Malware Detection Using Machine Learning: A Comparative Analysis

Sameeruddin Mohammed, Fan Zhang, Faria Brishti
Department of Computer Science
Tuskegee University
Tuskegee, USA
e-mail: {smohammed8703 | fzhang9458 | fbrishti7995}@tuskegee.edu

Baiyun Chen
Computer Science Department
Tuskegee University
Tuskegee, USA
e-mail: bchen@tuskegee.edu

Fan Wu
Computer Science Department
Tuskegee University
Tuskegee, USA
e-mail: fwu@tuskegee.edu

***Abstract*—To address the growing challenges posed by Cyber threats, anti-malware organizations have increasingly turned to Machine Learning (ML). In recent years, machine learning algorithms have become indispensable for solving complex classification problems, outperforming traditional statistical methods by capturing intricate patterns in high dimensional data. However, selecting the optimal model requires rigorous evaluation in multiple performance metrics while ensuring stability across different data splits. In this study, we conducted a comprehensive assessment of eight machine learning algorithms. Random Forest (RF), Extreme Gradient Boosting (XGBoost), Support Vector Machine (SVM), Logistic Regression (LR), Naive Bayes, Light Gradient Boosting Machine (LightGBM), Decision Tree (DT), and k-Nearest Neighbors (KNN) using stratified 5-fold cross-validation. Our results reveal that RF, LightGBM, DT, and KNN achieve exceptional performance, with identical near-perfect scores in accuracy (0.9918), precision (0.9920), recall (0.9918), F1 score (0.9918) and Area Under the Receiver Operation Characteristic Curve (AUC-ROC) (0.9998), along with remarkably low variance (10^{-6} to 10^{-8}), demonstrating unparalleled robustness. The study highlights the superiority of tree-based ensembles and KNN in achieving high predictive power and stability, whereas classical algorithms such as logistic regression and naive Bayes lag. Despite XGBoost's reputation, its performance here is eclipsed by simpler tree-based methods. Our analysis underscores the importance of considering variance when evaluating model selection, particularly for critical applications where stability is paramount, and provides actionable insights for practitioners seeking reliable, high-accuracy classifiers.**

***Keywords-machine learning*; *malware detection*; *classification*; *model comparison*; *model evaluation*.**

I. Introduction

Cyber threats such as malware have become a significant challenge to digital security in recent years, affecting individuals, organizations, and critical infrastructure worldwide. As these threats evolve and become increasingly sophisticated, traditional signature-based detection methods are becoming less effective [1]. In response, Machine Learning (ML) has emerged as a powerful tool to automate malware detection, offering the ability to classify large volumes of data to identify patterns that might otherwise go unnoticed [2].

However, despite the growing use of machine learning, selecting the most appropriate algorithm for malware classification remains a difficult task due to the complexity of the data and the need for high accuracy and stability of the model in different data splits [3]. To address this challenge, this study conducts a comprehensive evaluation of eight widely used machine learning algorithms for malware detection, including Random Forest, Extreme Gradient Boosting (XGBoost), Support Vector Machine (SVM), Logistic Regression, Naive Bayes, Light Gradient Boosting Machine (LightGBM), Decision Tree, and k-Nearest Neighbors [4]–[6]. These models are assessed using 5-fold stratified cross-validation to ensure robust performance estimation across multiple data splits [7]. The evaluation is based on key performance metrics, including accuracy, precision, recall, F1 score, AUC-ROC, and variance, allowing a detailed comparison of the predictive power and stability of each model [4], [8].

For our experiments, we leverage a refined version of the Microsoft Malware Classification Challenge (BIG 2015) dataset [9], which contains feature-engineered representations of malware binaries [10]. The dataset encapsulates both static features, such as Portable Executable (PE) headers and entropy profiles, and dynamic features, including API call sequences and assembly opcode distributions [11]. These features enable robust classification of malware into distinct families. By analyzing attributes such as section-wise entropy differences `ent_q_diffs`, importing table dependencies (Imports) and opcoding frequencies, we aim to develop an interpretable machine learning model for malware detection [12]. The dataset's rich feature space not only facilitates accurate classification but also enables anomaly detection, providing insights into evolving malware evasion techniques [13].

The primary objective of this study is to identify the most effective machine learning model for malware classification by balancing predictive accuracy with model stability. While ensemble based methods, like Random Forest and XGBoost, are known for their strong predictive capabilities, their performance

must be assessed in comparison to simpler models, like Decision Tree and KNN, which may offer competitive results with lower computational cost. Furthermore, we explore the role of variance-aware evaluation, which is crucial in cybersecurity applications where model reliability across different datasets is essential. Our findings reveal that RF, LightGBM, DT, and KNN achieve near-perfect classification performance with minimal variance, demonstrating their robustness in malware detection tasks. In contrast, XGBoost and SVM exhibit slightly lower accuracy and higher variance, while LR and Naive Bayes perform moderately, struggling to capture complex decision boundaries in the data. These insights provide valuable guidance for researchers and practitioners in cybersecurity, helping them select reliable models for malware classification.

The structure of the paper is as follows. Section II reviews related work in machine learning-based malware detection. Section III briefly introduces the eight ML models utilized in this work. Section IV depicts the modeling procedure and results for the malware detection. Section V discusses the findings. We conclude with Section VI.

II. Related Work

The application of machine learning in cybersecurity, particularly for malware detection, has gained significant attention in recent years. Salem et al. [1] provided a comprehensive review of Artificial Intelligence (AI)-driven detection techniques, highlighting the evolution from traditional signature-based methods to sophisticated machine learning approaches. Similarly, Dasgupta et al. [2] conducted an extensive survey on machine learning applications in cybersecurity, emphasizing the critical role of automated detection systems in addressing the growing complexity of cyber threats.

Several studies have focused on comparative analysis of machine learning algorithms for malware classification. Rahul et al. [4] analyzed various machine learning models for malware detection, demonstrating the effectiveness of ensemble methods in capturing complex malware behavior patterns. Singh and Singh [5] assessed supervised machine learning algorithms using dynamic API calls, providing insights into the importance of feature selection and extraction techniques. Their work highlighted the challenges of balancing accuracy with computational efficiency in real-time detection systems.

The Microsoft Malware Classification Challenge dataset [11] has served as a benchmark for numerous studies in this domain. Aslan and Samet [9] provided a comprehensive review of malware detection approaches, categorizing methods into static, dynamic, and hybrid analysis techniques. Ghouti and Imam [10] specifically focused on malware classification using compact image features and multiclass support vector machines, demonstrating the potential of visual representation techniques. More recently, Connors and Sarkar [12] explored machine learning approaches for detecting malware in PE files, while Lin and Chang [13] addressed the interpretability challenges in ML-based automated malware detection models. These studies collectively underscore the ongoing evolution of machine learning techniques in cybersecurity applications, setting the foundation for our comprehensive comparative analysis of eight state-of-the-art algorithms.

III. Methods

Classification algorithms, a cornerstone of machine learning, have demonstrated exceptional performance across various domains, including cybersecurity applications such as malware detection [3], [14]. Beyond cybersecurity, these algorithms play a crucial role in disease diagnosis [15], where they help detect conditions like cancer [16], [17], diabetes [18], [19], and cardiovascular diseases [20] through medical imaging and clinical data analysis [21]. In finance, classification models are widely used for fraud detection, identifying suspicious transactions and preventing financial crimes [22]. Additionally, they contribute to spam filtering in email systems, sentiment analysis in natural language processing, and customer churn prediction in business analytics [23]. The versatility and effectiveness of classification algorithms make them indispensable across diverse fields where pattern recognition and decision making are essential. This study evaluates eight state of the art classification models, namely, Random Forest (RF), XGBoost, LightGBM, Support Vector Machine (SVM), Logistic Regression, Naive Bayes, Decision Tree, and k-Nearest Neighbors (KNN) to predict malware classes using static and dynamic features. Performance is assessed via five metrics: Accuracy, Precision, Recall, F1-Score, and AUC-ROC, with variance analysis across stratified 5-fold cross-validation to quantify stability.

Given a labeled dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ where $\mathbf{x}_i$ represents feature vectors (e.g., API calls, entropy values) and $y_i \in \{0, 1\}$ denotes benign/malicious labels, we formalize each model's prediction $\hat{y}$ for a new sample $\mathbf{x}$.

A. Random Forest

RF is an ensemble method that aggregates predictions from multiple decision trees, reducing overfitting through majority voting. For malware detection, it has proven to be effective [6].

$$\hat{y} = \text{mode}\left(\{f_i(\mathbf{x})\}_{i=1}^{N}\right), \tag{1}$$

where f_i is the i-th tree's prediction, and N is the total number of trees in Equation (1). RF excels at handling high-dimensional feature spaces (e.g., API call sequences).

B. XGBoost

XGBoost iteratively improves predictions by combining weak learners (decision trees) with gradient descent optimization.

$$\hat{y} = \sum_{i=1}^{N} \gamma_i f_i(\mathbf{x}), \tag{2}$$

where γ_i is the learning rate. XGBoost's regularization (L1/L2 penalties) mitigates overfitting, critical for imbalanced malware datasets.

C. LightGBM

LightGBM uses histogram-based splitting for efficiency, optimizing memory usage for large-scale malware data.

$$\hat{y} = \sum_{i=1}^{N} \alpha_i f_i(\mathbf{x}), \tag{3}$$

where α_i weights leaf outputs. Its Gradient-based One-Side Sampling (GOSS) is ideal for sparse features (e.g., n-gram opcodes).

D. Support Vector Machine

SVM finds the optimal hyperplane to separate malware benign classes via maximum margin optimization.

$$\hat{y} = \text{sign}\left(\mathbf{w}^T \phi(\mathbf{x}) + b\right), \tag{4}$$

where $\hat{y}$ is the predicted class label for a given input $\mathbf{x}$, $\mathbf{w}$ is the weight vector learned by the SVM during training, $\mathbf{w}^T$ denotes the transpose of the weight vector $\mathbf{w}$, $\phi(\mathbf{x})$ is a non-linear transformation of the input vector $\mathbf{x}$ into a higher-dimensional feature space, performed using a kernel function, b is the bias term that shifts the decision boundary, and sign$(\cdot)$ is the sign function, which returns $+1$ if the argument is positive and -1 if it is negative. The kernel function $\phi(\cdot)$ enables SVM to handle non-linearly separable data by implicitly mapping inputs into a high dimensional space. A common choice is the Radial Basis Function (RBF) kernel. The effectiveness of SVM is highly dependent on the scaling of features, as it ensures that each feature contributes proportionally to the boundary of the final decision.

E. Logistic Regression

A linear model for probabilistic classification

$$\hat{y} = \mathbb{I}\left(\frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x}+b)}} \geq 0.5\right) \tag{5}$$

where $\mathbb{I}(\cdot)$ is the indicator function. It is Interpretable but limited to linear feature relationships.

F. Naive Bayes

Naive Bayes is a probabilistic classifier that Leverages Bayes' theorem with feature independence assumptions.

$$\hat{y} = \arg\max_y P(y) \prod_{j=1}^{d} P(x_j \mid y), \tag{6}$$

where $\hat{y}$ is the predicted class label for a given input instance, y represents a possible class label (e.g., malware or benign), $P(y)$ is the prior probability of class y, x_j is the j-th feature of the input vector $\mathbf{x}$, $P(x_j \mid y)$ is the conditional probability (likelihood) of observing feature x_j given class y, d is the total number of features in the input, and $\arg\max$ selects the class label y that maximizes the posterior probability. Naive Bayes is computationally efficient and effective for high-dimensional data. However, its performance can degrade when features are highly correlated, such as in the case of dependent API calls in malware behavior analysis.

G. Decision Tree

A single tree recursively partitions the feature space.

$$\hat{y} = f(\mathbf{x}; \theta), \tag{7}$$

where θ denotes split thresholds. It is prone to overfitting but useful for interpretability.

H. k-Nearest Neighbors

The k-Nearest Neighbors (KNN) algorithm classifies samples based on majority labels of the k closest training instances.

$$\hat{y} = \text{mode}\left(\{y_i \mid \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})\}\right), \tag{8}$$

where $\mathcal{N}_k(\mathbf{x})$ are the k-nearest neighbors. Sensitive to feature scaling and distance metrics (e.g., Hamming distance for binary features).

I. Performance Metrics

Five metrics evaluate model performance, with variance calculated across folds.

1. Accuracy is the proportion of correct predictions over total predictions [24].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{9}$$

where TP (True Positives) represents the number of correctly predicted positive instances; TN (True Negatives) is the number of correctly predicted negative instances; FP (False Positives) is the number of negative instances incorrectly predicted as positive; and FN (False Negatives) is the number of positive instances incorrectly predicted as negative [25].

2. Precision is the proportion of correctly predicted positive instances among all predicted positives [24].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

3. Recall is the proportion of actual positive instances that were correctly identified [24].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

4. The F1-Score is the harmonic mean of precision and recall [24].

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

5. AUC-ROC (Area Under the Receiver Operating Characteristic Curve) plots True Positive Rate (Sensitivity) against False Positive Rate (1-Specificity) across all classification thresholds. The closer the curve approaches the top-left corner (0,1), the better the model's discriminative ability [26].

6. The variance of each performance metric is calculated across cross-validation folds to assess the model's stability. A lower variance indicates a more consistent and reliable model, while a higher variance suggests performance fluctuations across different training sets [27].

TABLE I
RESULTS OF MODELS.

Model	Accuracy	Accuracy Variance	Precision	Precision Variance	Recall	Recall Variance	F1-Score	F1-Score Variance	AUC-ROC	AUC-ROC Variance
Random Forest	0.991833	2.57E-06	0.991972	2.39E-06	0.991833	2.57E-06	0.991814	2.57E-06	0.999819	1.90E-08
XGBoost	0.979296	9.65E-06	0.980132	8.70E-06	0.979296	9.65E-06	0.938477	0.000201884	0.999429	2.57E-08
SVM	0.979296	9.65E-06	0.980132	8.70E-06	0.979296	9.65E-06	0.938477	0.000201884	0.999429	2.57E-08
Logistic Regression	0.938575	0.000206963	0.943028	0.000117831	0.938575	0.00020696	0.938477	0.000201884	0.976317	3.25E-05
Naive Bayes	0.938575	0.000206963	0.943028	0.000117831	0.938575	0.00020696	0.938477	0.000201884	0.976317	3.25E-05
LightGBM	0.991833	2.57E-06	0.991972	2.39E-06	0.991833	2.57E-06	0.991814	2.57E-06	0.999819	1.90E-08
Decision Tree	0.991833	2.57E-06	0.991972	2.39E-06	0.991833	2.57E-06	0.991814	2.57E-06	0.999819	1.90E-08
KNN	0.991833	2.57E-06	0.991972	2.39E-06	0.991833	2.57E-06	0.991814	2.57E-06	0.999819	1.90E-08

IV. EXPERIMENTAL RESULTS

A. Dataset and Experimental Setup

The experiments were conducted using a refined version of the Microsoft Malware Classification Challenge (BIG 2015) dataset [9]. The dataset contains 21,741 samples with balanced class distribution across nine malware families and benign files. Feature engineering yielded 2,381 static features including PE header information, entropy profiles, import table dependencies, and assembly opcode frequencies. All models were evaluated using stratified 5-fold cross-validation to ensure robust performance estimation across different data splits [28].

B. Performance Evaluation Results

Table I presents the comprehensive performance evaluation of eight machine learning models across five key metrics. The results reveal distinct performance tiers among the evaluated algorithms.

Tier 1 - Exceptional Performers: RF, LightGBM, DT, and KNN achieved identical near-perfect performance with accuracy of 0.9918, precision of 0.9920, recall of 0.9918, F1-score of 0.9918, and AUC-ROC of 0.9998. These models demonstrated remarkably low variance (10^{-6} to 10^{-8}), indicating exceptional stability across cross-validation folds.

Tier 2 - Strong Performers: XGBoost and SVM achieved accuracy of 0.9793 with identical performance metrics. While still demonstrating strong classification capability, these models showed slightly higher variance ($\approx 10^{-6}$) compared to Tier 1 performers.

Tier 3 - Moderate Performers: Logistic Regression and Naive Bayes exhibited lower accuracy (0.9386) and significantly higher variance ($\approx 10^{-4}$), indicating less consistent performance across different data splits.

C. Statistical Significance and Stability Analysis

The variance analysis reveals critical insights into model reliability [29]. The exceptionally low variance ($< 10^{-6}$) observed in RF, LightGBM, DT, and KNN indicates these models maintain consistent performance regardless of training data variations—a crucial requirement for cybersecurity applications [27].

In contrast, the higher variance exhibited by Logistic Regression and Naive Bayes ($\approx 10^{-4}$) suggests potential instability when deployed across different malware datasets or network environments. This stability assessment is particularly important in cybersecurity where reliable performance across diverse threat landscapes is essential.

V. DISCUSSION

A. Model Performance Analysis and Implications

The superior performance of tree-based ensemble methods (Random Forest, LightGBM) and the Decision Tree can be attributed to their ability to capture complex, non-linear feature interactions inherent in malware behavior patterns [30]. These models effectively handle the high-dimensional feature space (2,381 features) without suffering from the curse of dimensionality.

Ensemble Method Advantages: Random Forest's bootstrap aggregating reduces overfitting while maintaining high accuracy. LightGBM's Gradient-based One-Side Sampling (GOSS) efficiently handles sparse features common in malware detection, such as n-gram opcodes and API call sequences.

KNN's Unexpected Success: The exceptional performance of KNN (identical to ensemble methods) suggests that malware and benign samples form distinct, well-separated clusters in the feature space. This clustering behavior indicates that the extracted features effectively capture discriminative patterns between malware families and benign software.

XGBoost Underperformance: Despite its reputation for strong performance, XGBoost's lower F1-score (0.9385) compared to simpler tree-based methods suggests potential overfitting or suboptimal hyperparameter configuration. This highlights the importance of hyperparameter optimization using techniques such as GridSearchCV or RandomizedSearchCV [31].

B. Linear Model Limitations

The moderate performance of Logistic Regression and Naive Bayes stems from their fundamental assumptions that are incompatible with malware detection requirements. Logistic Regression assumes linear decision boundaries, which cannot adequately model the complex, non-linear relationships between malware features and class labels. Similarly, Naive Bayes relies on the feature independence assumption, which is violated in malware analysis where features such as API call sequences

and opcode patterns exhibit strong dependencies. However, these models offer computational efficiency and interpretability advantages, making them suitable for resource-constrained environments or scenarios requiring explainable decisions.

C. Practical Deployment Considerations

Computational Complexity: While ensemble methods provide superior accuracy, they introduce computational overhead. Real-time malware detection systems may require model optimization or hardware acceleration for practical deployment.

Scalability Analysis: KNN's instance-based learning requires storing all training samples, making it memory-intensive and computationally expensive for large-scale deployments. Despite its excellent accuracy, scalability concerns limit its practical applicability.

Model Selection Recommendations: For production environments, Random Forest and LightGBM offer the optimal balance of accuracy, stability, and computational efficiency.

D. Challenges and Limitations

Several challenges were encountered during this study:

Feature Engineering Constraints were evident as this study relied primarily on static features extracted from malware samples. Incorporating dynamic behavioral features such as API call sequences and network traffic patterns could further enhance classification performance.

Dataset Generalization presents another concern since while the Microsoft dataset provides a solid foundation, real-world malware detection faces continuously evolving threats. Future work should evaluate model performance on contemporary malware samples and emerging attack vectors.

Class Imbalance Considerations must also be addressed, as although our refined dataset maintains balanced class distribution, real-world scenarios typically exhibit significant class imbalance where benign samples vastly outnumber malware instances. Addressing this through cost-sensitive learning or advanced sampling techniques represents an important future direction.

Hyperparameter Optimization limitations were apparent since the current study employed default hyperparameters for most algorithms. Systematic hyperparameter tuning using GridSearchCV or RandomizedSearchCV could potentially improve performance, particularly for XGBoost and SVM models.

E. Future Research Directions

Future investigations should explore several promising directions. Deep learning integration through evaluating Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for malware detection represents a natural evolution of this work. Dynamic feature incorporation by including behavioral analysis features such as API call sequences and runtime behavior patterns could significantly enhance detection capabilities. Adversarial robustness assessment against malware samples specifically designed to evade detection systems presents a critical research challenge. Additionally, real-time performance optimization through developing lightweight model variants suitable for edge computing and real-time detection systems would address practical deployment requirements in cybersecurity environments.

VI. Conclusion

This study systematically evaluated eight machine learning models for malware detection using stratified 5-fold cross-validation, assessing both accuracy and stability through variance analysis. The results demonstrate clear performance hierarchies among the evaluated algorithms with significant implications for cybersecurity applications.

Tree-based models, particularly RF, LightGBM, DT, and KNN, achieved exceptional performance with accuracy of 0.9918 and AUC-ROC of 0.9998, while maintaining minimal variance ($< 10^{-6}$). These models demonstrated remarkable stability across data splits, effectively capturing complex feature interactions in malware behavior patterns. Conversely, Logistic Regression and Naive Bayes underperformed with accuracy of 0.9386 and higher variance ($\sim 10^{-4}$) due to their linear assumptions, which fail to model complex malware characteristics.

Notably, KNN's exceptional performance suggests that malware and benign samples form distinct clusters in the feature space, validating our feature engineering approach. XGBoost's moderate F1-score (0.9385) indicates potential overfitting, highlighting the importance of hyperparameter optimization for gradient boosting algorithms.

For practical deployment, we recommend Random Forest and LightGBM due to their optimal balance of accuracy, stability, and computational efficiency. Our analysis emphasizes the critical importance of variance-aware evaluation alongside accuracy metrics for cybersecurity applications, ensuring consistent performance across diverse threat environments.

Future research should focus on integrating deep learning approaches, incorporating dynamic behavioral features, and developing adversarial robustness against evolving evasion techniques to advance practical malware detection capabilities.

Acknowledgement

The work is partially supported by the National Science Foundation under NSF Awards Nos. 2100134, 2131228, 2209637, 2234911, and 2417608. Any opinions, findings, or recommendations, expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] A. H. Salem, S. M. Azzam, O. E. Emam, and A. A. Abohany, "Advancing cybersecurity: A comprehensive review of AI-driven detection techniques", *Journal of Big Data*, vol. 11, no. 1, p. 105, 2024. DOI: 10.1186/s40537-024-00957-y.

[2] D. Dasgupta, Z. Akhtar, and S. Sen, "Machine learning in cybersecurity: A comprehensive survey", *The Journal of Defense Modeling and Simulation*, vol. 19, no. 1, pp. 57–106, 2022. DOI: 10.1177/1548512920951275. eprint: https://doi.org/10.1177/1548512920951275.

[3] H. Tan, "Machine learning algorithm for classification", *Journal of Physics: Conference Series*, vol. 1994, no. 1, p. 012 016, Aug. 2021. DOI: 10.1088/1742-6596/1994/1/012016.

[4] Rahul, P. Kedia, S. Sarangi, and Monika, "Analysis of machine learning models for malware detection", *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 23, no. 2, pp. 395–407, 2020. DOI: 10.1080/09720529.2020.1721870.

[5] J. Singh and J. Singh, "Assessment of supervised machine learning algorithms using dynamic api calls for malware detection", *International Journal of Computers and Applications*, vol. 44, no. 3, pp. 270–277, 2022. DOI: 10.1080/1206212X.2020.1732641. eprint: https://doi.org/10.1080/1206212X.2020.1732641.

[6] F. Zhang, B. Chen, F. Brishti, S. Mohammed, F. Wu, and L. Bai, "Predicting energy star scores for diverse building types using machine learning", *International Journal On Advances in Systems and Measurements*, vol. 17, no. 3, pp. 176–188, Dec. 2024, ISSN: 1942-261X.

[7] M. Shi *et al.*, "A novel electronic health record-based, machine-learning model to predict severe hypoglycemia leading to hospitalizations in older adults with diabetes: A territory-wide cohort and modeling study", *PLoS Medicine*, vol. 21, no. 4, e1004369, 2024.

[8] I. Abdessadki and S. Lazaar, "A new classification based model for malicious pe files detection", *International Journal of Computer Network and Information Security*, vol. 9, no. 6, p. 1, 2019.

[9] Ö. A. Aslan and R. Samet, "A comprehensive review on malware detection approaches", *IEEE Access*, vol. 8, pp. 6249–6271, 2020. DOI: 10.1109/ACCESS.2019.2963724.

[10] L. Ghouti and M. Imam, "Malware classification using compact image features and multiclass support vector machines", *IET Information Security*, vol. 14, no. 4, pp. 419–429, 2020. DOI: https://doi.org/10.1049/iet-ifs.2019.0189. eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-ifs.2019.0189.

[11] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ahmadi, "Microsoft malware classification challenge", *arXiv preprint arXiv:1802.10135*, 2018.

[12] C. Connors and D. Sarkar, "Machine learning for detecting malware in pe files", in *2023 International Conference on Machine Learning and Applications (ICMLA)*, IEEE, 2023, pp. 2194–2199.

[13] Y. Lin and X. Chang, "Towards interpreting ml-based automated malware detection models: A survey", *arXiv preprint arXiv:2101.06232*, 2021.

[14] K. Mohammed, "Harnessing the speed and accuracy of machine learning to advance cybersecurity", *arXiv preprint arXiv:2302.12415*, 2023.

[15] N. G. Nia, E. Kaplanoglu, and A. Nasab, "Evaluation of artificial intelligence techniques in disease diagnosis and prediction", *Discover Artificial Intelligence*, vol. 3, no. 1, p. 5, 2023, ISSN: 2731-0809. DOI: 10.1007/s44163-023-00049-5.

[16] M. J. Iqbal *et al.*, "Clinical applications of artificial intelligence and machine learning in cancer diagnosis: Looking into the future", *Cancer Cell International*, vol. 21, no. 1, p. 270, 2021, ISSN: 1475-2867. DOI: 10.1186/s12935-021-01981-1.

[17] T. Saba, "Recent advancement in cancer detection using machine learning: Systematic survey of decades, comparisons and challenges", *Journal of infection and public health*, vol. 13, no. 9, pp. 1274–1289, 2020.

[18] S. Kaul and Y. Kumar, "Artificial intelligence-based learning techniques for diabetes prediction: Challenges and systematic review", *SN Computer Science*, vol. 1, no. 6, p. 322, 2020, ISSN: 2661-8907. DOI: 10.1007/s42979-020-00337-2.

[19] B. F. Wee, S. Sivakumar, K. H. Lim, W. K. Wong, and F. H. Juwono, "Diabetes detection based on machine learning and deep learning approaches", *Multimedia Tools and Applications*, vol. 83, no. 8, pp. 24 153–24 185, 2024.

[20] T. Ullah *et al.*, "Machine learning-based cardiovascular disease detection using optimal feature selection", *IEEE Access*, vol. 12, pp. 16 431–16 446, 2024.

[21] A. Ogunpola, F. Saeed, S. Basurra, A. M. Albarrak, and S. N. Qasem, "Machine learning-based predictive models for detection of cardiovascular diseases", *Diagnostics*, vol. 14, no. 2, p. 144, 2024.

[22] A. Rahman *et al.*, "Machine learning and network analysis for financial crime detection: Mapping and identifying illicit transaction patterns in global black money transactions", *Gulf Journal of Advance Business Research*, vol. 2, no. 6, pp. 250–272, 2024.

[23] M. R. Hasan, R. K. Ray, and F. R. Chowdhury, "Employee performance prediction: An integrated approach of business analytics and machine learning", *Journal of Business and Management Studies*, vol. 6, no. 1, p. 215, 2024.

[24] J. C. Obi, "A comparative study of several classification metrics and their performances on data", *World Journal of Advanced Engineering Technology and Sciences*, vol. 8, no. 1, pp. 308–314, 2023.

[25] A. A. Theodosiou and R. C. Read, "Artificial intelligence, machine learning and deep learning: Potential resources for the infection clinician", *Journal of Infection*, vol. 87, no. 4, pp. 287–294, 2023, ISSN: 0163-4453. DOI: https://doi.org/10.1016/j.jinf.2023.07.006.

[26] D. J. Hand, "Measuring classifier performance: A coherent alternative to the area under the ROC curve", *Machine learning*, vol. 77, no. 1, pp. 103–123, 2009.

[27] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection", *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[28] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection", in *Proceedings of the 14th international joint conference on Artificial intelligence*, vol. 2, 1995, pp. 1137–1143.

[29] C. Sammut and G. I. Webb, "Cross-validation", in *Encyclopedia of machine learning*, Springer, 2010, pp. 249–249.

[30] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms", *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[31] S. Raschka, "Model evaluation, model selection, and algorithm selection in machine learning", *arXiv preprint arXiv:1811.12808*, 2018.