exceptional effectiveness in detecting spyware, particularly in the face of more intricate threats. By advancing our understanding of spyware and providing a potent detection mechanism, this research equips cybersecurity professionals with a valuable tool to combat this persistent online menace.

Corresponding Author:

Mosleh M. Abualhaj
Department of Networks and Cybersecurity, Faculty of Information Technology
Al-Ahliyya Amman University
Amman, Jordan
Email: m.abualhaj@ammanu.edu.jo

1. INTRODUCTION

Cyberattacks are when people or groups intentionally do bad things to take advantage of weak spots in computer systems, networks, or digital devices. They can do this for many reasons, but they usually want to get in without permission, steal data, stop operations, or cause harm. There are many distinct varieties of cyberattacks, each of which has its own unique goals and strategies. Phishing, social engineering, and malware attacks are some of the more popular types of cyberattacks [1]-[3].

Malware programs are made to sneak into target systems to steal information, interfere with or harm operations, or grant illegal access. Malware can spread via various channels, including hacked software downloads, malicious websites, and contaminated email attachments. Malware can take many forms,

including Trojan horses, worms, ransomware, and spyware [4]-[6]. The purpose of spyware is to stealthily gather data from a victim's computer or device without the victim's knowledge or consent. This data includes keystrokes, surfing patterns, personal information, and more. Spyware is frequently used for nefarious activities like identity theft, data theft, and unapproved monitoring [6]-[8]. Spyware is hitting most organizations and is spreading like wildfire. Most internet users—nearly 80%—have spyware on their computers [9].

Reputable anti-spyware software, safe browsing practices, caution when downloading software from untrusted sources, and firewalls are all necessary to defend against spyware attacks. These techniques must, however, keep up with the widespread and quick propagation of spyware. For example, firewalls and antivirus programs miss 80% of keylogger detections [10]. Using sophisticated techniques, including heuristic analysis, is recommended to identify spyware. Modern cybersecurity relies heavily on heuristic analysis to identify and neutralize threats that sidestep traditional signature-based defenses. Heuristic analysis is a proactive and dynamic method of detecting and combating spyware that continuously changes to evade established security measures by concentrating on behavior and action patterns. Machine learning (ML) techniques are frequently used in heuristic analysis to increase accuracy and adjust to changing threats. These techniques can continuously improve their detecting capacities by learning from prior activity [11]-[13].

The fundamental building blocks of ML systems are ML algorithms, which let computers learn from data and make judgments or predictions without needing to be explicitly programmed. These algorithms can be divided into many types based on their purpose and use. They are made to find patterns, connections, and insights in data. Common types of ML algorithms are K-nearest neighbors (KNN), linear support vector classifier (SVC), Naive Bayes (NB), gradient boosting tree (GB tree), and decision tree (DT) [14]-[16]. This paper will use a customized version of DT algorithm to detect spyware.

2. RELATED WORKS

The work by Javaheri *et al*. [6] presents an innovative approach to the identification, monitoring, and counteraction of concealed and disguised forms of malicious software, which encompass spyware and ransomware, including keyloggers, screen recorders, and blockers. This research method is grounded in a dynamic behavioral analysis approach, involving in-depth and transparent hooking of kernel-level routines. The linear regression, JRIP, and J48 DT algorithms were utilized as classifiers to identify and categorize three distinct types of malwares. The authors delineate the foundational architecture of an anti-spyware program designed to detect and thwart spyware activities by tracing their paths, terminating running processes, removing executable files, and restricting network connections. The efficacy of the proposed method was assessed based on its accuracy in detecting real-world samples of spyware, employing receiver operating characteristic (ROC) curves for analysis. Additionally, the success rate of effectively countering active malware was considered. Our method exhibited a remarkable capability to identify spyware, achieving an accuracy rate of approximately 93% with an error rate of around 7%. Furthermore, the suggested method demonstrated a commendable success rate of approximately 82% in effectively eradicating spyware infections within an operating system.

Almashhadani *et al*. [17] address the problem of contemporary host-based detection techniques that demand host infection to detect abnormalities and identify ransomware. Using Locky as a case study, Almashhadani *et al*. [17] thoroughly examine crypto ransomware network traffic and provide an advanced ransomware detection technique. PCAP files were collected and examined, and a special testbed environment was constructed. Locky's PCAP files from the malware capture facility project (MCFP) dataset were also gathered and thoroughly examined. According to the investigation, Locky has several network activities from which possible behavioral traits may be extracted. With the use of transmission control protocol (TCP), hyper text transfer protocol (HTTP), domain name system (DNS), and netbios name service (NBNS) traffic, a total of eighteen features were retrieved. These characteristics are instructive and have the ability to distinguish traffic produced by a compromised host. Additionally, a multi-classifier network-based technique for detecting ransomware is suggested and prototyped. This technique operates at both the packet and flow levels. Extensive experimental investigations confirm the efficacy of the derived characteristics with high detection accuracy of 97.92% and 97.08%, respectively, for each level.

Kono *et al*. [18] have addressed the challenge of detecting and responding to the growing variety of unknown malware types. To tackle this issue, the authors have proposed a method for detecting malware infections focusing on registry accesses and malicious processes executed on Windows-based host PCs. The proposed malware detection method leverages URSNIF, known for its ability to add registry values for automatic execution during system startup, along with its high failure rate of registry access. In experimental tests conducted using URSNIF banking spyware, the authors calculated a high failure rate in registry accesses and confirmed the detection results through specific access checks. This method was applied to eight

instances of URSNIF, revealing a registry access failure rate of approximately 35.3% and identifying six specific registry accesses. These findings suggest the potential for this method to serve as a new countermeasure against malware, particularly within the URSNIF subspecies. Moreover, it holds promise for applications in detecting other types of malwares. Thus, the future direction involves increasing the number of specimen tests and enhancing the overall effectiveness of this proposed approach.

3. METHOD

The suggested spyware attack detection model is deliberated in this section. First, the Obfuscated-MalMem2022 dataset, which has been utilized in the suggested model, will be presented. Then, the operations that is implemented on the used dataset to prepare it for training and testing the suggested model is discussed. Finally, the DT ML algorithm that is utilized to detect the spyware attack is presented. Figure 1 displays the steps implemented to detect the Spyware attack by the DT algorithm.

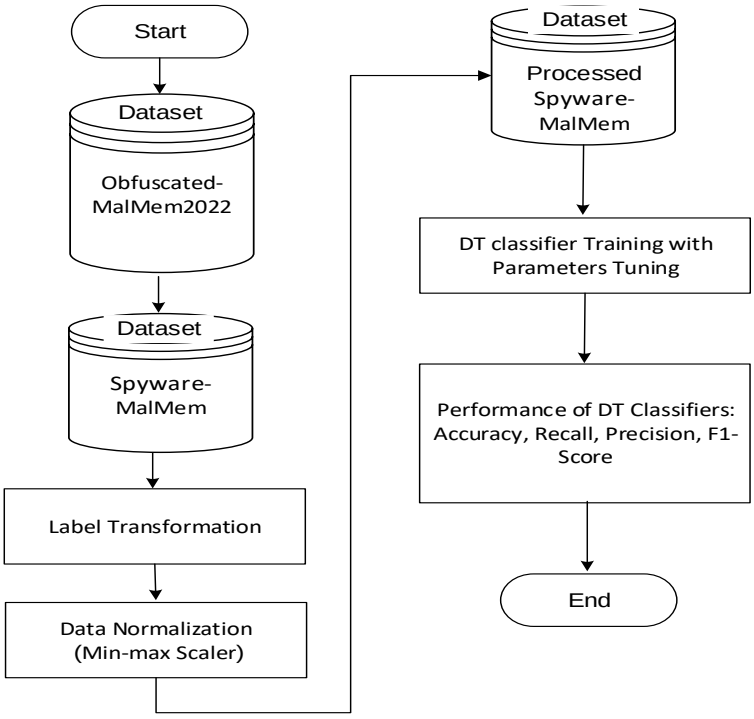

Figure 1. Spyware detection model

3.1. Obfuscated-MalMem2022 dataset

The Obfuscated-MalMem2022 dataset consists of three types of malware: Trojan Horse, spyware, and ransomware. This study emphasis on Spyware only. Consequently, all Trojan Horse and ransomware records is removed. The resultant dataset is named Spyware-MalMem. The Spyware-MalMem dataset consists of 10020 records divided into five types of Spyware: 180Solutions (2000 records), Coolwebsearch (2000 records), Gator (2200 records), Transponder (2410 records), and TIBS (1410 records). Besides, the Spyware-MalMem dataset consists of 29298 records of benign data [19].

3.2. Spyware-MalMem preprocessing

Preprocessing the data within the Spyware-MalMem dataset is crucial to prepare it for ML or data analysis tasks. Two main steps in preprocessing the Spyware-MalMem dataset are transformation and normalization. Transformation involves converting the data into a more suitable format or representation. In the context of the Spyware-MalMem dataset, one key transformation may be applied to convert the output column from categorical into numeric values [14], [20]. First, the five types of spyware, which are 180Solutions, Coolwebsearch, Gator, Transponder, and TIBS, will be replaced by malware [19]. Now, the output column will have only two values: benign and malware. Then, label encoding method might be used to

convert these categorical two values (benign and malware) into numeric values. Particularly, label encoding method will convert the benign to 0 and malware to 1.

Normalization involves adjusting the data to ensure that it adheres to a specific scale or distribution. Normalization is a critical preprocessing step that ensures data is appropriately scaled, enabling ML algorithms to perform better, handle outliers more effectively, and achieve faster convergence. However, it's essential to choose the normalization technique that best fits the characteristics of your data and the requirements of your specific ML task. In the context of the Spyware-MalMem, min-max method might be used to scale the features to a specific range between 0 and1. It involves subtracting the minimum value from each data point and dividing it by the range (max-min). This ensures that all features fall within the specified range [14], [20]. Tables 1 and 2 show samples of the Spyware-MalMem dataset before and after preprocessing, respectively.

Table 1. Sample of the spyware-MalMem dataset before preprocessing

Data samples	Output
45, 17, 10.55555556, 0, 202.8444444, 1694, 38.5	Benign
47, 19, 11.53191489, 0, 242.2340426, 2074, 44.12765957	Benign
40, 14, 14.725, 0, 288.225, 1932, 48.3	Benign
39, 15, 11.41025641, 0, 220.7692308, 1563, 40.07692308	Spyware
37, 15, 10.13513514, 0, 214.9189189, 1446, 39.08108108	Spyware
39, 15, 10.71794872, 0, 217.6410256, 1552, 39.79487179	Spyware

Table 2. Sample of the spyware-MalMem dataset after preprocessing

Data samples	Output
0.143712575, 0.155172414, 0.577520434, 0, 0.15162838, 0.369275153, 0.670560748	0
0.155688623, 0.189655172, 0.64338607, 0, 0.189252506, 0.506310855, 0.796932901	0
0.113772455, 0.103448276, 0.858793019, 0, 0.233182114, 0.455102777, 0.890625073	0
0.107784431, 0.120689655, 0.635178935, 0, 0.168749764, 0.322033898, 0.705971412	1
0.107784431, 0.120689655, 0.635178935, 0, 0.168749764, 0.322033898, 0.705971412	1
0.107784431, 0.120689655, 0.588475549, 0, 0.165761767, 0.318067075, 0.699637797	1

3.3. DT for spyware detection

DT classifier is a supervised ML method commonly used for solving classification problems. In cybersecurity, DT classifiers serve various purposes, including intrusion detection, email and spam filtering, and malware detection [21], [22]. In this study, DT classifier is harnessed for the efficient detection and classification of spyware. Leveraging DT classifier for spyware detection offers several compelling benefits. First, DT is inherently interpretable. The decision-making process is straightforward, enabling security analysts to understand and visualize the classification process of spyware easily. This transparency is invaluable for distinguishing spyware detection false positives from false negatives and elucidating the rationale behind spyware classification. Second, compared to other machine-learning models, DT classifier impose minimal processing overhead. This allows endpoint security solutions to perform real-time or near-real-time spyware detection. Third, DT can reveal the relative importance of specific features of spyware. Security experts can identify which characteristics of executables most indicate spyware presence. Finally, well-structured DT have the potential to scale efficiently, accommodating a substantial number of spyware features and samples without compromising system performance [21]-[23].

When used for identifying spyware, the DT method involves several essential processes executed during both the training and prediction phases. During the training phase, six primary steps are carried out. First, the DT decides which spyware features to split. It assesses all available spyware features at each internal node of the tree to determine the feature that provides the most accurate split. The second step involves setting the spyware features split threshold, if necessary. For continuous or ordinal features, the DT determines an ideal threshold value that effectively splits the spyware data into two groups. The third step revolves around creating internal nodes of the DT. The tree is populated with internal nodes by selecting a feature and establishing the appropriate threshold. The fourth step involves splitting the data into two subsets; one meets the specific criteria, and the other does not. The fifth step is the recursive construction of the tree, extending it further to represent the decision boundaries in the data effectively. Lastly, the sixth step involves creating leaf nodes, which signify the projected class labels (i.e., spyware or benign) [23]-[26].

During the prediction phase, three primary steps are carried out. First, the DT traverses the tree. When a new data point requires classification, it starts at the root node and moves down the tree, evaluating the decision rules at each successive node. If a condition is false, it moves to the right child node; if true, it

proceeds to the left child node (e.g., when a feature is less than the threshold). The second step involves reaching a leaf node. The traversal continues until one of the leaf nodes is reached, where the projected spyware or benign label is determined. In the third step, the predicted class label is returned, serving as the outcome of the classification process [23]-[26].

Constructing an effective DT classifier for spyware detection presents a significant challenge, primarily revolving around the selection of hyperparameters. Optimal hyperparameter choices are crucial in order to strike the right balance, preventing overfitting while achieving the best results in spyware detection. Several key hyperparameters significantly influence the performance of DT in the context of spyware detection. The 'criterion' hyperparameter is a measure of split quality in a DT. It determines how the tree should evaluate potential splits. The 'max depth' hyperparameter sets the maximum allowable depth of the DT. It limits the number of levels from the root node to the leaf nodes. The 'min samples split' hyperparameter is the minimum number of samples required to split an internal node. The 'min samples leaf' hyperparameter represents the minimum number of samples required to be in a leaf node. The 'max features' hyperparameter limits the number of features to consider when making a split. The 'splitter' hyperparameter determines the strategy for node splitting. The 'max leaf nodes' hyperparameter restricts the maximum number of leaf nodes in the DT. The 'min impurity' hyperparameter decrease sets a threshold for impurity decrease. A node is split only if the impurity decrease exceeds this threshold. The 'class weight' hyperparameter allows you to assign weights to classes [26]-[30]. The values chosen to detect spyware are summarized in Table 3.

Table 3. DT hyperparameters value for spyware detection

Hyperparameter	Used value	Justification
Criterion	Gini	'Gini' successfully creates decision boundaries that differentiate between spyware and benign data. It achieves this by identifying feature combinations and thresholds that minimize impurity, reducing the likelihood of mixing spyware and benign data within the same nodes of the DT. 'Gini 'guarantees that the DT can identify traits and patterns suggestive of Spyware, resulting in a more precise and trustworthy detection.
Maximum depth	20	Using '20' is reasonable for the spyware-MalMem dataset, which has 55 features, intricate patterns, and 35,000 records. This depth strikes a compromise between overfitting and model complexity, enabling the tree to provide complex decision boundaries and good generalization to previously unknown data.
Min samples split	2	The spyware-MalMem contains complex feature relationships, a substantial size of 35,000 records, and imbalanced class distributions. Thus, using '2' allows the model to capture subtle patterns within the data.
Min samples leaf	1	The '1' value is useful for managing the unbalanced spyware-MalMem dataset and spotting complex patterns and correlations within the dataset. It also helps to extract fine-grained differences from the intricate spyware-MalMem dataset.
Max features	Auto	The 'auto' value balances the bias and variance of the spyware-MalMem dataset, maintains feature diversity, prevents overfitting, and helps manage computational resources efficiently.
Splitter	Best	Using the 'best' value for spyware detection will maximize information gain, keep features' importance well-balanced, and prevent overfitting.
Max leaf nodes	None	Using the "None" value will minimize the risk of underfitting. It also helps capture important details and subtleties in the data, especially when detecting spyware.
Min impurity decrease	0.0	Using the '0.0' value for spyware detection will allow unbiased and comprehensive dataset exploration. Considering the complex nature of spyware behavior, this approach captures intricate and diverse features that may be crucial for accurate malware detection.
Class weight	Balanced	Because the classes in the spyware-MalMem dataset are not spread out evenly, it makes sense to use the 'balanced' value to look for spyware. This choice automatically adjusts class weights to address the imbalance, focusing on accurately detecting the minority class (spyware) and reducing the risk of false negatives.

4. RESULTS AND DISCUSSION

The proposed model was conducted on an HP Desktop with a 12th-generation Intel Core i7 processor, Windows 11, NVIDIA GeForce RTX 3060, 16 GB DDR5-4000 RAM, and 1 TB SSD hard drive. Python was used to implement a DT classifier for spyware detection the K-fold cross-validation technique is used to assess the performance and generalizability of the DT classifier for spyware detection. The spyware-MalMem is divided into five folds (parts), and then iteratively train and test your model five times, using a different part as the test set in each iteration [14], [31]. The confusion matrix is used to evaluate the performance of the proposed classification model. It is very suitable for this work where we have two classes (binary classification) benign or spyware. Confusion matrix consists of four components: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Using these four components, we can calculate various performance metrics to assess the quality of detecting spyware using DT. These metrics include accuracy, precision (positive predictive value), recall (sensitivity or true positive rate), and F1-score [14], [30].

4.1. Accuracy

Accuracy is one of the most straightforward metrics used to evaluate the performance of a classification model. It's defined as the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances in the dataset. The accuracy of spyware detection using DT is compared to the accuracy of other classifiers, as shown in Figure 2. As can be seen, the DT classifier managed to achieve the greatest level of accuracy, which was 99.97%. The NB classifier, on the other hand, had the lowest accuracy of all the classifiers, coming in at 98.41%.

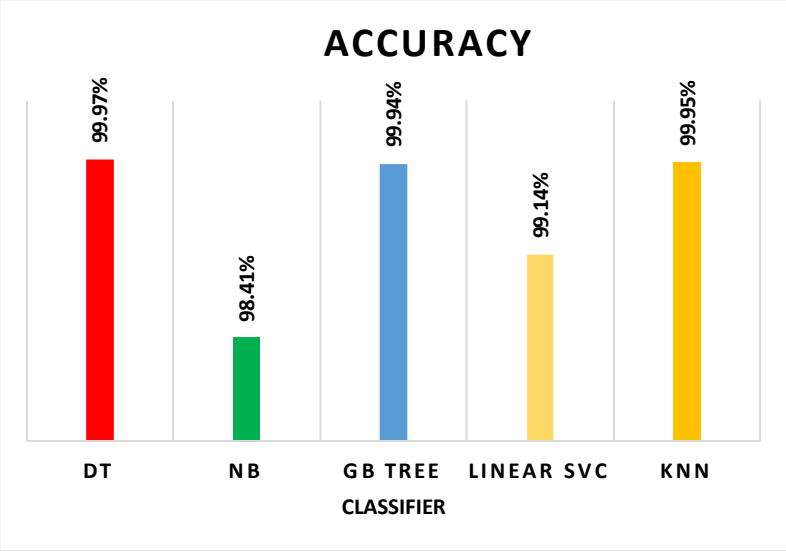

Figure 2. Accuracy of spyware detection

4.2. Precision

Precision also known as positive predictive value, is a metric that evaluates the accuracy of positive predictions made by a classification model. It quantifies the proportion of true positive predictions among all instances that the model predicted as positive. Clarification regarding the precision of spyware detection using DT is provided in Figure 3, which compares DT to different classifiers. As can be seen, the DT and GB tree classifiers attained a precision of 99.97%, which was the maximum possible result. The NB classifier, on the other hand, came in with the lowest precision result of 97.02% out of all the classifiers.

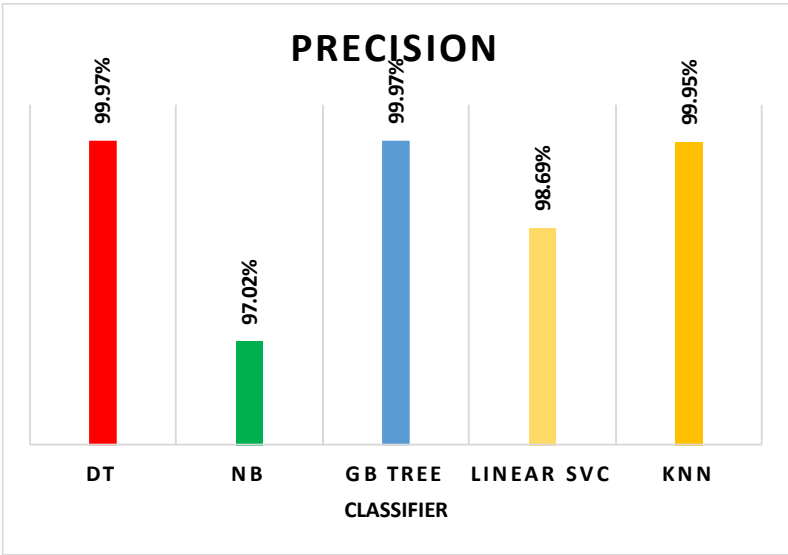

Figure 3. Precision of spyware detection

4.3. Recall

Recall measures the ability of a classification model to correctly identify all positive instances out of all actual positive instances. It quantifies the proportion of true positive predictions among all actual positive instances. Figure 4 illustrates more clearly how well DT compares to other classifiers when detecting spyware. As can be seen, the DT classifier successfully attained the maximum recall percentage of 99.97%. The linear SVC classifier, on the other hand, had the lowest recall of all the classifiers, coming in at 99.57%.

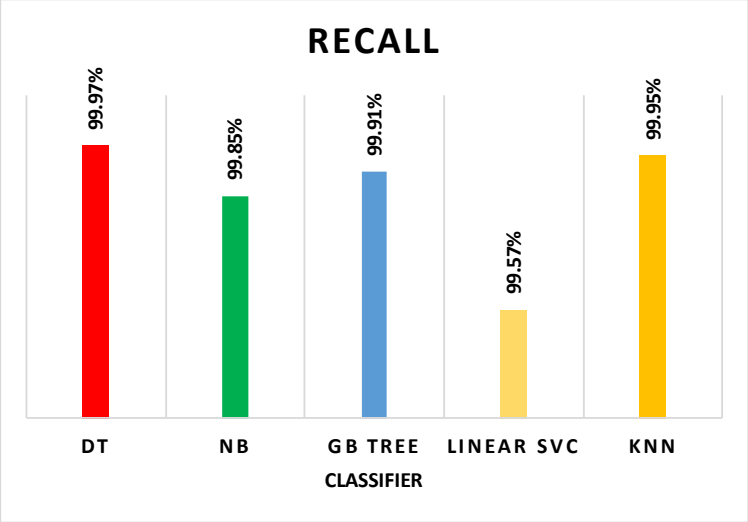

Figure 4. Recall of spyware detection

4.4. F1-score

F1-score is a widely used performance metric in ML, especially in cases where precision and recall are both important, and you want to balance the trade-off between them. The F1-score of spyware detection with DT compared to other classifiers is shown in Figure 5. As can be seen, the DT classifier managed to achieve the greatest level of F1-score, which was 99.97%. The NB classifier, on the other hand, had the lowest F1-score of all the classifiers, coming in at 98.42%.

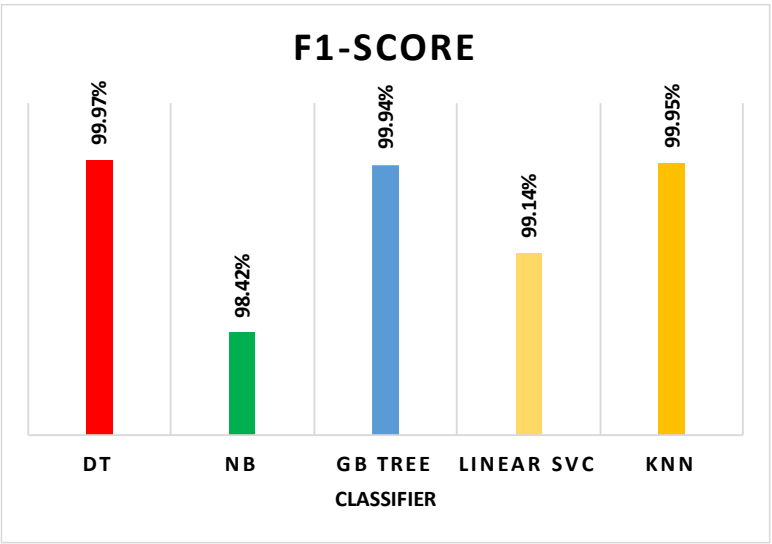

Figure 5. F1-score of spyware detection

5. CONCLUSION

In cybersecurity, spyware is an ever-present and pervasive threat, extensively explored in this research. The study sheds light on the profound implications of spyware, emphasizing the array of harms it inflicts on individuals and businesses, which include privacy breaches, data compromises, and operational disruptions. Moreover, this research has developed a robust methodology for spyware identification. The primary classifier used, DT, has consistently proven its exceptional effectiveness. The meticulous tuning of DT's hyperparameters emerges as a critical factor in optimizing the precision of spyware threat identification, ensuring even the most intricate spyware intrusions are pinpointed with remarkable accuracy. Validation of the proposed methodology, using the Obfuscated-MalMem2022 dataset, reaffirms its efficacy and correctness. The DT classifier consistently outperforms other ML classifiers, including KNN, linear SVC, NB, and GradiGB tree. This unwavering superiority results in an extraordinary accuracy rate of 99.97%, attesting to the unparalleled efficacy of DT in spyware identification. By equipping cybersecurity experts with a potent detection mechanism and deepening our understanding of the widespread impact of malware, this research contributes significantly to the ongoing battle against spyware.

ACKNOWLEDGEMENTS

The authors would like to express their heartfelt gratitude to Al-Ahliyya Amman University for sponsoring this research.

REFERENCES

[1] J. Lei *et al*., "A reinforcement learning approach for defending against multiscenario load redistribution attacks," *IEEE Transactions on Smart Grid*, vol. 13, no. 5, pp. 3711–3722, Sep. 2022, doi: 10.1109/tsg.2022.3175470.

[2] T. K. C. Alves, R. Das, and T. H. Morris, "Embedding encryption and machine learning intrusion prevention systems on programmable logic controllers," *IEEE Embedded Systems Letters*, vol. 10, no. 3, pp. 99–102, Sep. 2018, doi: 10.1109/les.2018.2823906.

[3] M. Abualhaj, A. Abu-Shareha, Q. Shambour, A. Alsaaidah, S. Al-Khatib, and M. Anbar," Customized K-nearest neighbors' algorithm for malware detection," *International Journal of Data and Network Science*, vol. 8, no. 1, pp. 431-438, 2024, doi: 10.5267/j.ijdns.2023.9.012.

[4] W. Peng, F. Li, X. Zou, and J. Wu, "Behavioral malware detection in delay tolerant networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 53–63, Jan. 2014, doi: 10.1109/tpds.2013.27.

[5] S. Şen, E. Aydogan, and A. I. Aysan, "Coevolution of mobile malware and Anti-Malware," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2563–2574, Oct. 2018, doi: 10.1109/tifs.2018.2824250.

[6] D. Javaheri, M. Hosseinzadeh, and A. M. Rahmani, "Detection and elimination of spyware and ransomware by intercepting Kernel-Level system routines," *IEEE Access*, vol. 6, pp. 78321–78332, Jan. 2018, doi: 10.1109/access.2018.2884964.

[7] H. Badih, Y. Alagrash and J. Rrushi, "A Blockchain and Defensive Deception Co-design for Webcam Spyware Detection," *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, Calgary, AB, Canada, 2020, pp. 593–600, doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00104.

[8] Q. Han, V. S. Subrahmanian, and Y. Xiong, "Android malware detection via (Somewhat) robust irreversible feature transformations," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3511–3525, Jan. 2020, doi: 10.1109/tifs.2020.2975932.

[9] N. Dekker, "Malware Statistics in 2022: The Evolving Cyber Threat-eftsure," *Eftsure*, Sep. 06, 2022. https://eftsure.com/statistics/malware-statistics (Accessed: 10/9/2023)

[10] Ctelecoms, "Everything You Need To Guard Against A Keylogger!," *Ctelecoms* https://www.ctelecoms.com.sa/en/Blog267/Everything-You-Need-To-Guard-Against-A-Keylogger! (Accessed: 10/9/2023)

[11] C. Wang and H. Zhu, "Representing Fine-Grained Co-Occurrences for Behavior-Based fraud detection in online payment services," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 301–315, Jan. 2022, doi: 10.1109/tdsc.2020.2991872.

[12] Y. Ding, X. Xia, S. Chen, and Y. Li, "A malware detection method based on family behavior graph," *Computers & Security*, vol. 73, pp. 73–86, Mar. 2018, doi: 10.1016/j.cose.2017.10.007.

[13] A. Parizad and C. J. Hatziadoniu, "Cyber-Attack detection using principal component analysis and noisy clustering algorithms: a collaborative Machine Learning-Based framework," *IEEE Transactions on Smart Grid*, vol. 13, no. 6, pp. 4848–4861, Nov. 2022, doi: 10.1109/tsg.2022.3176311.

[14] M. M. Abualhaj, A. A. Abu-Shareha, M. O. Hiari, Y. Alrabanah, M. Al-Zyoud, and M. A. Alsharaiah, "A Paradigm for DoS Attack Disclosure using Machine Learning Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 3, Jan. 2022, doi: 10.14569/ijacsa.2022.0130325.

[15] S. Choudhary and A. Sharma, "Malware Detection & Classification using Machine Learning," *2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3), Lakshmangarh, India*, Feb. 2020, doi: 10.1109/iconc345789.2020.9117547.

[16] C.-W. Chen, C.-H. Su, K.-W. Lee, and P.-H. Bair, "Malware Family Classification using Active Learning by Learning," *2020 22nd International Conference on Advanced Communication Technology (ICACT), Phoenix Park, Korea (South)*, Feb. 2020, doi: 10.23919/icact48636.2020.9061419..

[17] A. O. Almashhadani, M. Kaiiali, S. Sezer, and P. O'Kane, "A Multi-Classifier Network-Based Crypto Ransomware Detection System: A case study of Locky ransomware," *IEEE Access*, vol. 7, pp. 47053–47067, Jan. 2019, doi: 10.1109/access.2019.2907485.

[18] K. Kono, S. Phomkeona, and K. Okamura, "An Unknown Malware Detection Using Execution Registry Access," *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan*, Jul. 2018, doi: 10.1109/compsac.2018.10281.
[19] M. Dener, G. Ok, and A. Orman, "Malware detection using memory analysis data in big data environment," *Applied Sciences*, vol. 12, no. 17, p. 8604, Aug. 2022, doi: 10.3390/app12178604.
[20] H. Al-Mimi, N. Hamad, M. Abualhaj, M. Daoud, A. Al-Dahoud, and M. Rasmi, "An Enhanced Intrusion Detection System for Protecting HTTP Services from Attacks," *Int. J. Advance Soft Compu. Appl*, vol. 15, no. 2, pp. 67-84, Jul. 2023, doi: 10.15849/IJASCA.230720.05.
[21] V. K. Singh and M. Govindarasu, "A Cyber-Physical Anomaly Detection for Wide-Area Protection Using Machine Learning," in *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3514–3526, Jul. 2021, doi: 10.1109/TSG.2021.3066316.
[22] E. Nowroozi, Abhishek, M. Mohammadi, and M. Conti, "An Adversarial attack analysis on Malicious Advertisement URL Detection Framework," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1332–1344, Jun. 2023, doi: 10.1109/tnsm.2022.3225217.
[23] J. Vijayarangam, S. Kamalakkannan, and J. A. Smiles, "A Novel Comparison of Neural Network and Decision Tree as Classifiers using R," *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, 2021, pp. 712–715, doi: 10.1109/I-SMAC52330.2021.9640882.
[24] M. Kolhar, F. Al-Turjman, A. Alameen, and M. M. Abualhaj, "A three layered decentralized IoT biometric architecture for city lockdown during COVID-19 outbreak," *IEEE Access*, vol. 8, pp. 163608–163617, Jan. 2020, doi: 10.1109/access.2020.3021983.
[25] W. B. Zulfikar, Y. A. Gerhana, and A. F. Rahmania, "An Approach to Classify Eligibility Blood Donors Using Decision Tree and Naive Bayes Classifier," *2018 6th International Conference on Cyber and IT Service Management (CITSM)*, Parapat, Indonesia, Aug. 2018, pp. 1–5, doi: 10.1109/citsm.2018.8674353.
[26] S.-E. Ryu, D.-H. Shin, and K. Chung, "Prediction model of dementia risk based on XGBOOST using derived variable extraction and hyper parameter optimization," *IEEE Access*, vol. 8, pp. 177708–177720, Jan. 2020, doi: 10.1109/access.2020.3025553.
[27] J. Song, L. Xiao, M. Molaei, and Z. Lian, "Sparse coding driven deep decision tree ensembles for nucleus segmentation in digital pathology images," *IEEE Transactions on Image Processing*, vol. 30, pp. 8088–8101, Jan. 2021, doi: 10.1109/tip.2021.3112057.
[28] W. Alawad, M. Zohdy, and D. Debnath, "Tuning Hyperparameters of Decision Tree Classifiers Using Computationally Efficient Schemes," *2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, Laguna Hills, CA, USA, Sep. 2018, pp. 168–169, doi: 10.1109/aike.2018.00038.
[29] F. Khan, S. Kanwal, S. Alamri, and B. Mumtaz, "Hyper-Parameter optimization of classifiers, using an artificial immune network and its application to software bug prediction," *IEEE Access*, vol. 8, pp. 20954–20964, Jan. 2020, doi: 10.1109/access.2020.2968362.
[30] R. G. Leiva, A. F. Anta, V. Mancuso, and P. Casari, "A novel Hyperparameter-Free approach to decision tree construction that avoids overfitting by design," *IEEE Access*, vol. 7, pp. 99978–99987, Jan. 2019, doi: 10.1109/access.2019.2930235.
[31] H. Al-Mimi, N. A. Hamad, and M. M. Abualhaj, "A Model for the Disclosure of Probe Attacks Based on the Utilization of Machine Learning Algorithms," *2023 10th International Conference on Electrical and Electronics Engineering (ICEEE)*, Istanbul, Turkiye, 2023, pp. 241–247, doi: 10.1109/ICEEE59925.2023.00051.

BIOGRAPHIES OF AUTHORS

Mosleh M. Abualhaj is a senior lecturer in Al-Ahliyya Amman University. He received his first degree in Computer Science from Philadelphia University, Jordan, in 2004, Master degree in Computer Information System from the Arab Academy for Banking and Financial Sciences, Jordan in 2007, and Ph.D. in Multimedia Networks Protocols from Universiti Sains Malaysia in 2011. His research area of interest includes VoIP, congestion control, cybersecurity data mining, and optimization. He can be contacted at email: m.abualhaj@ammanu.edu.jo.

Ahmad Sami Al-Shamayleh received the Master's degree in Information Systems from The University of Jordan, Jordan, in 2014, and the Ph.D. degree in Artificial intelligence from University of Malaya, Malaysia, in 2020. He is currently an Assistant Professor with the Faculty of Information Technology, Al-Ahliyya Amman University, Jordan. His research interests include: artificial intelligence, human computer interaction, IoT, Arabic NLP, Arabic sign language recognition, language resources production, the design and evaluation of interactive applications for handicapped people, multimodality, and software engineering. He can be contacted at email: a.alshamayleh@ammanu.edu.jo.

Alhamza Munther is an Assistant Professor in University Technology and Applied Sciences in Sur, Sultanate of Oman. He received his BSc. In computer and software engineering from the University of Technology in Baghdad in 2003. He received a Master's degree in Advanced Computer Networks from Universiti Sains Malaysia (USM) in 2012, and in March 2017, he was awarded a Ph.D. in Computer Engineering from Universiti Malaysia Perlis, Malaysia. His research focuses on overlay networks, multimedia distribution, traffic engineering, data mining, and machine learning. He can be contacted at email: alhamza.wardi@utas.edu.om.

Sumaya Nabil Alkhatib is a senior lecturer in Al-Ahliyya Amman University. She received his first degree in Computer Science from Baghdad University, Iraq, in June 1994 and Master degree in Computer Information System from the Arab Academy for Banking and Financial Sciences, Jordan in February 2007. Her research area of interest includes VoIP, congestion control, cybersecurity data mining, and optimization. She can be contacted at email: sumayakh@ammanu.edu.jo.

Mohammad O. Hiari is a lecturer in Al-Ahliyya Amman University. He received his first degree in Software Engineering from Philadelphia University, Jordan, in August 2004 and Master degree in Computer Science from Al Balqa Applied University, Jordan in February 2016. His research area of interest includes VoIP and cybersecurity data mining and optimization. He can be contacted at email: m.hyari@ammanu.edu.jo.

Mohammed Anbar (Member, IEEE) received the B.Sc. degree in Software Engineering from Al-Azhar University, Palestine, in 2008, the M.Sc. degree in Information Technology from Universiti Utara Malaysia, in 2009, and the Ph.D. degree in Advanced Internet Security and Monitoring from Universiti Sains Malaysia (USM), in 2013. He is currently a senior lecturer with the National Advanced IPv6 Centre (NAv6), USM. His current research interests include malware detection, intrusion detection systems (IDSs), intrusion prevention systems (IPSs), network monitoring, the internet of things (IoT), software-defined networking (SDN) security, cloud computing security, and IPv6 security. He can be contacted at email: anbar@usm.my.