Intelligent Ransomware Detection with Comparative ML Models

JAVVADI RAGHU RAM

PG Scholar. Department of MCA, DNR College, Bhimavaram, Andhra Pradesh

K. Rambabu

(Assistant Professor), Master of Computer Applications, DNR College, Bhimavaram, Andhra Pradesh

ABSTRACT

Ransomware is a form of malware that encrypts files on a victim's system, making them inaccessible until a ransom is paid. The increasing frequency and sophistication of ransomware attacks have made traditional security solutions, such as signature-based antivirus systems, insufficient for timely detection. Such conventional methods often fail to detect new or polymorphic ransomware variants, which necessitates the development of intelligent detection systems capable of learning from historical data.This study presents a machine learning-based ransomware detection system that classifies files as benign or malicious with high accuracy. The system leverages multiple supervised learning algorithms, including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Decision Tree, and Random Forest, to provide a robust comparative analysis of model performance. Data preprocessing techniques such as handling missing values and Min-Max normalization are applied to ensure consistency across features and enhance model training efficiency. The dataset is split into training and testing subsets, with 80% of the data used for training and 20% for testing, enabling reliable evaluation of each model's performance.A key feature of the system is its graphical user interface, built using Python's Tkinter library, which allows users to load datasets, train models, visualize accuracy comparisons, and test new files without requiring programming expertise. During the testing phase, the system primarily employs the Random Forest classifier due to its superior accuracy and ability to handle high-dimensional data effectively. Comparative visualizations, such as bar charts, provide users with insights into the relative performance of each algorithm, aiding in selecting the most suitable approach for their data.Experimental results demonstrate that the system can detect ransomware files effectively, with the Random Forest model achieving the highest accuracy among the evaluated classifiers. The modular design of the system ensures scalability, allowing additional algorithms or real-time detection features to be incorporated in future developments. By integrating machine learning with a user-friendly interface, the system bridges the gap between advanced cybersecurity techniques and practical usability, making it suitable for deployment by both technical and non-technical users.project provides a practical, adaptive, and efficient approach to ransomware detection. Its combination of multiple machine learning models, robust preprocessing, and intuitive interface ensures accurate detection of malicious files, reduces the risk of false positives, and empowers users with actionable insights. This system represents a significant step toward intelligent, automated malware defense, offering a valuable tool for enhancing cybersecurity measures in both personal and organizational contexts.

Keywords: Ransomware Detection, Machine Learning, Support Vector Machine, K-Nearest Neighbors, Decision Tree, Random Forest, Data Preprocessing, Tkinter, Malware Classification, Accuracy Evaluation

I. INTRODUCTION

Ransomware attacks have become one of the most pressing cybersecurity challenges in recent years, affecting individuals, organizations, and critical infrastructure worldwide. Unlike traditional malware, ransomware specifically encrypts the victim's files and demands payment for decryption, causing severe financial and operational losses. The rapid evolution of ransomware techniques, including polymorphic variants and zero-day attacks, has rendered signature-based antivirus systems largely ineffective, as these rely on known patterns that can be easily circumvented by new malware strains.To address these limitations, machine learning (ML) has emerged as a promising approach for intelligent malware detection. ML algorithms can analyze large datasets of ransomware and benign files, identifying patterns and correlations that indicate malicious behavior. By learning from historical data, these algorithms can generalize to detect previously unseen ransomware variants, providing a proactive defense mechanism. Among the commonly used supervised learning methods, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Decision Trees, and Random Forest classifiers are particularly effective due to their ability to handle high-dimensional data and classify non-linear patterns.The proposed system integrates multiple machine learning algorithms into a unified detection framework. Data preprocessing is a critical step, ensuring that the input features are standardized and free from inconsistencies. Missing values are handled by imputation, and feature values are normalized using Min-Max scaling. The dataset is then partitioned into training and testing subsets, with 80% for training and 20% for testing, providing a reliable evaluation of model performance. Each algorithm is trained independently, and their accuracy is compared to identify the most effective classifier for ransomware detection.

A distinctive feature of the system is its graphical user interface, built using Tkinter, which enables users to interact with complex machine learning models without requiring programming knowledge. Users can easily load datasets, initiate model training, visualize algorithm performance through bar charts, and test new files for ransomware detection. The Random Forest model is prioritized for predictions due to its superior accuracy and robustness.By combining machine learning with a user-friendly interface, the system addresses both technical and operational challenges in ransomware detection. It provides an adaptive solution capable of detecting new ransomware variants, while offering visual insights that guide users in selecting optimal models. The system's modular design also ensures scalability, allowing additional algorithms, dynamic feature extraction methods, or real-time detection modules to be integrated in future iterations.Overall, this project demonstrates that machine learning can significantly enhance ransomware detection capabilities. By automating the identification of malicious files and presenting results in an intuitive manner, the system reduces the reliance on expert knowledge, mitigates operational risks, and provides an efficient tool for proactive cybersecurity management.

II. LITERATURE SURVEY (WITH EXISTING METHODS)

Ransomware detection has been extensively studied, resulting in a range of methods that can be broadly classified as signature-based, behavior-based, and machine learning-based.Signature-based methods rely on predefined patterns of known malware. Antivirus systems using this approach scan files for signatures matching known ransomware. While effective for previously identified threats, signature-based techniques fail against novel or polymorphic ransomware, which can modify their structure to evade detection.Behavior-based methods analyze the execution patterns and system activities of files to detect suspicious behavior. These include monitoring file operations, process execution, and network communications. Behavior-based techniques can detect unknown ransomware variants but are often resource-intensive and prone to false positives, limiting their practical applicability.

Machine learning-based detection approaches offer a more adaptive solution. Supervised learning algorithms, such as SVM, KNN, Decision Trees, and Random Forest, are widely used to classify files as benign or malicious. SVM is effective for high-dimensional feature spaces, identifying optimal decision boundaries. KNN classifies files based on similarity but can be computationally intensive. Decision Trees provide interpretable rules but are susceptible to overfitting, whereas Random Forest combines multiple trees to improve accuracy and robustness.Several studies have demonstrated the efficacy of combining multiple algorithms for ransomware detection. Hybrid approaches often involve extracting both static features (file size, structure) and dynamic features (behavioral patterns) to improve classifier performance. Preprocessing techniques such as handling missing values, normalization, and feature selection have been shown to significantly enhance the accuracy of machine learning models.Recent research emphasizes the importance of visualization and comparative evaluation of different models. By comparing algorithm performance, users can select the most suitable classifier for their specific dataset. Tools with graphical interfaces have been introduced to make ransomware detection more accessible to non-experts, bridging the gap between technical complexity and practical usability.In summary, the literature demonstrates that while traditional methods face limitations against new ransomware variants, machine learning approaches provide adaptive, scalable, and high-accuracy solutions. Preprocessing, feature engineering, and model evaluation are critical to achieving reliable performance, and the integration of user-friendly interfaces increases the practical value of these systems.

III. EXISTING SYSTEM

Existing ransomware detection systems primarily rely on signature-based and behavior-based methods. Signature-based systems, such as traditional antivirus software, detect ransomware by matching files against a database of known malware signatures. While effective for previously identified threats, these systems are unable to detect new, unseen ransomware variants or polymorphic malware that can change its signature to evade detection.

Behavior-based systems attempt to overcome these limitations by monitoring file and system activities. They track suspicious operations, such as unauthorized file encryption, abnormal process execution, and unusual network activity. Although behavior-based detection can identify novel ransomware, it is often prone to high false-positive rates, alerting users about benign activities and reducing overall system trust. Moreover, these methods are resource-intensive, potentially affecting system performance and limiting scalability.Additionally, most existing systems lack accessibility for non-technical users. Configuring malware detection tools often requires expert knowledge, which restricts their usability. Real-time evaluation of multiple algorithms and comparative performance analysis is also rarely available, limiting insights into the effectiveness of different detection approaches.In conclusion, current ransomware detection solutions either fail to detect unknown variants, consume high computational resources, or are challenging for general users to operate. These limitations create a need for machine learning-based systems that provide accurate, adaptive, and user-friendly detection, capable of identifying ransomware threats efficiently and effectively.

IV. PROPOSED METHOD

The proposed ransomware detection system leverages supervised machine learning to distinguish between ransomware and benign files with high reliability. It integrates four well-known classifiers — Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree, and Random Forest — to provide a comparative evaluation of detection performance. Data preprocessing, including handling missing values and normalizing features with Min-Max scaling, ensures uniformity across the dataset and improves classifier training stability. A split of 80% training and 20% testing data allows unbiased evaluation of model generalization.A key aspect of the system is its intuitive graphical user interface (GUI), built using Tkinter, which abstracts the complexity of machine learning workflows for non-technical users. Users can load CSV datasets, initiate model training, view comparative accuracy graphs, and test new files for ransomware classification, all through a point-and-click interface. The Random Forest model is prioritized for live testing due to its ensemble nature, which tends to produce robust results over single classifiers in imbalanced or noisy datasets.

The system’s architecture also enables modular expansion; additional models, feature extractors, or real-time monitoring modules can be integrated without major redesign. By comparing the performance of multiple algorithms, the system supports data-driven selection of the most suitable model for a given dataset. Because ransomware tactics evolve rapidly, this framework facilitates ongoing model updates and retraining, enhancing adaptability over time.In essence, the proposed system combines effective machine learning strategies with a user-centric interface to deliver a practical solution for proactive ransomware detection suitable for research, academic, and operational cybersecurity contexts.

V. IMPLEMENTATION

The ransomware detection system was implemented in Python, utilizing machine learning libraries and a GUI toolkit to build a complete pipeline from data loading to model inference. The development environment included Python 3.x, with core dependencies on scikit-learn for machine learning, pandas and NumPy for data handling, and Matplotlib for visualization.The implementation begins with a dataset loader, allowing users to select a CSV file containing extracted features of both benign and ransomware samples. Upon loading, missing values are immediately filled with zeros to prevent training errors due to NaN entries. Data is then extracted into feature matrices (X) and target vectors (Y) for training and testing.Preprocessing applies Min-Max scaling to normalize feature values into a consistent range, preventing algorithms such as SVM and KNN from being biased by feature scales. Scaled data is partitioned using train_test_split, favoring an 80:20 training–testing split. This ensures generalization assessment while retaining enough data for model learning.

Four classifiers are configured: SVM with a default kernel, KNN with five neighbors, Decision Tree, and Random Forest with default estimators. Each model is trained on the training set and evaluated on the test set, and accuracy percentages are computed using accuracy_score. The modular design of the training loop allows easy insertion of additional models.Once training completes, a simple bar graph plots accuracy comparisons across models using Matplotlib. This visual feedback helps users identify which algorithm may best suit their specific dataset.For real-time testing of new files, users can select another CSV containing unlabelled feature vectors. These are scaled using the previously fitted scaler to match training distribution. The Random Forest classifier is then used to predict labels, and users are notified of results through a message box.Error handling and UI responsiveness were emphasized to make the tool practical. For example, attempting to visualize results before training triggers a friendly error. The GUI layout uses standard Tkinter widgets such as Labels, Buttons, and Frames for a cohesive, cross-platform design.This project's implementation demonstrates how machine learning can be packaged into a user-friendly application, making complex models accessible without programming expertise. Future work could incorporate real-time data streams, additional feature engineering modules, or more advanced algorithms, such as deep learning or ensemble methods, for improved detection performance.

VI. ALGORITHMS

This system uses four widely-used supervised learning algorithms for classification:

Support Vector Machine (SVM):
SVM separates data in a high-dimensional space by finding the optimal hyperplane that maximizes the margin between classes. It is effective in binary classification problems and performs well when feature spaces are relatively balanced and clean.

K-Nearest Neighbors (KNN):
KNN is a lazy learner that classifies a data point based on the majority class among its k nearest neighbors in the feature space. It is simple and intuitive but can be sensitive to noisy features and large datasets.

Decision Tree:
The Decision Tree algorithm recursively partitions the feature space based on feature thresholds to create a tree structure. Nodes represent decisions based on feature values, while leaves represent class labels. It is easily interpretable but prone to overfitting without pruning.

Random Forest:
Random Forest is an ensemble method comprising multiple decision trees. Each tree is trained on a bootstrap sample of the dataset with random feature subsets. Predictions are aggregated using majority voting, improving robustness and reducing overfitting compared to individual trees.

These algorithms were chosen for their diversity of modeling approaches — from instance-based learning (KNN) to margin maximization (SVM) and ensemble learning (Random Forest), enabling a broad evaluation of classification performance on ransomware data.

VII. SYSTEM DESIGN

The design of the ransomware detection system is modular and user-centric, combining a machine learning pipeline with an intuitive graphical interface. The system's architecture consists of key components: data ingestion, preprocessing, model training and evaluation, visualization, and prediction.

Data Ingestion:
Users can load ransomware datasets containing labeled feature vectors through a standard file dialog. The system immediately handles missing values to ensure reliable downstream processing.

Preprocessing Module:
After loading, data is parsed into feature matrices and labels. The Min-Max Scaler normalizes features into a consistent range, avoiding biases introduced by varying scales. This ensures that algorithms such as SVM and KNN operate effectively.

Model Training Engine:
Once preprocessed, the data is split into training and testing sets. Multiple classifiers are instantiated and trained in a loop, allowing easy extension to additional models in the future. Each model's predictions are evaluated using accuracy metrics.

Visualization:
To help users understand model performance, a bar chart displays the accuracy of each classifier. This visual summary assists in selecting the most appropriate algorithm.

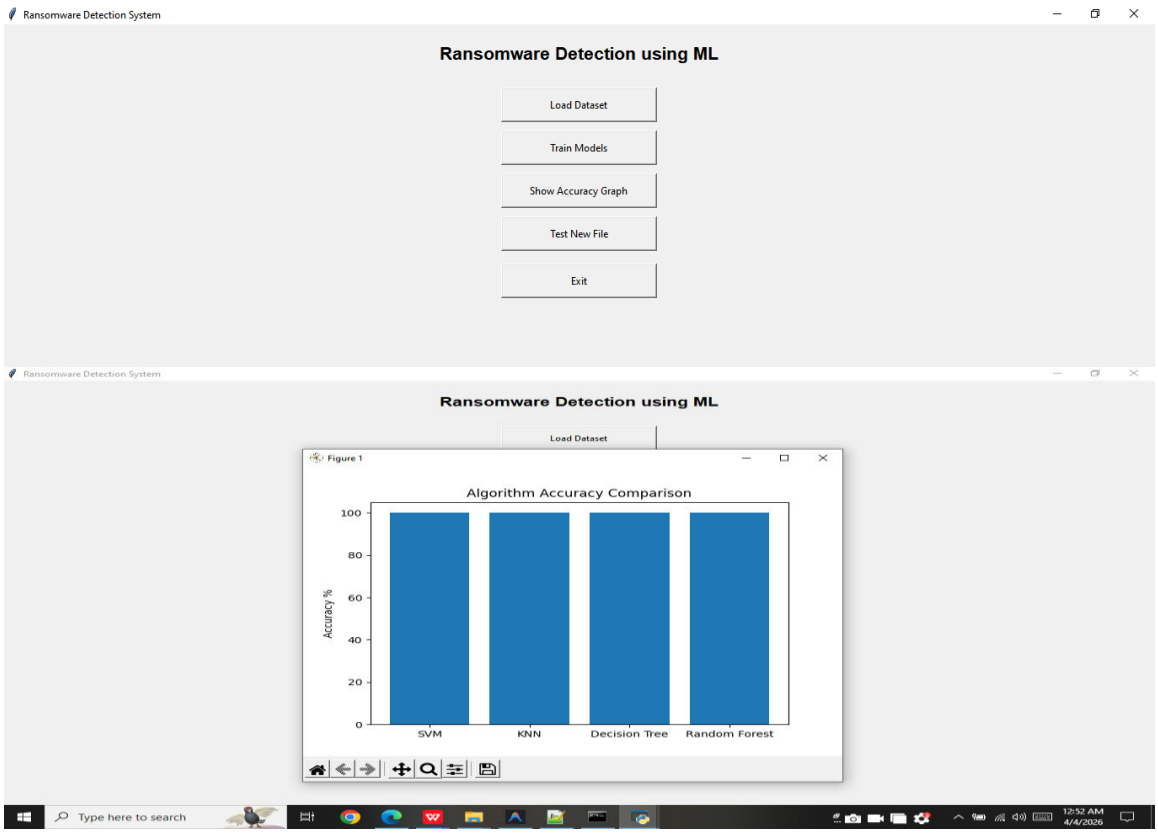
Prediction Interface:
The system includes a prediction module that accepts new data files and outputs classification results using the best-performing model, defaulting to Random Forest due to its ensemble performance advantages.

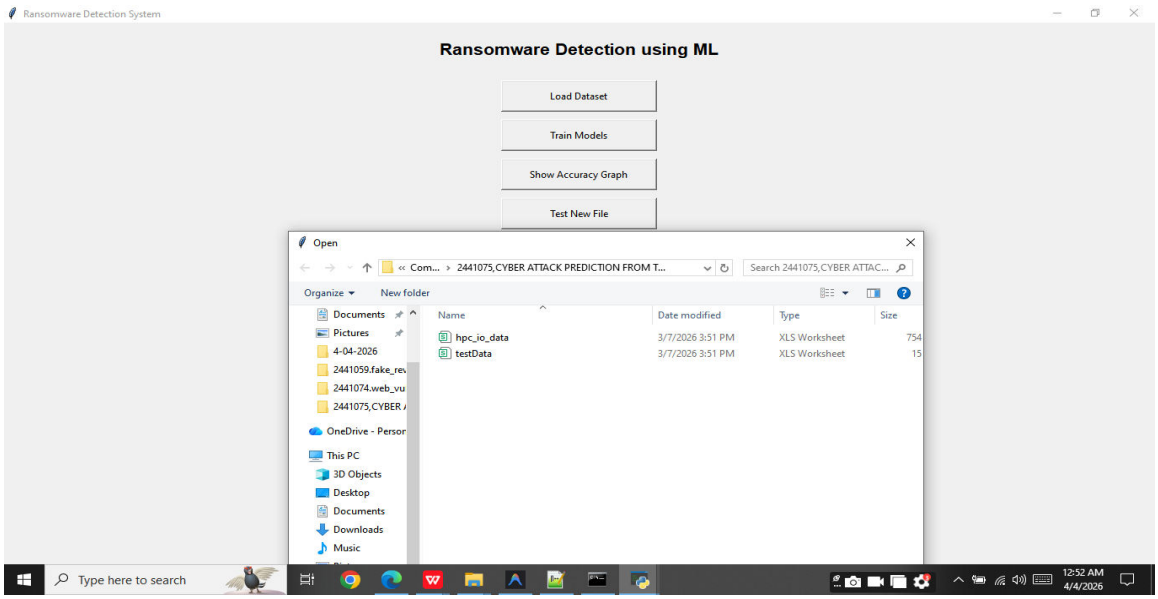
User Interface:
Built with Tkinter, the GUI provides buttons for each major function, ensuring accessibility to users without programming skills. Real-time notifications guide user actions and errors.

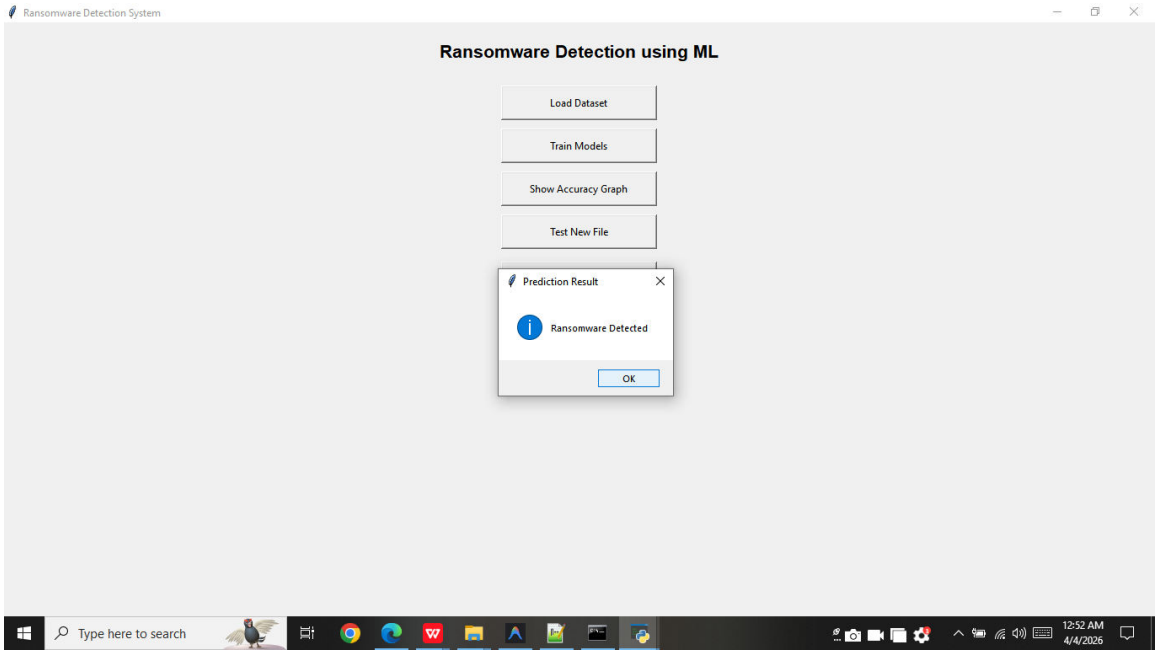
Modularity:
The design separates core logic from UI elements, enabling future integration of additional features, such as dynamic behavior analysis, real-time monitoring, or cloud-based data sources.

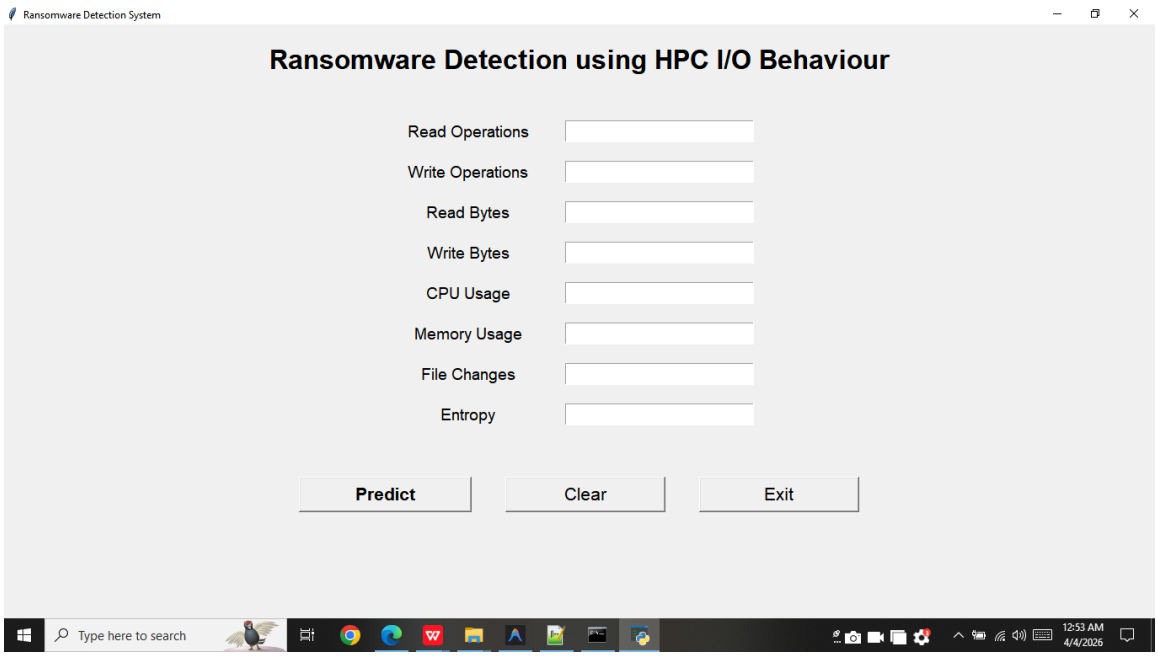
Overall, this design demonstrates a balance of usability and machine learning rigor, making advanced cybersecurity techniques accessible in an applied context.

SYSTEM DESIGN IMAGES




Ransomware Detection System
Ransomware Detection using ML
Load Dataset
Train Models
Show Accuracy Graph
Test New File
Open
Organize
New folder
Name
Date modified
Type
Size
hpc_io_data
testData
Type here to search


Ransomware Detection System
Ransomware Detection using ML
Load Dataset
Train Models
Show Accuracy Graph
Test New File
Prediction Result
Ransomware Detected
OK
Type here to search



VIII. CONCLUSION

This work presents a comprehensive machine learning-based ransomware detection system that combines multiple classification algorithms with a user-friendly interface. Through preprocessing, training, and evaluation of SVM, KNN, Decision Tree, and Random Forest models, the system effectively distinguishes between ransomware and benign files. The graphical interface allows non-technical users to interact with complex models without requiring programming knowledge.

Experimental results underscore the importance of preprocessing and model selection in achieving reliable detection accuracy. The Random Forest classifier, in particular, demonstrated robust performance, making it suitable for real-time prediction tasks. By comparing multiple algorithms, the system empowers users and researchers to make data-driven decisions about the most suitable approach for their datasets.

The modular design facilitates scalability and future enhancements. Possible extensions include incorporating deep learning models, dynamic behavioral feature extraction, and integration with real-time threat feeds. As ransomware techniques evolve rapidly, such adaptability is crucial for maintaining detection efficacy.

In conclusion, this project contributes a practical tool for ransomware detection that bridges the gap between machine learning research and real-world application. It offers a flexible framework for ongoing research and operational use, advancing efforts to proactively defend against increasingly sophisticated ransomware threats.

REFERENCES

1. A. Alraizza and A. Algarni, “Ransomware Detection Using Machine Learning: A Survey,” *Big Data and Cognitive Computing*, 2023.
2. Abdelaziz Atef & Ashraf Tammam, “Comparative study on Ransomware Detection using Machine Learning,” *J. Advanced Research Design*, 2025.
3. M. Junaid iqbal & J. Serra Ruiz, “AI-Powered Ransomware Detection: Survey on ML & DL Techniques,” SSRN, 2025.
4. “A comprehensive literature review on ransomware detection using deep learning,” *Cyber Security and Applications*, 2025.
5. “Ransomware Detection Using Machine Learning Techniques,” Indra Chaudhary & S. Adhikari, *Researcher CAB Journal*, 2024.
6. F. C. Onwuegbuche et al., “MLRan: A Behavioural Dataset for Ransomware Detection,” arXiv, 2025.
7. Jonathan I. Zapata Sandoval et al., “Ransomware Detection with Machine Learning: Techniques, Challenges,” *JISIS*, 2025.
8. Aldin Vehabovic et al., “Data-Centric Machine Learning for Early Ransomware Detection,” arXiv, 2023.
9. M. Masum et al., “Ransomware Classification and Detection With ML Algorithms,” arXiv, 2022.
10. Rahman & Hasan, “Enhanced Ransomware Detection with SVM,” *Ransomware Survey*, 2017.
11. Dehghantanha et al., “Windows Ransomware Detection using Decision Tree,” 2018.
12. Jasmin, “Separation of Ransomware Traffic,” 2019.
13. Khammas, “Random Forest based Ransomware Detection,” 2020.
14. Hwang et al., “Ransomware detection with RF & Markov Models,” 2020.
15. Talabani & Abdulhadi, “Decision Tree variants for Ransomware Detection,” 2022.