IJRCIT

International Journal of Research in Computer & Information Technology (IJRCIT),
Vol.8, Issue 4, Sept 2023
E-ISSN: 2455-3743
Available online at www.ijrcit.co.in

Machine Learning Framework for Malware Detection

Anushree Anand Pande

Abstract— Detecting malware presence in files is crucial due to its increasing volume, leading to significant issues for companies, such as data loss and operational challenges. Malware can notably hinder system performance by slowing operations and encrypting files in personal computers. This report presents a comprehensive exploration of a versatile framework employing machine learning algorithms tailored specifically for malware detection. These algorithms effectively differentiate between clean and infected files. The primary aim is to minimize false positives in the data. The paper proposes the use of three Machine Learning (ML) models - Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) - utilizing a publicly available malware dataset for training and testing these models. The final outcomes indicate that RF classifiers achieve an accuracy score of 99.12%, surpassing the performance of the other two classifiers.

Keywords— Malware detection, Machine Learning, Random Forest, Decision Tree, and K-Nearest Neighbors

I. Introduction

Security threats pose significant challenges, with a primary concern being malicious software, commonly known as malware. Its core objectives include surreptitiously gathering personal information while disrupting computer operations, causing inconvenience to users [1]. This category encompasses various types of malware, including viruses, worms, Trojans, rootkits, backdoors, spyware, and adware. Antivirus reports consistently reveal the creation of thousands of new malware strains daily, and these emerging threats have evolved to evade conventional detection methods such as signature-based, heuristic, and behavior-based techniques [2].

Signature-based detection involves searching for specific byte sequences within an object to identify known malware types. However, its limitation lies in its inability to detect newly developed or "zero-day" malware, as their signatures are not cataloged in the detection database [3]. Heuristic-based detection, devised to surpass signature detection's limitations, scrutinizes system behavior for anomalies rather than searching for predefined malware signatures. Although this method can detect newly spawned malware without known signatures, it comes with drawbacks such as impacting system performance and necessitating additional space [4].

Behavior-based detection focuses on a program's behavior during execution, categorizing normal execution as benign and abnormal execution as malware. Despite its emphasis on program behavior, this technique generates numerous false positives and false negatives [5]. A benign program might crash and be flagged as a virus, while a virus could execute like a regular program and be wrongly classified as benign.

II. Related Work

Several prior investigations employed machine learning (ML) methods, while others utilized deep learning (DL) techniques such as convolutional networks (CNN), recurrent neural networks (RNN), and long-short-term memory networks (LSTM) [6]-[7]. Some studies focused on desktop-related malware datasets, but the majority concentrated on mobile-related malware datasets.

According to Vinayakumar et al. [8], numerous ML and DL models were employed for detecting malware using the Ember dataset, comprising 70,140 benign and 69,869 malware records. Various models including K-nearest neighbors (KNN), Support Vector Machines (SVM), Random Forests (RF), AdaBoost, Logistic Regression (LR), Naïve Bayes (NB), and Deep Neural Network (DNN) were tested. They utilized the Adam optimization algorithm and trained the models for 200 epochs, with the LSTM model achieving the best performance at 98.9% accuracy.

Jeon and Moon introduced a DL-based malware detection system in 2020 [9]. They used a convolutional encoder to interpret opcode sequences extracted from Windows executable files, followed by recurrent neural networks (RNNs) for the detection process, achieving 96% detection accuracy and a 95% true positive rate.

Yazdinejad et al. [10] extracted opcodes for malware and benign activities from a dataset containing 200 benign and 500 malware records. They applied the LSTM model for constructing a malware detection system using 10-fold cross-validation, attaining a detection accuracy of 98%.

Darabian et al. [11] utilized opcodes and system calls in their study, employing a dataset with 1500 executable samples. They trained the CNN-LSTM model using this dataset, achieving a 99% detection accuracy with opcodes-based records, whereas system calls attained a detection rate

Manuscript Received June 30, 2023; Revised 15 September, 2023 and Published on September 30, 2023

Anushree Anand Pande, *Department of Computer Engineering, Dr Rajendra Gode Institute of Technology And Research Amravati, Maharashtra, India. Mail Id: anushreepande23@gmail.com*

DOI: 10.5281/zenodo.10597681

of 95%.

III. Proposed Methodology

The intended approach involves the utilization of diverse ML algorithms to address the issue of computer system malware present in files. These algorithms will be implemented by structuring a database aligned with the dataset. Subsequently, the dataset undergoes analysis and design procedures.

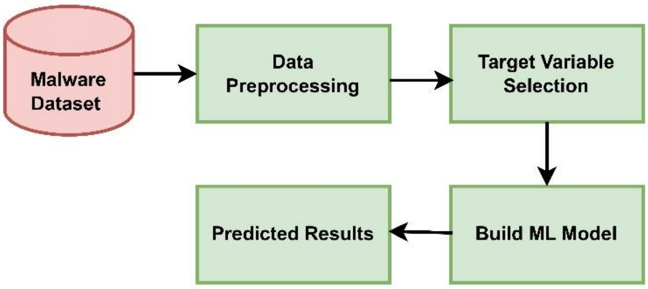

Figure 1: Block diagram of Malware detection

A. *Dataset Description*

Microsoft published a dataset of approximately 500 gigabytes for the Kaggle Microsoft Malware Classification Challenge (2015) [14], consisting of 21,653 assembly codes representing malware. We obtained the malware dataset from Kaggle Microsoft and gathered 7,212 benign programs specifically for the Windows platform (verified using virustotal.com) from our college's laboratory [12]. During our experimentation, we identified a scalability issue associated with dataset growth, causing increased time complexity, heightened storage requirements, and decreased system performance. To address these concerns, reducing the dataset becomes imperative.

To tackle the challenges, we considered two approaches for data reduction: Instance Selection (IS) and Feature Selection (FS). In our method, Instance Selection (IS) aims to diminish the number of instances (rows) within the dataset by identifying the most suitable instances. Conversely, Feature Selection focuses on selecting the most pertinent attributes (features) within the dataset. Both approaches are highly effective in data reduction as they filter and eliminate noisy data, leading to reduced storage requirements, improved time complexity, and enhanced classifier accuracy [13]-[14].

B. *Dataset Preprocessing*

In our examination of the malware dataset, we observed that any benign assembly surpassing the size of 147.0 MB was excluded from our analysis. Based on prior research, we identified 1808 unique opcodes [15], consequently utilizing these as features for machine learning in our approach. We proceeded by computing the frequency of each opcode within each malware and benign file, followed by calculating the total opcodes weight in each file type. We observed that 91.3% of malware files and 66% of benign files contained opcodes weight below 40000. To maintain the malware-to-benign ratio, we selected all files under 40000 weight. Following this filtration, 19,771 malware and 4,762 benign files remained for analysis.

The subsequent stage involved the removal of noisy data from the malware. We categorized the malware and benign files into intervals of 500 according to opcodes weight. If an interval lacked benign files, we removed malware files within that range. This process was repeated for intervals of 100, 50, 10, and 2 opcodes weights to eliminate malware noise. Eventually, the dataset was refined to include 6,010 malware and 4,573 benign files.

C. *Build ML Model*

Random Forest: Random Forest is a powerful machine learning algorithm used for classification and regression tasks. It operates by constructing a multitude of decision trees during training. Each tree in the forest is built on a random subset of the dataset and uses a random subset of features, adding randomness to prevent overfitting and enhance accuracy [16]. When making predictions, the Random Forest aggregates the predictions of individual trees to arrive at the final output. This ensemble technique is robust, versatile, and effective for handling large datasets with high dimensionality. It can handle missing values, maintain accuracy even with unbalanced data, and provide insights into feature importance [17].

Decision Tree: A Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. It operates by recursively partitioning the dataset into subsets based on the most discriminative features, creating a tree-like structure of decisions [18]. At each node of the tree, the algorithm selects the feature that best separates the data into distinct classes or groups. This process continues until a stopping criterion, such as a predefined tree depth or a minimum number of samples per leaf node, is met. Decision Trees are intuitive and easily interpretable, making them useful for understanding feature importance and how the model arrives at specific predictions [19]. However, they can be prone to overfitting, especially when the tree becomes too complex, which can be mitigated using techniques.

KNN: K-Nearest Neighbors (KNN) is a non-parametric supervised learning algorithm used for both classification and regression tasks. It works on the principle of similarity, where it classifies or predicts the value of a new data point based on its proximity to other known data points in the feature space [20]. The algorithm stores the entire training dataset and, when predicting for a new instance, identifies the K nearest neighbors to the new point using a distance metric (commonly Euclidean distance). The predicted class or value is determined by majority voting (for classification) or averaging (for regression) among the K nearest neighbors [21]. KNN is simple to implement and doesn't require model

DOI: 10.5281/zenodo.10597681

training, making it particularly useful for small to medium-sized datasets. However, it can be sensitive to the choice of K, requires a distance metric suitable for the data, and becomes computationally expensive with larger datasets due to its reliance on storing all training data. Additionally, handling categorical features or imbalanced datasets might require additional preprocessing for effective use with KNN.

IV. Result Analysis

In this study, the evaluation of performance is determined using various metrics: Accuracy, Precision, Recall, and F1-Score.

$$Accuracy = \frac{T\_P + T\_N}{T\_P + T\_N + F\_P + F\_N} \quad (1)$$

$$Precision = \frac{T\_P}{T\_P + F\_P} \quad (2)$$

$$Recall = \frac{T\_P}{T\_P + F\_N} \quad (3)$$

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

Table 1: Performance of ML Classifiers

Classifiers	Accuracy	Precision	Recall	F1-Score
RF	99.12	98.22	99.21	95.00
DT	98.59	97.00	98.00	96.00
KNN	97.56	97.00	96.00	95.00

Table 1 shows the performance of ML classifiers. It is clearly shows that random forest classifier gives better prediction accuracy of 99.12% as compared to other two classifiers.

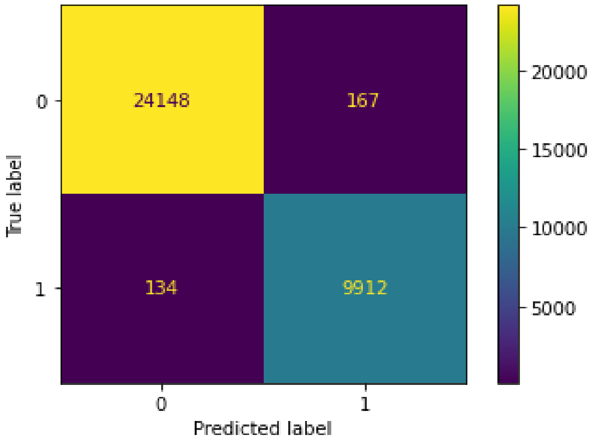

Figure 2: Confusion Matrix of RF Classifier

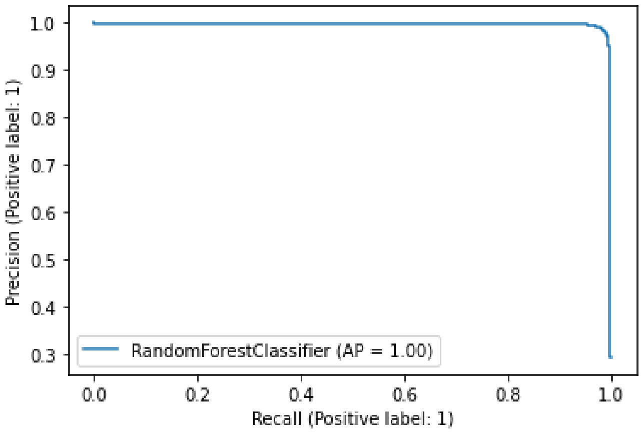

Figure 2: Accuracy Prediction of Recall Vs Precision of RF

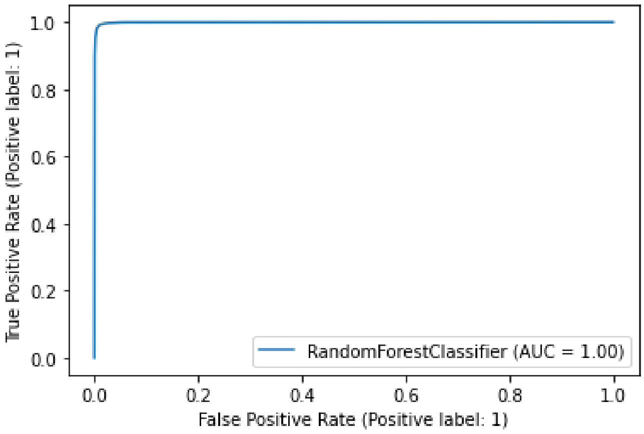

Figure 3: AUC of RF

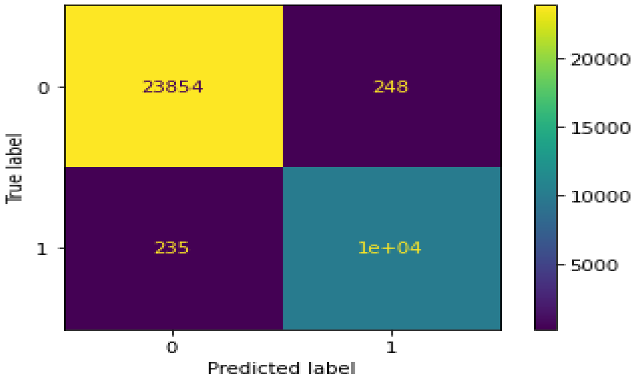

Figure 4: Confusion Matrix of Decision Tree

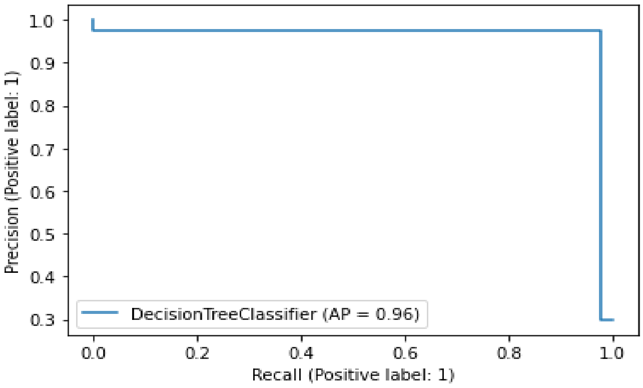

Figure 5: Accuracy Prediction of Recall Vs Precision of Decision Tree

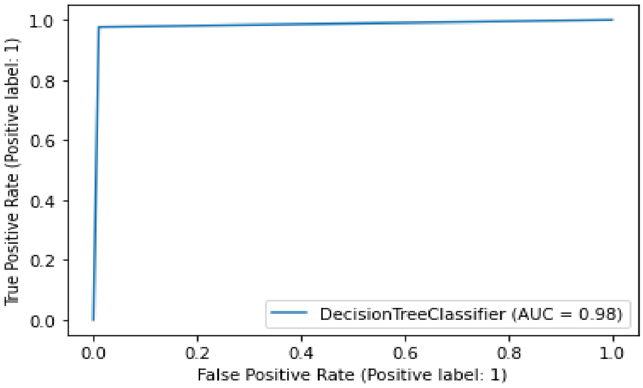

Figure 6: AUC of Decision Tree

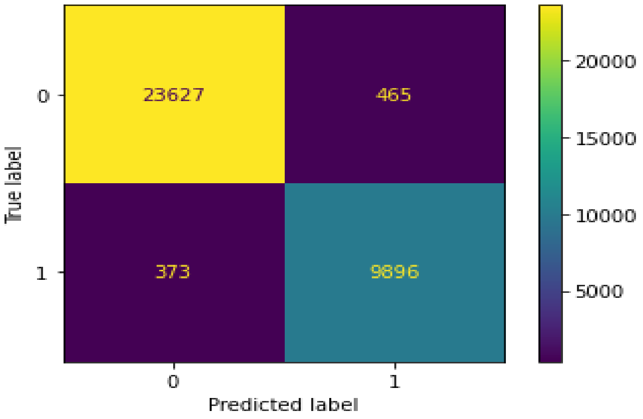

Figure 7: Confusion Matrix of KNN

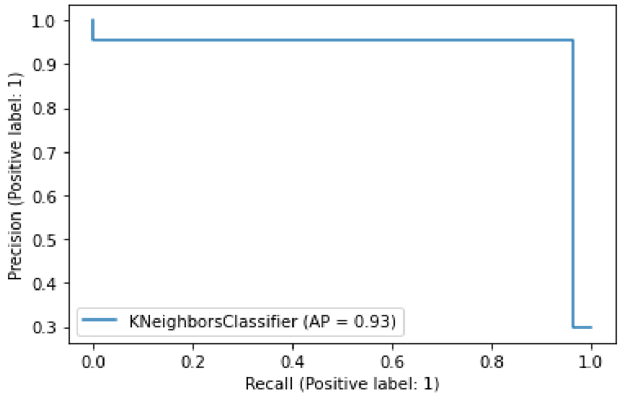

Figure 8: Accuracy Prediction of Recall Vs Precision KNN

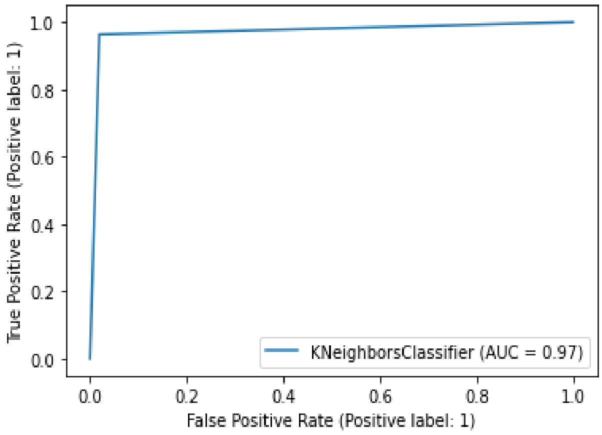

Figure 9: AUC of KNN

CONCLUSION

Detecting and eliminating malware from a system is crucial for overall system protection. This research primarily aims to employ machine learning techniques for identifying malware within the system. When applied meticulously, malware detection with stringent constraints should yield a zero false positive rate. However, the obtained results reveal that while the goals are closely approached, a non-zero false positive rate persists. This framework has led to a specific system segment being considered a competitive commercial product with varied deterministic expectation mechanisms. Consequently, employing machine learning methods significantly simplifies the process of identifying malware within files. Moreover, Decision Tree and Random Forest algorithms are highlighted as superior in efficiently detecting malware from extensive datasets. Enhancing these algorithms could potentially yield the desired outcomes in malware detection.

REFERENCES

[1] W. Hardy, L. Chen, S. Hou, Y. Ye and X. Li, "DL4MD: A deep learning framework for intelligent malware detection", International Conference on Data Mining (DMIN), 2016.

[2] A. Shalaginov, S. Banin, A. Dehghantanha and K. Franke, "Machine learning aided static malware analysis: A survey and tutorial", Cyber Threat Intelligence, pp. 7-45, 2018.

[3] S. Anderson and P. Roth. “EMBER: An Open Dataset for Training Static PE Malware”, arXiv, 1804.04637,2018.

[4] Nikam, U.V.; Deshmuh, V.M. Performance evaluation of machine learning classifiers in malware detection. In Proceedings of the 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), Ballari, India, 23–24 April 2022; pp. 1–5.

[5] Sethi, K.; Kumar, R.; Sethi, L.; Bera, P.; Patra, P.K. A novel machine learning based malware detection and classification framework. In Proceedings of the 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), Oxford, UK, 3–4 June 2019; pp. 1–13

[6] Abdulbasit, A.; Darem, F.A.G.; Al-Hashmi, A.A.; Abawajy, J.H.; Alanazi, S.M.; Al-Rezami, A.Y. An adaptive behavioral-based increamental batch learning malware variants detection model using concept drift detection and sequential deep learning. IEEE Access 2021, 9, 97180–97196.

[7] Sharma, S.; Krishna, C.R.; Sahay, S.K. Detection of advanced malware by machine learning techniques. In Proceedings of the SoCTA 2017, Jhansi, India, 22–24 December 2017.

[8] Chandrakala, D.; Sait, A.; Kiruthika, J.; Nivetha, R. Detection and classification of malware. In Proceedings of the 2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 8–9 October 2021; pp. 1–3.

[9] Gibert, D.; Mateu, C.; Planes, J.; Vicens, R. Using convolutional neural networks for classification of malware represented as images. J. Comput. Virol. Hacking Tech. 2019, 15, 15–28.

[10] Tahtaci, B.; Canbay, B. Android Malware Detection Using Machine Learning. In Proceedings of the 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), Istanbul, Turkey, 15–17 October 2020; pp. 1–6.

[11] Patil, R.; Deng, W. Malware Analysis using Machine Learning and Deep Learning techniques. In Proceedings of the 2020 SoutheastCon, Raleigh, NC, USA, 28–29 March 2020; pp. 1–7.

[12] Sethi, K.; Chaudhary, S.K.; Tripathy, B.K.; Bera, P. A novel malware analysis for malware detection and classification using machine learning algorithms. In Proceedings of the 10th International Conference on Security of Information and Networks, Jaipur, India, 13–15 October 2017; pp. 107–113.

[13] Hamid, F. Enhancing malware detection with static analysis using machine learning. Int. J. Res. Appl. Sci. Eng. Technol. 2019, 7, 38–42.

[14] Firdausi, I.; Lim, C.; Erwin, A.; Nugroho, A. Analysis of machine learning techniques used in behavior-based malware detection. In Proceedings of the 2010 Second International Conference on Advances in Computing, Control, and Telecommunication Technologies, Jakarta, Indonesia, 2–3 December 2010; pp. 201–203

[15] Saad, S.; Briguglio, W.; Elmiligi, H. The curious case of machine learning in malware detection. arXiv 2019, arXiv:1905.07573.

[16] Y. Dai, H. Li, Y. Qian, and X. Lu, “A malware classification method based on memory dump grayscale image,” Digital Investigation, vol. 27, pp. 30–37, 2018.

[17] J. Abawajy, A. Darem, and A. A. Alhashmi, “Feature subset selection for malware detection in smart IoT platforms,” Sensors, vol. 21, no. 4, p. 1374, 2021.

[18] Baset M. Master’s Thesis. Heriot-Watt University; Edinburgh, Scotland: 2016. Machine Learning for Malware Detection.

[19] Baghirov E. Techniques of Malware Detection: Research Review; Proceedings of the 2021 IEEE 15th International Conference on Application of Information and Communication Technologies (AICT); Baku, Azerbaijan. 13–15 October 2021; pp. 1–6.

[20] Akhtar M.S., Feng T. Malware Analysis and Detection Using Machine Learning Algorithms. Symmetry. 2022;14:2304. doi: 10.3390/sym14112304.

[21] Muppalaneni N., Patgiri R. Malware Detection Using Machine Learning Approach; Proceedings of the International Conference on Big Data, Machine Learning and Applications; Vancouver, BC, Canada. 29–30 May 2021; Singapore: Springer; 2021.

DOI: 10.5281/zenodo.10597681