Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Malware Detection using Machine Learning

Mr. Lochan Gowda M
Asst. Professor
Department of CSE, SJBIT

Likhitha Y A, Likitha M, Mythri R V
Department of Computer Science and Engineering,
SJB Institute of Technology,
Bengaluru, India

***Abstract* - Malware poses a risk to Windows computers, and antivirus software frequently fails. This project detects malware using machine learning, more precisely a Decision Tree. Without opening the programs, it examines information from executable files [PE], such as headers, sections, import tables, and entropy. The Decision Tree uses the CSV files to learn and make predictions. The end product is a quick, dependable system that can instantly determine if a file is malicious or safe.**

Keywords: Executable files [PE], machine learning, malware, and decision trees.

I. INTRODUCTION

Detecting malware effectively requires methods that can adapt to new and evolving threats. By analysing features extracted from PE files, it is possible to identify malicious files based on patterns rather than relying solely on known signatures, which traditional antivirus solutions often miss [5]. Using machine learning, the model learns from both safe and harmful files, creating clear decision rules that highlight which characteristics are most important for classification. This approach not only improves detection accuracy but also ensures safety, as files can be analysed without being executed. The workflow leverages CSV files containing structured PE features, allowing seamless integration into standard machine learning pipelines while maintaining efficiency and interpretability, making it easier for analysts to understand and trust the results [1][3][6][8].

Malware has emerged as one of the most urgent cybersecurity threats due to the exponential growth of software systems. Data theft, resource damage, and unauthorized access are all possible outcomes of malicious executables. Conventional antivirus programs don't detect new or obfuscated malware because they rely on known signatures [5].

Machine learning offers a promising alternative by learning patterns from large datasets of both malicious and legitimate files. By analysing structural properties of PE files, it is possible to classify new files without executing them, which ensures safety and efficiency [1][3].

This study focuses on using the Decision Tree algorithm, which provides clear, interpretable rules for classification, making it suitable for understanding which features contribute most to malware detection. The workflow relies on CSV files containing extracted PE features, enabling straightforward integration with standard ML pipelines [6][8].

II. LITERATURE SURVEY

Because of the ongoing rise in cyberthreats and the growing limitations of conventional antivirus software, malware detection has emerged as a crucial field of study. Signature-based techniques, which compare files to a database of recognized malware patterns, were the mainstay of early malware detection systems. These techniques work well for threats that have already been identified, but they have trouble identifying recently created, obfuscated threats. The significance of intelligent detection methods that can adjust to changing malware behaviour is highlighted by recent studies. In order to successfully identify complex and unknown threats, Rajendran et al. [1] stressed that modern cybersecurity systems must move beyond fixed signatures and adopt learning-based approaches. Their research demonstrates that in dynamic cyber environments, data-driven models greatly enhance detection capabilities.
Patil and Deng [2] studied both the machine learning-based methods and deep learning for malware analysis, and showed that the static analysis of PE/EXE files can be used to reach high confident in detecting without executing potentially suspicious programs. Their results demonstrate that feature extraction from executables can be used to identify malicious software relatively easily compared to legitimate programs.

Other researchers have been working on PE files analysis for malware detection. Jain and Sharma [8] demonstrated that PE file features, including headers, section information, and entropy values also present evident obverse between benign and malware. Statically analyzable structural features For these kinds of structure, they can be safely extracted and processed, thus static analysis is suitable to large-scale and real-time malware detection systems.

Newaz et al. [3] experimentally compared various machine learning models and revealed that tree-based classifiers are highly efficient on structured feature data. They made it clear

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

that models such as Decision Trees can provide high level of accuracy while maintaining interpretability, a key aspect to security analysts that for them is necessary to understand the decisions from classification.

Indumathi et al. [4] studied deep learning malware detection and obtained good classification performance. In their study, the authors also showed that deep learning models are computationally more expensive and require massive data which may restrict them to be used in lightweight detection systems. Such results advocate in favor of simpler supervised learning models in low-resource settings.

Microsoft's documentation on the Portable Executable format [5] is a key resource for learning about the internal structure of Windows executables. It details how PE headers, import tables and section layouts can be examined to retrieve useful snippets of malicious behaviour before you run anything.

Pedregosa et al. [6] presented scikit-learn, a framework used by many for machine learning in cybersecurity research. It also supports Decision Tree classifiers and evaluation metrics, making it applicable to CSV feature dataset-based malware detection.

Additionally, studies published by IJRASET [7] and Newaz et al. [11] further confirm that machine learning-based static analysis provides a fast, safe, and scalable solution for malware detection. These works highlight the importance of feature selection and proper evaluation metrics to achieve reliable performance on imbalanced malware datasets.

Overall, existing literature clearly supports the effectiveness of static PE file analysis combined with machine learning techniques for malware detection. While deep learning models show strong potential, Decision Tree classifiers remain a practical choice due to their simplicity, transparency, and low computational cost. These research findings strongly motivate the use of a Decision Tree–based approach for detecting malicious executables using CSV-formatted PE features.

III. METHODOLOGY

A. Data Collection

The dataset comprises malicious and legitimate Windows PE files. Malicious files were sourced from publicly available malware repositories, while legitimate files were collected from verified Windows system directories and open-source software. Each executable was verified for integrity, labelled manually, and cross-checked [2][3].

B. Data Preprocessing

1. **Feature Extraction:** PE headers, section characteristics and entropy values were extracted from each file.
2. **CSV Storage:** All extracted features were saved in structured CSV files to facilitate machine learning processing.
3. **Normalization:** Min-Max scaling was applied to numeric features to ensure uniformity and reduce bias during training.
4. **Train-Test Split:** The dataset was divided into training and testing sets (typically 80%-20%) to evaluate model performance [6][8].

C. Confusion Matrix

A confusion matrix provides an effective way to evaluate how accurately the Malware Detection System categorises the executable files (PE). Unlike the overall accuracy of the system, the confusion matrix separates the correct and incorrect classifications for each class. Thus, by examining the confusion matrix, we can make a better assessment of the Decision Tree model's accuracy in identifying malicious PE files versus benign ones [1][3].

A confusion matrix has four types of results:

- **True positives (TP):** These represent malicious PE files that are accurately identified by the Decision Tree as a malware file.
- **True negatives (TN)**: They are executable files that have been confirmed to be executed, and classified as benign, an accurate representation of the truth.
- **False positives (FP)**: It represent legitimate files that have been incorrectly identified as malware. A large number of false positive errors may cause users to lose trust in the detection process and trigger superfluous alerts.
- **False negatives (FN):** It represent malware files which have been incorrectly classified as benign, which presents the highest risk of compromise to a user's system as it allows malicious software to circumvent all means of protection against potential damages [2][8].

Predicted Class	Malicious	Legitimate
Actual Malicious	True Positive [TP]	False Negative [FN]
Actual Legitimate	False Positive [FP]	True Negative [TN]

Fig.1. Confusion matrix

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Using the confusion matrix, important evaluation metrics such as Precision, Recall, and F1-score can be accurately calculated. These metrics are especially important for malware detection tasks, where datasets are often imbalanced and accuracy alone may be misleading [3][6]. In real-world cybersecurity systems, reducing false negatives is critical to ensure that malware is detected before it can cause harm [1][11].

D. Performance Evaluation

To evaluate the performance of the models, we used common metric evaluation techniques.

- **Accuracy:** Percentage of accurately classified documents
- **Precision:** True positive (TP) to predicted positive (PP) ratio
- **Recall:** TP to actual positive (AP) ratio
- **F1 Score:** Average of precision and recall (harmonic mean) [1][3].

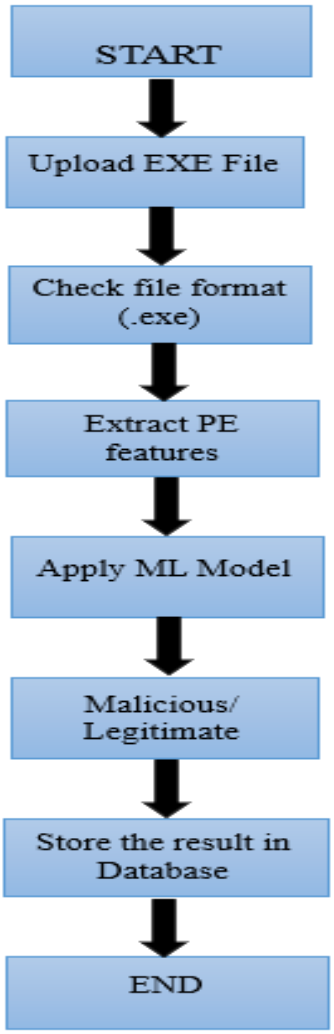

Fig.2. Flowchart

IV. Feature Extraction and Analysis

Static analysis allows safe extraction of executable features without running the program. Key features analysed in this project include:

This project will use static analysis instead of running an executable to safely extract all of the executable's executable features. The key features being analysed in this project are:

1. **PE Header Attributes:** This includes the basic metadata for the executable, such as entry point, file size and structure.

2. **Sections:** The PE file is separated into logical sections (Code, Data and resources). Sections allow us to identify how the executable is configured in the operating system.

3. **Import Tables:** These lists include Windows API calls as well as libraries that the PE executable will use.

4. **Entropy:** The amount of entropy in a PE file's sections will help to determine if the file has been packed/encrypted.

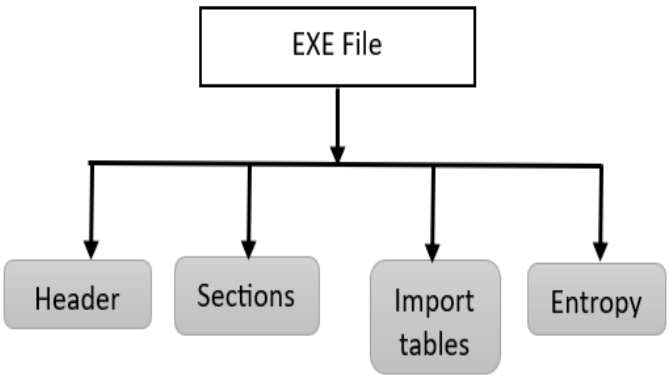

Fig.3. EXE File analysis

The extracted features were aggregated into CSV files. Feature importance analysis using the Decision Tree revealed that entropy values, and PE header anomalies were the most discriminative for detecting malware.

E. Decision Tree Classifier

The Decision Tree Algorithm was selected because it is simple to comprehend; it can utilize both numerical and categorical formats of data. The algorithm segments the dataset at intervals by using the values of the independent variables (features), therefore forming branches for every category until all items in the dataset have been categorized. Because of the splits and branches of the Decision Tree, it is visually intuitive for understanding the reasoning behind how the model arrives at an outcome. In addition to using categories, we made changes to settings that control the algorithm's growth to ensure more accurate predictions and reduce the risk of overfitting (the algorithm learning from the input dataset instead of identifying patterns within the dataset) by adjusting parameters such as the maximum depth of the tree and the minimum number of observations required in each branch. By altering these parameters, we increase the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

model's ability to generalize from the training dataset to new datasets that the model has not seen before; therefore, yielding more accurate predictions of an outcome when using a Decision Tree Model for classifying outcomes on a new dataset. Decision Tree Models are useful for classifying malware because of their ability to classify other types of complex data while remaining interpretable for why the model categorized a file as malicious or safe. [1][3][8]

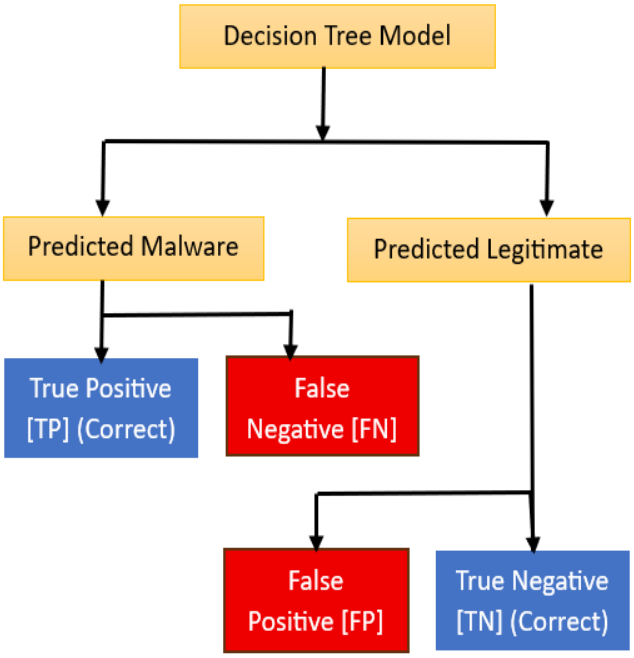

Fig.4. Decision Tree Model

V. RELATED WORK

Previous studies have highlighted the effectiveness of combining static analysis with machine learning for malware detection. Early works focused on signature-based detection, which cannot handle zero-day or polymorphic malware [1][2].

Recent research emphasizes the use of PE structural features for classification. Studies have shown that Decision Trees, Random Forests, and Support Vector Machines can achieve high accuracy by learning from these features [3][4][8]. CSV based pipelines simplify the preprocessing and training workflow, enabling rapid experimentation and reproducibility [6][7].

Moreover, combining multiple static features such as header anomalies, and entropy improves model generalization for previously unseen malware variants [2][11].

VI. RESULTS

The Decision Tree–based malware detection system demonstrated strong performance in classifying Windows Portable Executable (PE) files as benign or malicious. By using static analysis to extract features such as PE headers, section information, import tables, and entropy, the model was able to identify patterns that differentiate malware from legitimate programs without executing the files, ensuring safety during analysis [2][8]. Feature importance analysis revealed that anomalies in PE headers and high entropy values were the most discriminative indicators of malicious behaviour, consistent with findings from prior studies on PE structural features [3][1]. The Decision Tree classifier achieved high accuracy while remaining interpretable, allowing analysts to understand why a file was classified as malicious or safe [6][11]. Overall, the results indicate that this system provides a fast, reliable, and safe solution for real-time malware detection, making it suitable for deployment in Windows environments [8][4].

VII. CONCLUSION AND FUTURE WORK

The system will use user-defined attributes to characterize executable files. This allows for separating trusted or safe applications from potentially dangerous or malicious ones without running the application. This greatly reduces the user's possibility of having their system compromised or infected by a malicious executable, while still providing excellent levels of performance.

The analysis of the experimental results also indicates that PE header information and entropy values for the PE file format can both be significantly associated with malware or suspicious files. The Decision Tree also provides a high degree of accuracy and interpretability which would allow for this type of technology to be implemented in real-time malware scanning applications [1][3][8].

Future enhancements include:

- Integrating dynamic analysis features (runtime behaviours, system calls, network traffic) to improve detection of packed or heavily obfuscated malware.
- Using ensemble methods or deep learning to automatically learn complex feature representations.
- Expanding the CSV-based pipeline to include hybrid datasets from multiple repositories for better generalization.
- Developing a real-time detection system with automated dataset updates to handle evolving malware threats [2][4][11].

REFERENCES

[1] Rajendran, M. I. N. Mohamed, J. K. Jagadeesh, and B. Sampathkumar, “Cybersecurity Threat Detection Using Deep Learning,” IEEE Access, 2024.
[2] R. Patil and W. Deng, “Malware Analysis using Machine Learning and Deep Learning Techniques,” International Journal of Computer Applications, 2020.
[3] S. Newaz, H. M. Imran, and X. Liu, “Experimental Analysis of Malware Detection and Classification,” Journal of Cyber Security Technology, 2023.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

[4] Indumathi, R. Krishna Prasanna, G. Chamundeeswari, M. Preetha, and T. K. Tamilvani, "Detection of Malware Using Deep Learning," International Journal of Engineering Research & Technology (IJERT), 2021.
[5] Microsoft, "Portable Executable (PE) Format Documentation," Microsoft Docs.
[6] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, 2011.
[7] "Malware Detection Using Machine Learning Techniques," IJRASET.
[8] V. Jain and A. K. Sharma, "Static Analysis of PE Files for Malware Detection," IEEE Xplore, 2018.
[9] "Flask Web Framework Documentation," https://flask.palletsprojects.com/en/latest/.
[10] "SQLite Database Documentation," https://www.sqlite.org/docs.html.
[11] Sabila Newaz, Hasan Md Imran, Xingya Liu, "Detection Of Malware Using Deep Learning", 2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), University of Malaya, Kuala Lumpur, Malaysia, Sep 24-26, 2021.