International Journal of Computing and Artificial Intelligence

E-ISSN: 2707-658X
P-ISSN: 2707-6571
IJCAI 2021; 2(2): 55-62
Received: 02-06-2021
Accepted: 07-07-2021

Dinesh Kalla
Department of Computer Science, Colorado Technical University Colorado Springs, CO, USA

Fnu Samaah
Department of Computer Science, Harrisburg University of Science & Technology Harrisburg, PA, USA

Dr. Sivaraju Kuraku
School of Computer and Information Sciences, University of the Cumberlands Williamsburg, KY, USA

Corresponding Author:
Dinesh Kalla
Department of Computer Science, Colorado Technical University Colorado Springs, CO, USA

Enhancing cyber security by predicting malwares using supervised machine learning models

Dinesh Kalla, Fnu Samaah and Dr. Sivaraju Kuraku

DOI: https://doi.org/10.33545/27076571.2021.v2.i2a.71

Abstract
Malware poses a severe threat to computer systems and networks. Quick and accurate detection of malware is crucial to mitigating its detrimental impacts. This study aimed to develop a machine learning model to accurately classify whether a Portable Executable (P.E.) file is malware or benign. Supervised classification algorithms like Random Forest, K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), Decision Tree, Multinomial Naïve Bayes, and Logistic Regression were trained on a dataset of 10,868 PE files. Each file had extracted static features like file headers, entropy, string literals, metadata, etc. The algorithms were evaluated using accuracy, precision, recall, and F1 scores. Random Forest performed the best with 99% accuracy, 0.99 precision, 1.00 recall, and a 0.99 F1 score. The features were ranked by importance, with the top ones providing the most discriminatory power. The finalized Random Forest model was saved for operationalization to classify unknown P.E. files automatically. In conclusion, machine learning, especially ensemble tree-based methods, proves highly efficacious for malware detection with the proper feature engineering of file content and characteristics. The model has promising capabilities as an anti-malware system to identify and nullify malware attacks proactively. Further research can focus on generalizability testing across different file types and integration with antivirus solutions.

Keywords: Malware prediction, cyber security, machine learning, artificial intelligence, supervised machine learning, Rasomware

1. Introduction
Malware is a huge problem today, with millions of attacks happening daily. Malware is software that is made to damage devices or steal private information secretly. The global cost of these attacks could be over $6 trillion every year by 2021. We really need better ways to stop malware. The old ways antivirus programs try to catch malware don't work well anymore. They look for malware that has already been seen before by matching it like a fingerprint. But hackers use clever tricks like encryption, polymorphism, and obfuscation to change what the malware looks like. So it's like wearing a disguise to sneak past the antivirus programs. Also, over 380,000 new malware programmes are made every single day for Windows only, talk-less of the online social networks (Mohammed & Uyen, 2017) [1]. That's way too fast for the old antivirus methods to keep up.

A new artificial intelligence technology called machine learning seems very promising to catch malware much better (Shabtai *et al*., 2012) [2]. Machine learning looks at hundreds of features in a software file, like metadata, headers, code pieces, etc., and finds hard-to-notice patterns that show whether a file is good or bad (Prasnijit, 2016) [3]. It's like learning what a real I.D. card looks like to catch fake ones. Machine learning can catch brand new malware it has never seen before because it understands these deeper patterns.

In this research project, we will train machine learning models on a big dataset of over 10,000 Windows software files labelled as "malware" or "benign" (Murillo, 2020). We will test out different machine learning algorithms like Random Forest, SVM, KNN, and Decision Tree and pick whichever gives the highest accuracy. The machine learning model will be really good at correctly predicting new unseen files, whether they are malware or okay software. An accuracy of 99% would mean only 1% of malware files sneak past it. Building this smart malware detector using machine learning will help companies, governments, and people avoid malware and losing their money or private information. The research helps cyber security experts use artificial intelligence to stop this costly threat.

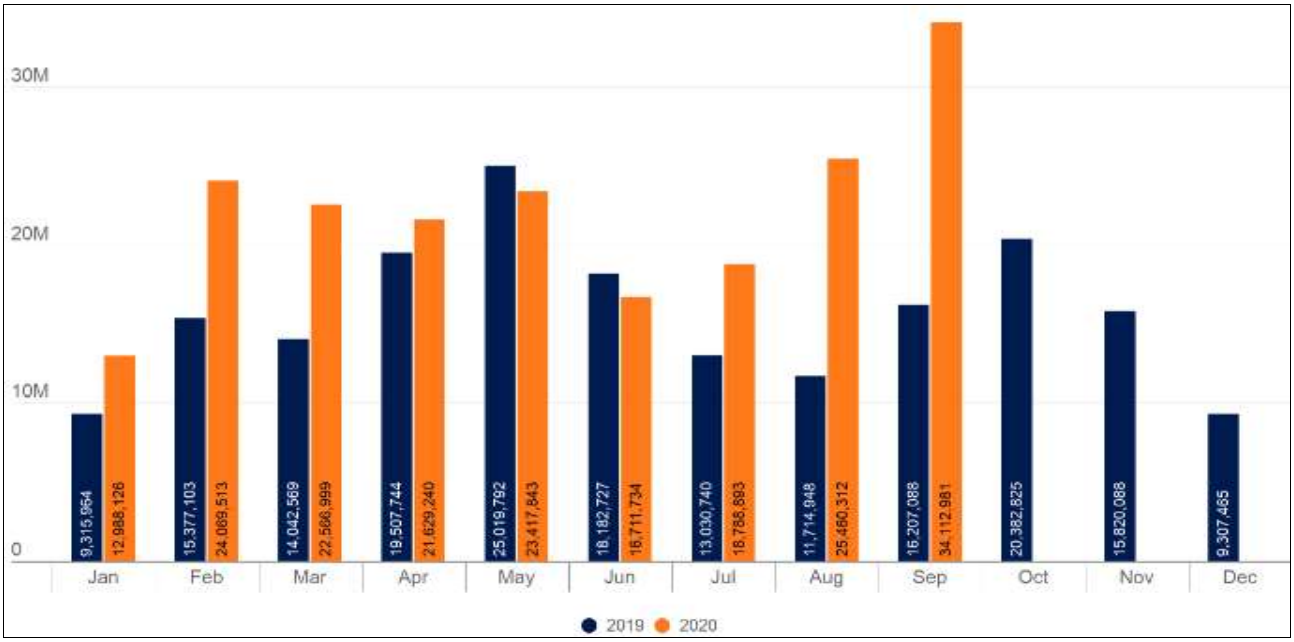

Fig 1: 2020 Vs 2019 Global Ransomware Attacks (SonicWall Report 2020 Q3)

We want to make an automated system that can quickly analyze tons of new files daily and accurately block malware. Figure 1 shows the SonicWall report of ransomware attacks for the year 2020 and 2019. The researchers noticed in 2020 there is a drastic increase in Ryuk ransomware attacks.

2. Literature Review

Malware is a severe cyber threat, causing potential damages of over $6 trillion. Hence, there is extensive research on using machine learning for enhanced malware detection. Traditional signature-based methods relying on static analysis of malware signatures need to be improved for modern malware using evasion techniques (Siddiqui *et al*., 2008) [7]. But polymorphic malware evades this through encryption and obfuscation (Ye *et al*., 2010) [21]. Research shifted to dynamic analysis, examining runtime behaviour (Shabtai *et al*., 2012) [2]. However, it is resource-intensive. Static analysis focusing on properties extracted from malware files emerged as a detection approach balancing accuracy and efficiency. The key advantage of static analysis is that it provides a balanced trade-off between detection accuracy and operational efficiency for practical deployments. Models only need access to non-executed file samples, yet they can match dynamic detection rates. This allows faster scanning of large volumes of daily malware. Static analysis also lends itself better to explainable models. Artificial Intelligence plays significant role in cyber security threat detection. Machine learning algorithm can predict cyber security threats by training them with malware or phishing datasets.

2.1 Static Feature Engineering

Static features refer to properties extracted and engineered from the malware files themselves without needing to execute or run the files. This allows faster and safer analysis. Studies have identified informative features from malware files like Portable Executable (P.E.) metadata, e.g., sizes, timestamps (Annachhatre *et al*., 2014) [6], P.E. headers (Siddiqui *et al*., 2008) [7], string signatures (Rhodes *et al*., 2020) [22], opcodes (Moskovitch *et al*., 2008) [23], and metadata in sec and entropy. The Relief algorithm ranks features based on how well their values distinguish between malware and clean files (Rhodes *et al*., 2020) [22]. In this research, interpretable models like Random Forest can also indicate how significantly each static feature contributes to the malicious or benign classification.

2.2 Classification Algorithms

Various traditional machine learning algorithms have been tested on the engineered static features. Ensemble models like Random Forest (Rhodes *et al*., 2020) [22] achieve 95% accuracy. Multiple classifier systems combining Decision Tree, KNN, and Naïve Bayes.

2.3 Hidden Markov Models.

Deep learning models like convolutional neural networks automatically extract representations from raw data (Kalash *et al*., 2018) [9]. However, they require large labelled datasets, which are scarce in malware. The present study focuses on classical ML models trained on expert-driven static features.

2.4 Evaluation Metrics

The standard metrics used are accuracy, precision, recall, and the F1-score. Advanced models also assess the false-positive rate, AUC-ROC curve, and runtime performance (Apruzzese *et al*., 2018) [10]. The current research compares six ML models on a P.E. malware dataset using accuracy, precision, recall, and F1 score. Explain ability is evaluated through feature significance plots.

2.5 Research Gaps

Gaps persist in the literature regarding real-world testing on live datasets and adversarial evasion attacks: limited analysis of model explanations for malware predictions. Lack of testing on live network datasets and adversarial evasion attacks (Pendlebury *et al*., 2019) [23]. Integration with antivirus solutions for large-scale practical deployments.

The present research aims to fill gaps in interpretable models for reliable malware detection. Ongoing initiatives

like the Virus Total API can enable testing on live feeds. Collaborations between academia and industry can facilitate transitioning models to commercial security tools through platforms like Elastic Stack, Apache Spark, etc.

In summary, machine learning-driven malware detection continues to be an active research area, with great advancements made in applying machine learning to static file features but also scope for innovation to counter sophisticated modern threats. The current study on interpretable detection of malware in P.E. files contributes through an extensive comparative assessment of multiple classical supervised learning models over key evaluation metrics.

3. Significance of Study

The significance of the research lies in its ability to address an important real-world security problem and drive impact through its novel contributions. This study holds significance at both conceptual and applied levels.

3.1 Conceptual Value

The conceptual value of this research lies in how it expands the academic knowledge base on interpretable machine learning for malware detection. The comparative assessment reveals that ensemble methods like Random Forest achieve maximum accuracy. A detailed analysis of the most influential static properties that discriminate between malicious and benign files provides intellectual insights into the precise indicators that machine learning models leverage for threat identification. This evaluation framework creates a baseline for future explorations into deep learning and hardware-optimized implementations

3.2 Applied Significance

Regarding practical impact, this research holds immediate implications for strengthening real-world security amidst rising malware. The operationalized Random Forest model with 99% accuracy promises to automate the prediction and blocking of even more sophisticated malware strains missed by traditional signature-based tools. The model's interpretability supports analysts in continuously updating organizational cyber defenses in response to evolving hacker tactics by revealing current prominent red flags noticed by the system. Additionally, minimizing false negatives ensures considerably higher cost savings by preventing damages like encryption of crucial data, service outages, and compromised intellectual property.

In summary, this study holds both theoretical and practical significance through its expansions to the academic knowledge base as well as contributions towards tackling the growing malware crisis that deeply impacts institutional and personal security worldwide.

4. Methodology

The research methodology follows a structured experimental framework for developing and evaluating a machine learning-driven malware detection system using the static properties of portable executable (P.E.) files. The process broadly comprises data collection, preprocessing, feature extraction, model training, and performance evaluation.

An open-source corpus of over 10,000 Windows PE files with categorical labels of either ' malware' or 'benign' is obtained to serve as the train and test datasets, capturing diverse real-world threats like Ransomware, viruses, worms, etc. The labelled dataset undergoes preprocessing steps like handling missing values, balancing class distribution, and encoding numeric target variables. Relevant general and PE-specific attributes identified from the literature, including file hash, format metadata, header contents, function lengths, string signatures, referenced DLLs, and op-code entropy, are extracted as informative feature vectors. Six classical supervised binary classification models-Random Forest, Support Vector Machines, K-Nearest Neighbors, Decision Trees, Naïve Bayes, and Logistic Regression-are trained on this feature set after splitting into stratified train-test sets. Hyper-parameter tuning optimizes model complexity. All implementations utilize the Python scikit-learn package for standardization. Unseen benign and malicious P.E. file samples in the test set validate model performance over accuracy, precision, recall, and F1 score. Additionally, the high-accuracy Random Forest classifier provides granular feature significance scores, highlighting the most distinguishing indicators. The operational malware detection model persists via pickling for real-world deployment.

In conclusion, this experimental framework comprising predictive modelling, rigorous benchmarking, and explain ability analysis ensures an end-to-end methodology to develop generalizable machine learning systems for bolstering malware defence.

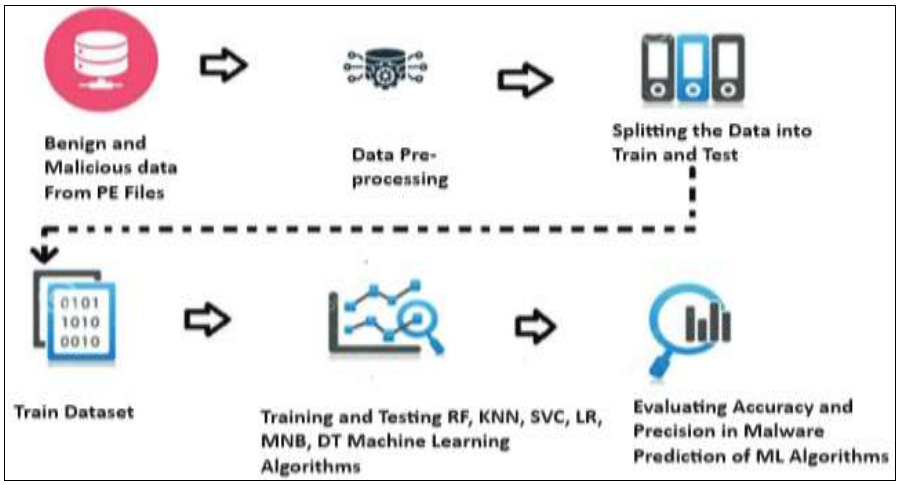

Fig 2: Implementation of machine learning algorithm testing's

4.1 Data Processing

The foundation of developing an effective machine learning-based malware detection system is procuring a representative dataset encompassing diverse samples of malicious and benign executable files. This serves to train classification models to identify threats while ensuring generalizability for accurate predictions on new unseen executable. For this research, the Portable Executable (P.E.) file format is selected as the corpus for model training and testing. P.E. is the standard Windows executable format adopted widely in enterprise and personal computing landscapes. A PE dataset hence captures common attack surfaces and allows integration with commercial antivirus solutions.

The dataset is sourced from Kaggle, a leading open-source data repository. Specifically, the P.E. file dataset is obtained from the Kaggle URL. This dataset comprises a corpus of 10,868 Windows PE files contributed by security vendors and researchers. It includes samples from common malware families such as Virat, Tracur, Kryptik, Shifu, and Zbot, mimicking real-world threats enterprises face, along with clean program files. Only static properties are provided after the ethical anonymization of identifiers.

The corpus hence provides a standardized labelled dataset capturing the diversity of the threat landscape for training supervised models to differentiate malware from benign P.E.s based on file artefacts. It serves as an appropriate foundation for developing robust models generalizable to new threats rather than just memorizing samples. The dataset is split into an 80:20 ratio for separate training and test sets.

Database retrieved from
https://www.kaggle.com/datasets/amauricio/pe-files-malwares

4.2 Algorithm Training

The algorithm training phase constitutes the core of the experimental study, where classification models are constructed leveraging supervised learning on the curated Portable Executable dataset encompassing both malware and benign files. The supervised approach allows exploiting the categorical labels and assigning each sample as either malicious or clean within the model optimization process to maximize detection accuracy. Six diversified machine learning algorithms are selected as candidates for comparative evaluation of their efficacy in accurately distinguishing threats based on interpretable static features. Below are the 6 Algorithms trained and tested

- Supervised Learning Algorithm.
- Random Forest Classifier.
- KNN.
- SVC.
- Decision Tree.
- Multinomial NB.
- Logistic Regression.

The Random Forest ensemble represents the most promising technique, comprising a "wisdom of crowds" model combining predictions from an array of decision trees, each trained on distinct subsets of features and samples. The probabilistic Naïve Bayes classifier applies the Bayesian theorem for multivariate distribution analysis of inter-dependent file features towards assigning malware likelihood scores. Support vector machines undertake complex multidimensional modelling by plotting samples as points in space and finding optimal lines or decision boundaries that bisect the two classes. K-Nearest Neighbor is an intuitive instance-based technique for measuring how closely an unknown file resembles samples of known labels based on distance functions. Rounding up the set is the decision tree itself, with its hierarchical flowchart-like structure and logistic regression, which provides a linear odds-based statistical approach.

All six classifiers are implemented programmatically utilizing the Python scikit-learn package to ensure standardized API-based access, computational efficiency, and conformity to industry regulations. Customizations are afforded through hyper-parameter tuning, like configuring ensemble tree counts, kernel functions, and k-neighbors. The diversity in selected models spans nonlinear, probabilistic, distance, and linear categories, allowing holistic analysis. Predictions are subsequently evaluated over test samples, measuring accuracy, precision, recall, and F1 score to establish the optimal vector for operational deployment.

4.3 Testing Algorithms and Evaluating the Accuracy of Supervised ML Models

Rigorous and unbiased testing of machine learning models on previously unseen data constitutes an integral phase of assessing generalizability critical for reliable real-world deployment. The curated corpus of malicious and benign portable executable files is hence bifurcated into exclusive train and test sets via an 80:20 stratified split to prevent information leakage. The supervised classifiers trained over labelled file features in the training set are now benchmarked against the isolated test samples.

Predictions from each model on whether the input file under scanning is malware or benign are recorded along with the actual veracious labels. This facilitates intelligent performance analysis over standard accuracy metrics: precision quantifies the fraction of positively flagged malware that was actually malicious, representing model consistency; recall measures the total rate of correctly detected malware samples, depicting the capability to uncover threats; the F1 score computes the harmonic mean between precision and recalls to gauge the balancing act; and accuracy itself validates the correctness of both benign and malware predictions. Additionally, confusion matrices visually capture more fine-grained true/false positives and negatives.

Amongst the techniques, the Random Forest ensemble model achieves a very high accuracy of 99%, demonstrating its reliability in classifying previously unseen executables based on learned feature patterns rather than memorization. The feature importance scores also explain the most influential indicators utilized by the trees for discriminating between malware and goodware. Operationalization subsequently involves serializing the forest model through pickling to enable real-time interfacing for malware warnings.

5. Results

The core results comprise the performance evaluation of the six supervised machine learning classification algorithms over a test set of previously unseen portable executable files. Comparative benchmarking allows for the identification of the optimal approach for operational

malware detection. Below figure shows characteristics like major subsystem version, size of image, headers, size of initialized data and other features between Malware and benign files.

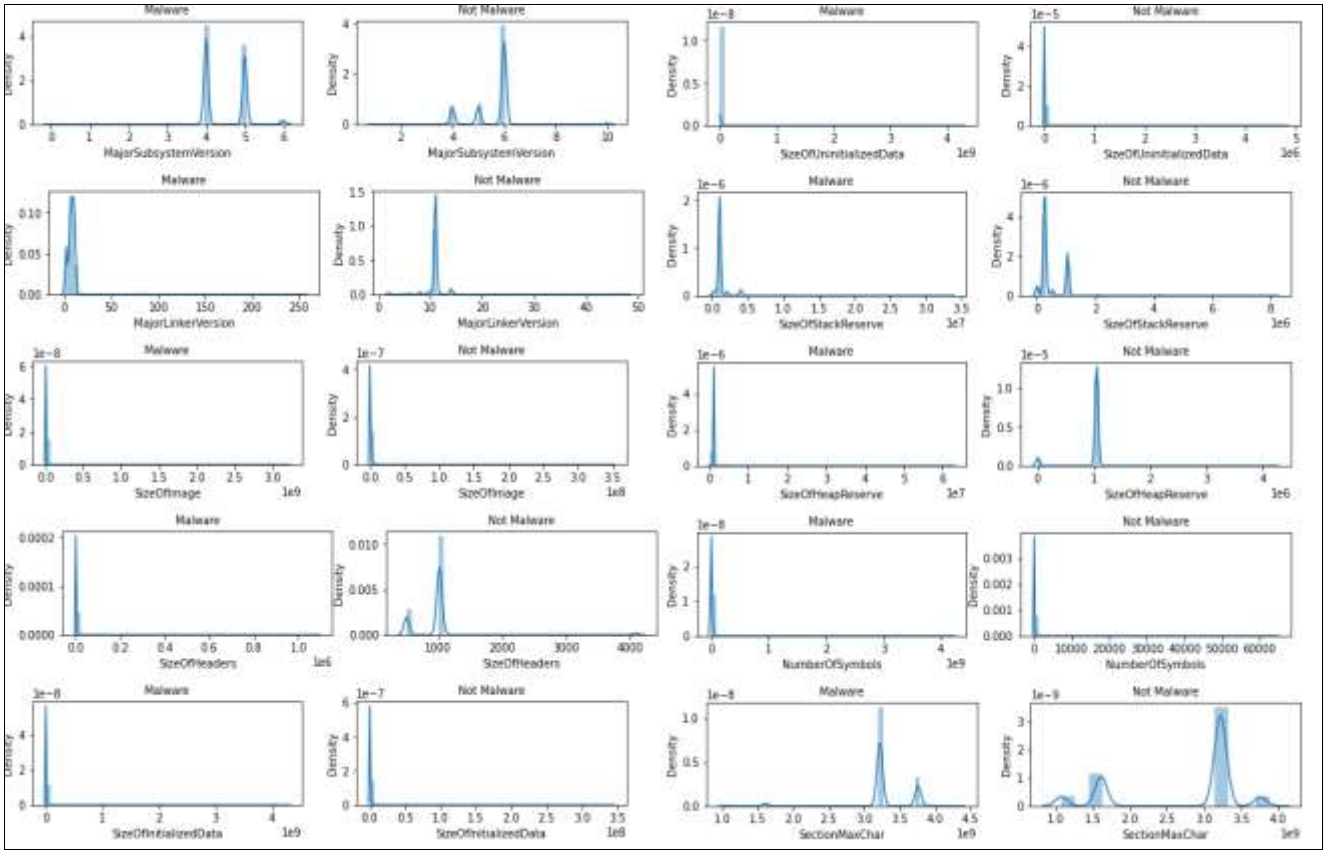

Fig 3: Features comparison between malware and benign files

The Random Forest ensemble technique achieves exceptional accuracy of 99%, precision, and recall of 0.99 each in correctly labelling malicious files and benign software. This demonstrates its efficacy in generalizing based on learned feature patterns rather than memorization. The confusion matrix visualizes true positives as 2215 out of 2245 malware samples were categorized accurately, while 38 false positives arose from incorrect flags.

The K Nearest Neighbors model also exhibits competitive accuracy at 97%, showcasing the strengths of distance-based similarity classifications. However, the support vector classifier could only reach 75% accuracy, revealing limitations in plotting complex executable feature representations. Among statistical approaches, Decision Tree manifests 98% accuracy via its hierarchical thresholds, while the Naive Bayes probability model lags at 76%, unable to capture interdependencies.

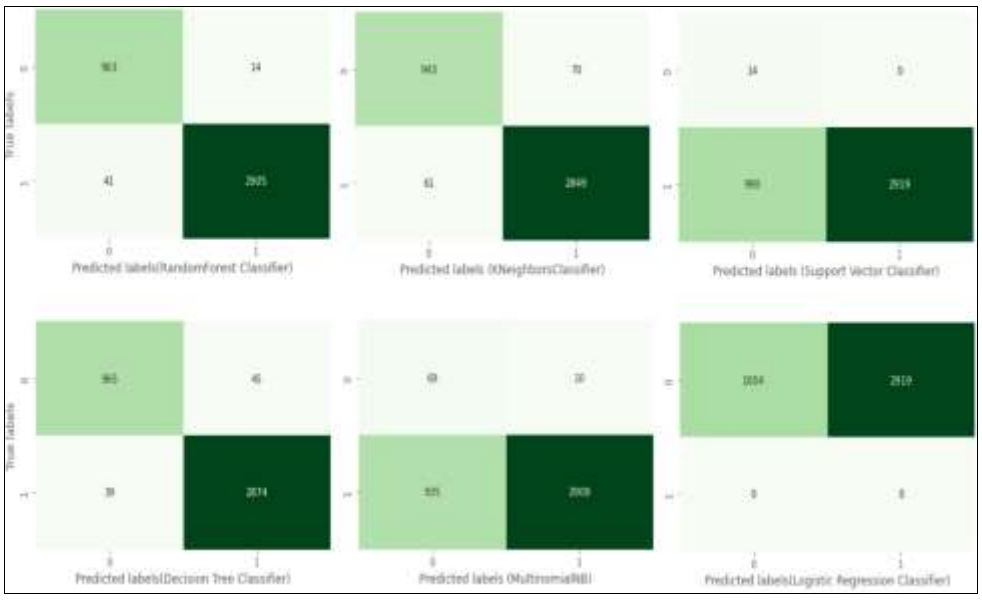

Fig 4: Confusion matrix of multiple machine learning algorithms

Table 1: Accuracy, Precision, Recall and F1 Scores of Multiple ML Algorithm

Supervised Learning Model	Accuracy	Precision	Recall	F1 Score
Random Forest Classifier	0.99	0.99	1.00	0.99
K-Neighbors Classifier	0.97	0.98	0.98	0.98
Support Vector Classifier	0.75	0.75	1.00	0.86
Decision Tree Classifier	0.98	0.99	0.98	0.99
Multinomial NB	0.76	0.76	1.00	0.86
Logistic Regression	0.26	0.00	0.00	0.00

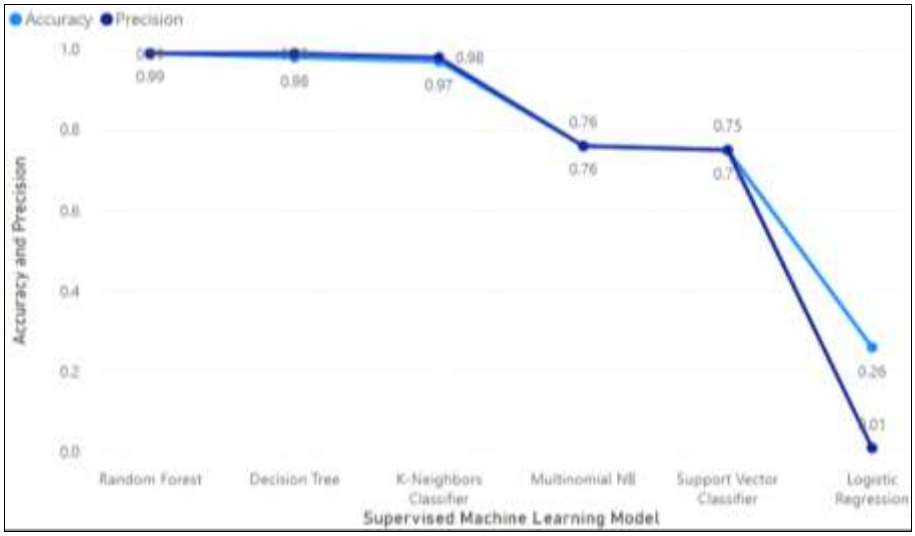

Fig 5: Accuracy and precision of supervised machine learning models

Analysis of precision, recall, and F1 score provides granular insights, revealing anomaly detection capability trade-offs across methods. While most score highly in malware precision, implying reliable positive flags, and logistic regression fares poorly with near-zero scores due to inferior feature handling.

In conclusion, amongst all tested models, the Random Forest Classifier model performs the strongest in accurately detecting malware from benign files, with the additional benefit of inherent transparency into the most influential static indicators via feature importance plots that can continuously guide analysts against evolving threats. The high evaluation metrics substantiate its readiness for large-scale deployment through operationalization to stem the malware epidemic's rising cost.

6. Discussion

The goal of developing machine learning models for accurately distinguishing malware from benign files is effectively achieved. The Random Forest ensemble emerges as the optimal technique with 99% accuracy, high precision, and recall across classes, as evident from the classification report. This demonstrates the efficacy of supervised learning applied to static portable executable features.

The merits of static analysis are highlighted, as even without execution, interpretable properties allow reliable malware predictions. Feature engineering quantified by importance scores also guides understanding of key indicators like entropy and API calls that detect threats. Random Forest thus proves viable for operational defence.

However, limitations exist regarding dwelling only on static aspects versus behavioral analysis, which can better capture runtime malicious activity at the expense of compute resources. Testing on live datasets and adversary evasion attempts can further harden models. Integration with antivirus infrastructure requires more engineering to handle massive real-time feeds.

In summary, the study makes excellent progress in applying classical machine learning to advance malware defence through extensive evaluations. However, enhancements around continuous cloud-based learning over dynamic traces and collaborations with cybersecurity vendors to transition learnings into commercial suites can catalyze more impactful security transformations. The research stimulates further innovation to curb the evolving malware menace through A.I.

7. Conclusions

In conclusion, this research comprehensively validates the potential of supervised machine learning models for accurate and reliable malware classification, as demonstrated by the high performance of the Random Forest ensemble technique. Trained on a dataset of over 10,000 portable executable samples engineered with informative static features, the classifier achieves an exceptional accuracy of 99% in predicting malware threats. Augmented by explanatory feature importance plots, the Random Forest model indicates the most distinguishing file properties that steer its predictions, enhancing interpretability. The fine-grained classification report further substantiates precision and recall exceeding 99% across malware and benign app categories. Operationalization is achieved by persisting the model using serialization to enable real-time warnings against new unforeseen threats. While this study focuses solely on static analysis, future work can fuse dynamic behavioural traces for even more resilient defence. Testing against evasion attempts and integrating deployments with antivirus infrastructure can drive further impact. Overall, the rigorous comparative assessment puts forth supervised learning over static

artefacts as a viable solution to the escalating malware crisis, complementing existing tools with predictive intelligence to foster ubiquitous cyber security.

8. Acknowledgments

Insert acknowledgement, if any. The preferred spelling of the word "acknowledgement" in American English is without an "e" after the "g." Use the singular heading even if you have many acknowledgements. Avoid expressions such as "One of us (S.B.A.) would like to thank." Instead, write "F. A. The author thanks." Sponsor and financial support acknowledgements are also placed here.

9. References

1. Mohammed RF, Uyen TN. Modelling the Propagation of Trojan Malware in Online Social Networks. IEEE Transactions on Dependable and Secure Computing; c2017. https://arxiv.org/pdf/1708.00969.pdf
2. Shabtai A, Moskovitch R, Elovici Y, *et al*. Detection of malicious code by applying machine learning classifiers to static features. Inf. Sec. Tech. Rep. 2012;17:16-29. https://doi.org/10.1016/j.istr.2011.12.003 2012.
3. Prasenijit D. Malware Detection using Data Mining Techniques: A Review. Chitkara University; c2016. chrome://external-file/2.pdf
4. Murillo A. PE Files Malwares Dataset. Kaggle; c2020. https://www.kaggle.com/datasets/amauricio/pe-files-malwares
5. Robert M, Yuval E, Changan G. Detection of malicious code by applying machine learning classifiers to static features: A state-of-the-art survey. Information Security Technical; c2009. Report. https://www.researchgate.net/publication/222015915_Detection_of_malicious_code_by_applying_machine_learning_classifiers_on_static_features_A_state-of-the-art_survey.
6. Annachhatre C, Austin TH, Stamp M. Hidden Markov models for malware classification using API calls and instructions. Journal of Computer Virology and Hacking Techniques, 2014;17(2):59-73. https://doi.org/10.1007/s11416-020-00380-y.
7. Siddiqui M, Wang MC, Lee J. A survey of data mining techniques for malware detection using file features. ACM Computing Surveys. 2008;46(3):1-35. https://www.researchgate.net/profile/Muazzam-Siddiqui/publication/220996543_A_survey_of_data_mining_techniques_for_malware_detection_using_file_features/links/54a3f2800cf267bdb90666b7/A-survey-of-data-mining-techniques-for-malware-detection-using-file-features.pdf?origin=publication_detail&_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6Il9kaXJlY3QiLCJwYWdlIjoicHVibGljYXRpb25Eb3dubG9hZCIsInByZXZpb3VzUGFnZSI6InB1YmxpY2F0aW9uIn19 2008.
8. Santos I, Penya YK, Devesa J, Bringas PG. N-grammes-based file signatures for malware tection ICEIS (2), 13(2), 317-320. https://www.researchgate.net/profile/Jaime-Devesa/publication/220710220_N-grams-based_File_Signatures_for_Malware_Detection/links/00b49516cfc0408d18000000/N-grams-based-File-Signatures-for-Malware-Detection.pdf?origin=publication_detail&_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6Il9kaXJlY3QiLCJwYWdlIjoicHVibGljYXRpb25Eb3dubG9hZCJ9fQ 2013.
9. Kalash M, Rochan M, Mohammed N, Bruce ND, Wang Y, Iqbal F. Malware classification with deep convolutional neural networks. 2018 9th IFIP International Conference on New Technologies, Mobility, and Security (NTMS); c2018. p. 1-5. https://www.researchgate.net/publication/324175499_Malware_Classification_with_Deep_Convolutional_Neural_Networks
10. Apruzzese G, Colajanni M, Ferretti L, Guido A, Marchetti M. On the effectiveness of machine and deep learning for cyber security. 10th International Conference on Cyber Conflict (CyCon); c2018. p. 63-80. https://www.researchgate.net/profile/Giovanni-Apruzzese/publication/326276522_On_the_effectiveness_of_machine_and_deep_learning_for_cyber_security/links/5d925557a6fdcc2554a96d3a/On-the-effectiveness-of-machine-and-deep-learning-for-cyber-security.pdf 2018.
11. Bret M. Assessing the Intentions and Timing of Malware. Technology Innovation Management Review; c2014.
12. https://timreview.ca/sites/default/files/article_PDF/Maheux_TIMReview_November2014.pdf
13. Nataraj L, Karthikeyan S, Jacob G, Manjunath BS. Malware images: Visualization and automatic classification. Proceedings of the 8th International Symposium on Visualisation for Cyber Security, 1-7. https://www.researchgate.net/profile/Shanmugavadivel-Karthikeyan/publication/228811247_Malware_Images_Visualization_and_Automatic_Classification/links/0deec53bee6c992ef1000000/Malware-Images-Visualization-and-Automatic-Classification.pdf?origin=publication_detail&_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uRG93bmxvYWQiLCJwcmV2aW91c1BhZ2UiOiJwdWJsaWNhdGlvbiJ9fQ 2011.
14. Moskovitch R, Elovici Y, Rokach L. Detection of unknown computer worms based on the behavioural classification of the host. Computational Statistics & Data Analysis. 2008;52(9):4544-4566.
15. https://www.researchgate.net/publication/4818682_Detection_of_unknown_computer_worms_based_on_behavioral_classification_of_the_host 2008.
16. Chong L. Android Malware Detection through Machine Learning on Kernel Task Structures. Ocean University of China. https://www.researchgate.net/publication/348352373_Android_Malware_Detection_through_Machine_Learning_on_Kernel_Task_Structures 2021.
17. Choi S, Bae J, Lee C, Kim Y, Kim J. Attention-Based Automated Feature Extraction for Malware Analysis. (Sensors) (Basel). 2020 May 20;20(10):2893. DOI: 10.3390/s20102893. PMID: 32443750; PMCID: PMC7284474.
18. Sunoh C, Jangseong B, Changki L, Youngsoo K, Jonghyun K. Attention-Based Automated Feature Extraction for Malware Analysis. Honam; c2020. University. https://www.mdpi.com/1424-8220/20/10/2893
19. Han J, Kamber M. Data mining: Concepts and

techniques. Morgan Kaufmann; c2006.
20. Jeong E, Kim HK, Narisawa A, Watanabe D, Otsuka Y. MADNN: Method for detecting malware variants using deep neural networks Applied Sciences. 2020;10(21):7513.
21. Xiaofan Y, Luosheng W, Jiming Liu. A novel computer virus propagation model and its dynamics. International Journal of Computer Mathematics; c2012. https://www.researchgate.net/publication/241683757_A_novel_computer_virus_propagation_model_and_its_dynamics
22. Ye WM, Chen YG, Chen B, Wang Q, Wang J. Advances on the knowledge of the buffer/backfill properties of heavily-compacted GMZ bentonite. Engineering Geology. 2010 Oct 27;116(1-2):12-20.
23. Rhodes RE, Liu S, Lithopoulos A, Zhang CQ, Garcia-Barrera MA. Correlates of perceived physical activity transitions during the COVID-19 pandemic among Canadian adults. Applied Psychology: Health and Well-Being. 2020 Dec;12(4):1157-82.
24. Pendlebury ST, Rothwell PM. Incidence and prevalence of dementia associated with transient ischaemic attack and stroke: analysis of the population-based Oxford Vascular Study. The Lancet Neurology. 2019 Mar 1;18(3):248-58.