

Android malware detection using the random forest algorithm

Anas El Attaoui, Norelislam El Hami, Younes Koulou

Science and Engineering Laboratory, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco

Article Info

Article history:

Received May 11, 2024

Revised Aug 20, 2024

Accepted Aug 31, 2024

Keywords:

Android malware detection

Artificial intelligence

Decision trees

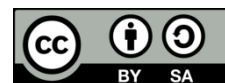
Machine learning

Random forest algorithm

ABSTRACT

The rapid growth in Android device usage has resulted in a significant increase in malware targeting this platform, posing serious threats to user security and privacy. This research tackles the challenge of Android malware detection by leveraging advanced machine learning techniques, with a particular emphasis on the random forest (RF) algorithm. Our primary objective is to accurately identify and classify malicious applications to enhance the security of Android devices. In this study, we employed the RF algorithm to analyze a comprehensive dataset of Android applications, where the classification of each application as either malware or benign is known. The method was rigorously tested, yielding impressive results: an average accuracy of 98.47%, a sensitivity of 98.60%, and an F-score of 98.60%. These metrics underscore the effectiveness of our approach. Moreover, we conducted a comparative analysis of the RF algorithm against other malware detection methods. The results demonstrate that the RF algorithm outperforms these alternative methods, offering superior detection capabilities and contributing to more robust Android security measures.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Anas El Attaoui

Science and Engineering Laboratory, National School of Applied Sciences, Ibn Tofail University

Kenitra, Morocco

Email: anaselatt045@gmail.com

1. INTRODUCTION

The Android operating system, which powers billions of devices worldwide [1], [2], faces major challenges in managing its enormous data volume. As the platform grows, the complexity of data management increases [3], [4], making it harder to ensure strong security. Consequently, Android has become a prime target for malicious actors aiming to exploit vulnerabilities and compromise user privacy and security [5]. With the rising sophistication and prevalence of Android malware, there is a critical need for effective detection methods to protect users and their sensitive information [6]. In response, researchers and cybersecurity experts are turning to machine learning (ML) techniques [7] to enhance malware detection capabilities on Android devices.

Several studies have explored Android malware detection using different machine learning approaches. Yang *et al.* [8] introduces a novel framework based on contrastive learning, which pretrains models without labeled data and achieves over 96% accuracy for general malware and over 98% for multiclass detection. Ma *et al.* [9] presents the multi-channel adaptive detection system (MCADS), a lightweight system with a two-layer structure consisting of a multidimensional perceptron (MLP) for initial analysis and a lightweight convolutional neural network (CNN) for further examination, reaching 98.12% accuracy. Additionally, Sahin *et al.* [10] employing multiple linear regression to analyze app permissions achieved 98.53% accuracy, demonstrating its effectiveness compared to other machine learning methods like support vector machines, k-nearest neighbors, Naive Bayes (NB), and decision trees (DT) across four datasets without requiring complex models. Lastly, Mat *et al.* [11] proposes an Android malware detection

system that uses permission features and Bayesian classification, analyzing 10,000 samples from AndroZoo and Drebin databases through static analysis. By applying information gain and chi-square feature selection algorithms, the system achieves a 91.1% detection accuracy rate.

Android malware evolves over time, making it clear that a single detection method is insufficient. Therefore, it is crucial to develop multiple detection strategies that can adapt to the changing nature of the data. While the random forest (RF) algorithm has been extensively utilized for various classification tasks due to its simplicity, flexibility, and robustness [12], its specific application to Android malware detection has not been comprehensively explored. This study aims to bridge that gap by providing a thorough evaluation of the RF algorithm in the context of Android malware detection, highlighting its strengths and potential limitations.

We will begin by elucidating the working principles of the RF algorithm and why it is particularly suited for malware detection. Following this, we will present an extensive Android malware dataset used in our analysis. Subsequent sections will detail our methodology for training and testing the algorithm, along with a comparative analysis of its performance against other state-of-the-art algorithms using key metrics such as accuracy, recall (sensitivity), and the F-score. Through this structured approach, our research will not only demonstrate the efficacy of the RF algorithm in detecting Android malware but also establish its relevance and superiority in this domain, providing valuable insights for future research and practical implementations.

2. METHOD

The RF algorithm is one of the most popular and effective machine learning techniques for classification and regression [13], [14]. This algorithm relies on the use of a set of DT, where each tree is randomly constructed from a subset of the training data [15]. By combining the predictions of multiple DT, the RF reduces overlearning and improves the robustness of the overall model [16].

2.1. Decision trees

The essence of the RF algorithm stems from DT, necessitating a comprehensive understanding of their mechanics for a thorough comprehension of their operation. DTs are extensively employed in machine learning, relying on heuristics derived from supervised learning methodologies. Their architecture is hierarchical, comprising nodes and leaves interconnected by branches. Graphically, the root sits atop the hierarchy, while the leaves form the bottom layer. Internal nodes, termed decision nodes, may encompass one or multiple rules, referred to as tests or conditions [17].

The potential values a variable can take in a DT are termed instances or attributes. Terminal nodes hold the predicted class, also known as the target variable. After construction, a DT can be represented as a collection of decision rules. In cases where the variable to be predicted is categorical, the tree manifests as a classification tree. Conversely, if the variable is quantitative, it assumes the shape of a regression tree. Figure 1 illustrates the layout of a DT [18].

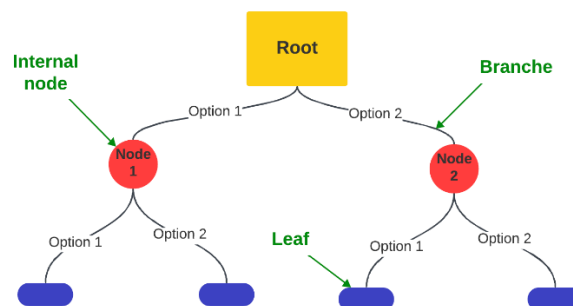


Figure 1. The structure of a DT

Making predictions using DT involves a systematic traversal of the tree structure based on the attributes of the input data. Beginning at the root node, the algorithm evaluates attributes at each internal node, selecting the appropriate branch based on the attribute's value in the input data. This process continues recursively until a terminal node is reached. At the terminal node, the prediction is made by determining the majority class for classification tasks or the average value for regression tasks, based on the training instances

associated with that node. This iterative process ensures that each input data instance is guided through the DT, ultimately resulting in a prediction based on the learned patterns within the data [19].

DT algorithms like Iterative Dichotomiser 3 (ID3), C4.5, and classification and regression trees (CART) are essential tools in machine learning, widely used for both classification and regression. The ID3 algorithm, one of the pioneers in this area, builds DT for classification by selecting attributes that maximize information gain [20]. C4.5, which builds upon ID3, brought several improvements, such as the ability to handle both continuous and discrete data, manage missing values, and implement pruning techniques to avoid overfitting [21]. CART is another prominent algorithm, distinct from ID3 and C4.5 in that it constructs binary trees, splitting the data into two groups at each node based on an attribute's value. For classification tasks, CART uses Gini impurity to choose the best splits, while for regression, it relies on mean squared error [22]. Despite their unique features, these algorithms all aim to recursively divide the feature space to generate decision rules that effectively predict outcomes.

Despite their effectiveness, DT have certain limitations that prompted the development of ensemble methods like RFs. One key limitation of DT is their tendency to overfit the training data, especially when the trees become deep and complex. Overfitting occurs when the model captures noise or irrelevant patterns in the data, leading to poor generalization of unseen data. DTs are also susceptible to instability, meaning small changes in the training data can result in significantly different tree structures. Additionally, DTs may struggle with capturing complex relationships in the data, particularly when features interact nonlinearly. To mitigate these issues, RF were introduced as an ensemble learning technique. By aggregating multiple DT trained on random subsets of the data and features, RF can reduce overfitting, improve stability, and capture more complex patterns in the data, thereby enhancing predictive performance [23].

2.2. Random forest algorithm

The RF is a supervised learning algorithm used for classification, regression, and other predictive analysis tasks. Its basic principle is based on the idea of combining several individual DT to form a robust and diverse «forest». Each DT is trained on a random sample of training data and uses a random subset of the available features to make decisions. When it's time to make predictions, each tree in the forest gives its own prediction, and the final prediction is determined by a majority vote or an average, depending on whether the task is classification or regression [24].

The RF algorithm comprises several steps to build an ensemble of DT for robust predictions. Firstly, it randomly selects subsets of the training data through a process called bootstrapping, creating multiple training sets for each tree. Next, DT are constructed on each bootstrapped dataset using a subset of randomly selected features at each node. This randomness ensures diversity among the individual trees in the forest. During tree construction, the algorithm recursively partitions the feature space based on the selected features, optimizing splits to maximize information gain or decrease impurity. Once all trees are grown, predictions are made by aggregating the outputs of individual trees. For classification tasks, the most frequent class among the predictions is chosen, while for regression tasks, the average of the predicted values is computed. The final prediction reflects the collective decision of the ensemble, providing robust and accurate predictions while mitigating the risk of overfitting inherent in individual DTs [25]. Figure 2 illustrates the workings of the RF algorithm.

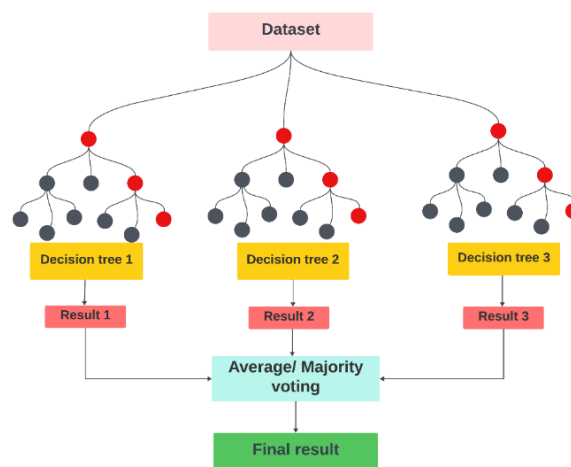


Figure 2. RF algorithm functioning

2.3. Different uses of the random forest algorithm

Numerous research endeavors have underscored the superior efficacy of the RF algorithm, particularly evident in its application for road traffic accident (RTA) prediction, as demonstrated in a study referenced as [26]. In regions characterized by high incidence rates of RTAs, such as low- and middle-income countries, the accurate forecasting of these events holds paramount importance. Comparing the performance of two prevalent modeling techniques, RF and support vector machine (SVM), utilizing data sourced from law enforcement agencies, the study revealed the RF algorithm's unparalleled performance in short-term RTA prediction. With a coefficient of determination (R²) reaching an impressive 0.91, the RF model significantly outshone the SVM model, which achieved an R² value of 0.86. This disparity highlights the RF robustness in real-time forecasting, offering invaluable insights for traffic management and public safety initiatives aimed at mitigating the impact of road accidents.

Demidova *et al.* [27] presents an innovative approach aimed at refining the classification performance of the SVM by incorporating it with the RF classifier. This novel hybridization strategy focuses on enhancing SVM's classification decisions, particularly for objects situated in specific subareas near the class-separating hyperplane, encompassing both accurately and inaccurately classified entities. Notably, the research suggests that employing this hybrid classification approach could extend the improved classification quality to new objects beyond the initial dataset. Furthermore, comparative analyses underscore the effectiveness of integrating auxiliary classifiers with SVM to augment classification outcomes. This finding not only underscores the adaptability of RF when integrated with other machine learning algorithms but also emphasizes its capacity to elevate their accuracy and efficiency, thereby enhancing overall classification performance.

Supsermpol *et al.* [28] delves into the assessment of companies' financial performance following their public listing on the stock market. Employing a class-based methodology, the research seeks to gauge the influence of two critical factors: the internal capabilities of companies before their initial public offering (IPO) and the capital raised during the IPO process. Utilizing a blend of statistical analysis, particularly logistic regression, and machine learning techniques, notably the RF algorithm, the study endeavors to accurately predict financial performance. Noteworthy findings from the investigation highlight the RF algorithm's superior performance over logistic regression in forecasting financial outcomes, underscoring the effectiveness of machine learning methodologies in augmenting predictive precision in financial analysis.

Lastly, Hoła and Czarnecki [29] delves into the application of machine learning algorithms, notably the RF algorithm and support vector machine, for the non-destructive assessment of moisture content in brick walls within historic structures. Leveraging a dataset comprising 290 sample sets and 7 predictor variables, encompassing factors like construction year, moisture levels, and salt concentration, the research unveils compelling insights. Notably, the RF algorithm emerges as the frontrunner, showcasing superior performance metrics. With a coefficient of determination (R²) reaching an impressive 0.968 and minimal values for root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) at 1.080%, 0.791, and 14.28%, respectively, the RF algorithm establishes itself as the optimal tool for precise, non-destructive evaluation of mass moisture content in historic brick walls. These findings not only bolster the efficacy of machine learning in heritage preservation but also underscore the significance of data-driven approaches in architectural conservation efforts.

The studies referenced above collectively underscore the versatility and effectiveness of the RF algorithm across diverse fields of application. Their findings consistently highlight the algorithm's ability to deliver superior results compared to alternative methods. This compelling evidence serves as the basis for our decision to incorporate the RF algorithm into our study on malware detection. By leveraging its proven track record of performance and adaptability, we aim to enhance the accuracy and efficiency of our malware detection system. This strategic choice aligns with the broader trend in the research community, affirming the RF status as a preferred solution for a wide range of machine learning tasks.

2.4. Data used

The dataset used in this study consists of feature vectors with 215 attributes extracted from a comprehensive set of 15,036 applications. Among these, 5,560 applications, sourced from the Drebin project, were identified as malware, while the remaining 9,476 were classified as benign. This dataset served as the foundation for developing and evaluating an innovative multilevel classifier fusion strategy for Android malware detection [30]. The features include API call signatures, intents, manifest permissions, and command permissions, and Figure 3 shows the count of each feature type in the dataset.

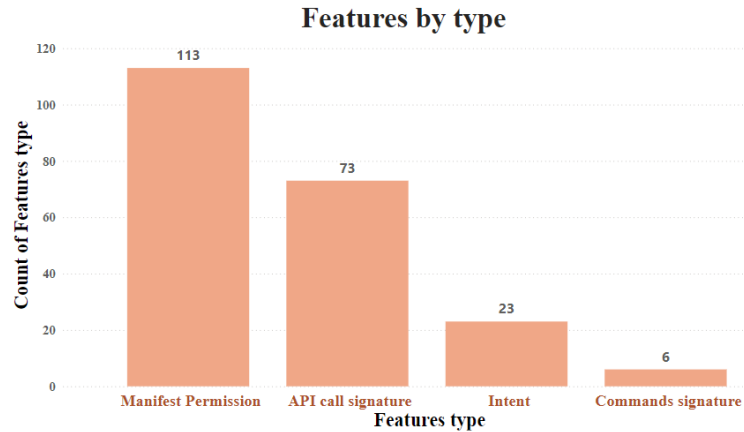


Figure 3. The number of features type in the dataset

2.5. Measurement metrics

After preparing the data beforehand, we advanced to implement our algorithm on a dataset representing 80% of the total, designated as the training set. Subsequently, to gauge the effectiveness of our model, we utilized the remaining 20% as a test set for making predictions. Following this, we computed various metrics, such as accuracy, sensitivity, and F-score, to evaluate the efficacy of our predictions.

Accuracy represents the percentage of correctly identified positive cases among all predicted positive cases, reflecting how well the model recognizes true positives. Sensitivity, also known as recall, measures the proportion of actual true positives that the model successfully predicts, indicating the model's effectiveness in detecting all true positives in the dataset. The F-score, on the other hand, merges accuracy and sensitivity into a single metric, balancing the model's ability to reduce false positives while effectively identifying true positives.

3. RESULTS AND DISCUSSION

The implementation of the RF algorithm yielded impressive results, evident from the performance metrics obtained. Across 100 iterations, prediction accuracy ranged from a commendable minimum of 98.47% to a remarkable maximum of 99.24%, with an average of 98.85%, as depicted in Figure 4. Similarly, the sensitivity of the results demonstrated resilience, with a range from a minimum of 98.37% to a maximum of 99.16%, and an average of 98.60%, as illustrated in Figure 5. Furthermore, the F-score exhibited consistent performance, ranging from a minimum of 98.37% to a maximum of 99.16%, with an average of 98.60%, as shown in Figure 6. These consistent and robust performances underscore the efficacy of the algorithm in addressing the problem at hand.

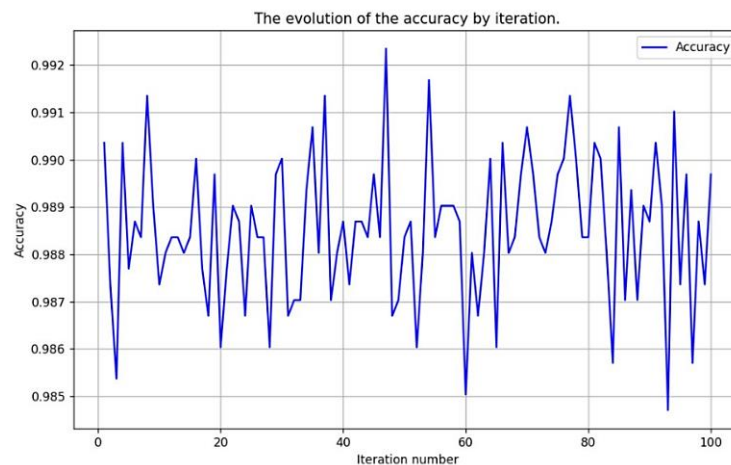


Figure 4. Accuracy in 100 iterations

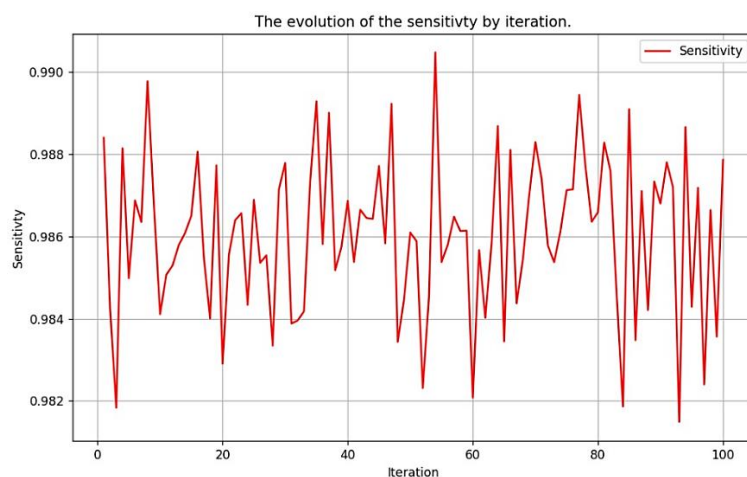


Figure 5. Sensibility (recall) in 100 iterations

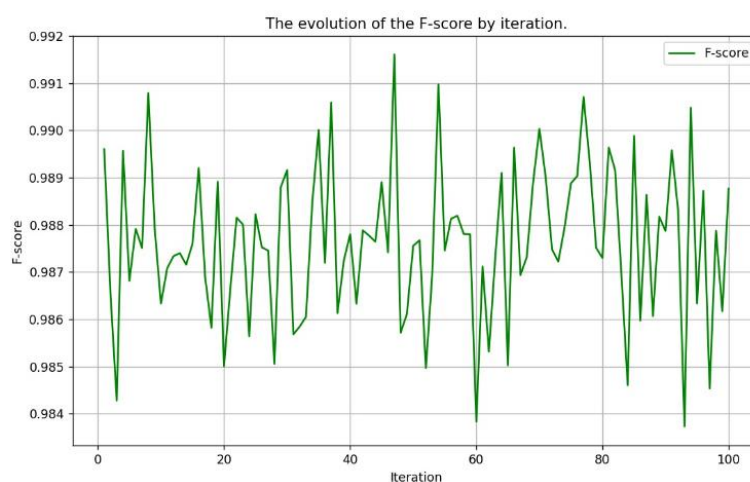


Figure 6. F-score in 100 iterations

Comparing our results with those from studies [8]-[11], the RF algorithm consistently demonstrates superior performance. The multiple linear regression methods in [8] achieved a high accuracy of 98.53%, yet this was slightly lower than the average accuracy of the RF. Similarly, the MCADS system in [9], with its innovative two-layer structure, achieved an accuracy of 98.12%, which is still below the RF average. The framework presented in [10] achieved over 98% accuracy for multiclass detection, but its accuracy for general malware detection was slightly over 96%, both falling short of the RF performance. Lastly, the system in [11], utilizing Bayesian classification and permission features, reached a notably lower accuracy of 91.1%, significantly trailing behind the RF metrics.

This study aimed to evaluate and enhance Android malware detection using the RF algorithm, demonstrating its superior performance compared to other methods. By achieving high accuracy, sensitivity, and F-score, the study underscores the importance of employing robust machine learning techniques to combat the growing threat of mobile malware. Despite these advancements, unanswered questions remain about the algorithm's performance across diverse malware types and its scalability in real-world scenarios, particularly given the evolving characteristics of Android malware over time, which can affect our algorithm's accuracy. Future research could explore the integration of the RF with other machine learning approaches, the impact of evolving malware strategies, and the application of the algorithm in different mobile environments to further improve detection accuracy and reliability.

4. CONCLUSION

In conclusion, this study thoroughly examined the application of the RF algorithm for Android malware detection, assessing its effectiveness through detailed experimentation on a large malware database. Our findings revealed that the RF algorithm significantly outperformed other machine learning techniques in terms of accuracy. This superior performance underscores the algorithm's robustness and reliability, making it a highly effective tool for identifying and combating Android malware. Accurate malware detection is essential for safeguarding user privacy and security. As malware threats become increasingly sophisticated and pervasive, the ability to precisely identify malicious software is critical in protecting users from potential harm. The study highlighted the significant benefits of employing advanced detection algorithms, which contribute to enhanced security measures and more reliable protection against evolving threats.

Moreover, given the constantly evolving nature of malware characteristics, threats are perpetually changing. This ongoing evolution requires malware detection systems to not only address current threats but also anticipate and adapt to future challenges. Maintaining effectiveness over time hinges on the ability to continuously update and refine detection methodologies. In light of these findings, the study emphasizes the importance of continual research and development in the field of malware detection. It is essential to invest in and advance detection technologies to ensure they remain effective in identifying new and emerging malware variants. Overall, this study highlights the pivotal role of the RF algorithm in enhancing Android malware detection and underscores the need for ongoing innovation to stay ahead of evolving cyber threats.





REFERENCES

- [1] D. Soi, A. Sanna, D. Maiorca, and G. Giacinto, "Enhancing android malware detection explainability through function call graph APIs," *Journal of Information Security and Applications*, vol. 80, p. 103691, 2024, doi: 10.1016/j.jisa.2023.103691.
- [2] A. T. Kabakus, "DroidMalwareDetector: a novel Android malware detection framework based on convolutional neural network," *Expert Systems with Applications*, vol. 206, p. 117833, 2022, doi: 10.1016/j.eswa.2022.117833.
- [3] A. El Attaoui, A. Boukhafla, S. Rhouas, and N. El Hami, "Custom application programming interface data extractor applied to the Klarna e-commerce dataset," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 31, no. 3, pp. 1624–1632, 2023, doi: 10.11591/ijeecs.v31.i3.pp1624-1632.
- [4] A. El Attaoui, S. Rhouas, and N. El Hami, "ETL applied to Klarna e-commerce dataset," in *2023 9th International Conference on Optimization and Applications, ICOA 2023 - Proceedings*, Oct. 2023, pp. 1–4, doi: 10.1109/ICOA58279.2023.10308808.
- [5] M. Kumar, S. Singh, U. Pilia, G. Arora, and M. Jain, "LSTM-based approach for Android malware detection," *Procedia Computer Science*, vol. 230, pp. 679–687, 2023, doi: 10.1016/j.procs.2023.12.123.
- [6] S. Rhouas, A. El Attaoui, and N. El Hami, "Optimization of the prediction performance in the future exchange rate," in *2023 9th International Conference on Optimization and Applications, ICOA 2023 - Proceedings*, 2023, pp. 1–6, doi: 10.1109/ICOA58279.2023.10308858.
- [7] S. Rhouas, A. El Attaoui, and N. El Hami, "Enhancing currency prediction in international e-commerce: Bayesian-optimized random forest approach using the Klarna dataset," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 14, no. 3, p. 3177, 2024, doi: 10.11591/ijece.v14i3.pp3177-3186.
- [8] S. Yang, Y. Wang, H. Xu, F. Xu, and M. Chen, "An Android malware detection and classification approach based on contrastive learning," *Computers and Security*, vol. 123, p. 102915, 2022, doi: 10.1016/j.cose.2022.102915.
- [9] R. Ma, S. Yin, X. Feng, H. Zhu, and V. S. Sheng, "A lightweight deep learning-based android malware detection framework," *Expert Systems with Applications*, vol. 255, p. 124633, 2024, doi: 10.1016/j.eswa.2024.124633.
- [10] D. O. Sahin, S. Akleyek, and E. Kilic, "LinRegDroid: detection of Android malware using multiple linear regression models-based classifiers," *IEEE Access*, vol. 10, pp. 14246–14259, 2022, doi: 10.1109/ACCESS.2022.3146363.
- [11] S. R. T. Mat, M. F. A. Razak, M. N. M. Kahar, J. M. Arif, and A. Firdaus, "A Bayesian probability model for Android malware detection," *ICT Express*, vol. 8, no. 3, pp. 424–431, 2022, doi: 10.1016/j.icte.2021.09.003.
- [12] T. P. Dinh, C. Pham-Quoc, T. N. Thinh, B. K. Do Nguyen, and P. C. Kha, "A flexible and efficient FPGA-based random forest architecture for IoT applications," *Internet of Things (Netherlands)*, vol. 22, p. 100813, 2023, doi: 10.1016/j.iot.2023.100813.
- [13] N. Jain and P. K. Jana, "LRF: A logically randomized forest algorithm for classification and regression problems[Formula presented]," *Expert Systems with Applications*, vol. 213, p. 119225, Mar. 2023, doi: 10.1016/j.eswa.2022.119225.
- [14] S. Giri *et al.*, "Revealing the sources of arsenic in private well water using random forest classification and regression," *Science of the Total Environment*, vol. 857, p. 159360, 2023, doi: 10.1016/j.scitotenv.2022.159360.
- [15] M. Wang *et al.*, "Wetland mapping in East Asia by two-stage object-based random forest and hierarchical decision tree algorithms on Sentinel-1/2 images," *Remote Sensing of Environment*, vol. 297, p. 113793, 2023, doi: 10.1016/j.rse.2023.113793.
- [16] X. Zhang *et al.*, "Improved random forest algorithms for increasing the accuracy of forest aboveground biomass estimation using Sentinel-2 imagery," *Ecological Indicators*, vol. 159, p. 111752, 2024, doi: 10.1016/j.ecolind.2024.111752.
- [17] W. Niu *et al.*, "Revealing suicide risk of young adults based on comprehensive measurements using decision tree classification," *Computers in Human Behavior*, vol. 158, p. 108272, 2024, doi: 10.1016/j.chb.2024.108272.
- [18] M. Ozcan and S. Peker, "A classification and regression tree algorithm for heart disease modeling and prediction," *Healthcare Analytics*, vol. 3, p. 100130, Nov. 2023, doi: 10.1016/j.health.2022.100130.
- [19] I. D. Mienye, Y. Sun, and Z. Wang, "Prediction performance of improved decision tree-based algorithms: a review," *Procedia Manufacturing*, vol. 35, pp. 698–703, 2019, doi: 10.1016/j.promfg.2019.06.011.
- [20] S. Chanmee and K. Kesorn, "Semantic decision trees: a new learning system for the ID3-based algorithm using a knowledge base," *Advanced Engineering Informatics*, vol. 58, p. 102156, 2023, doi: 10.1016/j.aei.2023.102156.
- [21] P. Li, F. Xiong, X. Huang, and X. Wen, "Construction and optimization of vending machine decision support system based on improved C4.5 decision tree," *Heliyon*, vol. 10, no. 3, p. e25024, 2024, doi: 10.1016/j.heliyon.2024.e25024.
- [22] G. S. Kori and M. S. Kakkasageri, "Classification and regression tree (CART) based resource allocation scheme for wireless sensor networks," *Computer Communications*, vol. 197, pp. 242–254, 2023, doi: 10.1016/j.comcom.2022.11.003.





- [23] X. Zhou, P. Lu, Z. Zheng, D. Tolliver, and A. Keramati, "Accident prediction accuracy assessment for highway-rail grade crossings using random forest algorithm compared with decision tree," *Reliability Engineering and System Safety*, vol. 200, p. 106931, 2020, doi: 10.1016/j.ress.2020.106931.
- [24] T. Wahyuningsih, D. Manongga, I. Sembiring, and S. Wijono, "Comparison of effectiveness of logistic regression, naive bayes, and random forest algorithms in predicting student arguments," *Procedia Computer Science*, vol. 234, pp. 349–356, 2024, doi: 10.1016/j.procs.2024.03.014.
- [25] L. F. Pordeus, A. E. Lazzaletti, R. R. Linhares, and J. M. Simão, "Notification oriented paradigm to digital hardware — a benchmark evaluation with random forest algorithm," *Microprocessors and Microsystems*, vol. 103, p. 104951, 2023, doi: 10.1016/j.micpro.2023.104951.
- [26] A. Gatera, M. Kuradusenge, G. Bajpai, C. Mikeka, and S. Shrivastava, "Comparison of random forest and support vector machine regression models for forecasting road accidents," *Scientific African*, vol. 21, p. e01739, 2023, doi: 10.1016/j.sciaf.2023.e01739.
- [27] L. A. Demidova, I. A. Klyueva, and A. N. Pylkin, "Hybrid approach to improving the results of the SVM classification using the random forest algorithm," *Procedia Computer Science*, vol. 150, pp. 455–461, 2019, doi: 10.1016/j.procs.2019.02.077.
- [28] P. Supsermpol, V. N. Huynh, S. Thajchayapong, and N. Chiadamrong, "Predicting financial performance for listed companies in Thailand during the transition period: A class-based approach using logistic regression and random forest algorithm," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 9, no. 3, p. 100130, 2023, doi: 10.1016/j.joitmc.2023.100130.
- [29] A. Hoła and S. Czarniecki, "Random forest algorithm and support vector machine for nondestructive assessment of mass moisture content of brick walls in historic buildings," *Automation in Construction*, vol. 149, p. 104793, 2023, doi: 10.1016/j.autcon.2023.104793.
- [30] S. Y. Yerima and S. Sezer, "DroidFusion: A novel multilevel classifier fusion approach for Android malware detection," *IEEE Transactions on Cybernetics*, vol. 49, no. 2, pp. 453–466, 2019, doi: 10.1109/tcyb.2017.2777960.

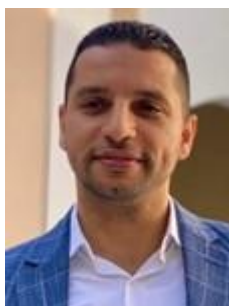
BIOGRAPHIES OF AUTHORS







Anas El Attaoui     he is currently pursuing a Ph.D. in Computer Science at the National School of Applied Science, Ibn Tofail University, Morocco, following his graduation with an engineering degree in Industrial Engineering from the same institution in 2019. His research focuses primarily on big data and machine learning. With numerous papers published in conferences and journals, he can be contacted at email: anaselatt045@gmail.com.



Norelislam El Hami     he is a professor of computer science at the National School of Applied Sciences in Kenitra, Ibn Tofail university, Morocco. He obtained a Diploma of State Engineer in 2000 from the Polytechnic Faculty of MONS (FPMS) in Belgium. The author's areas of specialization were computer science and telecommunications. In June 2012, he earned a Ph.D. in Computer Science from the National Institute of Applied Sciences of Rouen (INSA) in France, and Ph.D. in Applied Mathematics and Computer Science from Mohammed V-Agdal University, EMI, Rabat-Morocco. He can be contacted at email: norelislam@outlook.com.



Younes Koulou     he is a Ph.D. student in computer science from National School of Applied Science, Ibn Tofail University, Morocco. He received a master's degree in software quality at the Faculty of Sciences, Ibn Tofail University in 2010. At present he focuses on artificial intelligence. He can be contacted at email: younes.koulou@uit.ac.ma.