JIRBDAI

Journal of Institutional Research, Big Data Analytics and Innovation

TREE-LIKE ALGORITH MS FOR RANSOMWARE ATTACK DETECTION

Aro Taye Oladele[1], Ayinla Olatunji Mutiu[2], Koce Adayilo Kenneth[3], Mustapha Isah Olawale[4], Jimoh-Mahmud Oladayo Aishat[5], Musa Usman Bala[6], Tajudeen Olanrewaju Toyyib[7]

[1,6]Department of Data Science and Information Technology, Alhikmah University Ilorin, Nigeria
[3]Department of Computer Science, Federal Polytechnic Bida, Niger State, Nigeria
[6]Directorate of Information and Communication Technology, Ibrahim Badamasi Babangida University, Lapai, Niger State, Nigeria
[2,4,7] Department of Computer Science, Alhikmah University Ilorin, Nigeria
Corresponding Author's Email:taiwo774@gmail.com

ABSTRACT Ransomware attack remains a significant threat to cybersecurity affecting people, businesses, and governments. The prompt and accurate detection of these attacks is crucial in reducing their adverse effects. This paper developed a ransomware attack detection applying some selected tree-like learning algorithms, The study applied three feature selection techniques; selectKBest, recursive feature elimination, and feature importance to the Kaggle ransomware dataset to obtain reduced attributes before the classification phase, which involved the application of Decision Trees (DT), Extra Trees (ET), and LightGBM (LGBM). The model was evaluated in terms of accuracy, precision, recall, F1-score, and ROC AUC. ET and LGBM performed best in detecting ransomware, consistently outperforming the DT. ET achieved the highest F1-score value of 0.9959 and a near-perfect ROC-AUC of 0.9993 without feature selection, showing its ability to maintain a strong balance between precision and recall. LGBM excelled with an F1-score of 0.9957 and the highest ROC AUC of 0.9997, showcasing its superior power to differentiate between benign and malicious actions associated with ransomware. The DT model despite being slightly more effective than ET and LGBM still delivered competitive results, indicating that the simpler models can be valuable for the detection of ransomware, predominantly in applications where the priority is computational efficiency. The impact of feature selection on model performance was also explored, with the best results typically achieved without any feature selection. Finally, it was revealed that tree-like learning algorithms, particularly Extra Trees and LightGBM, are highly effective tools for detecting ransomware, providing a robust defence against the increasing ransomware threat.

KEYWORDS: Selectkbest, Feature Importance, Recursive Feature Elimination, Ransomware, Tree-like Algorithms.

I. INTRODUCTION

Ransomware can be classified as a type of hateful software that affects useful software, it has grown greatly since its early advent in the late 1980s [1]. Firstly employed for individual blackmail, ransomware has now heightened to corporate extortion, causing serious threats to business organizations and the overall internet ecosystem [2]. The attacks show significant security threats to private (confidential) and corporate data [3][4]. People using computer resources face some challenges such as monetary losses, authentication and privacy defilements, and damage to reputation due to effective ransomware attacks. These threats have developed more regular and classy in recent years, leading to financial losses, disruption of operations, and reputational damage to businesses[5]. Ransomware is performed by the encryption of important data and asking for a ransom for its release, making the old signature-based detection techniques ineffective [6] [7]. Hence, there is a need to explore more effective approaches to reduce this rapidly evolving threat. Ransomware encrypts the user's information or a complete system, rendering the information unuseful, and then asks for a ransom from the victim's computer in exchange for the decryption key [8][9]. These attacks have increased in recent years, leading to significant financial and reputational destruction to people and organizations. Detecting and

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

classifying ransomware is a crucial task for cybersecurity experts to conduct and protect against attacks [10]. The detection and classification of ransomware include identifying its features and behaviour, distinguishing it from other malware, and identifying its origin and attack vectors [11]. The application of learning algorithms in ransomware detection is an evolving study domain with several usage potentials in developing anti-ransomware systems [12].

Malware, including ransomware, can be identified quickly by its unpredictable actions using machine learning approaches, thereby improving security [13]. Machine learning allows computers to learn and detect patterns in massive amounts of data and make predictions based on those patterns (Kaur et al., 2023). In the context of ransomware detection and classification, machine learning can be used to analyze many aspects of ransomware attacks, such as the type of encryption employed, the attack vector, and the ransomware's behaviour [3]; this study examined machine-learning strategies to identify and classify ransomware attacks. Particularly, the developed ransomware detection model can analyze and predict different ransomware attacks. This paper presents a comparative analysis of some selected tree-like learning algorithms like Decision Trees (DT), Extra Trees (ET), and LightGBM (Light Gradient Boosting Machine) that affect ransomware attack detection and their performances.

II. RELATED WORK

Ayinla et al, [7] reviewed several studies on the application of machine-learning techniques to detect ransomware attacks in computer networks. Relevant literature was obtained from various research databases by employing keywords and search strings deeply related to the topic. A good number of papers were selected, and also arranged and considered. The literature was organized into many sections, organized sequentially from the most current to old papers. The year of publication for the reviewed papers ranges from 2017 to 2023. The review started with an exploration of some related concepts and then further moved to machine learning algorithms that have been proposed for the detection of ransomware attacks. The performances of the several learning techniques employed for ransomware attack detection in computer networks were started. The important aspect of ransomware detection was built using learning algorithms particularly Decision Tree (DT) [15]. The study explored the significance of DT in ransomware attack detection considering the distinctive potential of DT to identify intricate patterns within datasets. An approach was used to replace DT, emphasizing the importance of fine-tuning DT hyperparameters. The parameters optimization is crucial for the maximization of the DT's ability to detect ransomware threats correctly. The efficiency of DT in detecting ransomware was tested using the obfuscated MalMem2022 dataset. Results showed that the DT classifier consistently performed better than other classifiers in ransomware detection, including K-Nearest Neighbors, Gradient Boosting Tree, Naive Bayes, and Linear Support Vector Classifier. DT gave incomparable efficiency in differentiating between Ransomware and benign data, as evidenced by its remarkable accuracy of 99.97%.

A survey of ransomware detection, a timeline of assaults, and details on their nature [3]. Was also provided in the work a detailed study on current methods for detecting, avoiding, reducing, and recovering from ransomware attacks. An analysis of studies between 2017 and 2022 is another advantage of the study. This gave readers current information on the most recent developments in ransomware detection and highlighted the advancements in methods for combating ransomware attacks. Hossain et al., [17] came up with a new ransomware detection method in an android environment using traffic analysis to address the limitations. The traffic characteristics were obtained by applying Particle swarm optimization (PSO). Classification of the selected traffic features for the data traffic was achieved using a decision tree and random forest classifiers. The ransomware cyberattacks were examined in two distinct cases. In the first case, the ransomware traffic; in the second case, the location of a specific form of malware traffic in benign traffic. The best detection rate of ransomware was recorded in a random forest, whereas the decision tree gave the best result for detecting the types of ransomware. The improvement of accuracy improvements were 2.26% and 3.7% in the first and second scenarios, respectively. A novel approach was applied using static analysis to detect ransomware [18]. The significant characteristic of the proposed method is the dispensing of the disassembly process by direct extraction of features from raw byte by application of frequent mining of pattern which effectively increased the time of detection. The Gain Ratio method was employed for feature selection which showed that 1000 features was the optimal number for the detection process. A random forest classifier with a comprehensive analysis of the effect of both tree and seed numbers on ransomware detection was applied. Results showed that tree numbers of 100 with a seed number of 1 gave the best results in terms of time-consuming and

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

accuracy. The experimental evaluation revealed that the proposed method could achieve a high accuracy of 97.74% for the detection of ransomware.

III. METHODOLOGY

A. The Approach to Ransomware Detection Model

This study approach made use of the stepwise method. The dataset was acquired to gather data for the evaluation of the ransomware detection model. The available online Kaggle ransomware dataset was employed with preprocessing before being passed to the learning algorithms. Some tree-like learning algorithms, like Decision Trees, Extra Trees, and LightGBM, were applied to classify the ransomware attacks. The tree-like algorithms have been identified to show robust patterns for ransomware detection due to their capacity to handle complicated data, nonlinear correlations in data, and interpretability. The sklearn train_test_split function was used to divide the dataset into training and test data in the 80:20 ratio, with 80% of the data used to train the model and 20% used for testing. The final training dataset contained 49988 samples, while the test dataset contained 12497 samples. Figure 1 depicts the framework of the tree-like detection model.

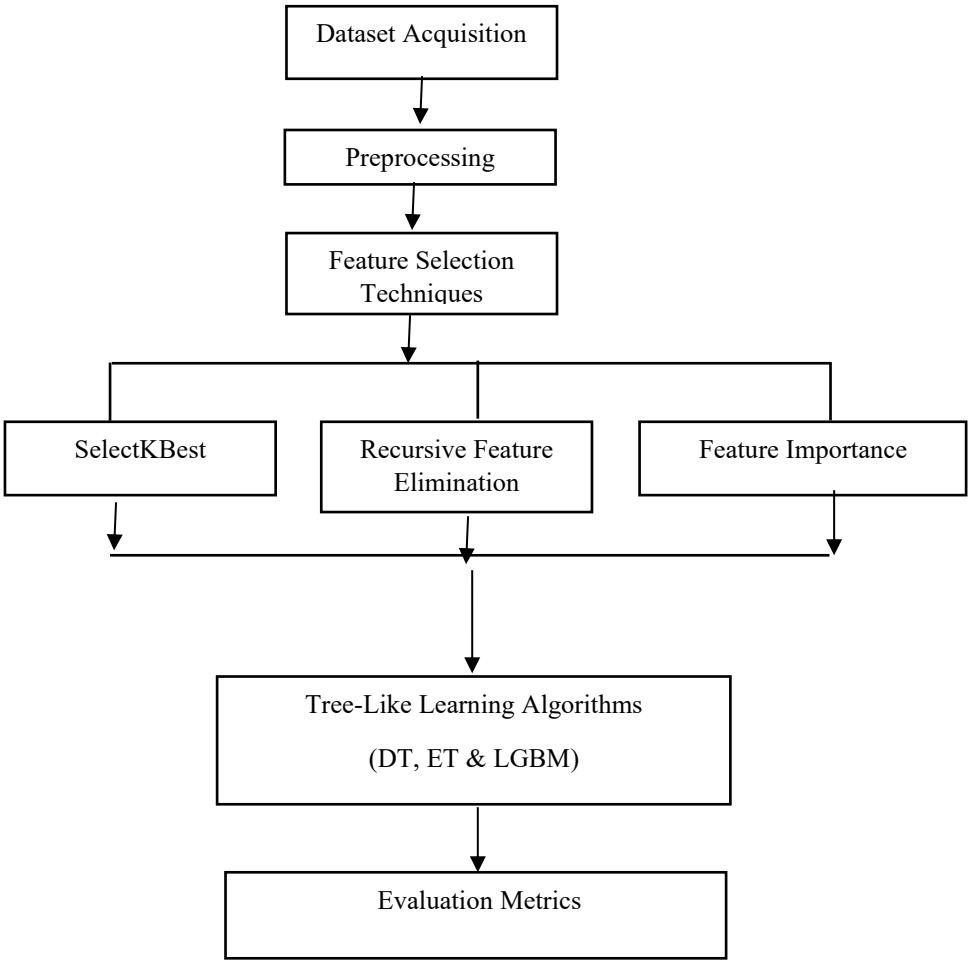


Figure 1. Ransomware Attack Detection Framework

B. Acquisition of Dataset

This work used Kaggle's ransomware dataset. The dataset was applied because of its ability to publish and sharing of datasets privately or publicly. The dataset provides adequate resources needed to store and process datasets. The dataset has 62485 unique values and 18 features. Table 1 shows the metadata from the ransomware detection dataset. The dataset contains 27118 legitimate collections, whereas the rest 35367 are

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

malicious collections. Figure 2 depicts the ratio of 43.499% legitimate and 56.60% non-legitimate data components. The selected dataset focuses on the changes that the system goes through when it is attacked by malware in terms of Debug Size, which is the size of the debug directory; Major Image Version, which is the version of the file; Major OS Version, Export RVA, Export Version, and IatRVA, which is the virtual address of the import file. The table below demonstrates the changes observed when a system or file malfunctions or becomes corrupted as shown in Table 1 Range index: 62485 entries, 0 to 62484 Data columns (total 18 Features):

Table 1 : Metadata: Ransomware Detection Dataset

#	Features	Non-Null Count	Dtype
0	FileName	62485 non-null	object
1	md5Hash	62485 non-null	object
2	Machine	62485 non-null	int64
3	DebugSize	62485 non-null	int64
4	DebugRVA	62485 non-null	int64
5	MajorImageVersion	62485 non-null	int64
6	MajorOSVersion	62485 non-null	int64
7	ExportRVA	62485 non-null	int64
8	Exportsize	62485 non-null	int64
9	latVRA	62485 non-null	int64
10	MajorLinkerVersion	62485 non-null	int64
11	MinorLinkerVersion	62485 non-null	int64
12	NumberOfSections	62485 non-null	int64
13	SizeOfStackReserve	62485 non-null	int64
14	DllCharacteristies	62485 non-null	int64
15	ResourceSize	62485 non-null	int64
16	BitcoinAddresses	62485 non-null	int64
17	Benign	62485 non-null	int64
dtypes: int64(16), object(2)			

C. Types of the Data Features

From the exploratory analysis of the dataset, it was also observed that the dataset has 17 input attributes and one target class. The data types in the dataset include two features with Object (object) while sixteen features are on integer types (int64). Index(['FileName', 'md5Hash', 'Machine', 'DebugSize', 'DebugRVA', 'MajorImageVersion', 'MajorOSVersion', 'ExportRVA', 'ExportSize', 'IatVRA', 'MajorLinkerVersion', 'MinorLinkerVersion', 'NumberOfSections', 'SizeOfStackReserve', 'DllCharacteristics', 'ResourceSize', 'BitcoinAddresses', 'Benign'], dtype='object'). The ransomware detection model is shown in Figure 2.

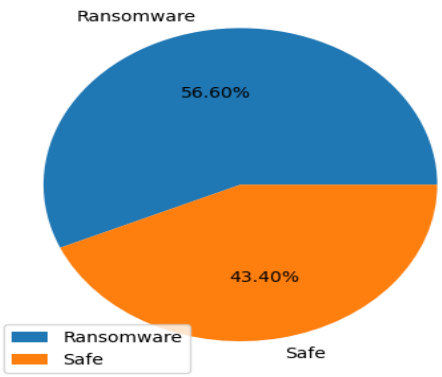


Figure 2. Ransomware Detection Dataset Ratio

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

D. Dataset Correlation Heatmap

All fields, apart from FileName and md5Hash, are numerical. The field md5Hash can be used to check if there are any duplicates as depicted in Figure 3.

Index(['Machine', 'DebugSize', 'DebugRVA', 'MajorImageVersion', 'MajorOSVersion', 'ExportRVA', 'ExportSize', 'IatVRA', 'MajorLinkerVersion', 'MinorLinkerVersion', 'NumberOfSections', 'SizeOfStackReserve', 'DllCharacteristics', 'ResourceSize',
'BitcoinAddresses'],
dtype='object')

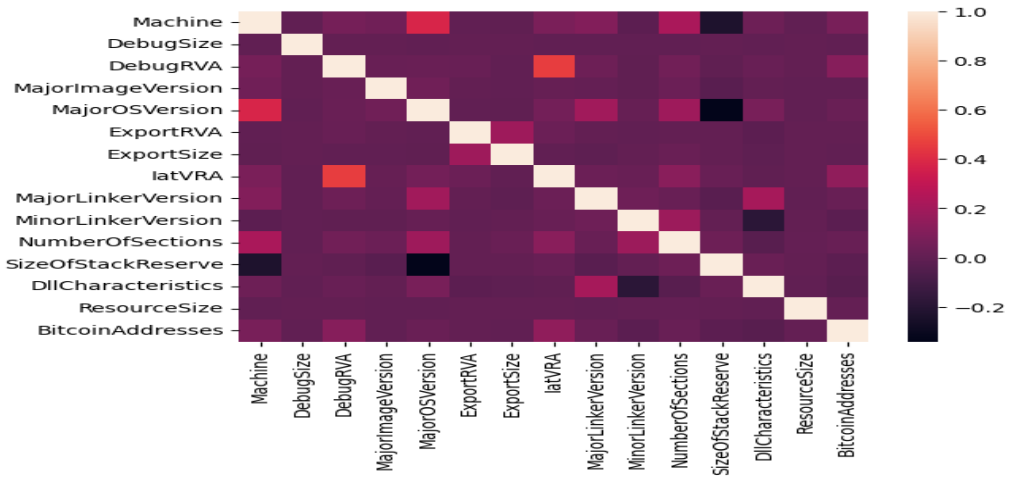


Figure 3: Heatmap to visualize the correlation

E. Preprocessing of Dataset

The dataset chosen for this study stands out for its cleanliness and clarity, with no null or missing values. The dataset's immaculate condition eliminates the requirement for preparation techniques that are generally used to handle such faults in data quality. Without any missing values to address in the study, the research can proceed to the analysis phase, where it can use the dataset's integrity to draw relevant insights and conclusions. The absence of null or missing values not only speeds the research process but also improves the reliability and validity of the dataset's conclusions. Researchers can confidently use the dataset in its original state, reducing the possibility of bias or distortion in the analysis. Thus, the decision to omit preprocessing measures emphasizes the resilience and applicability of the chosen dataset for the research objectives.

F. Algorithms for Feature Selection

After the dataset split step, three feature selection techniques were applied due to the robust nature selection of features prior the training of dataset. The featyres were fed th to Tree-like learning algorithms for classification. Each of the Algorithms employed is a Univariate Feature Selection (SelectKBest) from sklearn.feature_selection library, Recursive Feature Elimination (RFE), and Feature Importance from Tree-Based Models (Extra Trees) with k=12 for each.

SelectKBest

This is a filter-based feature selection strategy for machine learning. Filter-based feature selection approaches are independent of any particular machine learning algorithm. Instead, the traits are evaluated and ranked using statistical methods. It evaluates and ranks features using statistical tests such as the chi-square test, ANOVA F-test, and mutual information score as shown in Figure 4. Then, it selects the K features with the highest scores to include in the final feature set. It is easy to use and quickly reduces the feature set to a manageable size, which is essential when working with large datasets.

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

	Feature	Score
1	DebugSize	27526.344100
0	Machine	21846.190112
4	MajorOSVersion	8585.635523
6	MajorLinkerVersion	4787.720183
9	SizeOfStackReserve	4512.649134
10	DllCharacteristics	3660.450354
7	MinorLinkerVersion	631.743236
5	IatVRA	354.755710
2	DebugRVA	226.837739
11	BitcoinAddresses	144.136731
3	MajorImageVersion	115.383334
8	NumberOfSections	17.822210

Figure 4: Algorithm for Best Features(SelectKBest)

Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a wrapper-based strategy for repeatedly removing the least significant features based on a model's rating of feature relevance. The model is trained multiple times, and features are removed one by one until only the most relevant subset remains. RFE is appropriate for analyzing feature interactions since it considers how characteristics influence one another's impact on model predictions as shown in Figure 5.

	Feature	Score
1	DebugSize	0.449513
10	DllCharacteristics	0.309395
11	ResourceSize	0.070296
0	Machine	0.060031
4	MajorOSVersion	0.035670
8	MajorLinkerVersion	0.020695
9	NumberOfSections	0.019238
2	DebugRVA	0.013743
7	IatVRA	0.008243
3	MajorImageVersion	0.006579
6	ExportSize	0.004630
5	ExportRVA	0.001967

Figure 5: Algorithm for Best Features for (RFE)

Feature Importance

This technique is usually employed with models based on trees such as the Extra tree. It rates features based on how they help to reduce prediction error or impurity throughout the decision-making process. This strategy

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

provides significant insights into which features have the greatest influence on the model's predictions and is especially beneficial when using ensemble methods for classification or regression tasks.
The algorithm is shown in Figure 6.

	Feature	Score
11	DllCharacteristics	0.275769
10	Machine	0.242973
9	DebugSize	0.104864
8	MajorLinkerVersion	0.080610
7	MajorOSVersion	0.072884
6	SizeOfStackReserve	0.062130
5	NumberOfSections	0.053697
4	MinorLinkerVersion	0.033425
3	ResourceSize	0.020712
2	IatVRA	0.015728
1	MajorImageVersion	0.015199
0	DebugRVA	0.009081

Figure 6: Algorithm for Best Features(Feature Importance)

Tree-Like Learning Algorithms

Tree-like learning algorithms are machine-learning methods that utilize decision trees as the primary mechanism for classification. The tree-like learning algorithms are:

Decision Trees (DT)

Machine learning models such as decision trees are essential for both categorization and regression. They operate by recursively partitioning the dataset into subsets based on feature values that best distinguish the desired classes or values. Splits are often selected to maximize a criterion such as Gini impurity or information gain (entropy). Decision trees are simple to read and can handle nonlinear data effectively, but they are prone to overfitting, particularly when the tree grows too deep. They provide the foundation for several sophisticated models, including random forests and gradient boosting.

Extra Trees (Extremely Randomized Trees)

Extra Trees Classifier employs multiple decision trees to select features based on relevance ratings, reducing noise and irrelevant, and effectively handling large datasets. They are an ensemble method similar to Random Forests but with a different approach to split selection. Random Forests select the optimal split from a random group of features, whereas Extra Trees increase randomization by selecting the splitting point at random from the feature's value range. This additional randomization makes the model faster and less prone to overfitting than classic decision trees, while it may result in somewhat increased bias in some cases. Extra trees, like Random Forests, combine predictions from numerous trees to increase generality.

LightGBM (Light Gradient Boosting Machine)

LightGBM is an extremely efficient gradient-boosting framework that constructs trees consecutively to reduce prediction errors. Unlike typical gradient boosting methods, LightGBM employs a revolutionary methodology known as "leaf-wise" growth, in which it splits the leaf with the highest error reduction, resulting in deeper trees. This methodology, combined with enhancements such as histogram-based decision splits and efficient handling of categorical data, makes LightGBM significantly quicker and more memory-efficient than previous gradient-boosting approaches. Scalability and performance make it popular for large-scale datasets in competitive machine learning challenges. It is an optimized version of Gradient Boosting designed for efficiency and speed.

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

G. Performance Evaluation Metrics

Evaluating the machine learning model's performance for ransomware attack detection is vital to assess its effectiveness in identifying and preventing its spread. The evaluation metrics are as follows:

Accuracy

Accuracy measures the proportion of correctly classified instances (both positive and negative) out of the total number of instances. It is useful for balanced datasets, but in imbalanced datasets, it can give a false sense of performance if the majority class dominates the predictions. Equation (1).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Precision

Precision measures the proportion of true positive predictions among all instances predicted as positive (whether correctly or incorrectly). It quantifies the classifier's ability to avoid false positives. Equation (2)

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall (Sensitivity)

Recall measures the proportion of true positive predictions among all actual positive instances in the dataset. It quantifies the classifier's ability to capture all positive instances. Equation (3)

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

F1-Score

The F1-Score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall, making it useful for evaluating classifiers with imbalanced datasets as shown in Equation (3)

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

ROC Curve

The Receiver Operating Characteristic (ROC) curve is a graphical representation of the trade-off between true positive rate (Sensitivity) and false positive rate (1-Specificity) for different classification thresholds. Equation (3)

True Positive Rate (TPR)

Also known as **Recall** or **Sensitivity**. **True Positive Rate: TPR** measures the proportion of actual positives that are correctly identified by the model in Equation 4.

$$TPR = \frac{TP}{TP+FN} \quad (5)$$

False Positive Rate: FPR measures the proportion of actual negatives that are incorrectly identified as positive by the model.

$$FPR = \frac{FP}{FP+TN} \quad (6)$$

ROC Curve:

X-axis: False Positive Rate (FPR). Y-axis: True Positive Rate (TPR).

The Area Under the ROC Curve (AUC-ROC) quantifies the classifier's ability to distinguish between positive and negative instances, with higher values indicating better performance as shown in Equation (7)

$$\text{AUC} \approx \sum_{i=1}^{n-1} \left(\frac{TPR_{i+1}+TPR_i}{2}\right) X\ (FPR_i + 1 - FPR_i) \quad (7)$$

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

IV. RESULTS AND DISCUSSION

A. Experimental Analysis

All experimentation is conducted in Python a Windows-based System of Corei5 Processor, 500GB Hard Disk Drive (HDD), and 8GB RAM. The Python environment is an anaconda3 named Jupyter Notebook with various libraries such as Panda, sklearn, numpy, seaborn, Matplotlib, and selected packages.

B. Results of Developed Ransomware Detection

The results for the ransomware detection model were given according to different evaluation metrics used in this work. The following subsections discuss them in detail:

C. Result of Accuracy for Ransomware Detection Model

The accuracy results for various models and feature selection methods reveal that all models Decision Tree DT), Extra Trees (ET), and LightGBM (LGBM) perform best when no feature selection was used.

Table 2: Accuracy of the Ransomware Detection Model

Learning Algorithm	Attribute Selection			
	SelectKBest	Recursive Feature Elimination	Feature Importance	Without Feature Selection
DT	0.9910	0.9938	0.9933	0.9945
ET	0.9929	0.9954	0.9954	0.9965
LGBM	0.9926	0.9954	0.9951	0.9963

From Table 2, the Extra Trees model has the highest accuracy (0.9965), followed closely by LightGBM (0.9963). The Decision Tree model, while significantly less accurate, nevertheless achieves a high accuracy of 0.9945 without feature selection. Feature selection approaches such as Recursive Feature Elimination (RFE) and Feature Importance typically maintain good accuracy with just minor reductions when compared to the entire feature collection due to the decrease in the level of features for the classification phase. Which thus has contributed immensely to better performance of the detection model. SelectKBest consistently produces the lowest accuracy among all models, however, the loss is not considerable. This implies that, while feature selection can minimize complexity, it may also reduce the model's predictive power. Among the models, Extra Trees (ET) and LightGBM (LGBM) demonstrate superior accuracy, especially when using the full set of features. The Decision Tree (DT), although slightly less accurate, still performs well. The results imply that ensemble methods like ET and LGBM are more robust, particularly when all available features are utilized, making them preferable for scenarios where the highest accuracy is desired.

D. Results of the Precision for Ransomware Detection Model

The precision result of the developed model for ransomware detection is shown in Table 3

Table 3: Precision of the Ransomware Detection Model

Learning Algorithm	Attribute Selection			
	SelectKBest	Recursive Feature Elimination	Feature Importance	Without Feature Selection
DT	0.9868	0.9930	0.9926	0.9937
ET	0.9892	0.9945	0.9948	0.9963
LGBM	0.9894	0.9950	0.9959	0.9974

In Table 3, the precision scores are consistently high across all models, with the maximum precision obtained when no feature selection is used. LightGBM (LGBM) achieves the highest precision of 0.9974 without feature selection, followed by Extra Trees (ET) at 0.9963. The Decision Tree (DT) model, while significantly less accurate, still obtains a high precision of 0.9937 without feature selection. The precision has a slight

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

impact on the feature selection between models; both Recursive Feature Elimination (RFE) and Feature Importance maintain high precision, with minor changes from the entire feature set. However, SelectKBest produces the lowest precision across all models, particularly in the Decision Tree (DT) model, where it decreases to 0.9868. This implies that while feature selection can streamline the model, it may slightly diminish its ability to adequately identify positive instances. Among the models, LightGBM (LGBM) and Extra Trees (ET) consistently demonstrate superior precision, particularly without feature selection. The Decision Tree (DT) model, though slightly behind, still performs well. The results suggest that for tasks where precision is critical such as minimizing false positives the LGBM model, especially without feature selection, is the most reliable choice.

E. Results of the Recall for Ransomware Detection Model

The result of the recall obtained in detection is clearly shown in Table 4

Table 4: Recall of the Ransomware Detection Model

Learning Algorithm	Attribute Selection			
	SelectKBest	Recursive Feature Elimination	Feature Importance	Without Feature Selection
DT	0.9926	0.9926	0.9923	0.9935
ET	0.9945	0.9950	0.9947	0.9955
LGBM	0.9935	0.9945	0.9944	0.9941

From Table 4, the Extra Trees (ET) model has the highest recall (0.9955) even without feature selection. The Decision Tree (DT) and LightGBM (LGBM) models exhibit high recall rates, ranging from 0.9923 to 0.9945. The differences in recall between the models are small, showing that all models are capable of detecting relevant cases. Feature selection has little effect on recall, and most strategies produce similar results. Without feature selection, the Decision Tree (DT) model has the lowest variance and highest recall (0.9935). Extra Trees (ET) and LightGBM (LGBM) maintain high recall across all feature selection approaches, but SelectKBest results in a slight decrease in recall, particularly for LGBM (0.9935). This suggests that while feature selection can streamline models, it generally does not significantly affect their ability to identify positive instances. The Extra Trees (ET) model consistently delivers the best recall, particularly without feature selection, making it the most reliable model for maximizing recall. The Decision Tree (DT) and LightGBM (LGBM) models also perform well, though they show slight variations depending on the feature selection method used. Overall, the results indicate that all three models are robust in recall, with ET being the top performer, particularly when the full feature set is utilized.

F. Result of the F1-score for Ransomware Detection Model

The result of the F1-score is completely represented in Table 5

Table 5: F1-Score of the Ransomware Detection Model

Learning Algorithm	Attribute Selection			
	SelectKBest	Recursive Feature Elimination	Feature Importance	Without Feature Selection
DT	0.9897	0.9928	0.9919	0.9936
ET	0.9918	0.9947	0.9947	0.9959
LGBM	0.9914	0.9947	0.9928	**0**.9957

From Table 5, the F1 scores across all models are consistently high, with the highest scores achieved when no feature selection is applied. Extra Trees (ET) reaches the top F1 Score of 0.9959 without feature selection, closely followed by LightGBM (LGBM) at 0.9957, and the Decision Tree (DT) model is slightly lower but still achieves a strong F1 Score of 0.9936. selection methods show minimal impact on the F1 Score, with RFE and Feature Importance maintaining high scores close to the models' best performance. The F1-scores

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

slightly drop with SelectKBest across all models, especially in the DT model, which has the lowest score of 0.9897 with this method. This suggests that while feature selection can simplify the models, it might slightly reduce their overall steadiness between precision and recall. Among the models, Extra Trees (ET) and LightGBM (LGBM) deliver the best F1-scores, particularly without feature selection. The Decision Tree (DT) model, although slightly behind, still performs competitively. The results indicate that ET and LGBM are more effective in achieving a balanced trade-off between precision and recall, making them preferable for scenarios where both metrics are equally important.

G. Result of the ROC-AUC for Ransomware Detection Model

The result of the ROC-AUC is completely represented in Table 6.

Table 6: ROC-AUC of the Ransomware Detection Model

Learning Algorithm	Attribute Selection			
	SelectKBest	Recursive Feature Elimination	Feature Importance	Without Feature Selection
DT	0.9936	0.9938	0.9933	0.9944
ET	0.9985	0.9992	0.9993	0.9964
LGBM	0.9993	0.9997	**0.**9997	0.9960

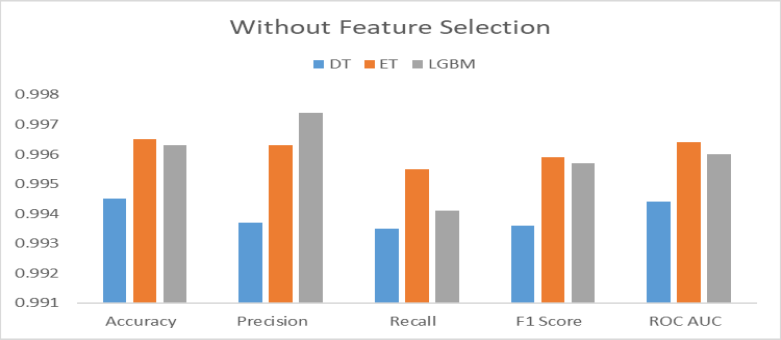
From Table 6, the ROC AUC scores are exceptionally high across all models, indicating strong discriminatory power. LightGBM (LGBM) achieves the highest ROC AUC of 0.9997 with both recursive feature elimination (RFE) and feature importance; the Extra Trees (ET) model follows closely, with a score of 0.9993 using feature importance. The Decision Tree (DT) model, while still strong, has slightly lower scores, with the highest being 0.9944 without feature selection. Feature selection methods, particularly RFE and Feature Importance, result in the highest ROC AUC scores for the LightGBM (LGBM) and Extra Trees (ET) models. SelectKBest slightly lowers the ROC AUC for all models, but the difference is minimal. The Decision Tree (DT) model shows consistent performance across all feature selection methods, with only slight variations. This suggests that advanced feature selection methods can enhance the discriminatory power of ensemble models like ET and LGBM. LightGBM (LGBM) consistently delivers the best ROC AUC scores, especially with RFE and feature importance, making it the most effective model for distinguishing between classes. Extra Trees (ET) is also highly competitive, particularly with feature importance. The Decision Tree (DT) model, while still robust, lags slightly behind the ensemble models. Overall, LGBM and ET are the top performers in ROC AUC, indicating their superior ability to correctly rank positive instances higher than negative ones.

H. Comparative Analysis of the Evaluation Metrics

A comparative study of the three selected tree-like algorithms used was conducted. The results are represented in the following subsections.

Comparative Analysis of Algorithms without Feature Selection

The result of the comparative study without feature selection techniques is represented in Figure 7



https://doi.org/10.5281/zenodo.16108642

Figure 7: Comparative Analysis without Feature Selection

From Figure 7, it was shown that the highest accuracy of 0.9963 was recorded in LGBM, the highest precision of 0.9974 was obtained in LGBM, the highest recall of 0.9955 was recorded in Extra trees, the highest F1-score of 0.9959 was obtained in ET, and highest ROC-AUC of 0.9964 was recorded in ET.

I. Comparative Analysis of SelectKBest

The result of the comparative study for selectKBest features is represented in Figure 8

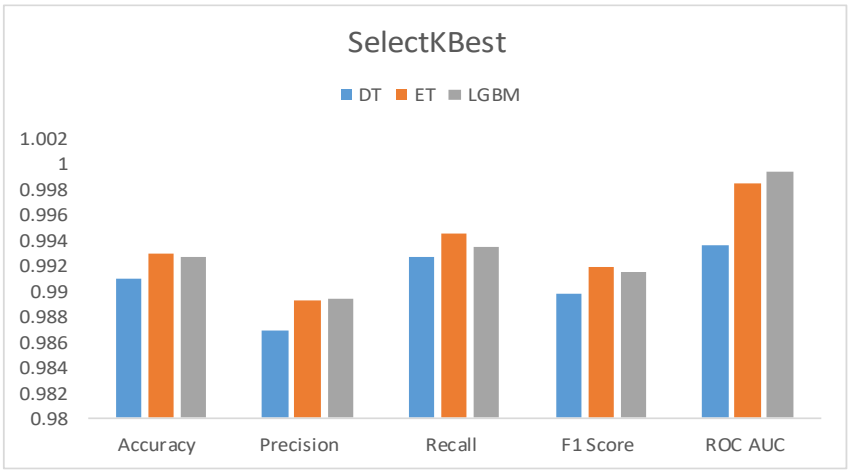


Figure 8: Comparative Analysis for SelectKBest

From Figure 8, it was shown that the highest accuracy of 0.9929 was recorded in Extra trees, the highest precision of 0.9894 was obtained in LGBM, the highest recall of 0.9945 was recorded in Extra trees, the highest F1-score of 0.9918 was obtained in ET and highest ROC-AUC of 0.9993 was recorded in LGBM.

J. Comparative Analysis of Recursive Feature Elimination(RFE)

The result of the comparative study for RFE is represented in Figure 9

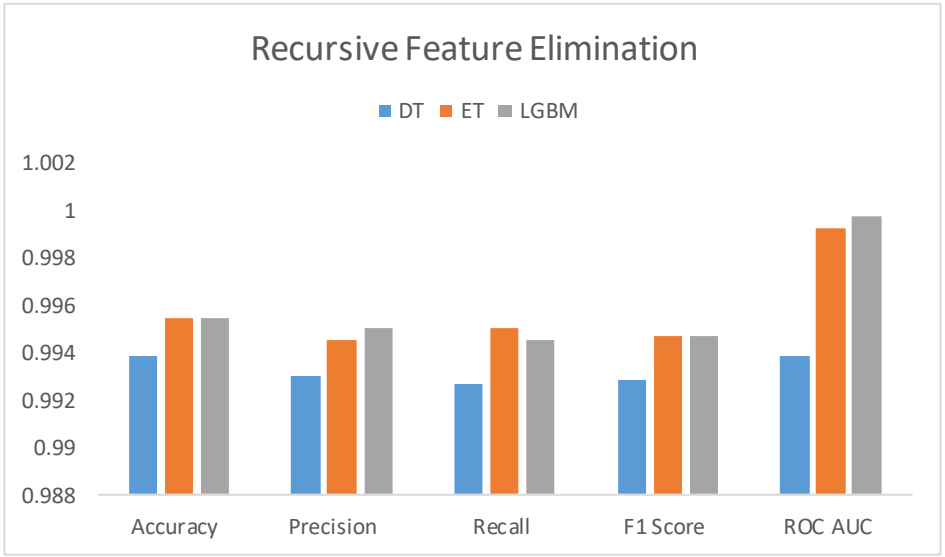


Figure 9: Comparative Analysis for RFE

VOLUME 1, Issue 2, 2025
https://doi.org/10.5281/zenodo.16108642

From Figure 9, it was shown that the highest accuracy of 0.9954 was recorded in Extra trees and LGBM the highest precision of 0.9950 was obtained in LGBM, the highest recall of 0.9950 was recorded in Extra trees, the highest F1-score of 0.9947 was obtained in ET and LGBM, and highest ROC-AUC of 0.9997 was recorded in LGBM.

K. Comparative Analysis of Feature Importance (FI)

The result of the comparative study for feature importance is represented in Figure 10

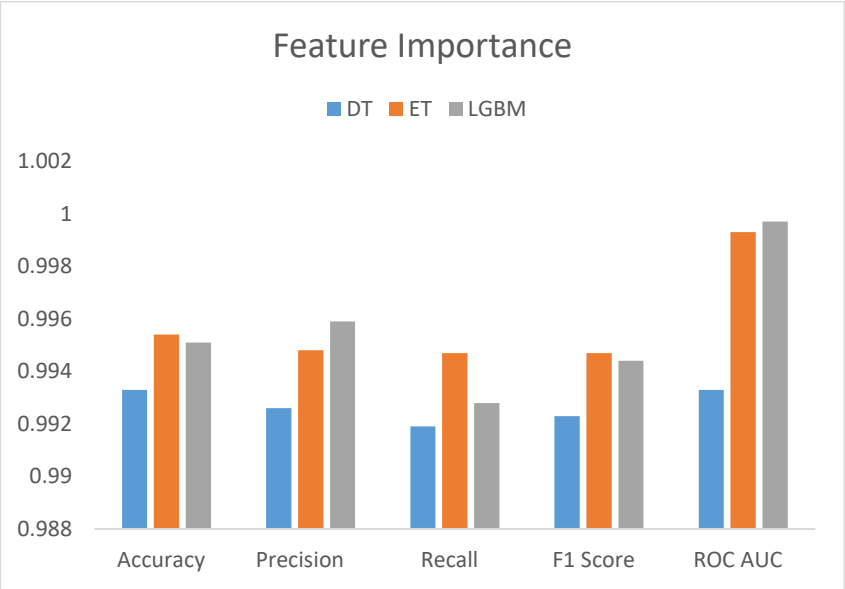


Figure 10: Comparative Analysis of Feature Importance

Summary of Comparative Analysis for Evaluation Metrics

The results suggest that while feature selection methods like RFE and feature importance can improve model performance, especially for DT, the highest performance is generally achieved without feature selection. ET and LGBM models are particularly effective, with ET being slightly more robust across different feature selection methods. If computational resources allow, skipping feature selection might provide the best results, especially with complex models like ET and LGBM. However, when feature selection is necessary, RFE and feature importance are preferable methods.

V. CONCLUSION

The analysis in this work highlights that the Extra Trees (ET) model outperforms all other metrics evaluated, especially when no feature selection was applied. ET performed better in accuracy, precision, recall, F1-score, and ROC AUC, making it the most reliable and effective model. The LightGBM (LGBM) model also delivered exceptional performance, particularly in terms of ROC AUC and precision, making it a strong contender and a viable alternative to ET. This study underscores that while feature selection methods like RFE and Feature Importance can significantly enhance model performance, particularly for simpler models like the Decision Tree (DT) the best overall performance was consistently achieved when no feature selection was used. For complex models like ET and LGBM, using the full set of features generally yields the best results. However, in situations where feature selection is necessary, either due to computational limitations or the need to simplify the model, RFE and feature importance emerged as the most effective methods.

REFERENCES

[1] A. Gunuganti, “Journal of Engineering and Applied Ransomware Evolution and Defense Strategies,” *J. Enginnerring Appl. Sci. Technol.*, vol. 4, no. 4, pp. 1–4, 2022.

[2] T. McIntosh, A. S. M. Kayes, Y. P. P. Chen, A. Ng, and P. Watters, “Ransomware Mitigation in the Modern Era: A Comprehensive Review, Research Challenges, and Future Directions,” *ACM Comput. Surv.*, vol. 54, no. 9, pp. 1–43, 2022, doi: 10.1145/3479393.

[3] A. Alraizza and A. Algarni, “Ransomware Detection Using Machine Learning: A Survey,” *Big Data*

https://doi.org/10.5281/zenodo.16108642

Cogn. Comput., vol. 7, no. 3, pp. 1–24, 2023, doi: 10.3390/bdcc7030143.

[4] G. Nagar, "The Evolution of Ransomware: Tactics, Techniques, and Mitigation Strategies," *Int. J. Sci. Res. Manag.*, vol. 12, no. 06, pp. 1282–1298, 2024, doi: 10.18535/ijsrm/v12i06.ec09.

[5] L. Y. Connolly, D. S. Wall, M. Lang, and B. Oddson, "An empirical study of ransomware attacks on organizations: An assessment of severity and salient factors affecting vulnerability," *J. Cybersecurity*, vol. 6, no. 1, pp. 1–18, 2020, doi: 10.1093/CYBSEC/TYAA023.

[6] U. Adamu and I. Awan, "Ransomware prediction using supervised learning algorithms," in *Proceedings - 2019 International Conference on Future Internet of Things and Cloud, FiCloud 2019*, 2019, pp. 57–63. doi: 10.1109/FiCloud.2019.00016.

[7] J. . Ayinla, M.O., Oyelakin, A.M., & Olomu, "A Comprehensive Review on Machine Learning Techniques for Protein Family Prediction," *LAUTECH J. Comput. Informatics*, vol. 4, no. 1, pp. 1–22, 2024, doi: 10.1007/s10930-024-10181-5.

[8] S. Aurangzeb, M. Aleem, M. A. Iqbal, and M. A. Islam, "Ransomware: A Survey and Trends," *J. Inf. Assur. Secur.*, vol. 12, no. January 2020, pp. 1–12, 2017, [Online]. Available: www.mirlabs.net/jias/index.html

[9] M. Cen, F. Jiang, X. Qin, Q. Jiang, and R. Doss, "Ransomware early detection: A survey," *Comput. Networks*, vol. 239, no. April 2023, p. 110138, 2024, doi: 10.1016/j.comnet.2023.110138.

[10] S. Jawad and H. M. Ahmed, "Ransomware Detection and Prevention Using Machine Learning and Honeypots: A Short Review," *Iraqi J. Comput. Commun. Control Syst. Eng.*, vol. 24, no. 2, pp. 29–40, 2024, doi: 10.33103/uot.ijccce.24.2.3.

[11] L. Albshaier, S. Almarri, and M. M. H. Rahman, "Earlier Decision on Detection of Ransomware Identification: A Comprehensive Systematic Literature Review," *Inf.*, vol. 15, no. 8, 2024, doi: 10.3390/info15080484.

[12] A. Redhu, P. Choudhary, K. Srinivasan, and T. K. Das, "Deep learning-powered malware detection in cyberspace: a contemporary review," *Front. Phys.*, vol. 12, pp. 01–29, 2024, doi: 10.3389/fphy.2024.1349463.

[13] M. Aljabri *et al.*, "Ransomware detection based on machine learning using memory features," *Egypt. Informatics J.*, vol. 25, pp. 1–18, 2024, doi: 10.1016/j.eij.2024.100445.

[14] S. Kaur, C., Chandel, R., Brar, T P., Sharma, "Machine learning and its applications: A review," *CGC Int. J. Contemp. Technol. Res.*, vol. 5, no. 2, pp. 57–60, 2023, doi: 10.1109/ICBDACI.2017.8070809.

[15] M. M. Abualhaj *et al.*, "A fine-tuning of decision tree classifier for ransomware detection based on memory data," *Int. J. Data Netw. Sci.*, vol. 8, no. 2, pp. 733–742, 2024, doi: 10.5267/j.ijdns.2024.1.005.

[16] A. Alraizza and A. Algarni, "Ransomware Detection Using Machine Learning: A Survey," *Big Data Cogn. Comput.*, vol. 7, no. 3, pp. 1–19, 2023, doi: 10.3390/bdcc7030143.

[17] M. S. Hossain *et al.*, "Android Ransomware Detection from Traffic Analysis Using Metaheuristic Feature Selection," *IEEE Access*, vol. 10, pp. 128754–128763, 2022, doi: 10.1109/ACCESS.2022.3227579.

[18] B. M. Khammas, "Ransomware Detection using Random Forest Technique," *ICT Express*, vol. 6, no. 4, pp. 325–331, 2020, doi: 10.1016/j.icte.2020.11.001.

https://doi.org/10.5281/zenodo.16108642