Available online www.unicrossjournals.com

**UNICROSS JOURNAL OF SCIENCE AND TECHNOLOGY, UJOST**

**RESEARCH ARTICLE VOL. 4(1) MARCH, 2025 ISSN:2814-2233**

Date Approved March 31, 2025 Pages 65 – 78

# DEVELOPMENT OF A WIRELESS NETWORK RANSOMWARE DETECTION SYSTEM USING A STACK ENSEMBLE ALGORITHM

**Owoade, A. A., Adeyemi, O. J., Odulaja, G. O and Ogunsanwo, G. O.**
Computer Science Department
Tai Solarin University of Education, Ijebu Ode

owoadeaa@tasued.edu.ng, tayo009@hotmail.com, odulajago@tasued.edu.ng, ogunsanwogo@tasued.edu.ng

## Abstract

Over the years, ransomware in wireless networks has become a significant computer network threat and a cybersecurity challenge, resulting in financial losses for both individuals and enterprises. Many system-based detection methods fail to provide timely alerts as they require a system to be infected with a threat before anomalies can be detected. To address these limitations, the research explores the use of a static file-based technique for ransomware detection by using Stack Ensemble model on a dataset of stack data and binary features of files. The dataset has 18 features including DebugSize, ResourceSize and BitcoinAddresses which are good indicators of ransomware. A Stack Ensemble model was built (with Random Forest (RF)), Support Vector Machine (SVM) and Gradient Boost (GB)) to leverage the strengths of multiple classifiers to improve detection accuracy. The proposed model of (Random Forest (RF) and Support Vector Machine (SVM) as base learner and Gradient Boost (GB) as meta learner performed well with an accuracy of 99.60%, precision 99.50% recall 99.46% and F1score of 99.48%. These results show that the Stack Ensemble model is effective in distinguishing ransomware files from benign files with good generalization across different data distributions. This paper advances static malware detection by providing an accurate and scalable way to detect ransomware. The results show that combining machine learning algorithms with file based feature analysis can complement traditional host based and network based-detection systems to have proactive defense against evolving ransomware threats.

Keywords**:** Ransomware detection, Machine learning, Stack Ensemble model, Cybersecurity, Hostbased detection.

## 1.0 Introduction

The digital landscape has become increasingly vulnerable to malicious cyber threats, which Ransomware is now among the most widespread and harmful malware types, targeting people, companies, and the government organizations. Ransomware encrypts victims' data and demands ransom payments, often in cryptocurrencies to restore access. This category of malware has evolved rapidly, focusing on important industries like healthcare, finance, and government institutions, resulting in significant economic and operational disruptions (Sathya *et al*, 2023). Cybersecurity solutions are continually challenged to keep pace with these threats, as modern ransomware employs sophisticated techniques to evade detection and propagate through networks (Mourad Benmalek, 2024). To be effective, ransomware must infiltrate the host system, encrypt the host data, and then demand a payment from the victim. These steps guide the attacks of the ransomware are:

- **Step 1. Infection and Distribution Vectors**

Immediately ransomware gained access to the host, it started to infect the host systems though some particular infection vectors are typically preferred by ransomware operators.

- **Step 2. Data Encryption**

Ransomware starts encrypting data as soon as it obtains access to the host machine. Its encryption features are based on operating system orientation. In this exercise files are accessed, encrypted using a "controlled key," and the encrypted file versions are substituted for the originals. Many of ransomware variations strike with caution in order for the host not to detect them in their early stage, so they maintain stability of the host until they gain full access.

- **Step 3. Ransom Demand**

When the encryption procedure is complete, the ransomware is prepared to demand a payment. often asks for a set amount of bitcoin in exchange for access to the victim's files. If the ransom is paid, the ransomware operator will either provide a copy of the private key used to secure it or the symmetric encryption key itself, as seen in Figure 1.

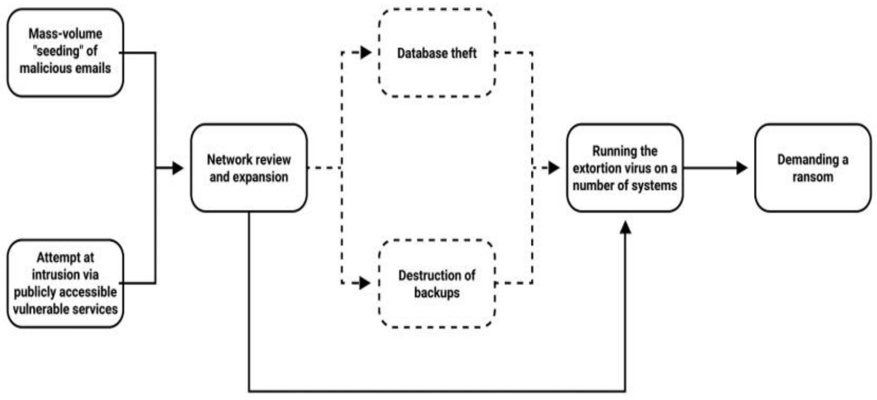

Mass-volume "seeding" of malicious emails
Attempt at intrusion via publicly accessible vulnerable services
Network review and expansion
Database theft
Destruction of backups
Running the extortion virus on a number of systems
Demanding a ransom

Fig. 1 Stages of ransomware attacks (source: SI-CERT. (2021))

Several ML algorithms are currently being used to successfully develop detection system for ransomwares because event-based, statistical-based, and data-centric approaches traditional methods of ransomware detection are insufficient to fight and failed in many cases, putting in place the best possible protection and security by using ML approaches against such sophisticated malevolent attacks have shown promises and raised hope of secure host system The use of tradition ML algorithms has shown limited results in their performance metrics results while Stack Ensemble models has shown greater results and promises in the areas of classification and detection cases. To increase overall performance, ensemble learning combines the predictions of several models. This approach makes use of the notion which a collection of failing leaners can band together to become a strong learners. (Masum *et al*, 2022 & Kuma *et al*,2024). In this paper, the Stack Ensemble model for ransomware detection is constructed using the machine learning methods: Random Forest (RF), Gradient Boost (GB), and Support Vector Machine (SVM). In order to identify new ransomware assaults, Zahoora et al. (2022) developed a Cost-Sensitive

Pareto Ensemble technique called CSPE-R. They did this by employing deep learning-based unsupervised feature extraction and a cost-sensitive pareto ensemble classifier. The suggested framework converts the underlying changing feature space into a more consistent and core semantic feature space by taking advantage of the unsupervised deep Contractive Auto Encoder (CAE) varying semantic spaces are explored at varying levels of detail using the suggested CSPE-R ensemble technique. The fundamental relevance between the different kinds of ransomware assaults is then determined by training heterogeneous base estimators across these extracted subspaces. Next, a new estimator selection method based on the Pareto Ensemble is used to strike a cost-sensitive balance between false positives and false negatives. The study proposed framework was divided as: deep features extraction and ransomware detection. Ransomware detection phase was subdivided into three: (1) base estimators training, (2) estimators' selection and, (3) estimators' aggregation strategy. The generalization aptitude is evaluated by weighing the proposed method on zero-day ransomware. The results indicate that reduced feature sets (100 and 500) maintain high performance compared to a full set of 16,382 features. Statistical analysis shows p-values below 0.01 confirming the framework's improved diversity and generalization capabilities. Overall, CSPE-R demonstrates robust detection performance with a focus on minimizing false positives while maximizing true positives.

Da Silva et al., (2024), this study develops an ensemble machine learning approach to enhance ransomware detection by combining disjoint data sets, as this method allows the models to observed a broader range of features that reflect the real-world behaviour of ransomware as the integration will helps in detecting various ransomware families and variants. The study's findings showed that the ensemble approach along with the dataset combination strategy produced better detection rates than applying the models in isolation. This performance evaluation highlights the effectiveness of the methods used in the research.

Kumar et al, (2024), the study employs an ensemble approach combining three base classifiers: Random Forest, Naïve Bayes, and Adaptive Boosting (AdaBoost) to detect ransomware attacks. AdaBoost improves classification accuracy by iteratively adjusting the weights of misclassified instances, NB is a probabilistic classifier based on the Bayes theorem presuming feature independence, while RF builds several decision trees and aggregates their predictions. The evaluation of the efficiency of the collaborative method is done utilizing a number of indicators, such as ROC curve analysis, F1 score, recall, accuracy, and precision. The suggested ensemble approach obtains a classification accuracy of 92.2%, precision of 86.7%, recall of 85.8%, and F1 score of 84.9%, according to the experimental results on the UGRansome dataset. By effectively utilizing the advantages of several base classifiers, the ensemble technique improves detection capabilities and strengthens cybersecurity defenses against ransomware threats.

Singh et al, (2023) present RANSOMNET+ in their study, a hybrid model designed to classify ransomware attacks effectively. Using pre-trained transformers in Convolutional Neural Networks (CNNs), utilizing the benefits associated with both architectures to record intricate structures and characteristics in data. The model, RANSOMNET+ includes interpretability features that enhance comprehension of its procedure for making decisions, incorporating analyses such as outlier identification, feature relevance, and feature distributions. When compared to well-known benchmarks, namely ResNet 50 and VGG 16, the model scored better in terms of F1 score, accuracy, precision, and recall. RANSOMNET+ exhibits remarkable performance metrics, attaining an F1 score of 97.64%, a precision of 99.5%, and a recall of 98.5%. Additionally, the model exhibits great testing accuracy (99.1%) and training accuracy (99.6%).

Moreira et al. (2024) in their study combining structural features in a thorough study to identify new ransomware families introduces a static analysis approach for detecting ransomware by examining key structural features in Windows executable files. The method combines the strengths of three machine learning methods: eXtreme Gradient Boosting (XGB), Random Forest (RF), and Logistic Regression (LR) through an ensemble soft voting model. By analyzing features like section entropy, function calls, imported DLLs, and header fields, the approach aims to spot newly emerging ransomware families effectively.

To test the model, the researchers compiled a dataset of 2,675 binary samples. This includes a training set with 1,023 ransomware samples from 1,134 benign applications across 25 groups, alongside a testing set featuring 15 current ransomware families comprise 385 samples, and 133 benign applications. Method of SHapley Additive exPlanations (SHAP) was applied to ensure transparency and build trust in the model's predictions, this offering insights into how decisions were made. With a weighted average accuracy of 97.53%, precision of 96.36%, recall of 97.52%, and F-measure of 96.41%, the suggested model showed excellent performance metrics in the Detection of New Ransomware Families (DNRF). With an average scanning and prediction time of 0.37 seconds, it demonstrated its flexibility in responding to changing ransomware threats while requiring little processing power. These findings demonstrate the model's applicability as an extra layer in antivirus software.

In their paper, Ransomware Detection Framework utilizing Soft Voting-Based Ensemble model, Gupta et al. (2023) use five base learners: decision trees, random forests, Adaboost, ExtraTree, and XGBoost. Their approach aims to improve the ability to identify ransomware assaults, addressing issues of low accuracy and high false positive rates found in existing methods. The model achieved an impressive accuracy of 98.42% when evaluated on the CIC-AndMa12017 ransomware dataset outperforming other models like k-NN, Random Forest, and AdaBoost.

Boyd et al. (2024) presented a novel methodology for automated ransomware detection via Deep Behavioral Sequence Mapping (DBSM), a machine learning-based approach that provides a reliable and flexible way to recognize ransomware by examining intricate trends in behavior rather than static indicators. Even when obfuscation tactics are used, DBSM analyzes dynamic, sequential actions to detect ransomware, in contrast to conventional methods that depend on pre-established rule sets or signatures. The study evaluations clearly show that DBSM successfully reduced false positives and captured subtle behavioral signals indicative of ransomware, demonstrating its robustness in real-time cybersecurity environments. Its detection accuracy of 97.2% is significantly higher than the accuracy rates of 85.6% for signature-based methods, 88.3% for anomaly-based methods, and 83.9% for heuristic methods. According to the study, the model's throughput and latency were also assessed; DBSM achieved a peak throughput of 620 samples per second and an average delay of 56.4 milliseconds per sample. This suggests that DBSM is capable of processing data quickly which makes it appropriate for real-time detection in settings with high demand.

In order to address the gaps in this research, two articles were used as the reference point in addressing the gap. Take for instance in Ismail et al., (2023) who employ ensemble-based machine learning algorithms to improve the detection of cyberattacks in Wireless Sensor Networks (WSNs). To increase detection accuracy and resilience against different kinds of attacks, the authors created a model that integrates several classifiers. The method overcomes the drawbacks of single-model systems, including high false-positive rates and little flexibility in response to changing threats, by utilizing ensemble learning. The results of the study show that ensemble approaches can greatly improve intrusion detection systems' dependability in WSNs. And also in Tabbaa et al. (2023), In this study, Tabbaa et al. (2023) present an online ensemble learning architecture designed specifically for WSN attack detection. The suggested methodology analyzes streaming data using both homogeneous and heterogeneous ensembles, allowing for real-time intrusion detection. The work shows that online ensemble learning can successfully improve detection performance while preserving computational efficiency by tackling issues like idea drift and resource limitations present in WSNs. Through the use of ensemble learning techniques, both publications improve detection accuracy and system resilience against complex cyberthreats, advancing ransomware detection in wireless networks.

## 2.0 Methodology

### 2.1 Method of data Collection and Preprocessing

A thorough process was required for the creation of the Stack Ensemble model, beginning with dataset collection, data preprocessing, feature extraction, model training, Stack technique training, and model assessment. For this research work, development of Stack Ensemble model for detection of ransomware dataset was collected from Kaggle online repository located online. The dataset contained details about ransomware attacks activities (Benign and No Benign) was downloaded. After the dataset collection, the dataset was preprocessed with feature selection methods to find and pick pertinent features from the dataset's original features.

### 2.1.1 Method of Collection of Relevant Data and Preprocessing

The dataset consists of 62485 records with 18 attributes among which are 17 input features while the remain one as target variable. The dataset was collected from Kaggle repository (‘https://raw.githubusercontent.com/securycore/MLRD-Machine-Learning) downloaded as a cvs file. Each malware was classified as either Benign or No Benign based on the attributes used to describe the dataset. The dataset was preprocessed in order for it to fit for the model building. Duplicate entries were identified and removed to ensure the quality of the dataset missing values were addressed using imputation techniques, though the dataset exhibited an imbalance in the target class distribution with a significant majority being Benign. To enhance the target class, the Synthetic Minority Oversampling Technique (SMOTE) was used. In order to guarantee that the dataset would offer a solid basis for training and testing machine learning models in ransomware detection, encoding and standardization techniques are also used. The dataset contains both nominal values (such as FileName and md5Hash) and numerical values (such as DebugSize, DebugRVA, MajorImageVersion, MajorOSVersion, ExportRVA, ExportSize, IatVRA, MajorLinkerVersion, MinorLinkerVersion, NumberOfSections, SizeOfStack-Reserve, DllCharacteristics, ResourceSize, BitcoinAddresses, and Benign's present).

### 2.2 Stack Ensemble Model

The predictive outcomes of the stack ensemble results of the machine learning algorithms (Random Forest (RF), Gradient Boost (GB), and Support Vector Machine (SVM)) employed in this work were combined in the Stack Ensemble model created in this study using a stacking ensemble learning approach. These algorithms were initially separated as based learners before being employed as Stack-learners with the ransomware detection dataset, as seen in Figure 2 to create the Stack ensemble model.

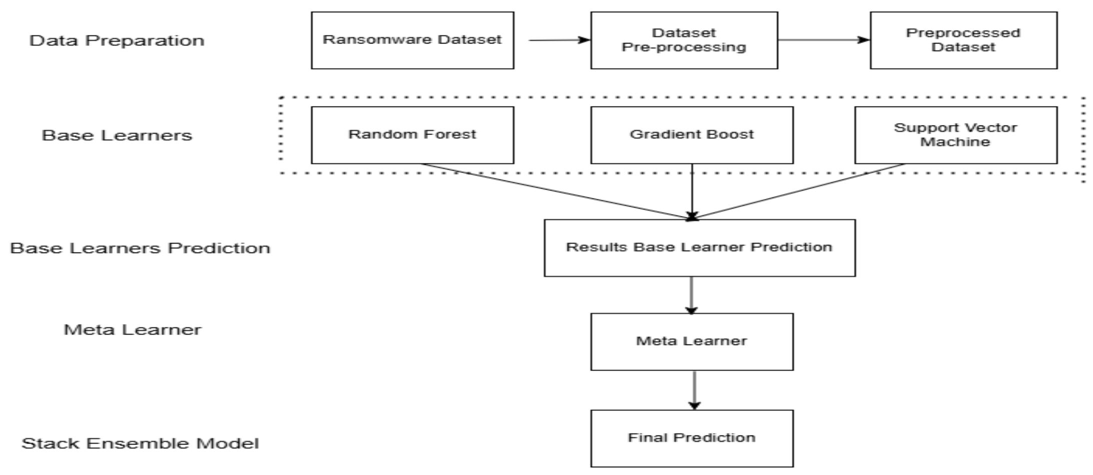


Fig. 2. Framework of stack-ensemble model for ransomware detection

To create the final prediction for the creation of the Stack-ensemble model that was necessary for the detection of ransomware. Figure 2 illustrates how the Stack-classifier (Logistic Regression) used the predictions of the three base learners' ensemble models (Random Forest, Gradient Boost, and Support Vector Machine). Because of its ease of use, computational effectiveness and capacity to produce calibrated probability for classification tasks, logistic regression was selected.

### 2.2.1 Steps involved in developing the Stack Ensemble model:

**Training Base Models**: The training dataset was split into two subsets: one subset was used to train the base models (Random Forest, Gradient Boosting, and SVM) on the original feature set and the other to test the model.

**Stacking Phase**: The predictions from the base models on the training data were collected now served as a new feature set (referred to as the Stack-feature set) for the Stack-learner.

**Stack-Learner Training:** Logistic Regression was trained on the Stack-feature set to learn how to combine the predictions of the base models effectively. This Stack-learner utilized the complementary strengths of the base models to achieve higher accuracy and robustness.

**Final prediction:** the trained Stack-learner final generates the final prediction using the base ensemble models' predictions as input as shown in Figure 2.

### 2.2.2 Random Forest

A range of unpruned regression or classification trees known as Random Forests are created by randomly selecting samples of the training data. The induction process involves the selection of random features. Furthermore, it integrates tree predictors, making each tree in the forest reliant on the values of a randomly chosen vector with the same distribution and independent sampling. The following sequences comprise the random forest classifier:

- Bootstrap Sampling Random Forest uses bootstrap aggregation (bagging) to create multiple subsets of the training data. From the training dataset D of size N, random subsets D1, D2,...,Dk of size N' (usually N′=N) are created by sampling with replacement. Di∼D for i = 1, 2, ..., k
- Feature Subset Selection for each decision tree, a random subset of mm features is selected from the total MM features at each split.m= square root (M) (for classification, default value)
- Formation of Decision Trees Using the random subset of characteristics and one of

the bootstrap samples, each decision tree is constructed separately. Until a stopping requirement (such as a minimum number of samples per leaf) is satisfied, the tree is grown to its maximum depth. The optimal feature and splitting threshold are determined at each split using the Gini impurity or information gain criterion.

- Ensemble of Trees Once all kk decision trees are built, they collectively form the Random Forest model.
- Prediction for Classification: Each tree provides a prediction for an input xx, and the forest outputs the class with the majority vote

### 2.2.3 Gradient Boost Classifier

Unlike random forests which rely on simple averaging, the gradient boosting classifier is an ensemble technique that builds its models sequentially. Gradient Boosting's concept is to incrementally introduce new models into the ensemble, each of which is intended to correct the ensemble's previous mistakes.
A new weak model also known as a base learner, is trained at each iteration. This model uses a gradient-descent-based optimization technique to reduce the ensemble's error and the process keeps going until a predetermined stopping criterion, like a maximum number of iterations is satisfied.

The fundamental idea behind this technique is to construct the new base-learners in a way that maximizes their correlation with the ensemble's overall negative gradient of the loss function. Even though the loss functions can be selected at random, the learning process would result in sequential error-fitting results if the error function was the conventional squared-error loss.

### 2.2.4 Support Vector Machine

One of the most effective techniques for creating a classifier is SVM. With its most powerful mathematical model, it has developed into a reliable paradigm for regression and classification. In order to predict labels from one or more feature vectors, the algorithm seeks to create a decision border between two classes. This decision boundary often referred to as the hyperplane, is oriented as far away from the nearest data points from each class as feasible. When the hyperplane has the greatest distance to the closest training data points x, also known as the support vectors, distinct separation is accomplished since the classifier's generalization error decreases with increasing margin

SVM aims to maximize the distance between hyper-planes in order to reduce the cost during model formulation. A good separation is achieved by the hyperplane ($\boldsymbol{w}^{T}\ \boldsymbol{x}\ +b\ =\ 0$) that has the largest distance 2/ $||w||$ to the neighboring data points of either classes at opposite ends. As shown in the Figure 2. The margin is the perpendicular distance between the two dashed lines $\boldsymbol{w}^{T}\ \boldsymbol{x}\ +b\ =1$ and $\boldsymbol{w}^{T}\ \boldsymbol{x}\ +b\ =-1$ and classification record any point above the hyperplane $\boldsymbol{w}^{T}\ \boldsymbol{x}\ +b\ =0$ is classified as +1 and any point below the hyperplane $\boldsymbol{w}^{T}\ \boldsymbol{x}\ +b\ =0$ is classified as −1. The expression of letters in the Figure 3:

W = Weight vector (perpendicular to the hyperplane).
X = Feature vector.
B = Bias term.

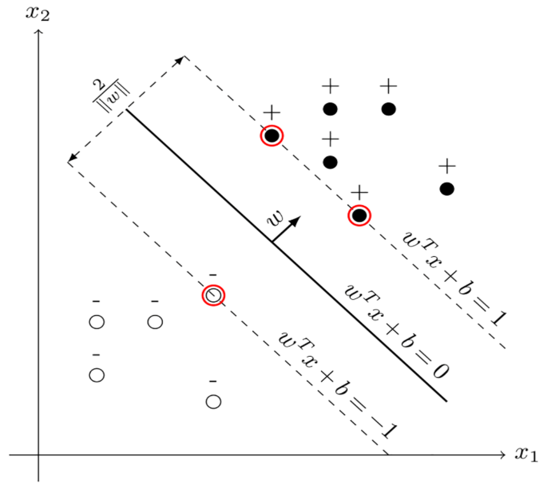


Fig. 3 Support vector machine with separating hyperplanes

### 2.3 Environment for Model Simulation

After determining which Stack ensemble model of machine learning algorithms was required to create the ransomware detection model, the model was simulated using data gathered from the online Kaggle repository. The model was developed in a simulation environment called the Google Collaboratory (CoLab) is powered by the cloud platform that is free and offered by Google which enables users to write and run Python code in a Jupyter Notebook environment. Python is the preferred language for machine learning development in this research due to its ease of use, large library, robust integration, and adaptability.

### 2.3.1 Utilizing a Confusion Matrix to Extract Evaluation Results

Confusion matrix was chosen as one of the indicators for performance to assess the efficacy among the algorithms which are used to develop the ransomware detection model because of its ability to highlight the model's strengths and weaknesses, indicate how well the model's predictions match the actual results and provide flexible and detailed evaluation results that go beyond overall accuracy. As seen in Figure 4, a confusion matrix is often depicted by a square box that shows the anticipated along the vertical axis and the actual detection along the vertical. False Positives (FP) and False Negatives (TN) are the other cells, whereas all corrected classifications known as True Positives (TP) and True Negatives (TN) are located along the diagonal from the north-west corner to the south-east corner.

Confusion matrix usually breaks down the predictions into four categories which helps to identify specific types of errors the models is making:

- **True Positives (TP)**: Benign cases were accurately identified as such.
- **True Negatives (TN):** It was accurately predicted that there would be no benign cases.
- **False Positives (FP):** No benign cases were mislabeled as benign cases.
- **False Negatives (FN):** No Benign instances were anticipated for Benign cases.

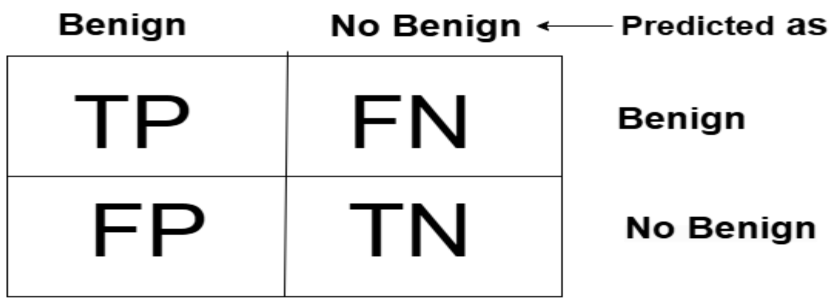


Fig. 4 Confusion matrix for assessing model achievement

The following is an explanation of the metrics' definition and phrases.

- The percentage of real Benign (or No Benign) instances that are correctly identified is known as the True Positive (TP) rates (sensitivity/recall).

$$TP \text{ (Benign)} = \frac{TP}{TP + FN}$$

$$TP \text{ (}no\text{ Benign)} = \frac{TN}{TN + FP}$$

- The percentage of real no Benign (or Benign) cases that are misclassified is known as the 1-specificity/false alarms is the rate of False Positives (FPs).

$$FP \text{ (Benign)} = \frac{FP}{FP + TN}$$

$$FP \text{ (}no\text{ Benign)} = \frac{FN}{FN + TP}$$

- Precision is the percentage of correctly classified anticipated Benign (or non-Benign) cases.

$$Precision \text{ (Benign)} = \frac{TP}{TP + FP}$$

$Precision$ ($no$ Benign) = $TN$

$FN+TN$

- Accuracy is the percentage of all forecasts that came true.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

## 3.0 Results

### 3.1 Findings from Pre-Processing and Data Collection

The dataset used in this study was gathered from a Kaggle online repository. The dataset had 6248 entries with 18 attributes, including the target class that determined if ransomware was present or not. As indicated in Table 1, the dataset included records for 27118 benign cases and 35367 no benign cases. Both nominal and quantitative values were utilized for labeling among the 18 attributes found in the dataset that was gathered. The presence of Machine was one of the numerical labels found. NumberOfSections, SizeOfStackReserve, DllCharacteristics, ResourceSize, BitcoinAddresses, MajorImageVersion, MajorOSVersion, ExportRVA, ExportSize, IatVRA, MajorLinkerVersion, MinorLinkerVersion, Benign and nominal labeling are FileName and md5Hash.

Table 1. Distribution Benign and No Benign in the Dataset

| Class Label | Frequency | Percentage (%) |
|---|---|---|
| Benign | 27118 | 43.40 |
| No Benign | 35367 | 56.60 |
| **Total** | **62485** | **100.0** |

The Benign indicates 43.4% and 56.6% for the No Benign. This dataset displays varying variability with attributes like FileName showing high diversity (62.485 distinct values) and others like Machine being narrowly distributed (6 values) as shown in Table 2. The Most Common column highlights attributes dominated by a single value while the Next Most Common reveals distribution nuances. Attributes with high distinct counts contribute richness but risk sparsity while low distinct counts may lack predictive power. Skewed distributions observed in this dataset was treated with SMOTE and scandalization to remove class imbalances which make the dataset fit for the modelling building.

Table 2. Frequency Distribution of the Data Collected Based on the Attributes.

| Attribute | Distinct | Most Common | Next Most Common |
|---|---|---|---|
| FileName | 62485 | 0124e21d-018c-4ce0-92a3-b9e205a76bc0.dll (1) | VirusShare_a964c930e8bce78a619b518c1f957dd0 (1) |
| md5Hash | 62485 | 79755c51e413ed3c6be4635fd729a6e1 (1) | a964c930e8bce78a619b518c1f957dd0 (1) |
| Machine | 6 | 332 (50624) | 34404 (11685) |
| DebugSize | 26 | 0 (36521) | 28 (12719) |
| DebugRVA | 10395 | 0 (36549) | 4096 (1603) |
| MajorImageVe | 49 | 0 (36618) | 10 (9493) |

| | | | |
|---|---|---|---|
| rsion | | | |
| MajorOSVersion | 17 | 4 (30758) | 5 (12474) |
| ExportRVA | 11673 | 0 (42008) | 4192 (297) |
| ExportSize | 2885 | 0 (42034) | 68 (706) |
| IatVRA | 5727 | 4096 (11705) | 0 (10680) |
| MajorLinkerVersion | 89 | 6 (12400) | 9 (12275) |
| MinorLinkerVersion | 117 | 0 (42915) | 10 (10654) |
| NumberOfSections | 26 | 5 (15304) | 3 (14124) |
| SizeOfStackReserve | 29 | 1048576 (43323) | 262144 (13218) |
| DllCharacteristics | 86 | 0 (23745) | 32768 (7331) |
| ResourceSize | 9083 | 0 (5230) | 3120 (1645) |
| BitcoinAddresses | 2 | 0 (61360) | 1 (1125) |
| Benign | 2 | 0 (35367) | 1 (27118) |

Figure 5 shows the correlation between the features and the "Benign" column which is likely to be 0 or 1. Correlation values range from -1 to 1 where positive is direct and negative is inverse. "Machine" has a moderate positive correlation (0.55) with "Benign" so it might be a good indicator. "DebugSize" and "DebugRVA" have no correlation with "Benign" so they don't seem to matter much for classification. The diagonal line of red values is the self-correlation for each feature. "ResourceSize" and "BitcoinAddresses" have a negative correlation with "Benign" so they are good indicators for malicious samples.

Some features like "iatRVA" and "ExportSize" have low inter feature correlation so they are contributing unique information. "DllCharacteristics" has complex relationship with other features and has low to moderate correlation. The heatmap helps in identifying feature dependencies and redundancies which is important for feature selection in machine learning. The varying intensity of blue and red helps in making informed decision for predictive modeling. The model was able to identified features that were more shows how important it is to look at feature interactions in cybersecurity datasets.

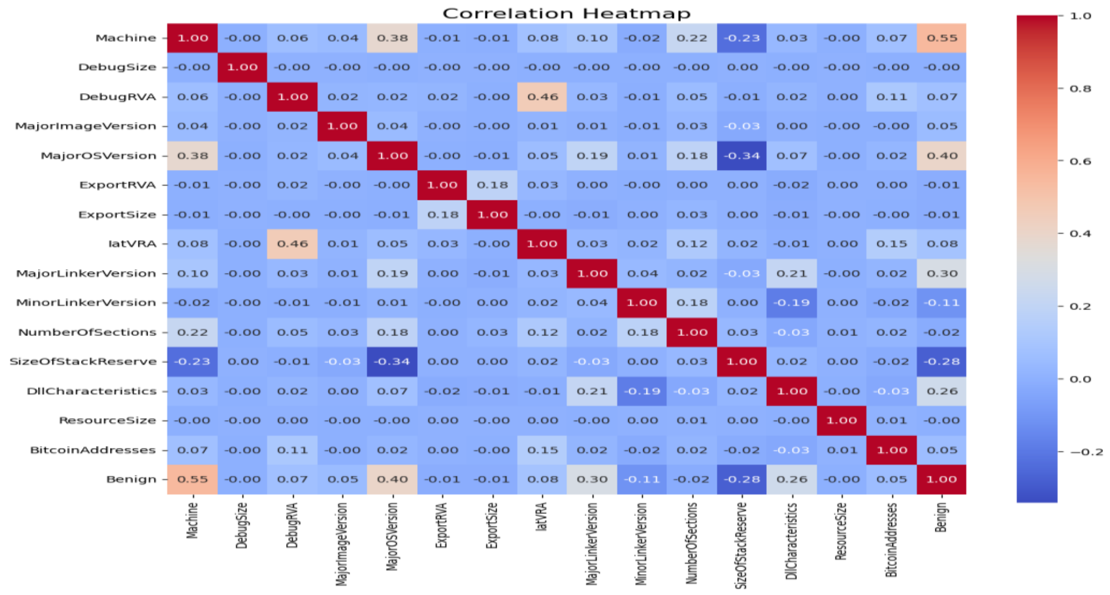


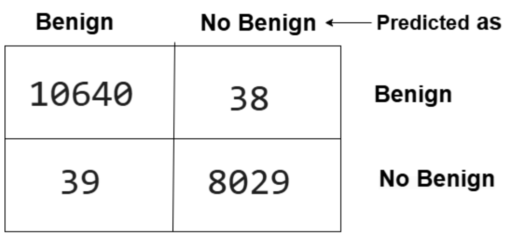
Fig. 5 show the heatmap of benign and other features

### 3.2 Findings from the Development and Simulation of the Stack Ensemble Model

The Stack ensemble model, which is necessary for ransomware detection according to the framework, is presented in this section after the data identification and description procedure. The model was developed using the CoLab simulation environment. This environment relied on the Python programming language and Jupyter notebook technology. By selecting two of the identified ML algorithms and employing the third as the meta classifier, the ensemble created from ML algorithms functions as both the base classifiers and the meta classifiers. This procedure was carried out three times using any two machine learning algorithms as basis classifiers and a stack classifier for the third.

#### 3.2.1 Findings from the Stack Ensemble Models Simulation

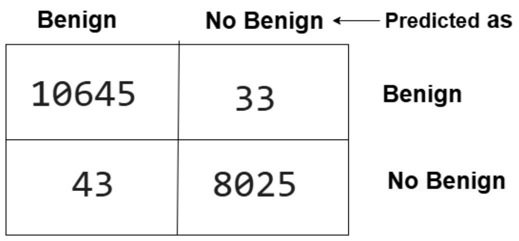
For the first set of ensemble models, the stack model was simulated in the Colab environment using Support Vector Machine (SVM) as the meta classifier and Random Forest (RF) and Gradient Boost (GB) as the base classifiers. Here, the meta classifier (SVM) was built using the predictions made by the basic classifier (RF and GB) as input to create the first final stack ensemble model for ransomware classification. With a performance accuracy of 99.59%, 18,669 out of 18,746 records in the dataset were correctly classified. As seen in figure 6, the model accurately predicted 10,640 cases out of 10,678 non-target cases and 8,029 instances out of 8,068 target samples, with just 38 non-target cases and 39 target cases misclassified.



| Benign | No Benign ← Predicted as | |
|---|---|---|
| 10640 | 38 | Benign |
| 39 | 8029 | No Benign |

Fig. 6. Outcome of stack ensemble model with SVM as a meta learners and RF and GB as base learners

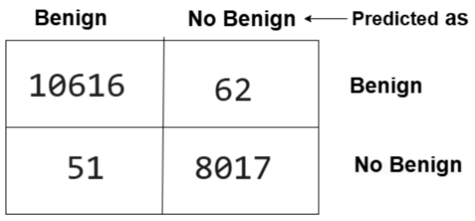
The strategy correctly predicted positive class with 8025 of the 8068 target samples that No Benign correctly identified and 10645 for actual benign which is nearly the entire sample. Only 33 cases of Benign were misclassified as No Benign, and 41 cases of No Benign were misclassified as Benign. This performance accuracy was achieved by the second stack ensemble mode which was created using Random Forest (RF), Support Vector Machine (SVM) as the base learner and Gradient Boost (GB) as the meta learner according to Figure 7.



| Benign | No Benign | ← Predicted as |
|---|---|---|
| 10645 | 33 | Benign |
| 43 | 8025 | No Benign |

Fig. 7. Result of stack ensemble model employing RF and SVM as base learners and GB as a meta learner

A 99.40% performance accuracy was obtained from the third stack ensemble model build using Gradient Boost (GB), Support Vector Machine (SVM) as the base learner, and Random Forest as the meta learner. Of these, 113 cases were misclassified meaning that 62 cases of Benign were classified as No Benign and 51 cases of No Benign were classified as Benign. Figure 8 shows that 10616 cases of Benign and 8017 cases of No Benign were correctly classified.



| Benign | No Benign | ← Predicted as |
|---|---|---|
| 10616 | 62 | Benign |
| 51 | 8017 | No Benign |

Fig. 8. Result of stack ensemble model using GB and SVM as base learners and as a meta learner

### 3.2.2 Result of Key Performance Metrics for Stacking Ensemble Models

Table 3 presents the key performance metrics results for the three stacking ensemble models using the combinations of Gradient Boosting (GB), Support Vector Machine (SVM), and Random Forest (RF) as base and meta classifiers.

Table 3. Highlights the outstanding performance of all three stacking ensemble models

| Models | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| RF + GB (Base), SVM (Meta) | 99.59% | 99.48% | 99.46% | 99.47% |
| RF + SVM (Base), GB (Meta) | 99.60% | 99.50% | 99.46% | 99.48% |
| GB + SVM (Base), RF (Meta) | 99.40% | 99.23% | 99.37% | 99.30% |

The Table 3 highlights the outstanding performance of all three stacking ensemble models each achieving an accuracy above 99%. The second model using RF and SVM as base classifiers with GB as the meta-classifier achieved the highest overall accuracy at 99.60%.

## 4.0 Conclusion

By combining Random Forest (RF), Gradient Boosting (GB), and Support Vector Machine (SVM), the study introduced a stack-ensemble model for ransomware detection. Two algorithms functioned as base learners, and the third as the meta-learner. The dataset which included characteristics pertinent to ransomware behavior allowed the stack-ensemble model to be evaluated.

The algorithm successfully classified ransomware and benign occurrences with an accuracy of 99.59%, according to the results. The confusion matrix revealed 10,640 benign and 8,029 ransomware instances were correctly identified demonstrating the model's robustness. The stack-ensemble with RF and GB as base classifiers and SVM as the meta-classifier consistently outperformed other configurations.

Among the features, the model identified DllCharacteristics, DebugRVA, and DebugSize as the most significant features, with importance scores of 0.209, 0.132, and 0.130, respectively.

These alongside MajorLinkerVersion (0.115) and ResourceSize (0.082) played a vital role in detecting ransomware.

The findings underline the promise of stack-ensemble models in enhancing ransomware detection by using the complementary strengths of different algorithms. The combination of robust feature learning and meta-classification enhanced the model's accuracy and reliability, providing a practical solution for dynamic and evolving ransomware threats. Future work could explore alternative configurations and additional datasets to further refine the approach.

## 4.1 Real World Application of the Proposed Model

Ransomware attacks have become one of the most significant cybersecurity threats, especially in wireless networks used in businesses, healthcare, finance, and other critical infrastructures. A wireless network ransomware detection system using a stack ensemble algorithm offers a robust defense mechanism by improving accuracy in detecting and mitigating ransomware threats. Below are some real-world applications of such a system:

(i) Corporate and Enterprise Networks

Application: Wireless networks are essential to the smooth functioning and communication of large corporations. By encrypting crucial data and requesting a payment, ransomware attacks have the potential to completely destroy businesses.
Benefit: Real-time detection of suspicious network activity by a stack ensemble model improves detection accuracy and lowers false positives.

(ii) Hospitals and Healthcare Systems
Application: Healthcare management systems, patient data, and medical gadgets are all supported by wireless networks in hospitals. Critical healthcare services may be disrupted by ransomware attacks on these networks.
Benefit: The system guards against the encryption of patient records and guarantees early identification of ransomware activity, averting possible medical infrastructure shutdowns.

(iii) Banking Networks and Financial Institutions
Application: Wireless networks are used by banks and financial institutions for communication and transaction. Data leaks and significant financial losses could result from a ransomware assault.
Benefit: By adding an extra degree of protection, the technology makes sure that wireless transactions are safe and that ransomware is found before it spreads.

(iv) IoT Networks and Smart Cities
Application: For traffic control, security, and utility monitoring, smart cities rely on wireless networks and networked IoT devices. In these settings, ransomware has the potential to create significant disruptions.
Benefit: Critical infrastructure security is ensured by the stack ensemble algorithm's ability to effectively monitor network traffic patterns and identify ransomware threats in real-time.

There are numerous real-world uses for a stack ensemble algorithm-powered wireless network ransomware detection system in industries where data security is crucial. Its capacity to effectively and precisely identify ransomware guarantees continuous operations in a variety of sectors, including government, healthcare, business, and finance, ultimately improving cybersecurity in wireless environments.

### 4.1.1 Uniqueness of Ensemble Combination

The stack ensemble algorithm's capacity to integrate several machine learning models, adjust to novel threats, and deliver high accuracy with few false positives makes it especially well-suited for wireless network ransomware detection. For safeguarding wireless environments, it provides a complete and scalable cybersecurity solution by combining supervised, unsupervised, and deep learning models.

## References

Sathya, T., N, K., S, S., Upodhyay, D., & Muzafar, H. (2023). Bitcoin heist ransomware attack prediction using data science process. E3S Web of Conferences, 399, 04056.

Mourad Benmalek (2024). Ransomware on cyber-physical systems: Taxonomies, case studies, security gaps, and open challenges. Internet of Things and Cyber-Physical Systems 4 (2024), 186–202

SI-CERT. (2021). Advanced ransomware attacks. SI-CERT. Retrieved January 22, 2025

Masum, M., Faruk, M. J. H., Shahriar, H., Qian, K., Lo, D., & Adnan, M. I. (2022). Ransomware classification and detection with machine learning algorithms. Conference: IEEE CCWC 2022: IEEE Annual Computing and Communication Workshop and Conference at: USA

Kumar, S., Rajkumar, N., Rayudu, K., Vasanthi, A., Tharani, B., & Kumar, S. R. (2024). Enhancing ransomware detection in cybersecurity: A comprehensive ensemble approach. Journal of Electrical Systems, 20(3), 5315-5324

Zahoora, U., Khan, A., Rajarajan, M., Khan, S. H., Asam, M., & Jamal, T. (2022). Ransomware detection using deep learning based unsupervised feature extraction and a cost sensitive pareto ensemble classifier. Scientific Reports, 12(1).

Da Silva, C. M. R., de Castro, P. A. L., & César, C. de A. C. (2024). Ransomware Detection: Ensemble Machine Learning Models Using Disjoint Data, 1–6.

Kumar, S., Rajkumar, N., Rayudu, K., Vasanthi, A., Tharani, B., & Kumar, S. R. (2024). Enhancing ransomware detection in cybersecurity: A comprehensive ensemble approach. Journal of Electrical Systems, 20(3), 5315-5324

Singh, A., Mushtaq, Z., Abosaq, H., Mursal, S. N. F., Irfan, M., & Nowakowski, G. (2023). Enhancing Ransomware Attack Detection Using Transfer Learning and Deep Learning Ensemble Models on Cloud-Encrypted Data. Electronics.

Moreira, C., Moreira, D., & Sales, C. (2024). A comprehensive analysis combining structural features for detection of new ransomware families. Journal of Information Security and Applications.

Boyd, C. A., Johansson, S., & McAllister, C. J. (2024). A Novel Methodology for Automated Ransomware Detection via Deep Behavioral Sequence Mapping.

Ismail, S., El Mrabet, Z., & Reza, T. (2023). An ensemble-based machine learning approach for cyber-attacks detection in wireless sensor networks. *Applied Sciences, 13*(1), 30.

Tabbaa, M., Ifzarne, A., & Hafidi, M. (2023). An online ensemble learning model for detecting attacks in wireless sensor networks. *Computing and Informatics, 42*(4), 1013-1038.