Enhancing XGBoost Performance in Malware Detection through Chi-Squared Feature Selection

Salma Rosyada[1], Fauzi Adi Rafrastara[2]*, Arsabilla Ramadhani[3], Wildanil Ghozi[4], Warusia Yassin[5]
Department of Informatics Engineering[1], [2], [3], [4], Fakulti Teknologi Maklumat dan Komunikasi[5]
Universitas Dian Nuswantoro[1], [2], [3], [4], Universiti Teknikal Malaysia Melaka[5]
Semarang - Indonesia[1], [2], [3], [5], Melaka - Malaysia[4]
111202113382@mhs.dinus.ac.id[1], fauziadi@dsn.dinus.ac.id[2], 111202113384@mhs.dinus.ac.id[3],
wildanil.ghozi@dsn.dinus.ac.id[4], s.m.warusia@utem.edu.my[5]

***Abstract*— The increasing prevalence of malware poses significant risks, including data loss and unauthorized access. These threats manifest in various forms, such as viruses, Trojans, worms, and ransomware. Each continually evolves to exploit system vulnerabilities. Ransomware has seen a particularly rapid increase, as evidenced by the devastating WannaCry attack of 2017 which crippled critical infrastructure and caused immense economic damage. Due to their heavy reliance on signature-based techniques, traditional anti-malware solutions struggle to keep pace with malware's evolving nature. However, these techniques face limitations, as even slight code modifications can allow malware to evade detection. Consequently, this highlights weaknesses in current cybersecurity defenses and underscores the need for more sophisticated detection methods. To address these challenges, this study proposes an enhanced malware detection approach utilizing Extreme Gradient Boosting (XGBoost) in conjunction with Chi-Squared Feature Selection. The research applied XGBoost to a malware dataset and implemented preprocessing steps such as class balancing and feature scaling. Furthermore, the incorporation of Chi-Squared Feature Selection improved the model's accuracy from 99.1% to 99.2% and reduced testing time by 89.28%, demonstrating its efficacy and efficiency. These results confirm that prioritizing relevant features enhances both the accuracy and computational speed of the model. Ultimately, combining feature selection with machine learning techniques proves effective in addressing modern malware detection challenges, not only enhancing accuracy but also expediting processing times.**

Keywords*— *malware detection, XGBoost, chi-squared, machine learning, feature selection.

I. INTRODUCTION

In today's Internet era, users face increasing security threats, including malware, which can lead to the loss or unauthorized alteration of critical information [1]. Malware, or malicious software, is a harmful code designed to compromise computer systems and negatively impact users by hijacking devices, deleting files, stealing data, spamming, and downloading additional dangerous programs. This malware category includes various forms, such as viruses, Trojans, rootkits, ransomware, worms, botnets, spyware, adware, keyloggers, and numerous variants that continuously spread online. The range of malicious activities perpetrated by malware is extensive and constantly expanding as new threats emerge [2].

Worms are self-propagating malware programs that exploit system vulnerabilities to spread rapidly across networks without human intervention. As such, they can cause significant harm by consuming bandwidth, overloading servers, and potentially stealing or deleting data. Unlike viruses, which typically require user interaction to spread, worms can replicate autonomously and infect multiple systems, making them a persistent and dangerous threat [2], [3]. An example is the ILOVEYOU virus, which spread rapidly via email in May 2000, disrupting the email systems of major companies such as Microsoft and Ford Motor Company. This worm affected approximately 45 million users and resulted in an estimated $10 billion in economic damages within ten days [4].

Malware has continuously evolved, becoming increasingly complex and difficult to defend against. This evolution has been accompanied by the rise of social engineering as a key method for malware distribution, notably in the spread of ransomware [5]. A particularly alarming category of malware is ransomware. It uses encryption to lock files and folders, rendering them unusable until the victim pays a ransom, typically demanded in Bitcoin. It is often delivered through social engineering tactics like phishing, spear phishing, smishing, and BEC attacks. Once the data is encrypted, attackers demand payment to restore access to the files [6].

On May 12, 2017, a ransomware variant known as WannaCry globally impacted numerous organizations across various sectors, including government and healthcare [7]. The ransomware was particularly notorious for its ability to propagate across diverse computer systems and networks, notably affecting the UK's National Health Service and severely disrupting its operations [8]. In order to regain access to their decrypted data, victims were pressured to pay a ransom, typically ranging from $300 to $600. The attackers exclusively accepted Bitcoin as payment, threatening to permanently delete the victim's data if the ransom remained unpaid within the given timeframe [9].

Traditional anti-malware solutions for operating systems like Windows, Android, or other OSs rely on signature-based methods to detect malware by analyzing file signatures, such as cryptographic hashes and byte patterns. However, this approach

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.xx.xx, Copyright ©2020
Submitted : , Revised : , Accepted : , Published :

has significant limitations, as minor alterations to a malware's signature, such as code modifications or changes in instructions, can prevent detection. Consequently, these limitations hinder the ability of anti-malware tools to identify modified malware and effectively address zero-day attacks, exposing vulnerabilities in current malware detection systems and cybersecurity [10]. In response to these limitations in detecting advanced malware variants, this study proposes an enhanced machine-learning approach that utilizes XGBoost combined with Chi-Squared Feature Selection. Enhanced malware detection accuracy and effectiveness are the intended outcomes of this approach, which involves strategically selecting key features and optimizing model performance.

Unlike traditional gradient boosting, XGBoost leverages regularization techniques to create a robust ensemble of predictive models. It combines the strengths of these individual models, each correcting the weaknesses of the others to achieve higher accuracy [11]. Known for its accuracy and efficiency, XGBoost has been widely studied and consistently shown to achieve high accuracy in various research applications [12], [13].

Various methods have been developed to improve accuracy in malware detection. Rafrastara et al. [14] used kNN with Information Gain, achieving 97.0% accuracy with the Manhattan distance. Lu et al. [15] combined Deep Belief Networks and Gate Recurrent Units with Random Under-Sampling, reaching 97.79% accuracy. Abujazoh et al. [16] found that Decision Tree with Chi-square achieved 98.53% accuracy in high-dimensional data. Elayan & Mustafa [17] used GRU for Android malware detection, achieving 98.2% accuracy, outperforming traditional methods like SVM and kNN.

By utilizing the Chi-Square feature selection method, this study seeks to optimize the accuracy and efficiency of XGBoost for malware detection. This expectation is consistent with previous findings demonstrating the significant impact of feature selection on achieving optimal performance.

II. RESEARCH METHODS

This study comprises several stages, as shown in Figure 1, which details the processes implemented throughout the research. The study involves several stages: dataset collection and preparation, where issues in the downloaded dataset are identified and resolved (detailed in subsection B); pre-processing, where raw data is cleaned and transformed for optimal model building (detailed in subsection C); modeling, where techniques such as Random Forest, k-Nearest Neighbors, Naïve Bayes, and XGBoost are used (detailed in subsection D); and evaluation, where 10-fold cross-validation is used to maximize dataset usage, counteract overfitting, and validate (detailed in part E). Accuracy, F1-score, training, and testing duration are all measured as part of the evaluation.

A. Hardware and Software

The smooth execution of research relies on both hardware and software. Effective software requires adequate hardware, and vice versa [18]. Orange Data Mining (https://orangedatamining.com/) was the primary software used for data analysis in this study. The analysis was performed on a computer with an Intel i7 processor, 16 GB of RAM, and an NVIDIA GeForce RTX 2070 graphics card to handle the computational demands of the tasks.

B. Dataset Collection & Preparation

Initially, the UCI Machine Learning Repository acquired two malware datasets (VxHeaven, VirusTotal) and one goodware dataset. These datasets exhibit variations in their feature counts, requiring careful handling during the subsequent analysis to avoid biases and ensure accurate results. Further details are provided in Table 1.

TABLE I. DETAILS OF THE DATASET USED

Dataset Name	Malware static and dynamic features VxHeaven and VirusTotal Data Set.
Number of Files	3 (consisting of goodware, malware from VirusTotal, and malware from VxHeaven files).
Number of Rows	Goodware: 595; VirusTotal: 2955; VxHeaven: 2698.
Number of Features	Goodware: 1086; VirusTotal: 1087; VxHeaven: 1087 (including labels).
Number of Classes	2 (0 and 1).
Missing Values	None.

Table 1 shows that the Goodware dataset contains 595 records with 1086 features labeled '0'. The VirusTotal dataset has 2955 records with 1087 features, and the VxHeaven dataset has 2698 records with 1087 features labeled ‘1’. To consolidate these datasets, features not present in all files were removed. Specifically, SafeArrayPtrOfIndex and _vbaVarIndexLoad were removed from VirusTotal and VxHeaven, reducing their features to 1085. Feature 1 was removed from Goodware, aligning it with the malware datasets. The VirusTotal and VxHeaven files are then combined. The Data Preparation stage is followed by the Preprocessing stage.

C. Pre-Processing

At this stage, the prepared dataset undergoes further processing before modeling, involving Class Balancing, Feature Scaling, and Feature Selection, as shown in Fig. 1. Pre-processing is crucial in the knowledge discovery process. It ensures that data mining algorithms can effectively learn and identify valuable patterns by focusing on essential attributes of the input data. This stage includes cleaning, filtering, normalizing, annotating, and transforming raw data, enhancing its suitability for data mining analysis and improving prediction model accuracy [19].

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024

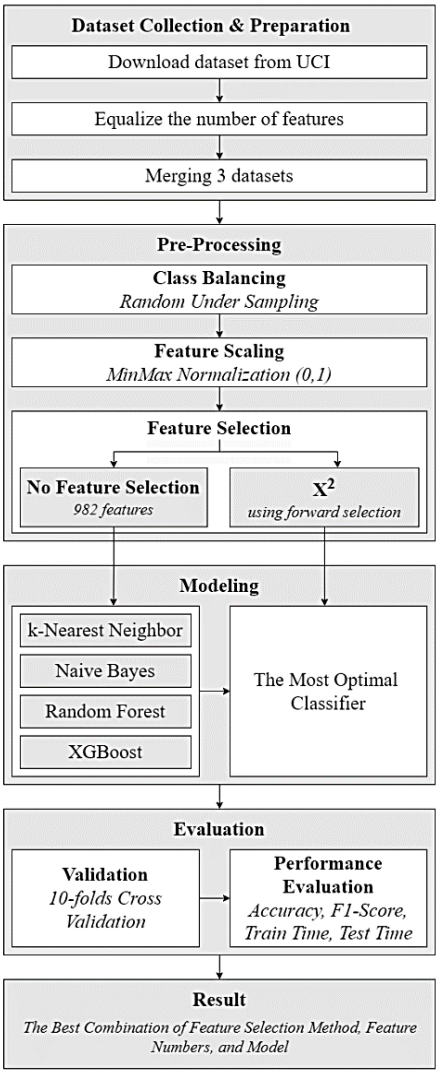

Fig. 1. Research Stages

As depicted in Fig. 1, the dataset in this study is imbalanced, with 5,653 malware instances and 595 goodware instances, resulting in a 1:9.5 ratio. This medium imbalance necessitates balancing the dataset. The study uses random sampling with fixed instances in Orange Data Mining to address this issue. Random sampling is a versatile technique in data analysis, essential for approximate query processing systems. It allows for efficient approximate answers with minimal error [20]. In this study, class balancing is achieved by selecting 595 records from the malware dataset (labeled '1') to match the 595 records in the goodware dataset (labeled '0'), resulting in a balanced 1:1 ratio.

Following class balancing, the next step involves feature scaling using Min-Max Normalization. This process involves scaling all features to a range between 0 and 1, ensuring that each feature contributes equally to the analysis [18], [21]. The formula for min-max normalization is:

$$v_i = \left(\frac{x_i - x_{min}}{x_{max} - x_{min}} \left(v_{max} - v_{min} \right) \right) + v_{min} \quad (1)$$

Equation 1 defines the min-max normalization process, where x_i stands for the original value in the *i*-th row, while v_i is the corresponding normalized value. This transformation uses the feature's minimum value x_{min} and maximum value x_{max} to rescale the data to a new range, with v_{max} (i.e., 1) as the upper bound and v_{min} (i.e., 0) as the lower bound [18]. Figure 2 illustrates the changes in values before and after applying MinMax normalization, where all values are transformed into a range between 0 and 1.

	dd	db
1	1409	439
2	1492	187
3	4639	1003
4	3497	392
5	248	18
6	3146	307
7	63	1
8	1207	513
9	9257	1888
10	321	129

	dd	db
1	0.00572881	0.0117430
2	0.00606627	0.0050021
3	0.0188616	0.0268297
4	0.0142183	0.0104858
5	0.00100834	0.0004815
6	0.0127912	0.0082121
7	0.00025615	0.0000267
8	0.0049075	0.0137224
9	0.0376377	0.0505029
10	0.00130514	0.0034507

Fig. 2. A comparison of the data distribution before (left) and after (right) applying Min-Max normalization

Feature selection is the final step in the pre-processing stage, involving experiments in Orange Data Mining to identify the smallest number of features that achieve the highest accuracy. This technique is essential for dimensionality reduction, as it helps to identify the most relevant features for machine learning applications. By reducing the dataset size and eliminating redundant or non-essential features, feature selection enhances model performance, speeds up the training process, and simplifies model development [22].

This study uses the Chi-squared method for feature selection, a widely used non-parametric statistical test. The Chi-squared test is significant for nominal dependent variables, but does not measure the strength of associations. It is unaffected by the order of categories and focuses on differences between groups. Additionally, it can be adapted for interval or ratio data converted into ordinal categories [23]. The formula for calculating feature selection using the Chi-square method is provided in Equation 2.

$$x^2 = \sum \frac{(O_i - E_i)}{E_i} \quad (2)$$

O_i represents the actual observed value (observed frequency), E_i is the expected value or expectation (expected frequency), and i refers to each category or group.

This feature selection method is then implemented on the most optimal classification algorithms discussed in the following section.

D. Modeling

Four machine learning algorithms—K-Nearest Neighbors (KNN), Random Forest, Naive Bayes, and XGBoost—are trained on the pre-processed data. Their performance will be evaluated and compared without feature selection to understand

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024

their inherent capabilities with the given dataset.

KNN predicts the category of a new data instance by examining its proximity to previously categorized instances. The algorithm first locates the 'k' nearest neighbors to a new instance. Then, it uses a majority vote among these neighbors to assign a category to the new instance [24]. To determine 'nearest' neighbors, KNN calculates the distance between new data points and those already classified. While various distance metrics can be used, the Euclidean distance is the most common choice for this calculation [18]. Euclidean distance formula is presented in Equation 3.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \quad (3)$$

Random Forest enhances predictive accuracy by combining multiple decision trees from bootstrapped data subsets. Until a stopping requirement is satisfied, this procedure repeatedly divides the data at each node according to chosen features. Combining the various forecasts of each decision tree yields the final prediction, especially when there is little association between them [25].

For classification, the Naive Bayes algorithm uses a probabilistic approach. It assumes strong independence among features, implying that the value of one feature does not influence or depend on the values of other features in the dataset [26]. It relies on Bayes' theorem for making predictions, as represented in Equation 4.

$$P(H|E) = \frac{P(E|H) \times P(H)}{P(E)} \quad (4)$$

Bayes' theorem calculates the updated probability of a hypothesis (P(H|E)) after considering new evidence (E). P(E|H) quantifies how likely the evidence is to occur given the hypothesis. P(H) represents the initial belief in the hypothesis before observing any evidence. Finally, P(E) is the probability of the evidence occurring, independent of the hypothesis [26].

XGBoost is a high-performing and accurate ensemble learning method that integrates the power of gradient boosting with a tree-based architecture. It comprises multiple decision trees, each functioning as a weak learner, with boosting techniques enhancing its performance to create a strong learner. In classification tasks, while the overall predictive accuracy of each weak classifier may be relatively low, it can achieve significantly high accuracy in specific aspects of the data [27].

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), + f_k \in \Gamma \quad (5)$$

Equation 5 formalizes the prediction process in XGBoost. The final prediction $\hat{y}_i$ for a given input x_i is derived by adding the contributions of 'K' individual regression trees. Each *k*-th tree produces a prediction score of f_k, and these scores are accumulated to arrive at the final output. Γ represents the space of all possible regression trees [28].

E. Evaluation

Given the limitations of the dataset, 10-fold cross-validation is utilized to ensure a thorough evaluation of the models. This technique partitions the dataset into ten distinct subsets. The data is segmented into ten subsets. For each iteration of the cross-validation process, a single subset is held out for validation while the model is trained on the remaining nine. This process is repeated ten times to ensure a comprehensive evaluation and maximize data utilization [18].

The algorithms' performance undergoes a rigorous evaluation process using a visualization tool known as a confusion matrix. This matrix is essential for gaining insights into the model's predictions and comprehending its classification accuracy. True Negatives (TN), True Positives (TP), False Negatives (FN), and False Positives (FP) are the four categories of categorization outcomes that are distinguished by the confusion matrix [18]. Every category denotes a distinct classification process result. For example, TP denotes the cases in which the model correctly detects a positive case, whereas FN denotes the cases in which a positive case is incorrectly categorized as negative. It is crucial to comprehend these categories in order to evaluate the model's dependability. Equation 6 illustrates how the confusion matrix produces a crucial metric, accuracy, which assesses the general validity of the model's predictions. [29].

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \times 100\% \quad (6)$$

A balanced performance statistic, the F1-score takes into account both recall (Equation 8) and precision (Equation 7). This gives a more comprehensive picture of model effectiveness and is particularly crucial when one metric is much lower [30]. Equation 9 is utilized to determine the F1-score.

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

$$F1 - Score = \frac{2 \times (Precision \times Recall)}{Precision + Recall} \quad (9)$$

III. RESULT AND DISCUSSION

This study draws upon three datasets from the UCI Machine Learning Repository. These datasets include benign software files (goodware), a diverse set of malware samples compiled by VirusTotal, and an additional malware dataset sourced from VxHeaven. These distinct datasets comprehensively represent both benign and malicious software for our analysis. After merging the two malware files, the combined malware dataset has 1,087 features, including the label, while the goodware dataset has 1,086 features, also including the label. To standardize the datasets, two features (_vbaVarIndexLoad and SafeArrayPtrOfIndex) were removed from the malware dataset, and one feature (Feature 1) was removed from the goodware dataset. These adjustments aligned the feature sets in both datasets.

The goodware dataset contains 595 records, while the malware dataset has 5,653 records, leading to class imbalance. This study addressed the issue by randomly sampling 595 records from the malware dataset. The malware and goodware datasets were then merged, resulting in 1,190 records. After merging, the number of features was reduced to 983, retaining only the common features from both datasets.

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024

Four machine learning algorithms—Random Forest, kNN, Naive Bayes, and Extreme Gradient Boosting—were assessed without feature selection after the data was prepared. According to preliminary findings, XGBoost performed better than the other models, obtaining an F1-Score and accuracy of 99.1% (see Table 2). The Chi-Square approach of feature selection was used to improve performance even more. This led to a significant improvement, as seen in Table 3, where XGBoost achieved 99.2% accuracy and F1-Score with just 38 characteristics. This demonstrates how feature selection works: enhancing the model's performance by incorporating only the most pertinent features led to increased accuracy and fewer features, which increased computational efficiency.

TABLE II. PERFORMANCE COMPARISON OF VARIOUS CLASSIFIERS WITHOUT FEATURE SELECTION

Algorithm	Accuracy	F1-Score
kNN	95.9%	95.9%
Naïve Bayes	93.1%	93.1%
Random Forest	98.3%	98.3%
XGBoost	**99.1%**	**99.1%**

TABLE III. PERFORMANCE OF XGBOOST WITH FEATURE SELECTION

Number of Features	XGBoost with X^2	
	Accuracy	F1-Score
33	99.1%	99.1%
34	99.1%	99.1%
35	99.1%	99.1%
36	99.1%	99.1%
37	99.1%	99.1%
38	**99.2%**	**99.2%**
39	99.2%	99.2%
40	99.2%	99.2%
41	99.1%	99.1%
42	99.2%	99.2%

Reducing the number of features not only improved accuracy and F1-score but also significantly enhanced the model's efficiency. This dimensionality reduction led to faster training and prediction times, which can be observed in Fig. 3 and Table 4.

TABLE IV. PERFORMANCE COMPARISON OF XGBOOST WITH AND WITHOUT FEATURE SELECTION

Number of Features	Feature Selection	Accuracy	F1-Score	Train Time	Test Time
All Features (982)	No Feature Selection	99.1%	99.1%	2.102	0.168
38 Features	X^2	99.2%	99.2%	0.746	0.018
Time Reduction				64.50%	89.28%

The results in Table 4 demonstrate that employing all 982 features without feature selection resulted in an accuracy and F1-score of 99.1% for the XGBoost model. However, this came at the cost of increased computational time, with training requiring 2.102 seconds and testing 0.168 seconds. By incorporating Chi-square (χ^2) feature selection and reducing the features to 38, the model achieved a comparable accuracy and F1-score of 99.2% while significantly reducing training time to 0.746 seconds and testing time to 0.018 seconds. This represents a substantial improvement in efficiency, with a 64.50% reduction in training time and an 89.28% reduction in testing time.

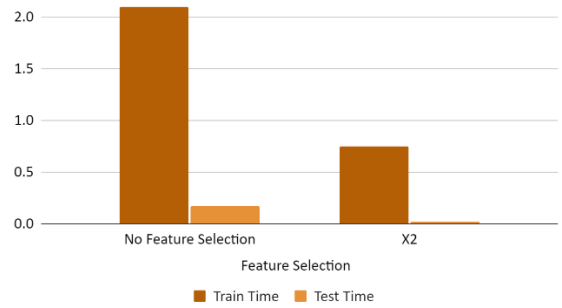

Fig. 3. Diagram of Time Reduction

As visualized in Fig. 3, Chi-square feature selection markedly decreases training and testing time. This highlights the impact of feature selection on enhancing computational efficiency while preserving high accuracy.

The experimental results, based on the same dataset, demonstrated that XGBoost, combined with Chi-squared feature selection, achieved the highest accuracy compared to previous methods in the literature. Table 5 below provides a performance comparison between this model and the results from existing studies.

TABLE V. PERFORMANCE COMPARISON OF OUR PROPOSED WORK WITH STATE-OF-THE-ARTS

No.	Methods	Number of Features	Accuracy	F1-Score
1.	**XGBoost + X^2 (proposed)**	**38**	**99.2%**	**99.2%**
2.	kNN + IG [14]	32	97.0%	97.0%
3.	DT + X^2 [16]	30	98.53%	98.53%

A thorough analysis of the accuracy gain attained by various feature selection techniques and algorithms is provided in Table 5. Using 38 features and combining XGBoost with Chi-square feature selection yields the best results, with an accuracy and F1-score of 99.2%.

Several feature selection techniques and models were investigated, but none were able to equal XGBoost's performance. The lowest scores, 97.0% for accuracy and F1-score, were obtained by kNN with Information Gain (32 features). With a 98.53% accuracy rate and a 98.53% F1-score, the Decision Tree (DT) with Chi-square (30 features) outperformed the others. XGBoost consistently outperformed the other algorithms, proving its supremacy for this malware detection task, even though DT performed well. This implies that the best model for achieving high accuracy is XGBoost, which has a larger feature set and effectively uses Chi-square feature selection.

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024

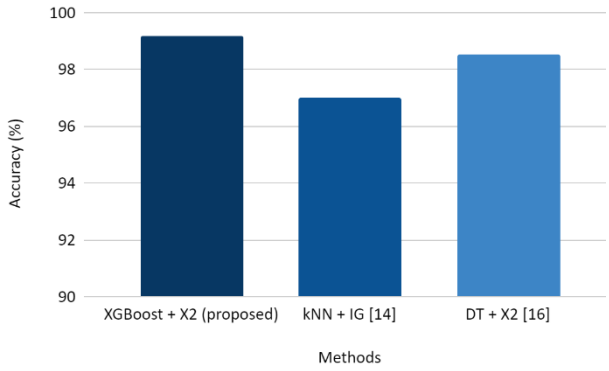

Fig. 4. Diagram of Performance Analysis Result

Fig. 4 illustrates the performance comparison among three methods: XGBoost with Chi-square, kNN with Information Gain, and Decision Tree with Chi-square. The results show that XGBoost with Chi-square achieves the highest accuracy, indicating its effectiveness for this dataset. In contrast, kNN with Information Gain has the lowest accuracy among the three methods. At the same time, the Decision Tree with Chi-square provides competitive results, performing better than kNN but slightly below XGBoost. This comparison underscores the superior performance of XGBoost combined with Chi-square feature selection.

In evaluating the model's effectiveness for malware detection, the confusion matrix plays a multifaceted role. It provides a visual representation of the model's predictions—arranging them into TP, TN, FP, and FN—and facilitates a deeper understanding of its performance. This detailed breakdown yields insights into the model's classification patterns, overall accuracy, and specific areas of success and failure. Moreover, the confusion matrix is essential for computing key metrics such as F1-score, recall, and precision. Fig. 5 displays the confusion matrix for this study.

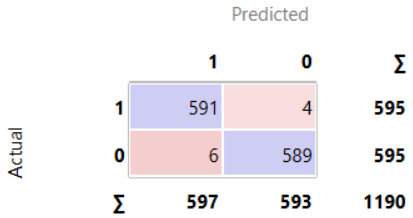

Fig. 5. Confusion Matrix of XGBoost

Analysis of the confusion matrix in Fig. 5 reveals that the model correctly classified 591 instances as positive (True Positives) and 589 as negative (True Negatives). However, there were also misclassifications, as four instances were incorrectly predicted as positive (False Positives), and six were incorrectly predicted as negative (False Negatives). The total number of samples encompassing correct and incorrect classifications across both classes is 1190.

V. CONCLUSION

This study showed how the Chi-Square Feature Selection method significantly improved XGBoost's malware detection ability. The accuracy of the model rose from 99.1% to 99.2% after feature selection was added, and testing time was significantly decreased. This emphasizes how crucial it is to give pertinent features top priority in order to maximize accuracy, F1-Score, and computational efficiency. The findings show that using Chi-Squared Feature Selection significantly boosts the effectiveness of machine-learning models intended for malware detection. To further improve real-time detection systems, future research could look at hybrid models or other feature selection techniques such recursive feature elimination.

REFERENCES

[1] N. A. Azeez, O. E. Odufuwa, S. Misra, J. Oluranti, and R. Damaševičius, "Windows PE Malware Detection Using Ensemble Learning," *Informatics*, vol. 8, no. 1, p. 10, Feb. 2021, doi: 10.3390/informatics8010010.

[2] N. Pachhala, S. Jothilakshmi, and B. P. Battula, "A Comprehensive Survey on Identification of Malware Types and Malware Classification Using Machine Learning Techniques," in *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India: IEEE, Oct. 2021, pp. 1207–1214. doi: 10.1109/ICOSEC51865.2021.9591763.

[3] N. Dutta, N. Jadav, S. Tanwar, H. K. D. Sarma, and E. Pricop, "Introduction to Malware Analysis," in *Cyber Security: Issues and Current Trends*, vol. 995, in Studies in Computational Intelligence, vol. 995. , Singapore: Springer Singapore, 2022, pp. 129–141. doi: 10.1007/978-981-16-6597-4_7.

[4] N. Adeel, R. Kumar, K. N. S. Akella, V. Manickam, M. W. Khan, and S. V. Nandury, "Measuring the Implications of Email Viruses Through a Unified Model of Cyber Security," in *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, Gautam Buddha Nagar, India: IEEE, Sep. 2023, pp. 614–621. doi: 10.1109/IC3I59117.2023.10398148.

[5] R. Vanness, M. M. Chowdhury, and N. Rifat, "Malware: A Software for Cybercrime," in *2022 IEEE International Conference on Electro Information Technology (eIT)*, Mankato, MN, USA: IEEE, May 2022, pp. 513–518. doi: 10.1109/eIT53891.2022.9813970.

[6] A. M. Kovács, "Ransomware: a comprehensive study of the exponentially increasing cybersecurity threat," *IRD*, vol. 4, no. 2, pp. 96–104, Jun. 2022, doi: 10.9770/IRD.2022.4.2(8).

[7] M. Aljaidi *et al.*, "NHS WannaCry Ransomware Attack: Technical Explanation of The Vulnerability, Exploitation, and Countermeasures," in *2022 International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI)*, Zarqa, Jordan: IEEE, Nov. 2022, pp. 1–6. doi: 10.1109/EICEEAI56378.2022.10050485.

[8] C. M. Codreanu, "Exploring the need for human-centred cybersecurity. The WannaCry Cyberattack," vol. 15, no. 2, 2021.

[9] B. Fiore, K. Ha, L. Huynh, J. Falcon, R. Vendiola, and Y. Li, "Security Analysis of Ransomware: A Deep Dive into WannaCry and Locky," in *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA: IEEE, Mar. 2023, pp. 285–294. doi: 10.1109/CCWC57344.2023.10099114.

[10] A. Muzaffar, H. Ragab Hassen, M. A. Lones, and H. Zantout, "An in-depth review of machine learning based Android malware detection," *Computers & Security*, vol. 121, p. 102833, Oct. 2022, doi: 10.1016/j.cose.2022.102833.

[11] K. D. K. Wardhani and M. Akbar, "Diabetes Risk Prediction Using Extreme Gradient Boosting (XGBoost)," *join*, vol. 7, no. 2, pp. 244–250, Dec. 2022, doi: 10.15575/join.v7i2.970.

[12] J. Palša *et al.*, "MLMD—A Malware-Detecting Antivirus Tool Based on the XGBoost Machine Learning Algorithm," *Applied Sciences*, vol. 12, no. 13, p. 6672, Jul. 2022, doi: 10.3390/app12136672.

[13] R. Kumar and G. S, "Malware classification using XGboost-Gradient Boosted Decision Tree," *Adv. sci. technol. eng. syst. j.*, vol. 5, no. 5, pp.

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024

536–549, 2020, doi: 10.25046/aj050566.
[14] F. A. Rafrastara, C. Supriyanto, A. Amiral, S. R. Amalia, M. D. Al Fahreza, and F. Ahmed, "Performance Comparison of k-Nearest Neighbor Algorithm with Various k Values and Distance Metrics for Malware Detection," *mib*, vol. 8, no. 1, p. 450, Jan. 2024, doi: 10.30865/mib.v8i1.6971.
[15] T. Lu, Y. Du, L. Ouyang, Q. Chen, and X. Wang, "Android Malware Detection Based on a Hybrid Deep Learning Model," *Security and Communication Networks*, vol. 2020, pp. 1–11, Aug. 2020, doi: 10.1155/2020/8863617.
[16] M. Abujazoh, D. Al-Darras, N. A. Hamad, and S. Al-Sharaeh, "Feature Selection for High-Dimensional Imbalanced Malware Data Using Filter and Wrapper Selection Methods," in *2023 International Conference on Information Technology (ICIT)*, Amman, Jordan: IEEE, Aug. 2023, pp. 196–201. doi: 10.1109/ICIT58056.2023.10226049.
[17] O. N. Elayan and A. M. Mustafa, "Android Malware Detection Using Deep Learning," *Procedia Computer Science*, vol. 184, pp. 847–852, 2021, doi: 10.1016/j.procs.2021.03.106.
[18] C. Supriyanto, F. A. Rafrastara, A. Amiral, S. R. Amalia, M. D. Al Fahreza, and Mohd. F. Abdollah, "Malware Detection Using K-Nearest Neighbor Algorithm and Feature Selection," *mib*, vol. 8, no. 1, p. 412, Jan. 2024, doi: 10.30865/mib.v8i1.6970.
[19] Rishitha Venumuddala and J. Krishna, "Methodological approach for designing a data pre- processing tool on textual data," 2022, doi: 10.13140/RG.2.2.18627.27689.
[20] T. D. Nguyen, M.-H. Shih, D. Srivastava, S. Tirthapura, and B. Xu, "Stratified random sampling from streaming and stored data," *Distrib Parallel Databases*, vol. 39, no. 3, pp. 665–710, Sep. 2021, doi: 10.1007/s10619-020-07315-w.
[21] M. Shantal, Z. Othman, and A. A. Bakar, "A Novel Approach for Data Feature Weighting Using Correlation Coefficients and Min–Max Normalization," *Symmetry*, vol. 15, no. 12, p. 2185, Dec. 2023, doi: 10.3390/sym15122185.
[22] M. Büyükkeçeci and M. C. Okur, "A Comprehensive Review of Feature Selection and Feature Selection Stability in Machine Learning," *Gazi University Journal of Science*, vol. 36, no. 4, pp. 1506–1520, Dec. 2023, doi: 10.35378/gujs.993763.
[23] K. Kishore and V. Jaswal, "Statistics Corner: Chi-squared Test," *Journal of Postgraduate Medicine, Education and Research*, vol. 57, no. 1, pp. 40–44, Apr. 2023, doi: 10.5005/jp-journals-10028-1618.
[24] D. Sitanggang, A. S. Ginting, R. M. Simanjuntak, and N. Lumbantoruan, "EEG Signal Classification using K-Nearest Neighbor Method to Measure Impulsivity Level," *SISFOKOM*, vol. 13, no. 2, pp. 261–266, Jun. 2024, doi: 10.32736/sisfokom.v13i2.2154.
[25] J. Hu and S. Szymczak, "A review on longitudinal data analysis with random forest," *Briefings in Bioinformatics*, vol. 24, no. 2, p. bbad002, Mar. 2023, doi: 10.1093/bib/bbad002.
[26] L. M. Sinaga, Sawaluddin, and S. Suwilo, "Analysis of classification and Naïve Bayes algorithm k-nearest neighbor in data mining," *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 725, no. 1, p. 012106, Jan. 2020, doi: 10.1088/1757-899X/725/1/012106.
[27] K. Wang, M. Li, J. Cheng, X. Zhou, and G. Li, "Research on personal credit risk evaluation based on XGBoost," *Procedia Computer Science*, vol. 199, pp. 1128–1135, 2022, doi: 10.1016/j.procs.2022.01.143.
[28] S. Chehreh Chelgani, H. Nasiri, and A. Tohry, "Modeling of particle sizes for industrial HPGR products by a unique explainable AI tool- A 'Conscious Lab' development," *Advanced Powder Technology*, vol. 32, no. 11, pp. 4141–4148, Nov. 2021, doi: 10.1016/j.apt.2021.09.020.
[29] D. A. Anggoro, "Comparison of Accuracy Level of Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) Algorithms in Predicting Heart Disease," *IJETER*, vol. 8, no. 5, pp. 1689–1694, May 2020, doi: 10.30534/ijeter/2020/32852020.
[30] J. Asian, M. Dholah Rosita, and T. Mantoro, "Sentiment Analysis for the Brazilian Anesthesiologist Using Multi-Layer Perceptron Classifier and Random Forest Methods," *join*, vol. 7, no. 1, pp. 132–141, Sep. 2022, doi: 10.15575/join.v7i1.900.

p-ISSN 2301-7988, e-ISSN 2581-0588
DOI : 10.32736/sisfokom.v13i3.2293, Copyright ©2024
Submitted : October 9, 2024, Revised : November 11, 2024, Accepted : November 13, 2024, Published : November 21, 2024