Analyzing Ensemble Techniques for Improved Malware Detection Performance.

[1]D.Swetha, [2]Sudha Singaraju,[3]Dr.V.Goutham.

[1] Assistant Professor, Department of AI &DS ,Guru Nanak Institute of Technology, Ibrahimpatnam,Hyderabad, and Research Scholar at BESTIU, Anantapur, Andhra Pradesh.

[2] Senior Assistant Professor, Department of CSE, Geethanjali College of Engineering & Technology,Cheeryal,Hyderabad.

[3] Professor, Department of CSE,AVN Institute of Engineering & Technology, Ibrahimpatnam, Hyderabad.

ABSTRACT:

The Ensemble Learning-based Malware Detection (EL-MD) algorithm integrates multiple classifiers, which enhances accuracy, resilience, and robustness when dealing with imbalanced datasets. This advancement broadens the scope of possibilities through ensemble techniques by incorporating feature selection and around-scaled detection systems to combat future-generation malware variants. This contribution significantly improves reliability and efficiency in malware detection by addressing issues such as false positives and computational inefficiencies. An ensemble learning approach was executed through the voting of KNN and Naïve Bayes models. We employed both hard and soft voting schemes. The ensemble model achieved an accuracy of 97% and demonstrated high robustness, with a macro F1 score rising to 0.91, while class consistency was also enhanced. This approach proved particularly effective in managing occasional errors and optimizing precision and recall for unbalanced datasets.

Keywords: EL-MD , KNN, Naïve Bayes, Principal Component Analysis , Grey Wolf Optimization

1.INTRODUCTION

Malware stands at the forefront of the cybersecurity challenges that have arisen alongside the digital era, which has introduced unprecedented levels of connectivity and convenience. A wide-ranging category of software designed to inflict harm or exploit any programmable device, service, or network is known as malware. In the ever-evolving landscape of cyber threats, malware remains a significant hurdle faced by cybersecurity professionals. To safeguard digital infrastructure, comprehensive detection systems are essential due to the rapid proliferation of malware variants. Traditional methods that rely on signatures, while effective in recognizing established threats, often

fall short in detecting new, unidentified, or polymorphic malware. Cybercriminals continuously create new malware variants, taking advantage of software and system vulnerabilities at an alarming pace. This constantly shifting and dynamic threat landscape demands sophisticated, flexible, and robust defense strategies. Traditional approaches to malware detection have largely relied on heuristic and signature-based techniques. While signature-based methods are effective in identifying known threats, they struggle to detect new or variant malware since they depend on existing databases of recognized malware signatures. Although heuristic methods, which utilize behavior or attribute-based techniques to detect malware, offer some protection against unknown threats, they are prone to high rates of false positives. These limitations highlight the necessity for more advanced and adaptable detection strategies that can effectively address the complexities of modern malware. Machine learning (ML) offers the ability to recognize existing malware by identifying patterns and anomalies within datasets.

ML Models have become essential in addressing the limitations of traditional malware detection methods. Unlike the signature-based approach, which depends on established patterns, ML-based detection systems utilize a vast database of records to identify intricate patterns and anomalies that may indicate malicious activity. Techniques such as KNN and Naïve Bayes can classify based on features including API Calls, permissions, or even behavioral traits. Ensemble methods, like Random Forest and Gradient Boosting, enhance detection accuracy and generalization by integrating multiple classifiers to mitigate biases. Concurrently, CNNs identify malware through image representations of binaries, while RNNs are utilized to analyze logs and network traffic data for anomalies. By employing machine learning, contemporary malware detection systems offer adaptive and scalable solutions to address new and obfuscated threats. This transition from traditional detection methods to AI-driven, data-centric approaches is crucial for effectively managing the evolving and dynamic threat landscape.

2. LITERATURE REVIEW:

This section delves into research studies that utilize both traditional and advanced machine learning models for the detection of malware, such as KNN, Naïve Bayes, and Random Forest. The reviews extensively discuss improvements in accuracy, feature selection, and the dependency on datasets. Ahmed et al. [1] proposed a unique method for detecting malware by combining texture feature extraction techniques (SFTA, LBP, Gabor) with classifiers (KNN,

SVM, RF). The combination of KNN with SFTA and Gabor showed significant accuracy improvements when tested on the MaleVis and Malimg datasets. This technique enhances model performance by addressing malware detection through feature extraction, leading to reliable detection results from efficient texture-based feature extraction. However, a limitation is the need for manual intervention, which requires human engagement. To further improve the method, the authors recommend future research to focus on data-balancing strategies and the integration of deep learning models. Although the method is effective, automation and scalability remain significant challenges. Umar et al. [2] introduced a behaviorally-driven malware detection technique based on machine learning, utilizing KNN and Random Forest. This method outperforms conventional approaches with high accuracy rates of 96.77% and 98.19% for KNN and Random Forest, respectively. The approach was tested on a dataset consisting of 3,540 malicious and 6,999 benign files, demonstrating good predictive performance. Its primary benefit lies in its ability to precisely identify malware by analyzing behavioral patterns. However, a limitation is the reliance on feature scaling and the specific dataset used. Future research could explore alternative machine learning strategies to enhance detection performance. Despite scalability and data dependency issues, the method offers good accuracy overall. Islam et al. [3] presented a weighted voting ensemble model for classifying Android malware based on dynamic features. Tested on the CCCS-CIC-AndMal-2020 dataset, the model outperformed recent research with an accuracy of 95.0%. One advantage of the approach is its improved accuracy achieved through simplifying the dataset using feature reduction techniques like Principal Component Analysis and outlier handling. However, a significant drawback is the intensive feature engineering required by the model. The success of the model depends on meticulous feature selection and data preparation, which can be time-consuming. To validate the method's broader applicability and enhance its robustness, the authors plan to apply it to new datasets in future work. Although the strategy performs well overall, rigorous data preprocessing and engineering are essential to maximize outcomes.

Mahindru and Sangal [4] developed a framework for detecting Android smartphone malware, which utilizes three ensemble techniques and five machine learning algorithms. The suggested framework focuses on feature selection to enhance detection accuracy and was evaluated on a total of 67,538 malware apps and 1,94,659 benign apps. The Nonlinear Ensemble Decision Tree Forest (NDTF) technique within the framework achieved an impressive 98.8% detection rate. Findings indicated that selected features resulted in higher accuracy and fewer misclassifications compared to using all extracted features, outperforming traditional anti-virus

scanners and other methods. Future research may explore expanding the model to determine the number of permissions required to identify malicious or benign apps, while acknowledging the limitation of binary categorization of programs. Raniyah [5] addressed the challenge of identifying subtle variations of cyberattacks by introducing a semi-supervised machine learning approach for intrusion detection systems (IDSs). The suggested approach improved performance through the utilization of fivefold cross-validation and k-nearest neighbor (KNN) hyperparameter adjustment. By classifying unlabeled data using the statistical information of its k-nearest neighbors, the method increased detection rates and decreased false alarms when tested on the NSL-KDD dataset, outperforming conventional KNN-based IDS. While exhibiting improved accuracy, the method faces challenges in real-time attack detection and managing varying dataset sizes, prompting future research to focus on incorporating more features and structures for enhanced real-time detection and identification of novel attack types. Swarna Priya et al. [6] presented a Deep Neural Network (DNN) classifier for intrusion detection in the Internet of Medical Things (IoMT) environment, utilizing a hybrid PCA-GWO architecture. The integration of Principal Component Analysis (PCA) and Grey Wolf Optimization (GWO) in the suggested approach reduced dimensionality and selected significant features, leading to a 15% increase in accuracy and a 32% decrease in time complexity for healthcare systems. Experiments using several machine learning classifiers on a benchmark dataset demonstrated better performance and highlighted the potential for future research to expand the method to handle multi-class problems and optimize it for dynamic IoMT contexts. Prabhat Kumar et al. [7] proposed a unique distributed intrusion detection system (IDS) for enhancing the security of IoT networks using fog computing. The system utilizes a combination of Gaussian Naive Bayes, XGBoost, and k-nearest neighbors as separate classifiers, with a Random Forest for final classification, and achieved up to 99.99% detection rates when tested on the UNSW-NB15 and DS2OS datasets. While the approach effectively addresses the drawbacks of centralized IDS by distributing processing among fog nodes, it is limited to specific attack types, despite achieving low false alarm rates and high detection accuracy. Future research aims to enhance security parameter verification in IoT systems and incorporate deep learning approaches.

3.RESEARCH METHODOLOGY:

In this study, some of the notable ensemble techniques applied are Random Forest, Gradient Boosting, and Voting Classifiers. Each of these techniques follows a different approach to aggregating models. In Random Forest, multiple decision trees are trained on different bootstrap

samples of the dataset, and the final output is derived from averaging the output of individual trees (or voting in classification). To increase the variation between the trees, random feature selection is applied during tree construction. On the other hand, Gradient Boosting constructs its ensemble in a stage-wise fashion, emphasizing the correction of failures from previous models. It progressively introduces and weights models based on their predictions, leading to an aggregate strong model that minimizes the desired loss function.

Another practical way is applying ensemble learning using voting mechanisms to combine the prediction information from multiple heterogeneous classifiers, like KNN, Naïve Bayes, and decision trees. With hard voting, a class label is predicted based on the majority vote on all the combined outputs of classifiers. Essentially, if $h_1(x), h_2(x), \ldots, h_T(x)$ are the predictions from T classifiers, the ensemble prediction $H(x)$ is defined as:

$$H(x) = arg\max_{y} \sum_{t=1}^{T} \mathbb{I}(h_t(x) = y) \qquad (3.9)$$

Where $\mathbb{I}$ is the indicator function, and is the set of all known classes. With soft voting, a model gives a probability of belonging to a certain class, and the model with the highest average probability for each class is selected:

$$H(x) = arg\max_{y} \sum_{t=1}^{T} P_t(y \mid x) \qquad (3.10)$$

Where $P_t(y \mid x)$ is the probability that classifier h_t assigns to class y.

Ensemble learning's resilience to dataset abnormalities, such as imbalance, noise, and overlapping class distributions, is its main advantage when it comes to malware detection tasks.
Certain classifiers, for instance, might perform poorly for minority classes but quite well for majority classes.To counteract this bias and produce more reliable forecasts for the variety of malware types, models with varying strengths might be combined.
Additionally, ensemble approaches can handle the complexity of the more potent models (Random Forest and GradientBoosting) and "leak" the strength of the weaker ones (Naive Bayes).

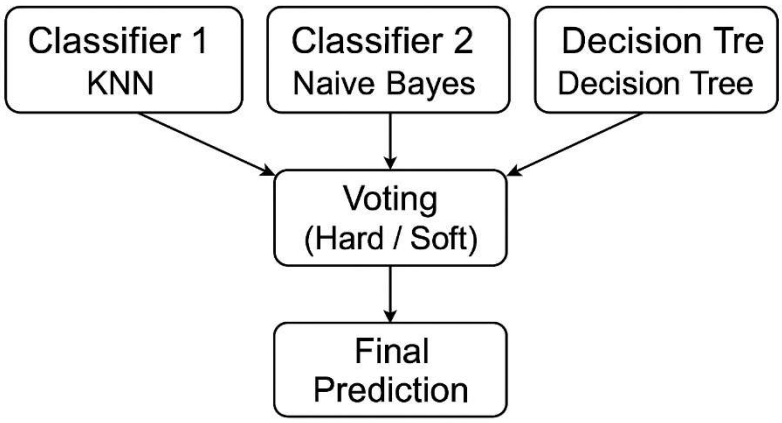


Figure 1: Ensemble Learning Architecture Combining Classifier Outputs via Hard or Soft Voting for Final Prediction

The basic framework of an ensemble learning model, which combines the predictions of several base classifiers to generate a final prediction, is depicted in Figure 3.5.

Classifier 1 (KNN), Classifier 2 (Naive Bayes), and Classifier 3 (Decision Tree) are the three source classifiers at the top level of the diagram. Each classifier represents an algorithm class of classifiers.

These models generate predictions on a particular input after being fitted independently on the same feature space.

Aspect	**Configuration / Description**
Ensemble Type	Voting-based ensemble using Voting Classifier from Scikit-learn
Base Classifiers	K-Nearest Neighbours (KNN), Naïve Bayes (Gaussian / Multinomial variants)
Voting Methods	Hard Voting (majority class), Soft Voting (average probability)
Probability Estimates	Enabled via predict_proba() in both KNN and Naïve Bayes
Hyperparameter Settings	Base classifiers are configured with optimized parameters from prior sections.
Weighting Scheme	Uniform weights assigned to all classifiers (future work may include accuracy-weighted voting)
Cross-Validation	10-fold cross-validation applied separately to both voting modes
Ensemble Output	Final class label selected based on majority vote (hard) or highest probability average (soft)

Integration	Ensemble applied after hybrid feature selection on reduced-dimensional input.
Reproducibility	Ensemble is defined in a modular code block with a fixed random seed for consistent trial evaluation.

I used the trained KNN and Naïve Bayes models to instantiate the VotingClassifier in the implementation, and depending on how it operated, I set the voting parameter to "hard" or “soft”. For the voting, the base classifiers were not given any additional weight.However, future research could explore the idea of adopting weighted voting schemes, where classifiers contribute to the remaining validation accuracies.The baseline scores and all ensemble outcomes were stored for future comparisons and performance increase analysis.

Compared to the individual classifiers, the ensemble modeling approach produced higher overall accuracy, precision, and F1-score.

Additionally, it showed greater stability (i.e., less variance) between folds, particularly when soft voting was used.

This validated the model combination in the malware detection challenge; a complementary approach can be used to learn heterogeneous behavior in malware classes.

Other researchers were able to replicate the ensemble configuration and use it with different classifiers or datasets thanks to the modularized source code.

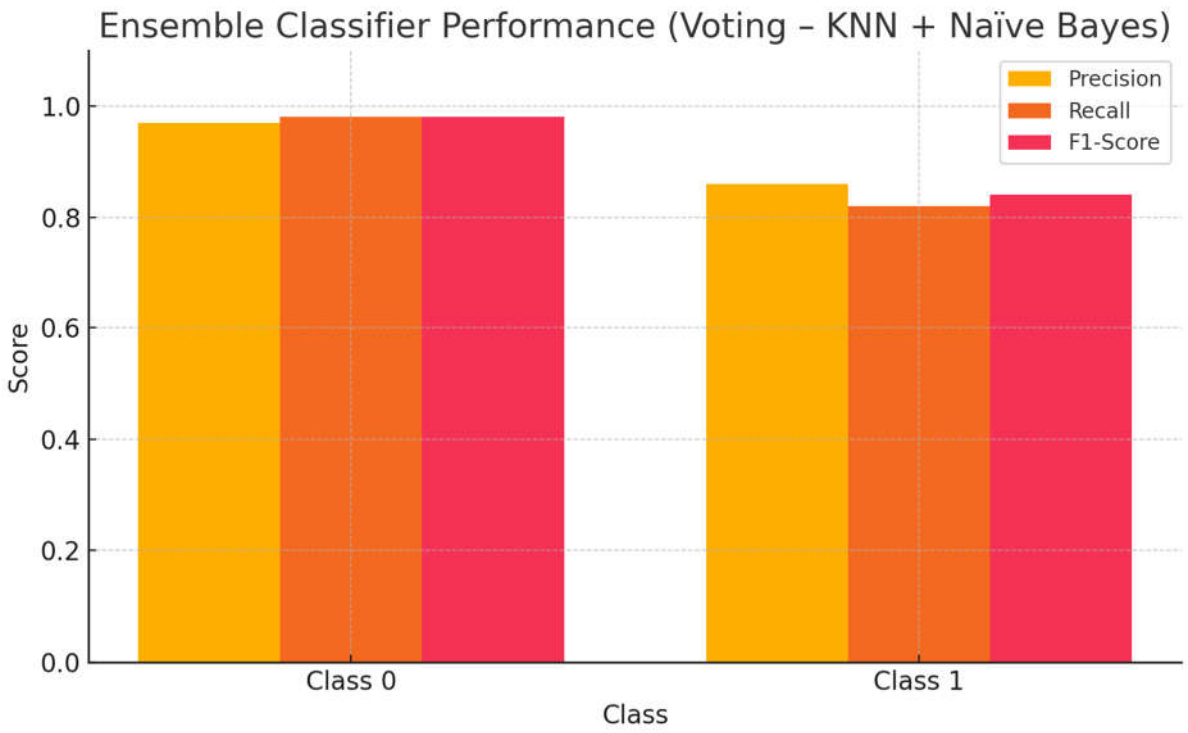
4. RESULTS AND DISCUSSION:

Table 4.1: Ensemble Classifier Performance Using Optimized Features (Voting – KNN + Naïve Bayes)

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Class 0	0.97	0.98	0.98	900
Class 1	0.86	0.82	0.84	100
Accuracy	–	–	0.97	1000
Macro Average	0.91	0.90	0.91	–
Weighted Average	0.97	0.97	0.97	–

Table 4.1 shows the classifier statistics of the ensembled model formed by the KNN and NBC using a voting scheme. The model's performance is demonstrated using the GA-optimized feature set, and

the results are reported for both types of malware over the main performance measures: precision, recall and F1 score



The classification accuracy of the ensemble model, which combines KNN and Naïve Bayes classif iers via voting with the aid of a GAoptimized feature set, is graphically described in Figure 4.1. In order to compare the behavior of the ensemble model with respect to both classes, the graph displays the class-wise precision-recall-f1 score for Class 0 and Class 1.

In Class 0, the model's precision, recall, and F1score were all consistently high at 0.97, 0.98and 0. 98respectively.The model's ability to identify and classify the majority class with minimal error is further supported by these findings.More significantly, Class 1 performs better as the ensemble approach attains 0.86 precision, 0.82 recall, and 0.84 F1-score.

Our ensemble can use the complimentary strength among members, as evidenced by the notable improvement in minority class malware recognition between the baseline and single models.

6. CONCLUSION:

The combination of KNN and Naïve Bayes also demonstrated a strong outcome, as did Ensemble, which combined the predictions of KNN and Naïve Bayes via voting. The ensemble's balanced accuracy at the macro level was 97%, allowing for a sensible tradeoff between robustness and precision.It is robust in managing noisy and unclear samples and effectively corrects errors related to minority class identification.

7. REFERENCES:

[1] Ismail Taha Ahmed, Baraa Tareq Hammad, And Norziana Jamil. (2024). A Comparative Performance Analysis of Malware Detection Algorithms Based on Various Texture Features and Classifiers. *IEEE*. 12, pp.11500 - 11519. http://DOI:10.1109/ACCESS.2024.3354959.

[2] Zeeshan Umar, Muhammad Zunnurain Hussain, Muhammad Zulkifl Hasan,SummairaNosheen,Ali Moiz Qureshi, Adeel Ahmad Siddiqui, Zaima Mubarak, Saad Hussain Chuhan, Muzzamil Mustafa,Muhammad Atif Yaqub, Afshan Bilal, and Tanveer Abbas. (2024). Analysis of Behavioral Artifacts of Malware for its Detection using Machine Learning. *IEEE*., pp.1. http://DOI:10.1109/I2CT61223.2024.10543310.

[3] Rejwana Islam, Moinul Islam Sayed, Sajal Saha, Mohammad Jamal Hossain, and Md Abdul Masud. (2023). Android malware classification using optimum feature selection and ensemble machine learning. *Elsevier*. 3, pp.100-111. https://doi.org/10.1016/j.iotcps.2023.03.001.

[5]Halim, Z., Yousaf, M. N., Waqas, M., Sulaiman, M., Abbas, G., Hussain, M., … Hanif, M. (2021). An effective genetic algorithm-based feature selection method for intrusion detection systems. Computers & Security, 110, 102448. doi:10.1016/j.cose.2021.102448.

[6]Mahindru, A., & Sangal, A. L. (2021). HybriDroid: an empirical analysis on effective malware detection model developed using ensemble methods. The Journal of Supercomputing, 77(8), 8209–8251. doi:10.1007/s11227-020-03569-4.

[7]Wazirali, R. (2020). An Improved Intrusion Detection System Based on KNN Hyperparameter Tuning and Cross-Validation. Arabian Journal for Science and Engineering. doi:10.1007/s13369-020-04907-7.

[8]R.M., S. P., Maddikunta, P. K. R., M., P., Koppu, S., G., T. R., Chowdhary, C. L., &Alazab, M. (2020). An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. Computer Communications. doi:10.1016/j.comcom.2020.05.048.

[9] Kumar, P., Gupta, G. P., & Tripathi, R. (2020). A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks. Journal of Ambient Intelligence and Humanized Computing. doi:10.1007/s12652-020-02696-3.

[10] L. Dhanya, R. Chitra, and A.M. Anusha Bamini. (2022). Performance evaluation of various ensemble classifiers for malware detection. *Elsevier*. 62(7), pp.4973-4979. https://doi.org/10.1016/j.matpr.2022.03.696.

[11] Ahmed S. Shatnawi, Qussai Yassen, and Abdulrahman Yateem. (2022). An android malware detection approach based on static feature analysis using machine learning algorithms. *Elsevier*. 201, pp.653-658. https://doi.org/10.1016/j.procs.2022.03.086.

[12] DURMUŞ ÖZKAN ŞAHİN, SEDAT AKLEYLEK, AND ERDAL KILIÇ. (2022). LinRegDroid: Detection of Android malware using multiple linear regression models-based classifiers. *IEEE*. 10, pp.14246 - 14259. http://DOI:10.1109/ACCESS.2022.3146363.

[13] Xinrun Zhang, Akshay Mathur, Lei Zhao, Safia Rahmat, Quamar Niyaz, Ahmad Javaid, Xiaoli Yang. (2022). An early detection of android malware using system calls based machine learning model. *ACM*. .(92), pp.1-9. https://doi.org/10.1145/3538969.3544413