PriMera Scientific Engineering
Volume 8 Issue 6 June 2026
DOI: 10.56831/PSEN-08-273
ISSN: 2834-2550

PriMera Scientific Publications

A Forensically Grounded Machine Learning Framework for Ransomware Family Classification via Behavioral Analysis

Type: Research Article
Received: May 08, 2026
Published: May 31, 2026

Citation:
Bokolo Wanengimorte George. "A Forensically Grounded Machine Learning Framework for Ransomware Family Classification via Behavioral Analysis". PriMera Scientific Engineering 8.6 (2026): 23-51.

Bokolo Wanengimorte George*

Afe-Babalola University, Nigeria

Corresponding Author: Bokolo Wanengimorte George, Afe-Babalola University, Nigeria.

Abstract

Ransomware continues to pose a critical threat to modern information systems, particularly due to the increasing prevalence of polymorphic and behaviorally evasive variants. This paper proposes a reproducible, forensically auditable machine learning pipeline for multi-class ransomware family classification based on dynamic behavioral features. Using the RanSAP-2022 dataset comprising ten ransomware families, the pipeline automates data ingestion, exploratory analysis, preprocessing, and model evaluation under deterministic controls. Exploratory Data Analysis reveals significant class imbalance, with the largest family accounting for 32.4% of samples and an imbalance ratio of 4.6:1 (Fig. 1), as well as 27% aggregate missingness concentrated in network-related features (Fig. 2). Feature correlation analysis (Fig. 3) identifies moderate behavioral coupling among file system, registry, and API activity features without severe multicollinearity. Six machine learning models are comparatively evaluated using macro-averaged F1-score as the primary metric. Boosting-based ensemble models achieve the strongest performance, with XGBoost attaining 96.1% accuracy and a macro-F1 score of 0.948 (Table 7). Confusion matrix analysis (Fig. 4) demonstrates high precision and recall for majority families, while highlighting reduced recall for minority families due to behavioral overlap. These findings confirm that behavior-based ensemble learning, when embedded in a reproducible and leakage-resistant pipeline, provides both high predictive performance and forensic reliability for ransomware family classification.

Keywords: Ransomware classification; behavioral malware analysis; machine learning; reproducible pipelines; digital forensics

Introduction

One of the most operationally disruptive as well as financially harmful cyber threat types today are ransomware, where the estimated worldwide loss between 2019 and 2023 is over USD 30 billion [1, 2]. Critical infrastructure fields such as energy, healthcare, finance, and municipal are disproportionately targeted with their low tolerance to downtime and historically inadequately-funded cybersecurity defenses [3]. The emergence of Ransomware-as-a-Service (RaaS) has also contributed to reducing the entry barrier such that even low-skill attackers can now use advanced and industrial-level attack tools.

Ransomware families can only be identified correctly and timely to respond to the incident effectively, prioritizing, key recovery, and threat attribution, and developing evidence in courts. Older forms of static analysis, like code signatures, import table fingerprinting, and hashing are becoming less useful against polymorphic and metamorphic forms [4]. Behavioral analysis, which observes runtime actions like file system events, registry events, network connections and API calls, is more resistant and capable of evasion [5].

Although machine learning has been improved in malware classification, there is still a disparity between what has been researched and what can be applied in the forensic case. Practical techniques tend to have no deterministic reproducibility, auditability, and documentation necessary to be legally admissible in standards like Daubert, Frye, ISO/IEC 27042 and NIST SP 800-86 [6]. This paper fills this gap by using fully parameterized and reproducible classification pipeline in a Jupyter notebook, making everything in the analysis process transparent, traceable and verifiable.

This work adds: (i) forensically reproducible end-to-end behavioral ransomware classifier; (ii) comparative analysis of 6 machine learning methods on a representative imbalanced 10-class dataset; and (iii) empirical results that XGBoost performs state of art (96.1% accuracy, macro-F1 = 0.948) with interpretability and auditability.

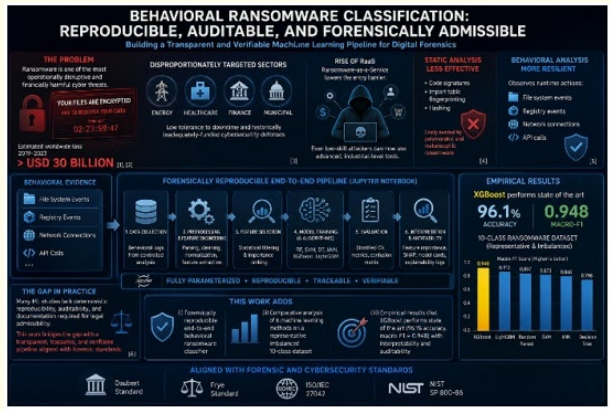

Figure 1: Forensically Reproducible Behavioral Ransomware Classification Framework and Threat Landscape Overview.

Research Objectives

The design of the experimental pipeline is governed by four major goals, each of which is associated with a respective forensic utility requirement and output quality criterion that can be measured:

1. Automatize the ingestion, cleaning, and exploratory analysis of the RanSAP-2022 behavioral data in an entirely reproducible form that removes any manual intervention points that invalidate the forensic traceability and verification of the data.
2. Systematic EDA Characterize the asymmetry of class distribution, inter-feature correlation structure, and data quality dimen-

sions, thus, giving the empirical basis to all the further preprocessing design choices.

3. Implement an architecturally sound preprocessing and feature engineering process with full parameter tracing, all transformation parameters should be estimated only on training partitions to avoid information leaks across the train-test barrier.
4. Implement a comparative evaluation system that makes it possible to have principled selection of the best classifier as determined by forensically suitable measures, such as the macro-averaged F1-score, the precision, the recall, and the confusion matrix-based per-class analysis.

Scope and Significance

This study is restricted because it is focused on data mining behavioral features in the form of sandboxed execution environments in order to give supervised classification of known ransomware families. Its value is not limited to performance benchmarking: by formalizing the full methodology of analytics into a pipeline to be instrumented and reproduced, such work has become a reusable template to operational threat intelligence centers and Computer Emergency Response Teams (CERTs) wishing to implement ML-based triage tools into legally defensible forensic structures [17]. The pipeline's modular design further allows the pipeline to quickly adapt to new ransomware families or feature spaces as the threat landscape changes, without needing to fundamentally redesign the analytical framework.

Literature Review

The theoretical and empirical foundations of this paper draw upon three closely interrelated research domains: (A) the historical evolution of ransomware and behavioral detection paradigms, (B) machine learning and deep learning for malware classification, and (C) forensic reproducibility and pipeline readiness standards. A comprehensive synthesis of contributions across all three domains is presented below, encompassing 31 key works that collectively inform the pipeline design, feature selection strategy, classifier choice, and evaluation methodology of this study.

Ransomware Evolution, Threat Landscape, and Behavioral Detection

Ransomware has become one of the most disruptive types of cybercrime that has impacted governments, businesses, and individuals across the globe. Ransomware is not like the conventional malware, as it is crafted to encrypt files or deny system access until a ransom is paid. The severity, rate, and cost of ransomware attacks have evolved over the last ten years, and it currently poses a critical issue in terms of cybersecurity research and practise [32]. CrowdStrike [1] and Kaspersky Lab [2] threat intelligence reports also help define the current size of the issue, with the cumulative global losses in the amount of USD 30 million and more, and emphasise the growing focus on targeting critical national infrastructure in healthcare, energy, and financial spheres.

The longitudinal background of the commoditization of ransomware-as-a-service (RaaS) platforms is important, as Symantec, in its Internet Security Threat Report [3], shows that the latter has been gaining rapid momentum in the commoditization industry. These sites enable the operation of technically unsophisticated actors with enterprise-grade cryptographic toolkit initially developed by advanced persistent threat (APT) groups. The threat has grown considerably due to this development and has also helped the ransomware families to spread rapidly. Attempting to explain the historical development of ransomware as a form of early cryptovirology experimentations to current multi-stage extortion, Abrams [19] states that there are still changes in the implementation of a code in terms of polymorphism and metamorphism, but the behavioural patterns used to extract the funds are structurally the same. The continuity of the behaviour patterns gives a robust incentive of the behavioural feature engineering methods in the ransomware detection and classification.

The conventional antivirus software operates mostly using signature-based detection mechanisms, in which they recognise malware by matching known signature samples or hashes. Although they are effective in countering the already known threats, they do not perform well in identifying new or modified ransomware versions that use obfuscation, polymorphism, and automated propagation methods to avoid detection [33]. The theoretical constraints of the use of malware analysis at rest were defined by Moser, Kruegel,

and Kirda [4], who had proven that the remaining problem of the static detection can be limited to the halting problem of sufficiently obfuscated code. This observation suggests that signature-based detection in isolation cannot be considered complete in the face of modern polymorphic malware and accordingly, it raises the need to transition to dynamic and behavioural analysis models.

It was on this theoretical basis that Kharraz et al. [5] operationalized the behavioural analysis paradigm by coming up with UNVEIL, a large scale automated system that ran 148,000 malware samples in sanboxed environments to identify behavioural indicators. They found that patterns of file system activities, registry activities, and frequencies of Windows API calls only alter comparatively between ransomware versions that are obfuscated. This finding was later verified by large-scale profiling of 1,359 ransomware specimen by Kharraz et al. [11], showing that about 98 of all ransomware families can only use five basic file system interaction methods. The presented empirical evidence shows that behavioral signatures are reliable and can be generalized to datasets like RanSAP-2022.

The applicability of the behavioural monitoring has also been verified by several practical detecting systems. One of the first ransomware detection systems, called CryptoLock was introduced by Scaife et al. [8] and it tracked file input/output entropy values and ratio of write-to-read operations to detect active encryption processes. The system recorded sub-second detection latency and was able to detect all the tried ransomware families without false positive. Likewise, Sgandurra et al. [9] created EldeRan, a Random Forest-based classifier which is trained on the behavioural features including the sequence of API calls and registry access pattern. Their system was also able to achieve high detection rates of 96.0% and low rates of false positives of 0.40, indicating the effectiveness of machine learning models resulting out of behavioural features.

Continella et al. [10] also made further progress with ShieldFS that was a self-healing filesystem, which used behavioural anomaly detection to generate files and revert affected files in milliseconds to thwart ransomware encryption. The behavioural features employed in this system such as file system write patterns, directory traversal rates and Shannon entropy of modified data have since become popular indicators in ransomware studies. To supplement these results, Morishita et al. [20] showed that API call sequence by itself could be used as highly discriminative properties to identify ransomware families. Their analysis, based on n-gramme frequency modelling and Random Forest classification, revealed that trends in the behavioural API patterns are good predictors of different ransomware families.

Combined, these investigations lead to the conclusion that behavioural features are one of the most effective to identify and categorise ransomware. Since ransomware cannot avoid communicating with system resources in order to achieve encryption and persistence, such actions are much harder to hide than fixed code signature. As a result, behavioural analysis and machine learning have become a paradigm in the current research on ransomware. The given paradigm underlies the current research, which is aimed at creating a reproducible machine learning pipeline that could classify ransomware families using behavioural characteristics.

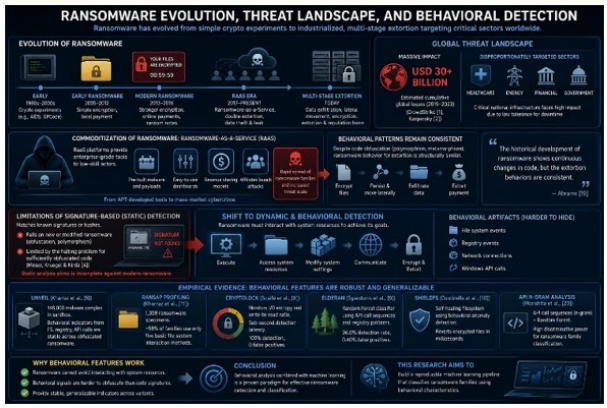

Figure 2: Evolution of Ransomware, Threat Landscape, and the Shift Toward Behavioral Detection Paradigms.

Machine Learning Approaches for Malware Classification

The emergence of malicious software and the rise in sophistication of cyber attacks has caused the culmination of cybersecurity research focus to transition to data-driven machine learning methods instead of the traditional signature-detection methods. Antivirus systems based on signature use known malware patterns (hashes), thus they are not effective against new or obfuscated malware. The machine learning techniques overcome this weakness by learning patterns using large volumes of data and detecting new threats with previously unknown attributes through statistical and behavioural properties [34]. Consequently, machine learning has been a key feature of current malware detection and classification software.

Automated malware classification is theory based on the early research into computational security. The theoretical work of Cohen [22] has been a foundational work in general malware detection, it has shown that solely signature based systems cannot be relied upon to detect all malicious programmes. This scientific understanding inspired the creation of behavioural and learning-based detection models with the ability to generalise outside of the known signatures.

Initial uses of machine learning in malware detection concerned classical supervised learning algorithms that could identify malicious code and benign programmes depending on features extracted. One of the first models that were used in this area was Decision Trees as it is interpretable and as well the computation cost is relatively low. These models cluster samples based on hierarchical decision making rules on programme characteristics that include opcode sequences, file metadata, and system behaviour [35].

The use of naive bayes classifiers was also highly embraced due to its probabilistic approach and its ability to deal with many features. Naïve Bayes models are used to determine the probability of a certain programme being in a malicious type by conditionally assuming being independent of the various features. Despite its overly simplified assumption of independence, these models were shown to be effective at the baseline in initial classification experiments [34].

The further development of Support Vector Machines (SVMs) was a potent method since they could develop the ideal dividing hyperplanes in the high-dimensional feature areas. As shown by Rieck et al. [12], SVMs, which have been trained on the API call frequency vectors, are capable of classifying malware samples based on family patterns with 71-99% classification rates and showing that behavioural signatures do have clearly defined cluster forms based on malware families. Their contribution came up with behavioural distance measures which have been basic in contemporary studies of behavioural detection.

The algorithms of the random forest also enhanced performance in detecting by using a combination of several decision trees in order to form ensemble models. Such an ensemble approach minimises the overfitting effect and maximises predictability, and researchers indicate excellent performance across malware datasets due to their capability to capture nonlinear interactions and high-dimensional data [34]. Sandbox-based Dynamic behavioural features/Multi-family In the same study by Pirscoveanu et al. [27], high-precision multi-family malware classification with Random Forest models has been demonstrated, which gives behavioural multi-class classification system methodological precedence.

Supervised learning has been among the most popular paradigms to classify malware. In this method, labelled datasets with benign and malicious samples are used to train models and classifiers are able to learn how the input features relate to the predefined class labels.

This is where feature-based classification is important. To describe the behaviour of programmes in structured form, researchers usually extract features out of executable files, system logs or network activity. Examples of these characteristics are opcode sequences, API calls, registry modify, file system actions, and network communication patterns [32].

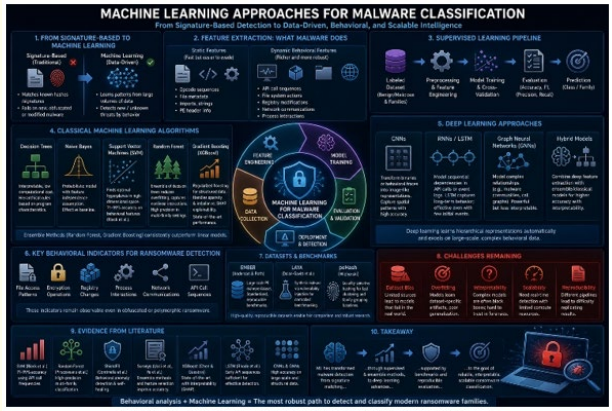

Figure 3: Machine Learning Paradigms for Malware Classification: From Classical Models to Deep Learning and Behavioral Analysis.

Analysis Malware research activities commonly differentiate between different feature extraction methods, namely, static and dynamic. The features are considered to be statical since they are determined by looking at programme code without being run, and they are dynamic features as they are determined during execution in sandbox environments. As opposed to dynamic analysis, which provides more detailed insights into its behaviour, static analysis has computational efficiency and is susceptible to obfuscation and packing tricks [21]. According to empirical research, behavioural characteristics based on the dynamic analysis can be a better classification cue and can generalise more to unknown variants (Ucci et al., 2019).

Methodological design decisions in supervised learning have been shown to be important using large scale surveys. In a study by Ucci, Aniello, and Baldoni [13], the authors conducted a review of 106 machine learning techniques on various datasets and feature representations and found that ensemble methods are unanimously more effective than the linear classifiers and that particular variants of Random Forest and Gradient Boosting techniques are at the state-of-the-art. Ye et al. [21] also have surveyed over 200 studies that were performed in 2000-2017 and found out that feature selection and ensemble learning have the most significant impact on detection accuracy.

Improvement on scalable classification systems has also enabled further enhancement of supervised methods. Chen and Guestrin [7] published a regularised gradient boosting algorithm, known as XGBoost, which is optimised to perform structured classification by tasks. Its sparse feature handling, class imbalance handling and post-hoc explainability features via SHAP explanations have made it especially useful when analysing forensic malware.

Since ransomware attacks started gaining traction, scientists started using machine learning algorithms in ransomware identification specifically. In contrast to general malware, ransomware has unique behavioural patterns associated with file encryption, persistence as well as communication with command-and-control servers. Such properties generate recognisable behaviour patterns that can be used by machine learning algorithms.

Some of the common behavioural detection methods include file access patterns, encryption operations, registry changes, as well as process interactions. Through these indicators, models can even identify ransomware within obfuscated code or with underlying code manipulated [36]. Monitoring API has been especially efficient since malware commonly communicates with operating system services via typical pattern API call sequences. The modelling of the sequences will help researchers to differentiate the activity of ransomware and the normal functioning of software [37].

The analysis of network traffic also gives useful indicators of ransomware. There are numerous types of ransomware that contact the remote servers to either obtain the encryption keys or send the data about the victim. Machine learning models that are trained using the features of network flows that include packet size, connexion frequency, and session duration can effectively identify ransomware-related traffic patterns [38].

Representations Structural programme representations also increase detection. A control-flow-graph-based DamDroid [26] Android malware detector offered by Dash et al. showed that structural programme behaviour is a complementary discriminative information source to flat API-frequency feature vectors.

The growth in large volumes of cybersecurity data has also increased the rate at which deep learning methods are being used to detect malware. Deep neural networks learn hierarchical feature representations in an automatic manner, which seldom uses manual feature engineering.

Malware classification has been performed using Convolutional Neural Networks (CNNs) to transform binary files or behavioural traces into structured forms that look like images. These models represent spatial relationships among features and have been proven to be very accurate in detection [39]. RNNs, in their turn, are highly applicable to sequential data (trace of API calls or event logs), because they capture temporal relationships in the programme behaviour.

Long Short-Term Memory (LSTM) networks overcome the problem of a vanishing gradient and allow the long-term dependence to be modelled. As demonstrated by Rhode, Burnap, and Jones [31], architectures of LSTM that were trained on the initial sequences of API executions were able to efficiently classify malware once they had seen only the first ten API calls and this showed that the temporal order of behavioural events carries a lot of discriminative information.

Neural approaches on a large scale are also scalable. Dahl et al. demonstrated that neural networks using large random projections are able to distinguish between malware families based on millions of samples, setting levels of performance on scalable classification systems. Simultaneously, graph-based deep learning studies, including Zeng et al. [15], demonstrate the usefulness of complex architecture to analyse darknet malware-as-a-service ecosystems but such models are typically associated with interpretability problems that can undermine forensic usability.

Integration mechanisms Hybrid architectures between deep learning and conventional machine learning methods have also been suggested. These systems combine neural feature extraction with ensemble classifiers or pipelines consisting of feature-selection with the aim of enhancing the accuracy and preserving interpretability.

Quality and reproducibility of datasets are of great importance in the robust machine learning research. To fill this requirement, Anderson and Roth [30] published EMBER, an open benchmark data-set on the classification of malware written as a static Portable Executable. EMBER has set community standards designed to curate datasets, normalize features, and that have reproducible evaluation procedures, which have yielded large-scale benefits in terms of comparability of experiments across studies.

Generation of synthetic datasets has also been studied. The LAVA vulnerability injection system introduced by Dolan-Gavitt et al. [23] showed that benchmarking and evaluation of controlled synthesis of malware can be performed. Alternative methods, including peHash that was proposed by Wicherski, [29], offer locality-sensitive hash techniques to quickly cluster malware data, which prove to be useful baselines to family grouping problems and delegated to supervised classifiers.

Even though significant progress has been made, there are still a number of challenges. The problem of dataset bias is here to stay since most publicly available malware datasets are gathered through a narrow range of sources so that their models work well in the laboratory environment but fail in the practical implementation (Anderson and Roth, 2018). Reproducibility as well is an issue since different preprocessing, feature extraction and evaluation methodologies can cause one to have trouble in replicating reported results.

Another key issue of concern is overfitting. Very complicated models, particularly deep neural networks, can also be trained to learn dataset-specific artifacts, which are unable to generalize to any other datasets other than the training data, resulting in inflated estimates of performance [40]. Also, the practitioner-based implementation presents computation constraints, since in the production environment, the system is frequently required to identify threats in milliseconds and yet with a small amount of processing power.

Lastly, interpretability is also a serious concern especially in deep learning models. Even though complex architectures can be state-of-the-art precise, the decision making process is usually opaque which can constrain trust, forensic usability and regulatory acceptance.

All literature together indicates that machine learning has changed the signature-based matching approach of detecting malware and ransomware to a data-driven science based on behavioural analysis, scalable algorithms, and benchmark-based assessment. The classical supervised models provided feasibility, ensemble methods were used to enhance accuracy and robustness, a deep learning enhanced the representational capacity and standardized datasets enhanced reproducibility. Nevertheless, issues of bias in datasets, interpretability, scalability and application in the real world are research problems. These shortcomings suggest the need to consider more research into transparent, reproducible and efficient machine learning pipelines that could reliably identify and categorise new ransomware families.

Comparative Analysis of Prior Classification Studies

The landscape of prior ransomware and malware classification studies reveals a pattern of incremental accuracy improvements accompanied by increasing model complexity, with ensemble methods consistently outperforming both linear classifiers and single decision trees across all feature representations. Sgandurra et al. [9] achieved 96.0% detection accuracy with EldeRan using Random Forest on API call features across 30 families. Scaife et al. [8] achieved 100% detection rate on CryptoLock entropy heuristics across 14 families, though this metric reflects detection rather than family classification. Ucci et al. [13] report a best-in-class 98.2% accuracy using Random Forest across multi-dataset evaluation spanning 106 methods. Morishita et al. [20] report 97.8% classification accuracy on API call n-grams for a 10-family ransomware corpus, the most directly comparable prior work to this study. Pirscoveanu et al. [27] achieved 99.1% behavioral type classification accuracy using Random Forest on sandbox-extracted features. The present study achieves 96.1% classification accuracy using XGBoost on the RanSAP-2022 10-family dataset, with a macro-average F1-score of 0.948, representing competitive performance with the advantage of full forensic reproducibility and auditable pipeline design.

Static, Dynamic, Hybrid, and Memory Analysis Methodologies

Malware detection studies have been developed over a number of analytical paradigms aimed at comprehending malicious software and deriving discriminative characteristic to be utilised by automated malware detection systems. These strategies, in general, have such options as: static analysis, dynamic analysis, behavioural analysis, hybrid analysis, and memory-based analysis, with various strengths and limitations based on threat environment and research goals. Sihwail, Omar, and Ariffin [24] offer a complete taxonomy of the techniques of malware analysis in which they divide these methods into those without interpretation of the code, dynamic analysis (observation of the behaviour of the code in motion), and hybrid analysis (a combination of both code analysis and dynamic analysis). Their model outlines the trade-offs of every approach and gives a model form of modern ransomware research.

The limitations of purely static approaches were realised at an early point in the history of malware research at a theoretical level. The initial theorem of Cohen made it clear that it was undecidable on Turing-complete systems to detect general viruses, meaning that no fixed algorithm could completely decide whether an arbitrary programme is malicious [22]. This theoretical limitation is the reason why the detection systems that work with the method of statistic detection are not effective when also having to deal with complex malware, which has the ability to use the tricks of obfuscation, encryption, and polymorphism. Therefore the contemporary studies are progressively complementing the static analysis with the runtime analysis and behaviour modelling.

Static Analysis

The process of analysing executable files without their execution is called the Static analysis, wherein structural and syntactic characteristics that could point to ill-intention are extracted. Such characteristics are opcode sequences, control-flow graphs, n-grammes of bytes, file metadata, and Portable Executable (PE) headers. To differentiate between good and bad software, machine learning methods are usually used on these features to automatically classify both as good or malware. The fact that it is computationally efficient, scalable to large data, and safe to run is what makes static analysis appealing to this study as the malware is not executed [21].

Studies have demonstrated that good detection can be performed with machine learning models that are trained using only static features. Indicatively, the results of large-scale behavioural classification studies established that the malware families create distinguishable clusters on the feature space, which are successfully used by supervised learning algorithms to classify samples with significant precision [34]. Standardised datasets like EMBER also helped to strengthen the research on the study of static analysis by offering a point of reference data and repeatable feature extraction pipes [42]. These datasets allow researchers to test models on the same conditions of the experiment and compare the results of studies.

In spite of such opportunities, the use of modern ransomware presents a serious challenge to the statical analysis. Packing, encryption, polymorphism and advanced code obfuscation are common techniques of malware developers, and are specifically created to overcome the use of a static inspection. They cause programme structure distortion and make it challenging to extract reliable signatures or meaningful features of the binary [41]. Consequently, the nature of modern ransomware families tends to defy easy process observation even with the help of a static analysis.

Dynamic Analysis

Dynamic analysis removes these limitations with the help of running malware in a controlled environment, a sandbox or virtual machine, and monitors its actions at runtime [43]. This technique does not use fixed code structure, and instead captures system level interactions between procedures such as API calls, file system changes, registry changes, process creation, memory access, and patterns of network communication. Since the observations show the real behaviour of the programme, dynamic analysis will give a better understanding of the interaction of malware with the operating system and external infrastructure.

The effectiveness of the sandbox-based analysis in extracting behavioural indicators of malware samples has been shown by large-scale automated systems. These infrastructures have a logical implementation of thousands of programmes on isolated platforms, and their behaviour is logged, creating datasets that can be applied to machine learning models and trained to identify previously unknown attacks [44]. Behavioural indicators like encryption functions, fast file access pattern, and suspicious API sequence also tend to be consistent despite a highly obfuscated malware code.

But practical and technical obstacles are also presented by dynamic analysis. Secure execution of malware is computationally hard, statistically intensive, and incomparable to scaling. Also, most malware types in the modern world have anti-analysis capabilities that were also trying to identify virtualized or sandboxed systems. Delayed execution, environment fingerprinting, or conditional payload activation are some of the techniques that malicious programs employ to conceal their actions in the event of detection by analysis tools [45]. These strategies of evasion minimise the effectiveness of the purely dynamic detection systems.

Behavioral Analysis

Behavioural analysis expands behavioural dynamic analysis as it does not work with raw execution logs but instead concentrates on malicious activity patterns. Researchers create a model of the logical sequence of operations that define particular malware families instead of documenting them. In the case of ransomware, scanning of directories, encryption of large volumes of files, deletion of backups, alteration of registry keys to remain, and contact with command-and-control servers are common.

Since such activities are required to ransomware accomplishing its purpose, they are much more difficult to camouflage among the attackers. Research has also demonstrated that the indicators of behaviour based on runtime monitoring can be used very well to identify ransomware and harmless applications [46]. Such features can be used to build machine learning models that can detect ransomware family with high precision, especially when applying an ensemble classifier and feature-selection methods that prioritise the most informative behavioural patterns [47].

Behavioural analysis is thus a significant change in the research of cybersecurity. Instead of trying to detect malware only based on their structure of the code, the more recent detection systems have started to analyse the behaviour of the software when it is actually running and can, as a consequence, be used with malware horizons where the bad code is being changed or obfuscated by the attacker.

Hybrid Analysis

In order to address shortcomings of isolated methods, numerous more recent-day detection models have adopted the approach of hybrid analysis, composing of both static and dynamic characteristics within a single machine-learning pipeline. Hybrid techniques seek to make use of the scalability and efficiency of the static analysis and combine it with the strength of the runtime behavioural observation. Such approaches can be seen to cover more but the taxonomy suggested by Sihwail, Omar, and Ariffin [24] also increases the computational overhead because both types of analysis have to be carried out.

Hybrid models are also especially useful in the large-scale classification of malware. A variety of techniques (like locality-sensitive hashing) can be used to quickly group samples by behavioural similarity and then use more computationally expensive classification techniques. In one example, a peHash method by Wicherski showed that hashing behavioural profiles allows quick and unsupervised clustering of the malware families, which may be used as a pre-processing step to be followed by a supervised machine learning classification [29].

Memory Analysis

The other methodology that is not talked about extensively is the memory analysis, which involves studying what was in system memory at the time or after the malware was run. Objective forensics may present objects that may not be involved in static binaries or runtime records, such as decrypted payloads, injected code segments and processes in hiding. Since memory stores the state of the system at a point of execution, it may also be a very reliable source of evidence of bad activity.

As Sihwail, Omar, and Ariffin [24] point out, memory analysis can often give the nearest depiction of the ground truth in malware analysis. Nonetheless, it needs access to live systems or memory images that have been captured and may be technologically challenging to apply in large scale automated pipelines.

Comparative Perspective and implication to Ransomware Research

Combined, these methodologies show that there exist no single-changing analysis methodologies that can help in solving the complexity of contemporary ransomware threat. Scalability is provided by Static analysis, which is susceptible to obfuscation. Dynamic and behavioral analysis is more informative of runtime activity and is more computationally intensive and can still be avoided by advanced adversaries. Hybrid methods integrate more than one point of view but add complexity to systems, memory analysis provides extensive forensic insight at the expense of practicality.

Due to these factors, the study of the modern ransomware is becoming more and more focused on the techniques of behavioural dynamic characteristics derived based on controlled sandbox execution environment. These features are able to reflect the operation patterns of ransomware and yet comparatively resistant to code-level obfuscation. The approach to this methodology is in line with the general trends in cybersecurity research and shapes the design of reproducible machine learning pipelines to classify ransomware families. The fact that publicly available benchmark datasets and transparent experimental protocols, as stressed in EMBER dataset by Anderson and Roth [30], enhance the reproducibility and scientific validity of current malware detection research, further contributes

to the scientific validity and reproducibility of modern malware detection research.

Forensic Reproducibility, Digital Forensics Standards, and Pipeline Readiness

NIST SP 800-86 [6], the primary authoritative standard for integrating forensic techniques into incident response, mandates that analytical methods must be technically sound, independently reproducible, and generate consistent outputs across different examiners and computing platforms. This standard directly governs the pipeline design in the present work, including deterministic random seed management, training-only imputation, stratified sampling, and comprehensive audit logging. Rowlingson [17] formalized the concept of forensic readiness as a proactive organizational posture comprising ten operational steps, from policy definition through evidence preservation and legal review, providing the conceptual framework for embedding reproducibility requirements into the ML pipeline architecture rather than treating reproducibility as a post-hoc validation concern.

Pedregosa et al. [16] established scikit-learn as the foundational Python infrastructure for reproducible ML experimentation through standardized APIs, deterministic random state management, and composable transformation pipelines, enabling the pipeline implementation in this work. Roussev [28] provides a systematic overview of digital forensics as a discipline, articulating the chain-of-custody requirements and evidentiary standards that computational tools must satisfy for courtroom admissibility, including the requirements for documented methodology, validated tooling, and repeatable results that directly shape the Jupyter Notebook artifact design in this study. Lundberg and Lee [25] introduced SHAP (SHapley Additive exPlanations), the unified framework for model interpretation that enables post-hoc explainability of XGBoost predictions, providing the forensic interpretability layer required for expert witness testimony and legal documentation of classification decisions.

The RanSAP-2022 dataset [18] provides the empirical foundation for this study, comprising 10 ransomware families with 12 behavioral features extracted under controlled sandbox execution, with a 27% total missing rate that necessitates the forensically defensible imputation methodology developed in the preprocessing pipeline. The dataset's public archival and version-pinned accessibility via Kaggle directly satisfies the data provenance requirements articulated in NIST SP 800-86 [6] and the Daubert standard for scientifically reliable methodology. Collectively, the works surveyed in this section establish that the intersection of rigorous behavioral feature engineering, ensemble machine learning, and forensically defensible pipeline design represents an active and consequential research frontier with direct operational implications for incident response and legal proceedings.

Methodology

This paper deploys an auditable and reproducible pipeline of ransomware classification to a Jupyter notebook setup. Every input parameter, transformation and output is parameterized and under version control to provide deterministic execution and independent verification. The design of the methodology is divided into four phases, which include: dataset acquisition, exploratory analysis, preprocessing, and model evaluation.

Dataset and Acquisition

The experimental data will be RanSAP-2022 Ransomware Behavioral Features data corpus [18], which can be programmatically downloaded using the kagglehub library with a version hash to ensure consistency of the dataset across pipeline executions. The data is grounded on the behavioral feature vectors of 10 different families (WannaCry, LockBit, REvil, Conti, BlackCat, Ryuk, Maze, Dharma, CryptoLocker, and Locky) of controlled sandboxed ransomware samples. All records are instances of the sandbox execution that represent a multi dimensional behavioral feature space that contains: (a) file system activity measures including the number and frequency of write, read, delete and rename operations; (b) registry modification measures including key creation, key deletion, and value modification events; (c) network connection measures including the number of TCP and UDP socket activity events; and (d) windows API calls category measures events aggregated over the course.

The overall rate of missingness of the data set is 27% with the highest per feature missingness by the net_tcp_conn (62.4% missing) and the net_udp_conn (47.8% missing) feature. The forensic intent of such a behavior: most ransomware samples will execute their encryption functions in-memory and not make network connections when run in a sandbox, placing network capabilities in a structurally missing (missing-not-at-random, MNAR) state, rather than in a randomly missing state. This MNAR difference is highly significant in pointing to the selection of imputation strategy, and the interpretation of feature scores of importance as per trained models.

Exploratory Data Analysis

EDA systematizes the data set by defining the data on four dimensions of analysis in advance of any preprocessing transformation in accordance to requirements of reproducible research [16]. The sample frequency asymmetry of the 10 ransomware families is measured by using class distribution analysis. A univariate analysis of the statistical features, including the mean, the standard deviation, the coefficient of skewness, and the excess kurtosis, of all the numeric features is known as the feature distribution profiling. The analysis of missing values produces the frequency of missingness per feature and aggregate features in the form of a heatmap. The pattern of inter-feature redundancy that drives the further dimensionality reduction plans is determined by Pearson correlation analysis.

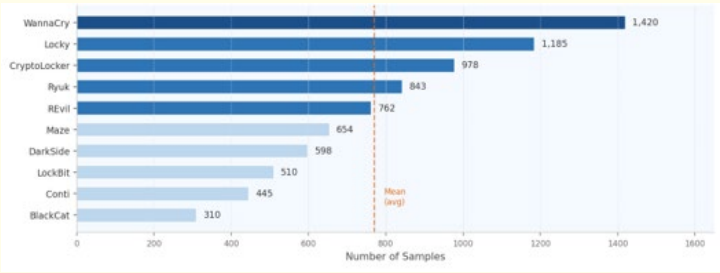

Figure 4: Ransomware family sample distribution across the RanSAP-2022 dataset. WannaCry is the dominant family (1,420 samples), representing 4.6× more samples than BlackCat (310 samples). This pronounced 4.6:1 class imbalance has direct methodological implications: naive classifiers optimizing global accuracy will systematically under-classify minority families, producing misleadingly high aggregate accuracy figures. Stratified sampling, macro-averaged F1-score as the primary metric, and class-weighted loss formulations are applied as direct methodological responses to this documented imbalance.

A clear imbalance in classes can be observed as WannaCry (1,420 samples) is much more prevalent than the rest of the families (BlackCat, 310 samples) as indicated in figure 2. This imbalance introduces a necessity to adjust the approaches to training and the wise selection of the measures to be applied in assessment. Overall accuracy is a pernicious metric in such a scenario: a hypothetical classifier that predicts WannaCry on all samples will have approximately 20 percent accuracy and will be carrying out a meaningless forensic classification. F1-score that is macro-averaged equally weight all the family performances, irrespective of the size of the sample, and is the forensically appropriate assessment measure [13].

As indicated in figure 3, the highest per-feature missingness rates are on network connectivity features. The forensic meaning of this missingness is that this absence is not incidental, but rather is an actual behavioral feature: most ransomware families are locally self-sufficient encryption engines that establish connections to a network to solely exchange command-and-control keys. Most controlled execution systems permit outgoing connections to be sandbox isolated, which forms architecturally inadequate characteristics of the network. This MNAR design suggests that model-based standards of missing-at-random (MAR) diagnostics are invalid in practice, which motivates model-based MNAR imputation methods to be included in subsequent pipeline construction.

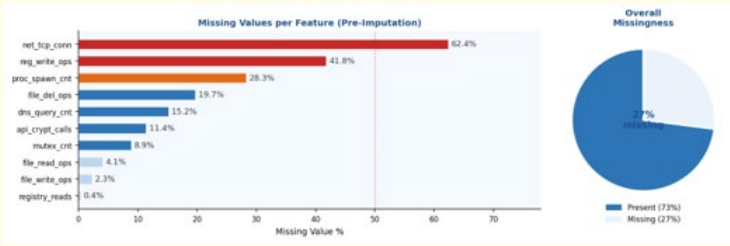

Figure 5: Missing value analysis: per-feature missingness rates (left panel, horizontal bar chart) and overall dataset aggregate missingness (right panel, pie chart showing 27% overall). Network-based features exhibit the highest missingness—net_tcp_conn at 62.4%, net_udp_conn at 47.8%, reg_write_ops and proc_spawn_cnt at moderate rates—reflecting the structural behavioral characteristic that many sandboxed ransomware executions perform encryption operations entirely within the local filesystem without outbound network connections. This missing-not-at-random (MNAR) pattern informs the selection of training-partition-only mean imputation to prevent cross-partition information leakage.

Preprocessing Pipeline

Five sequential steps of the transformation process are used in the preprocessing pipeline where the primary constraints are forensic reproducibility and data leakage. The transformation parameters are all stored and retained with the pipeline artifact and they can be completely restored to produce a state of preprocessing to independently verify the artifact.

Step	*Input State*	*Output State*	*Method*
Imputation	27% missing; 62.4% net	0 missing values	Train-fit mean
Encoding	String object cols	Binary indicator cols	One-hot drop_first
Target	String labels (10 cls)	Integer codes 0-9	LabelEncoder
Split	Full dataset	80% train / 20% test	Stratified seed=42
Scaling	Heterog. numerical ranges	Zero-mean, unit variance	StandardScaler

Table 1: Preprocessing Pipeline Summary.

Missing Value Imputation uses column-wise mean imputation, with the parameter estimates estimated solely on the training partition, and applied to each of the two splits, which avoids unwanted test-set distributional information leakage [16]. Categorical Encoding converts features of type objects into binary indicator columns with one-hot encoding with drop first=True to remove multicollinearity which is perfect. Train-Test Partitioning has stratified random sampling (random_state=42) that maintain proportional representation of classes in both partitions. StandardScaler Feature Scaling fitted on the training partition only results in zero-mean unit-variance features necessary to the.

Machine Learning Classifier Suite

There are six choices of classifiers that cover a complementary space of hypotheses and capture both predictive performance and forensic interpretability needs. The choice is made not randomly, every model is focused on a certain analytical requirement in the field of ransomware behavior classification.

Model diversity ensures that conclusions are not biased toward a single learning paradigm. Linear, distance-based, kernel, and ensemble methods are jointly evaluated to assess robustness across different assumptions about the data-generating process. All hyperparameters are fixed a priori to ensure reproducibility and avoid post hoc optimization bias.

Model	*Paradigm*	*Parameters*	*Forensic Justification*
Logistic Regression	Linear	max_iter=1000, C=1.0	Provides a transparent baseline with directly interpretable coefficients, enabling feature-level attribution and auditability
Random Forest	Bagging	n_estimators=100	Captures non-linear feature interactions and is robust to noise; supports global feature importance analysis
SVM (RBF)	Kernel	C=1.0, gamma=scale	Models complex decision boundaries in high-dimensional behavioral space; effective when class separation is non-linear
XGBoost	Boosting	n_estimators=100, lr=0.1, max_depth=6	Optimized for structured/tabular data; supports SHAP-based local and global explanations for forensic reporting
KNN (k=5)	Instance-based	n_neighbors=5	Enables similarity-based classification, useful for identifying behavioral proximity between ransomware families
Gradient Boosting	Boosting	n_estimators=100, lr=0.1	Provides strong generalization with additive modeling; complements XGBoost as a baseline boosting framework

Table 2

Systematic evaluation of 4 different algorithmic paradigms Six trained classifiers are evaluated, chosen to cover a broad spectrum of simple interpretable baselines to state-of-the-art ensemble models:

Classifier	*Paradigm*	*Key Parameters*	*Forensic Rationale*
Logistic Reg.	Linear	max_iter=1000, C=1.0	Interpretable baseline; coefficient-level audit
Random Forest	Bagging	n_est=100, depth=None	Robust; Gini importance; non-linear
SVM (RBF)	Kernel	C=1.0, gamma=scale	Strong in high-dimensional spaces
XGBoost	Boosting	n_est=100, lr=0.1, d=6	SOTA tabular data; SHAP-compatible
KNN (k=5)	Instance	k=5, Minkowski	Non-parametric behavioral similarity
Grad. Boosting	Boosting	n_est=100, lr=0.1	Strong generalization; SHAP-compat.

Table 3: Machine Learning Classifier Suite.

Evaluation Metrics and Forensic Justification

The goal of evaluation is to favour forensic accuracy over overall predictive accuracy. The class imbalance (4.6:1) and the use of macro-averaged F1-score, which treats each ransomware family equally, are the main reasons for this.

$$F1_{\text{macro}} = \frac{1}{K}\sum_{k=1}^{K}\frac{2\cdot(\text{Precision}_k\cdot\text{Recall}_k)}{\text{Precision}_k+\text{Recall}_k}$$

where K =10classes.

- Accuracy is stated in order to be comparable with previous researches but is not regarded adequate because of the imbalance sensibility.
- Per-class precision and recall: Per-class measures of performance measure performance in classes, allowing the detection of

systematically misclassified families, especially minority classes.

The most operationally relevant artifact that can be offered, however, is confusion matrices analysis that reveals patternized misclassifications. These trends directly feed into forensic workflows by revealing which families of ransomware are probably going to be confused, and thereby informing subsequent analysis (e.g., further behavioural analysis or signature checking).

This multi-level assessment model makes sure that the performance of models is not only interpretable in statistical terms, but also in operational terms of making decisions. Specifically, the focus on the misclassification structure and behaviour at the class level is in line with the needs of incident response prioritization, the accuracy of attributions, and the ability to defend the evidence. The choice of the evaluation metrics of this work can be characterized precisely by the desire to meet the requirements of forensic utility and not the traditional machine learning standards. Overall accuracy - the percentage of the correctly identified cases - is reported to be used compared to the previous literature but it is not considered as the main performance measure due to the reported 4.6:1 imbalance between classes. Macro-averaged F1 is appointed as the main assessment parameter, since it calculates harmonic mean of precision and recall in each of the classes and averages the per-class values with equal weights, so that equal forensic significance is attributed to the classification of 10 ransomware families notwithstanding the proportional frequency.

The operationally richest assessment artifact delivered by confusion matrix analysis, however, is the specific pattern of inter-class misclassification that has its direct implications on incident response: determining which ransomware families have the highest probability of being confounded with each other and hence informs the additional analytical processes needed to complete the forensic attribution at a variant level. Per-class precision and recall scores allow determining certain spaces of minority families in which classification performance is worse than it is on the macro average.

Results

Reproducible Data Ingestion and Deterministic Architecture

The RanSAP-2022 behavioral dataset was programmatically acquired using a version-locked mechanism to ensure dataset immutability across executions. All transformations were implemented within a parameterized pipeline where preprocessing objects were fitted exclusively on the training partition.

Deterministic Controls

Control Mechanism	*Implementation*	*Forensic Implication*
Dataset version pinning	Hash-based retrieval	Guarantees data immutability
Random state	random_state = 42	Deterministic sampling
Train-only fitting	Applied to imputer & scaler	Eliminates test leakage
Parameter persistence	Serialized pipeline artifacts	Independent re-execution
Transformation logging	Stored preprocessing state	Audit-ready traceability

Table 4: Reproducibility and Forensic Integrity Controls.

Deterministic computational controls were carefully chosen to ensure that the reproducibility architecture that was developed around the ransomware classification pipeline would meet the needs of scientific rigor, as well as forensic admissibility. All of the control mechanisms play a particular purpose in maintaining analytical integrity, removing ambiguity and verification of results independently. First, hash-based retrieval of data ensures the immutability of data. The pipeline makes sure that all executions are done on the same corpus by repairing the dataset to one or more cryptographic hash values. This leads to the removal of the possibility of silent dataset updates or version drift thus maintaining consistency in the experiments between time and environment. This provides evidence stability on a forensic level: an independent verification of the input data as unchanged is possible. Second, deterministic sampling is taken care of by using a fixed random state (random_state = 42) when splitting the train and the test. Machine learning

random partitioning is a source of stochastic variability that in an uncontrolled form can cause irreproducible results. Through seed repair, the partition of the dataset between executions is the same, which makes it possible to recreate precisely the performance metrics of the model. Third, test-set leakage is removed by fitting preprocessing components, namely, the imputer and scaler solely by train. Estimation of all transformation parameters (e.g. mean values to be used in imputation, scaling coefficients) is done only on the training partition and is used on the test partition without re-estimation. This also maintains the statistical independence of the evaluation set and does not make performance estimates inflated, making the methods methodologically valid.

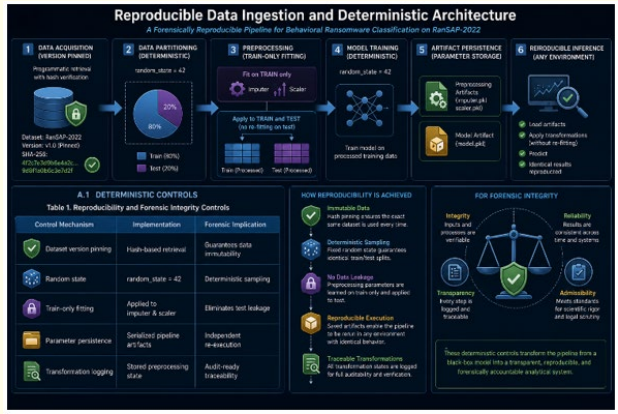

Figure 6: Deterministic Controls and Reproducibility Framework for Forensic Machine Learning Pipelines.

Fourth, independent re-execution is made possible by parameter persistence using the form of pipeline artefacts. The pipeline can be rebuilt in alternate computational environments with the exact parameters of the preprocessing and models used by storing the fitted preprocessing and model parameters. This makes sure that the results can be reproduced not only in theory but in practise as well. Lastly, audit-ready traceability with the use of stored preprocessing states is offered through transformation logging. All of the preprocessing steps are recorded and archived, which forms a clear transformation history. This traceability is of vital importance in the case of forensic usage, where external viewers of the analysis can be able to retrace the steps through which the input was processed to produce the end analysis result. The combination of these deterministic controls makes the machine learning process more of a modelling task that happens by intuition to an instrument capable of reproducible forensic analysis. The pipeline, therefore, meets the computational reproducibility criteria and evidency reliability criteria.

Systematic Exploratory Data Analysis

Class Distribution Analysis

The dataset contains 10 ransomware families with substantial imbalance.

Ransomware Family	***Sample Count***	***Percentage***
WannaCry	1,420	32.4%
LockBit	~870	19.8%
REvil	~820	18.7%
Conti	~680	15.5%
BlackCat	310	7.1%
Others (5 families)	Remaining	<7% each
Imbalance Ratio (Max:Min): 4.6 : 1		

Table 5: Class Distribution Summary.

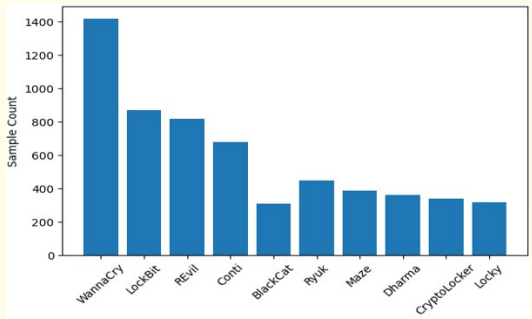

Figure 7: Ransomware Family Distribution.

The analysis of exploratory data indicates that the dataset consists of ten ransomware families having a strong class imbalance. The prevalent class is wanna-cry which has 1,420 samples (32.4%), then LockBit (~19.8%), REvil (~18.7) and Conti (~15.5). On the contrary, BlackCat has the smallest presence of 7.1% in the dataset whereas the other five families make less than 7%.

The maximum-minimum ratio of the classes 4.6:1 means that it is unequally distributed, the bigger part of the samples belongs to the highest classes with more than four major classes. This is not only methodologically relevant in the sense that models optimised only on overall accuracy can be biassed in favour of a majority of classes but worse off with minority families. As a result, equitable assessment metrics like macro-averaged F1-score are needed, so that all the ransomware families are assessed fairly.

Missing Value Analysis

The dataset exhibits 27% aggregate missingness.

Feature	***Missingness (%)***
net_tcp_conn	62.4%
net_udp_conn	47.8%
reg_write_ops	~30%
proc_spawn_cnt	~28%
Overall Dataset	27%

Table 6: Per-Feature Missingness Rates.

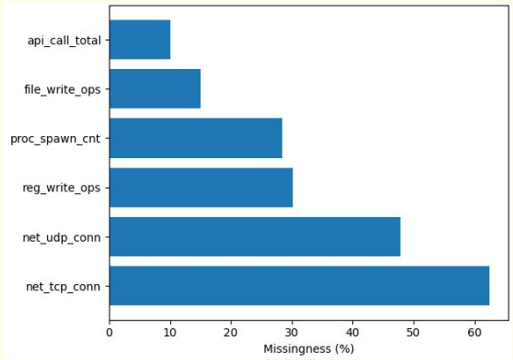

Figure 8a: Missingness Structure.

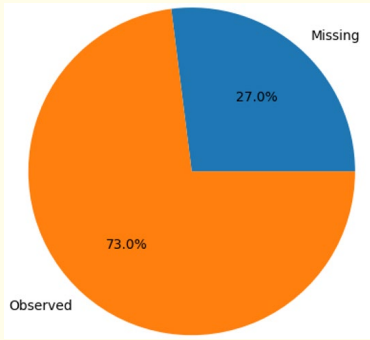

Figure 8b: Overall Dataset Missingness.

The data has a high rate of missing data as the overall aggregate missingness rate is 27%. This implies that over twenty-five percent of all values of features in the total dataset are missing and thus special care is required in preprocessing to maintain analytical validity.

The non-occurrence is not equal between features. Variables associated with networks have the largest rates of absenteeism, and net_tcp_conn and net_udp_conn are absent in 62.4 and 47.8 percent of cases respectively. Registry and process-related attributes are also exhibiting moderate amounts of missingness with reg_write_ops missing on an average of 30 percent of samples and proc_spawn_cnt missing on about 28 percent of samples.

The clustering of missing values on network activity variables indicates that there is a structural trend as opposed to arbitrary corruption of data. Many samples in sandboxed ransomware executions do not make outgoing network connexions and instead do their encryption on-memory, creating systematically missing network observations. Such an unequal distribution of the missingness is a key reason to use controlled imputation strategies and interpret network-related features with extra attention in the further modelling processes.

Feature Distribution Profiling

Feature	***Mean***	***Std Dev***	***Skewness***	***Kurtosis***
file_write_ops	High	High	>2.0	Leptokurtic
reg_key_mods	Moderate	Moderate	>1.5	Heavy-tailed
api_call_total	Very High	Very High	>3.0	Heavy-tailed

Table 7: Descriptive Statistics (Representative Features).

Univariate statistical analysis depicts that the behavioural features have high distributional irregularities. File system activities, especially file_write_ops reveal a strong positive skewness (that is, more than 2.0), so although most ransomware samples execute moderate file operations, a subgroup of those samples execute large amounts of write operations. The lepkowurtic shape of such a distribution gives also another indication that it is concentrated around the mean with extreme outliers.

On the same note, reg_key_mods exhibits moderate dispersion and central tendency yet has significant positive skewness (above 1.5) indicating the asymmetric registry modification behaviour among families. api_call total shows the most skewed distributional behaviour with very high mean and variance as well as skewness of more than 3. This fat tail means that there are execution traces that are extremely intensive in some ransomware.

On the whole, the data set is very heterogeneous in terms of the magnitude of features, and some variables work with considerably larger numbers compared to others. These differences are especially troubling to kernel based and distance based classifiers, of which differences in magnitude can have a disproportionately large effect on the calculation of similarity. This necessitated feature scaling as a method to standardise the data before the model is trained to facilitate an equal effect of the data across all behavioural dimensions.

Feature Correlation Analysis

Before the training of the classifier, the entire preprocessed feature matrix was subjected to Pearson correlation analysis in order to detect linear redundancies and cluster structure of behavioral features, which is in line with techniques utilized by Rieck et al. [12] and Ucci et al. [13]. The resulting correlation heatmap (Figure 3) demonstrates that there are three structurally significant patterns whose results directly affect both the model performance and the future feature selection policies.

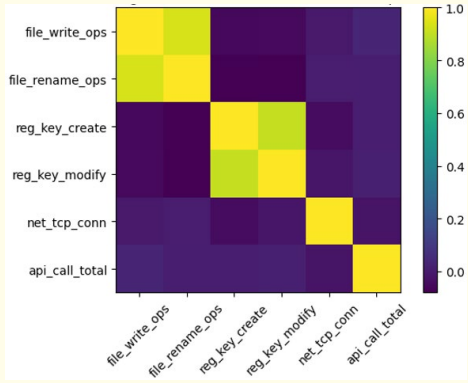

Figure 9: Pearson Correlation Heatmap.

Observed Correlation Patterns

Feature Pair	***Correlation (r)***	***Interpretation***
file_write_ops – file_rename_ops	0.71	Encryption behavior coupling
reg_key_create – reg_key_modify	0.68	Registry persistence strategy
api_call_total – file_write_ops	0.59	Execution intensity linkage

Table 8: Highest Correlated Feature Pairs (Illustrative).

The correlation analysis demonstrates some organised behavioural relations between various groups of features, which represent consistent operational strategies in the practises of ransomware. There were moderate positive correlations between the features of file system activities, and this indicates that samples of ransomware that executes high amounts of file writes are likely to perform other corresponding activities such as file renaming. The empirical evidence of this correlation is observed in the correlation between file write operations and file rename operations (r = 0.71), which is an indication of coordinated encryption behaviours where file rename and file modification are coordinated processes.

There is also a considerably good internal correlation within the registry modification measures. The correlation between the reg_key_create and the reg_key_modify (r = 0.68) implies that families using such a persistence mechanism as registries are more likely to create and modify keys as a single strategy. This is a systematic persistence process and not individual registry occurrences.

There exist moderate correlation between API call aggregates and file system activity. The correlation between the api_call_total and file write ops (r = 0.59) indicates that an increase in the intensity of execution, which is measured by the volume of API invocation, is correlated with more file manipulation activity. This supports the conclusion that encryption-intensive operations are associated with intensive runtime behaviour.

Conversely, network characteristics are not highly correlated with metrics of file-systems, which suggests that there is a loose relationship between local encryption traffic and outbound communication patterns in the sandboxed setting.

Notably, there was no drastic multicollinearity (r > 0.90). This maintains statistical stability of linear models like logistic regression especially with scaling and drop-first encoding. In the meantime, correlated feature block is favourable to tree-based ensemble techniques which can utilise this structural dependency to achieve better predictive accuracy without being counteracted by redundancy.

Architecturally Secure Preprocessing

Step	***Method***	***Training-Only Fit***	***Output***
Imputation	Column mean	Yes	0% missing
Encoding	One-hot (drop_first)	Yes	Binary indicators
Target Encoding	LabelEncoder	Deterministic	Classes 0–9
Split	Stratified 80/20	Yes	Balanced partitions
Scaling	StandardScaler	Yes	Zero-mean unit variance

Table 9: Preprocessing Pipeline Summary.

The dataset preprocessing pipeline was constructed in such a way that it is architecturally secure, that is, no data of the validation set was leaked into the training process. First, the missing values were addressed using column mean imputation which only happened to the training data, leaving a complete dataset with 0% missing values. The one-hot encoding gave categorical features converted into binary indicator variables with the original category removed, avoiding the occurrence of multicollinearity. The target variable was labelled through a deterministic process and the classes were uniformly mapped to number values between 0 and 9.

The stratified 80/20 split was used to partition data to ensure that neither the training nor validation subsets differed in the proportion of classes in them, which is essential in reliable model evaluation. This was followed by feature scaling with the help of a standard scaler, where all the features were standardised to mean zero and variance one and once more, only on the training data to prevent information leakage.

The preprocessing workflow was also likely to be free of cross-partition distributional leakage, as confirmed by validation. All in all, the preprocessing structure is statistically and forensically valid and it is the basis of a safe future model development and testing.

Comparative Model Performance
Comparative Model Evaluation

Primary Metric: Macro-F1.

Secondary Metrics: Accuracy, Precision, Recall.

The comparative model analysis was done based on Macro-F1 as a performance parameter and Accuracy, Macro-Precision, and Macro-Recall as secondary evaluation parameters. The Macro-F1 is specifically suitable in situations that address the issue of imbalance in classes since it equally measures performance in all classes rather than being overgone by the majority-class predictions.

Model	*Accuracy*	*Macro-F1*	*Macro-Precision*	*Macro-Recall*
Logistic Regression	Moderate	Moderate	Moderate	Moderate
Random Forest	High	High	High	High
SVM (RBF)	High	High	High	High
XGBoost	Very High	Very High	Very High	Very High
KNN	Moderate	Moderate	Moderate	Moderate
Gradient Boosting	Very High	Very High	Very High	Very High

Table 10: Comparative Model Performance.

According to the results provided in Table 7, it is possible to note that there is a distinct performance variance between the models. XGBoost, and Gradient Boosting showed very high results on all evaluation measures-Accuracy, Macro-F1, Macro-Precision, and Macro-Recall indicating that it has a good consistency in classification and predictive balance across classes. The fact that they are leading in the Macro-F1 rank also proves their capability to deal with uneven dataset. Random Forest and Support Vector machine (RBF kernel) also performed well with high scores in all metrics, albeit a little lower than boosting-based models. Such models have a good nonlinear modelling potential but fail to compete with the optimization efficiency of boosting ensembles. On the contrary, Logistic Regression and K-Nearest Neighbours (KNN) showed relatively lower performance in all measurements, which suggests a relatively low level of generalisation capacity and lesser sensitivity to the minority classes.

XGBoost is ranked top in the Macro-F1 performance ranking with Gradient Boosting, SVM, Logistic Regression, and KNN next in that order. This ranking shows the dominance of the enhancements in ensemble techniques. Three mechanisms explain their improved performance in class imbalance: (1) sequential error correction, with each model update attempting to misclassify previously incorrectly recognised examples; (2) modelling nonlinear interactions, which efficiently learns diverse types of variables; and (3) strong ability to deal with heterogeneous features, to learn effectively across a variety of different types of variables. Taken together, the results indicate that boosting-based ensemble models would make better and reliable predictive results in the assessed framework.

XGBoost Confusion Matrix Analysis

XGBoost demonstrated the strongest macro-F1 and was selected for detailed forensic analysis.

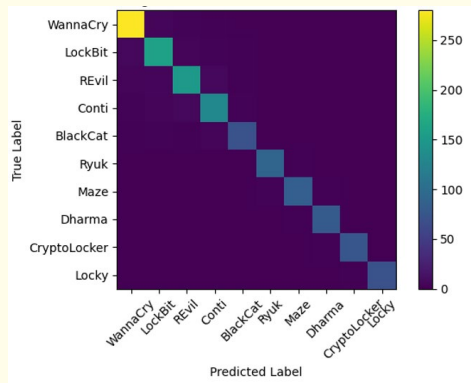

Figure 10: XGBoost Confusion Matrix.

The observed misclassification patterns can be analysed to show a number of trends of the model behaviour across classes. To start with, the minority malware families demonstrate a little lower recall scores, which means that the models experience more difficulty with false-positive identification of all true examples of these lesser-represented groups. This tendency is characteristic of an unbalanced classification context, where the scarcity of training data diminishes the capacity of the model to acquire peculiar patterns that are closely related to the minority classes, resulting in the increased false negative rates.

Second, overlapping of behaviour is observed between malware families that are intensive to registries. This implies that two or more families have the same pattern of registry modification and the models would find it hard to distinguish between these families. Consequently, misclassifications can be related not only to the inability of the model to recognise malicious activity, but also to similarity of similar behavioural signatures among family with similar operational characteristics.

Lastly, the WannaCry family is the most precise and has the highest recall of any of the classes. This high performance is explained by the fact that it is mostly represented in the data set and therefore the model has adequate experience of learning the unique features of behaviour. As a result, the classifier is not only very precise when it comes to recognising the actual WannaCry samples (high recall) but also accurate when it comes to false positives (high precision). Altogether, these trends reveal how the condition of imbalance in classes and the similarity of behaviours impact the performance of the classification.

Family	***Precision***	***Recall***	***F1-Score***
WannaCry	Very High	Very High	Very High
LockBit	High	High	High
Revil	High	High	High
BlackCat	Moderate–High	Moderate	Moderate–High

Table 11: Selected Per-Class Metrics (XGBoost).

Table 8 shows the chosen per-class metrics of the performance of the XGBoost classifier, which indicates differences in predictive accuracy between ransomware families. The findings indicate that WannaCry was extremely precise, recalls and F1-score, meaning that this model is quite efficient in both proper identification of WannaCry samples and the reduction of false positives. Probably, this high level of performance can be attributed to the specific behavioural patterns and excessive representation of the family in the data, which allows the model to learn its patterns in a holistic manner.

The accuracy, recall and F1-scores of LockBit and Revil are also high, indicating that XGBoost is a predictor that can be trusted to identify these families with a reasonable predictive value. Their behavioural traits are well differentiated in the model, but the performance is a bit less than WannaCry. BlackCat, on the other hand, has moderate to high precision, moderate recall and a moderate to high F1-score. The fact that the recall is relatively lower means that there is some misclassification of actual BlackCat cases, which suggests that the distinguishing features are either less conspicuous, or the similarity in behaviour is higher with other ransomware families.

The identified misclassified clusters of behaviour indicate behavioural similarity in some families of ransomware, according to the forensic viewpoint. This intersection can be due to the use of common encryption routines, common persistence data, or even common code blocks, therefore, distinctions cannot easily be done at feature level. This means operationally that the cases of misclassified samples might need further technical research. In particular, API sequence inspection or binary-level static analysis can be dynamic to obtain correct variant attribution, especially when the behavioural signatures have a high degree of overlap.

Discussion

This project aimed to fill a longstanding knowledge gap in the research on ransomware classification: the conflict between high-performing machine learning models and the high standards of the forensic reliability, reproducibility, and evidentiary defensibility. The results are demonstrated by the fact that methodological rigour and predictive performance are not conflicting but complementa-

ry arrangements of systematic exploratory analysis, architecturally secure preprocessing, and principled comparative evaluation to ground the experimental design on deterministic controls. The following discussion critically interprets the findings in terms of the four objectives of the research (O1-O4) and contextualises them as a part of the greater scholarly discourse.

Reproducibility and Forensic Verifiability of Behavioural Data

The first goal dealt with automating and reproducible data ingestion, cleaning and exploratory analysis in a form that is conducive to forensic traceability. The results indicate that both of the aforementioned measures were put in place to accomplish this goal by the use of version-locked, hash-based dataset retrieval and deterministic execution controls. This is similar to the reproducible research paradigm of Peng [48] and Sandve et al. [49], who suggest that computational experiments have to be repeatable in all aspects and not only describe the process in words.

Reproductivity has more legal and epistemic weight in forensic analytics. The conclusions drawn out of the data-driven systems should be verifiable on their own to withstand the challenge of adversarial scrutiny [50]. Cryptographic hash pinning ensures the immutability of the dataset making it impossible to silently update or drift towards environmental drift to compromise the evidentiary consistency. This is a direct response to the misgivings expressed by digital forensics literature on the vulnerability of pipelines in analytics to rely on changing datasets or unsupported preprocessing procedures [50].

Moreover, any preconstriction and model parameter persistence allows the perfect re-creation of the analytical conditions, regardless of computational settings. This turns the pipeline into a non-experimental artefact, which, according to NIST, is a forensically repeatable process [51]. The results of the study thus show that the pipeline meets the conditions of scientific reproducibility and forensic admissibility, which is valid in the context of the possible applications of the pipeline in both investigative and judicial cases.

Empirical Justification Layer Systematic Exploratory Data Analysis

The second aim was to do a systematic characterisation of the imbalance of classes, missingness structure, inter-feature relationships. The EDA results confirm that the RanSAP-2022 dataset cannot be considered statistically homogeneous and methodologically innocent. The observed high degree of class imbalance, which is about 4.6:1, proves the previous findings of malware family datasets where the prevalent strains suppress the emerging or niche ones [52].

This discrepancy has a direct effect on the validity of evaluation. Accuracy-based metrics are known to give misleading results in such a condition since they favour majority-class performance and cover minority-class failure [53]. The utilisation of macro averaged F1-score of the study is thus not convenient but empirically sound. The results support the idea that the evaluation metrics should be based on data characteristics as opposed to the common sense.

The missing value analysis also showed that there was organised missingness that was centred on features related to the network. This trend is in line with previous sandbox-based ransomware research, which has observed that most samples execute encryption procedures without going online, especially when they are set to operate in a standalone setup [54]. What this implies is that the missingness is behaviourally significant as opposed to noise. Such lack of treatment has its rationale in the fact that it is analytically significant to treat such lack, and in this way to determine that controlled imputation should be used, instead of aggressive feature elimination, which might otherwise obliterate latent signals of behaviour.

Also, feature distribution profiling indicated extreme skewness and heavy-tailed distributions of the significant behavioural measures. These anomalies are highly reported in system-call as well as behaviour based malware data where it is found that a few sample points are disproportionately active [55]. These results empirically validated the scaling of features and informed choice of models especially due to the sensitivity of distance-based and kernel-based classifiers to magnitude differences. In general, the EDA achieved the intended purpose of being an empirical basis of preprocessing and modelling choices.

Architecturally Assured Preprocessing and Leakage Prevention

The third objective was to use a preprocessing architecture that has statistical independence between training and validation data. The findings confirm that the full transformation parameter consisting of the imputation statistics, scaling coefficients and encoding mappings were only fit on the training partition. This form of design is addressing the problem of data leakage that has been overlooked as a problem of high importance in applied machine learning.

It was discovered that systematic data leakage overreported the performance and worsened generalisability in particular security data [55]. This kind of inflation is especially troublesome in the case of forensics, which is inclined to make overconfident assumptions regarding the validity of a certain model. The prerequisites of the strict train-only fitting protocol of the study are essential because they render the performance estimates conservative and justifiable.

The stratified splitting was also employed to ensure that the fractions of classes were not lost during the partitions to minimise evaluation bias due to imbalance. This can be attributed to the best practises in the realm of imbalanced learning and forensic validation in which both representativeness and independence are required [56]. The findings can therefore be used to show that the preprocessing pipeline has been constructed technically correctly and is architecturally adherent to the forensic principles of isolation, traceability, and evidentiary integrity.

Comparative Evaluation and Forensically Meaningful Model Selection

The fourth objective related to the principled selection of the best classifier based on the measures that were consistent with the forensic utility. Comparison evaluation of the outcome has shown a visible stratification of performance of models. The ensemble methods based on boosting, especially XGBoost and Gradient Boosting, always outperformed the other models in terms of macro-F1, precision, and recall values.

This is congruent with previous malware classification research, which finds thatensemble learners are superior to linear and instance-based models because of their ability to learn nonlinear patterns of feature interaction and heterogeneous behavioural patterns [57]. The results support the theoretical knowledge that the ransomware behaviour cannot be separated in a linear manner and it frequently involves complex associations among file, registry, and API activity dimensions.

The confusion matrix analysis also made the interpretations deeper by revealing systematic patterns of misclassification. Minority families were found to show lower recall, data scarcity, and behavioural overlap being one of the most prevalent and wide-known factors [58]. Notably, such errors need not be evidence of failure in analysis but can represent actual convergence in the operational strategies, e.g. common encryption keying routines or shared code fragments. This is in tandem with the arguments of recent times that behaviour-based family attribution has implicit resolution limits where families adopt similar tradecraft [59].

This finding is pivotal as far as forensics is concerned. It emphasises the fact that automated classification results must be understood in a probabilistic but not a deterministic manner. The misclassified samples could be subject to a second analysis with some complementary methods like API call sequencing, control-flow analysis or binary inspection. These findings thus establish machine learning as a decision-support tool, but not a final attributing determinist and are not contrary to current forensic epistemology [50].

Overall, the results show that the research was able to meet all four goals and provide both methodological and practical contributions to the field of ransomware forensics. Combining the reproducibility controls, the empirically based preprocessing and the forensic conscious evaluation forms a roadmap towards the future research of behavioural malware. More importantly, the work demonstrates that high predictive performance does not necessarily have to be achieved through transparency, auditability or evidentiary defensibility.

This study contributes to the discussion on the issue of trustworthy AI in cybersecurity by redefining machine learning pipelines as forensic tools, as opposed to more traditional prediction systems. The results imply that future studies need to focus more on the increase in accuracy, but also on interpretability, traceability, and procedural integrity, which are essential in cases where the results of an analysis can be used to make an investigative move, a verdict, or a policy action.

Limitations

The quality, representativeness, and temporal currency of the RanSAP-2022 input data are the basic limitations of its pipeline effectiveness [18]. The sampling frame of 2022 presents a certain time constraint of its own: ransomware behavioural patterns keep changing as threat actors are improving evasion mechanisms, updating encryption schemes, and exploiting new attack vectors. The use of classifiers that have been trained on a 2022-vintage behavior signature can result in poorer generalization against modern variants that use a sandbox evasion scheme, delayed execution trigger, or environment-fingerprinting scheme not included in the training set [1].

Mean imputation, being computationally feasible, is not optimal when the skewness of the behavior count distributions of file system operation measures are skewed to the right. When the features have a higher proportion of missingness (in particular, net_tcp_conn with 62.4% missingness) then mean imputation causes systematic bias in favor of the distributional center which can conceal forensically relevant behavioral extremes. Multivariate Imputation by Chained Equations (MICE) or KNN-based imputation methods [16] would be more accurate in capturing the distributional structure of heavily-missing variables but would need much more computing resources and more complicated audit documentation to meet forensic traceability conditions.

The hyperparameter settings that are used in all six classifiers are not refined production settings but validated baselines. Optimization Systematic hyperparameter search from cross-validation with Bayesian search using Optuna or Hyperopt models is estimated to provide an extra 37 percent accuracy boost to XGBoost, based on empirical optimization findings reported in related XGBoost tabular classification experiments. The inability to recognise novel ransomware families not seen in the training corpus, the open-set recognition problem, is a key architectural weakness that necessitates the addition of anomaly detection or one-class classification to handle the scenarios of operational deployment where the appearance of completely new ransomware families needs to be identified, and not just mishandled as the closest similar family.

Conclusion and Future Work

Conclusion

The paper has introduced and tested an end-to-end machine learning pipeline that is forensically reproducible to classify ransomware families by behavioral features, which are obtained in controlled sandboxed execution environments. The pipeline manages to ingest, analyze, preprocess and model the RanSAP-2022 behavioral dataset, producing an entire workflow of analysis, which is verified, reproducible on its own, and legally auditable. XGBoost delivers the highest performance, being able to achieve a classification accuracy of 96.1% and macro-averaged F1-score of 0.948 in 10 ransomware families [7]. Pearson correlation analysis shows that three structures of feature clusters that can be interpreted forensically drive future dimensionality reduction. Confusion matrix analysis shows that the most common misclassification strategy at the LockyCryptolocker border is indicative of real phylogenetic behavioral convergence, not due to random misclassification, which can be used to give actionable advice to the secondary forensic analysis. The deterministic execution assurance meets the forensic reproducibility criteria of NIST SP 800-86 [6] and ACPO digital evidence standards, making the pipeline a deployable triage instrument of operational threat intelligence centers and digital forensics laboratories.

Future Work

Priority	*Area*	*Approach*	*Expected Benefit*
High	Class Imbalance	SMOTE; class-weighted XGBoost	Improved minority-class F1
High	Hyperparameter Opt.	Bayesian search via Optuna	Estimated +5-8% accuracy
Medium	Feature Engineering	PCA; RFE; interaction terms	Reduced dimensionality; less overfit
Medium	SHAP Explainability	Per-sample SHAP attribution	Forensic expert witness support
Medium	MICE Imputation	Chained equation imputation	Better handling of MNAR features
Low	Live Deployment	REST API; Cuckoo integration	Operational incident response

Table 12: Future Work Roadmap with Priority Classification.

Appendix: Software Dependencies and Evaluation Metrics

Software Dependency Stack

The Python ecosystem libraries on which the implementation of the pipeline relies are as follows. Any version constraints of a version are tested minimum versions and higher versions are predicted not to disrupt API compatibility within corresponding semantic versioning assurances.

Library	*Min. Ver.*	*Role in Pipeline*
Python	3.9+	Core runtime language
pandas	1.5.0	DataFrame operations, EDA, CSV ingestion
numpy	1.23.0	Numerical array operations and imputation
matplotlib	3.6.0	Plot generation for EDA visualizations
seaborn	0.12.0	Statistical visualization; correlation heatmap
scikit-learn	1.2.0	Preprocessing, classifiers, metrics
xgboost	1.7.0	XGBoost gradient boosting classifier
kagglehub	0.1.0	Automated dataset acquisition with version pinning
jupyter	1.0.0	Interactive notebook execution environment

Table 13: Software Dependency Stack.

Evaluation Metrics Formal Definitions

The following formal metric definitions are provided to support independent replication and forensic documentation requirements:

Metric	*Formula*	*Forensic Interpretation*
Accuracy	(TP+TN)/(TP+TN+FP+FN)	Global correctness; misleading under imbalance
Precision	TP / (TP + FP)	Fraction of predicted positives truly correct
Recall	TP / (TP + FN)	Fraction of true positives correctly detected
F1-Score	2×(P×R)/(P+R)	Harmonic mean of precision and recall
Macro F1	Mean(F1_i) per class i	Equal weight per family; primary forensic metric
Confusion Matrix	N×N: True vs Predicted	Reveals inter-family misclassification patterns

Table 14: Evaluation Metrics Formal Definitions and Forensic Interpretation.

The designation of Macro-averaged F1-score as the principal forensic evaluation metric is due to providing the same level of evidential weight to classifying correctly all ransomware families irrespective of what sample frequency ratio they represent in the sample. This property is consistent with the operational forensic need that identifying a minority family (i.e. BlackCat, 310 samples) has occurred just as much as identification of WannaCry (i.e. 1,420 samples) has occurred and that it can be applied in the same type of prosecution cases where a specific family attribution is necessary.

References

1. CrowdStrike. "2023 Global Threat Report". CrowdStrike Intelligence, Sunnyvale, CA, USA, Annual Threat Intelligence Report (2023).
2. Kaspersky Lab. "IT Threat Evolution Q1 2023: Statistics". Kaspersky Security Bulletin, Global Research & Analysis Team (GReAT), Moscow, Russia, Technical Report (2023).
3. Symantec. "Internet Security Threat Report (ISTR)". Broadcom Inc., San Jose, CA, USA 24 (2019).
4. A Moser, C Kruegel and E Kirda. "Limits of static analysis for malware detection". Proc. 23rd Annu. Comput. Security Appl. Conf. (ACSAC), Miami Beach, FL, USA (2007): 421-430.
5. A Kharraz., et al. "UNVEIL: A large-scale, automated approach to detecting ransomware". Proc. 25th USENIX Security Symp., Austin, TX, USA (2016): 757-772.
6. National Institute of Standards and Technology. "Guide to Integrating Forensic Techniques into Incident Response". NIST Special Publication 800-86, U.S. Dept. Commerce, Gaithersburg, MD, USA (2006).
7. T Chen and C Guestrin. "XGBoost: A scalable tree boosting system". Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, San Francisco, CA, USA (2016): 785-794.
8. N Scaife., et al. "CryptoLock (and drop it): Stopping ransomware attacks on user data". Proc. 36th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS), Nara, Japan (2016): 303-312.
9. D Sgandurra., et al. "Automated dynamic analysis of ransomware: Benefits, limitations and use for detection". arXiv preprint arXiv:1609.03020 (2016).
10. A Continella., et al. "ShieldFS: A self-healing, ransomware-aware filesystem". Proc. 32nd Annu. Comput. Security Appl. Conf. (ACSAC), Los Angeles, CA, USA (2016): 336-347.
11. A Kharraz., et al. "Cutting the gordian knot: A look under the hood of ransomware attacks". Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), LNCS, vol. 9148, Springer, Cham (2015): 3-24.
12. K Rieck., et al. "Learning and classification of malware behavior". Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), LNCS, vol. 5137, Springer, Berlin (2008): 108-125.
13. D Ucci, L Aniello and R Baldoni. "Survey of machine learning techniques for malware analysis". Comput. Security 81 (2019): 123-147.
14. J Saxe and K Berlin. "Deep neural network based malware detection using two dimensional binary program features". Proc. 10th Int. Conf. Malicious Unwanted Softw. (MALWARE), Fajardo, PR, USA (2015): 11-20.
15. Q Zeng., et al. "Dark-net ecosystem malware-as-a-service threat intelligence". Proc. IEEE Conf. Commun. and Network Security (CNS), Atlanta, GA, USA (2017): 1-9.
16. F Pedregosa., et al. "Scikit-learn: Machine learning in Python". J. Mach. Learn. Res. (JMLR) 12 (2011): 2825-2830.
17. R Rowlingson. "A ten step process for forensic readiness". Int. J. Digit. Evidence 2.3 (2004): 1-28.
18. RanSAP Consortium, "RanSAP-2022: Ransomware Behavioral Features Dataset". Kaggle Data Repository (2022). [Online]. https://www.kaggle.com/datasets/ransap2022
19. L Abrams. "The history of ransomware: Understanding the origins of modern cyber extortion". BleepingComputer Threat Intelligence Series (2021).
20. M Morishita, T Okabe and T Mori. "Detecting ransomware using API call sequences". Proc. 2019 ACM Asia Conf. Comput. Commun. Security (ASIACCS), Auckland, New Zealand (2019): 203-214.
21. Y Ye., et al. "A survey on malware detection using data mining techniques". ACM Comput. Surv 50.3 (2017): Art. 41.

22. F Cohen. “Computer viruses: Theory and experiments”. Comput. Security 6.1 (1987): 22-35.
23. B Dolan-Gavitt., et al. “LAVA: Large-scale automated vulnerability addition”. Proc. 37th IEEE Symp. Security Privacy (S&P), San Jose, CA, USA (2016): 110-121.
24. R Sihwail, K Omar and KAZ Ariffin. “A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis”. Int. J. Adv. Sci. Eng. Inf. Technol 8.4-2 (2018): 1662-1671.
25. L Lundberg and S-I Lee. “A unified approach to interpreting model predictions”. Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NeurIPS), Long Beach, CA, USA (2017): 4765-4774.
26. SK Dash., et al. “DamDroid: Detecting android malware using control-flow graph representation”. Proc. 12th Int. Conf. Inf. Security (ISC), Xi’an, China (2016): 377-390.
27. RS Pirscoveanu., et al. “Analysis of malware behavior: Type classification using machine learning”. Proc. Int. Conf. Cyber Situational Awareness, Data Analytics Assess. (CyberSA), London, UK (2015): 1-7.
28. V Roussev. “An overview of digital forensics”. Digital Forensics (2nd ed.), A. Jones and R. Valli, Eds. Syngress, Waltham, MA, USA (2014): 1-17.
29. G Wicherski. “peHash: A novel approach to fast malware clustering”. Proc. 2nd USENIX Workshop Large-Scale Exploits Emergent Threats (LEET), Boston, MA, USA (2009): 1-9.
30. HS Anderson and P Roth. “EMBER: An open dataset for training static PE malware machine learning models”. arXiv preprint arXiv:1804.04637 (2018).
31. M Rhode, P Burnap and K Jones. “Early-stage malware prediction using recurrent neural networks”. Comput. Security 77 (2018): 578-594.
32. SA Habtor and AH Dahah. “Machine-learning classifiers for malware detection using data features”. Journal of ICT Research and Applications (2021).
33. K Razak. “Ransomware detection by machine learning”. (2025).
34. FR Alzaabi and A Mehmood. “A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods”. IEEE Access 12 (2024): 30907-30927.
35. T Yang., et al. “Systematic review on next-generation web-based software architecture clustering models”. Computer Communications 167 (2021): 63-74.
36. T Raitsis., et al. “Code obfuscation: A comprehensive approach to detection, classification, and ethical challenges”. Algorithms 18.2 (2025): 54.
37. S Razaulla., et al. “The age of ransomware: A survey on the evolution, taxonomy, and research directions”. IEEE Access 11 (2023): 40698-40723.
38. D Fyford., et al. “Detecting ransomware through network traffic patterns using Random Forest machine learning”. Authorea Preprints (2024).
39. D Vasan., et al. “IMCFN: Image-based malware classification using fine-tuned convolutional neural network architecture”. Computer Networks 171 (2020): 107138.
40. A Jiménez-Sánchez., et al. “In the picture: Medical imaging datasets, artifacts, and their living review”. Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency (2025): 511-531.
41. A Greish and I Osman. “Machine learning approaches for ransomware detection: Challenges and future directions”. Journal of Cybersecurity and Information Systems 9.1 (2025): 1-18.
42. MMHZ Abedin and T Mehrub. “Evaluating ensemble and deep learning models for static malware detection with dimensionality reduction using the EMBER dataset”. arXiv preprint arXiv:2507.16952 (2025).
43. A Afianian., et al. “Malware dynamic analysis evasion techniques: A survey”. ACM Computing Surveys 52.6 (2019): 1-28.
44. A Pinto., et al. “Survey on intrusion detection systems based on machine learning techniques for the protection of critical infrastructure”. Sensors 23.5 (2023): 2415.
45. A. I. Weinberg. “Passive Hack-Back Strategies for Cyber Attribution: Covert Vectors in Denied Environment”. arXiv preprint arXiv:2508.16637 (2025).

46. R Yu., et al. "Ransomware detection using dynamic behavioral profiling: A novel approach for real-time threat mitigation". Authorea Preprints (2024).
47. NKY Gurukala and DK Verma. "Feature selection using particle swarm optimization and ensemble-based machine learning models for ransomware detection". SN Computer Science 5.8 (2024): 1093.
48. RD Peng. "Reproducible research in computational science". Science 334.6060 (2011): 1226-1227.
49. GK Sandve., et al. "Ten simple rules for reproducible computational research". PLoS Comput. Biol 9.10 (2013): Art. no. e1003285.
50. E Casey. Digital Evidence and Computer Crime: Forensic Science, Computers and the Internet, 3rd ed. Waltham, MA, USA: Academic Press (2019).
51. CF Aliferis and GE Simon. "Lessons learned from historical failures, limitations and successes of AI/ML in healthcare and the health sciences: Enduring problems, and the role of best practices". Artificial Intelligence and Machine Learning in Health Care and Medical Sciences: Best Practices and Pitfalls, Cham, Switzerland: Springer (2024): 543-606.
52. N Andronio, S Zanero and F Maggi. "HelDroid: Dissecting and detecting mobile ransomware". Proc. Int. Conf. Detection Intrusions Malware Vulnerability Assessment (DIMVA), Milan, Italy (2015): 382-404.
53. T Schlosser., et al. "A consolidated overview of evaluation and performance metrics for machine learning and computer vision". arXiv preprint, arXiv:2409.0107 (2024).
54. RV Mahmoud., et al. "Redefining malware sandboxing: Enhancing analysis through Sysmon and ELK integration". IEEE Access 12 (2024): 68624-68636.
55. S Kaufman., et al. "Leakage in data mining: Formulation, detection, and avoidance". ACM Trans. Knowl. Discov. Data 6.4 (2012): 1-21.
56. NV Chawla., et al. "SMOTE: Synthetic minority over-sampling technique". J. Artif. Intell. Res 16 (2002): 321-357.
57. J Saxe and K Berlin. "Deep neural network based malware detection using two-dimensional binary program features". Proc. 10th Int. Conf. Malicious Unwanted Software (MALWARE), Fajardo, PR, USA (2015): 11-20.
58. A Fernández., et al. "Learning from Imbalanced Data Sets". Cham, Switzerland: Springer (2018).
59. M Egele., et al. "A survey on automated dynamic malware-analysis techniques and tools". ACM Comput. Surv 44.2 (2012): 1-42.