electronics

Article

A Study on Countermeasures against Neutralizing Technology: Encoding Algorithm-Based Ransomware Detection Methods Using Machine Learning †

Jaehyuk Lee [1], Jinseo Yun [2] and Kyungroul Lee [3,*]

[1] Process Development Team, Fescaro, Suwon 16512, Republic of Korea; gurtmggg@mokpo.ac.kr
[2] Faculty of Interdisciplinary Studies, Chonnam National University, Gwangju 61186, Republic of Korea; 202800@jnu.ac.kr
[3] Department of Information Security, Mokpo National University, Muan 58554, Republic of Korea
* Correspondence: carpedm@mnu.ac.kr; Tel.: +82-61-450-2713
† This work is a revised and supplemented paper based on the paper selected as an encouragement prize in the "2023 Cybersecurity Paper Contest" hosted by the National Intelligence Service and organized by the Korea Cyber Security Association.

Abstract: Ransomware, which emerged in 1989, has evolved to the present in numerous variants and new forms. For this reason, serious damage caused by ransomware has occurred not only within our country but around the world, and, according to the analysis of ransomware trends, ransomware poses an ongoing and significant threat, with major damage expected to continue to occur in the future. To address this problem, various approaches to detect ransomware have been explored, with a recent focus on file entropy estimation methods. These methods exploit the characteristic increase in file entropy that is caused by ransomware encryption. In response, a method was developed to neutralize entropy-based ransomware detection technology by manipulating entropy using encoding methods from the attacker's perspective. Consequently, from the defender's standpoint, countermeasures are essential to minimize the damage caused by ransomware. Therefore, this article proposes a methodology that utilizes diverse machine learning models such as K-Nearest Neighbors (KNN), logistic regression, decision tree, random forest, gradient boosting, support vector machine (SVM), and multi-layer perception (MLP) to detect files infected with ransomware. The experimental results demonstrate empirically that files infected with ransomware can be detected with approximately 98% accuracy, and the results of this research are expected to provide valuable information for developing countermeasures against various ransomware detection technologies.

Keywords: ransomware; information entropy; encoding algorithms; neutralization strategies; encoding; machine learning

Citation: Lee, J.; Yun, J.; Lee, K. A Study on Countermeasures against Neutralizing Technology: Encoding Algorithm-Based Ransomware Detection Methods Using Machine Learning. *Electronics* **2024**, *13*, 1030. https://doi.org/10.3390/electronics13061030

Academic Editors: Lixin Wang, Jianhua Yang and Qiang Ye

Received: 29 December 2023
Revised: 5 March 2024
Accepted: 6 March 2024
Published: 9 March 2024

1. Introduction

Ransomware is a combination of the words "ransom" and "ware" and is a type of malicious code that encrypts files stored in a user terminal and demands a ransom in exchange for decryption [1]. Since its first appearance recorded by Joseph Pope in 1989, new and varied ransomware has been constantly emerging [2]. According to the Korea Internet & Security Agency's Ransomware Trends Report for the second quarter of 2022, the number of ransomware programs detected in Korea was 61,582 in April 2022, 6561 in May, and 26,546 in June, with an average of approximately 40,000 to 60,000 ransomware programs detected per month [3].

In response, various studies are being conducted to detect and prevent ransomware, and as a representative study, ransomware detection based on entropy measurement, where the increase in file entropy serves as an indicator of ransomware, has arisen as a new method [4]. However, this method detects ransomware based on the increasing entropy

of files infected with ransomware; thus, it is difficult to detect ransomware if an attacker manipulates entropy. Attackers use encoding techniques such as base64 to manipulate the entropy of files in order to neutralize entropy-based ransomware detection methods [5,6]. Consequently, to prevent infections from ongoing ransomware development and minimize damage caused by it, a response to technologies that neutralize ransomware detection is essential.

Therefore, this article presents an efficacious ransomware detection technology using machine learning. The proposed methodology, based on datasets derived from the analysis of ransomware detection using encoding, verifies whether files infected with ransomware can be effectively detected, even if a technology—such as an encoding technique that renders ransomware detection methods inoperative—is applied.

The contributions of this article are as follows.

- In this study, we intensively surveyed entropy measurement-based ransomware detection mechanisms and organized datasets and environments for experiments to prevent and minimize damage caused by infection from continuously developing ransomware.
- Based on the results of an in-depth analysis of existing technologies, various machine learning models were applied to respond to ransomware detection to verify that detection is possible even if a technology neutralizing it is applied.
- Comparing and evaluating ransomware detection performance for neutralizing technology, which is a target of comparison for the proposed scheme, it was found that the proposed scheme can be used to detect ransomware more effectively.

2. Prior Knowledge and Related Works

2.1. Prior Knowledge

2.1.1. Entropy

The concept of entropy was first introduced by American computer scientist Claude Shannon, and the term expresses the amount of uncertainty or the degree of unpredictability [7]. In the context of computer engineering or information security, this term is mainly used to describe the expectation of information, also referred to as information entropy. In other words, information entropy refers to the uniformity of data, and if the data are uniform, they have high entropy. Conversely, if the data are not uniform and are deviated, they have relatively low entropy [8–10]. In this case, entropy can range from zero to eight.

2.1.2. Characteristics of Files Based on Ransomware Infection

This section describes the characteristics of files when a system is infected with ransomware and files are encrypted by it. First, encryption refers to the process of converting plaintext into ciphertext, using an encryption algorithm, so that the data cannot be accessed except by those with secret information related to specific data [11]. Thus, the main purpose of encryption is to ensure that only those who possess the secret key can generate the ciphertext and the decrypted text. Therefore, when generating a ciphertext, it should be manufactured in such a way that it is difficult for people who do not own the secret key to decrypt the ciphertext and infer the plaintext. For this purpose, when designing an encryption algorithm, it is designed so that all values are distributed with a certain probability, without repeating or biasing a particular value. In other words, an effective algorithm is one that generates ciphertext so that all data are distributed at a constant probability. To sum up, ciphertexts generated by encryption algorithm are characterized by high entropy [12].

2.1.3. Binary–Text Encoding Techniques

All files on a computer are expressed in binary format, and it is very difficult for people to understand only binary data. For this reason, a binary–text encoding technique has emerged which converts binary data into text so that people can identify it [5]. Representative binary–text encoding techniques include base32 [13], base64 [13], ascii85 [14], punycode [15], etc. All encoding techniques basically involve converting the existing binary

data into text, and, ultimately, all data in the file are converted. In other words, if an encoding technique is applied to the ciphertext assuming a ransomware infection, the distribution of data in the file will change, allowing the attacker to manipulate entropy and, ultimately, neutralize detection methods that measure the entropy of the file. In the end, the ransomware detection neutralization technology, which is the subject of analysis and performance evaluation in this article, neutralizes ransomware detection by manipulating entropy based on these characteristics of binary–text encoding techniques.

2.2. Related Works

In this section, we focus on the neutralization technology of the ransomware detection method. The technology detects a ransomware-infected file by measuring the entropy of the file based on the characteristics of the ciphertext, as mentioned in Section 2.1.2. However, if the binary–text encoding method is applied to the encrypted file, the entropy of that file can be manipulated, making the detecting method ineffective. Section 2.2.1. presents a study that neutralized ransomware detection by applying base64 encoding; this previous study first proposed the possibility of neutralizing ransomware detection. Section 2.2.2. describes the results of an analysis of the limitations of existing neutralization technology and introduces a study that proposes an optimal neutralization technology by determining the optimal encoding method for each file format.

2.2.1. Research on Neutralizing Ransomware Detection Using base64 Encoding Technique

In [5], in order to neutralize ransomware detection technology based on file entropy measurement, a method was developed to lower the entropy of the ciphertext to resemble plaintext by binary–text encoding. In this study, base64 encoding was applied after encrypting files in CSV, TXT, DOC, XLS, PPT, DOCS, XLSX, PPTX, PDF, and JPG file formats to nullify ransomware detection. The results confirmed that for all file formats, the entropy of base64-encoded files was relatively lower than the entropy of the ciphertext. Consequently, it was proven that if an encoding method such as base64 is applied to files infected with ransomware, it is possible to neutralize an entropy-measurement-based ransomware detection method, as shown in Figure 1.

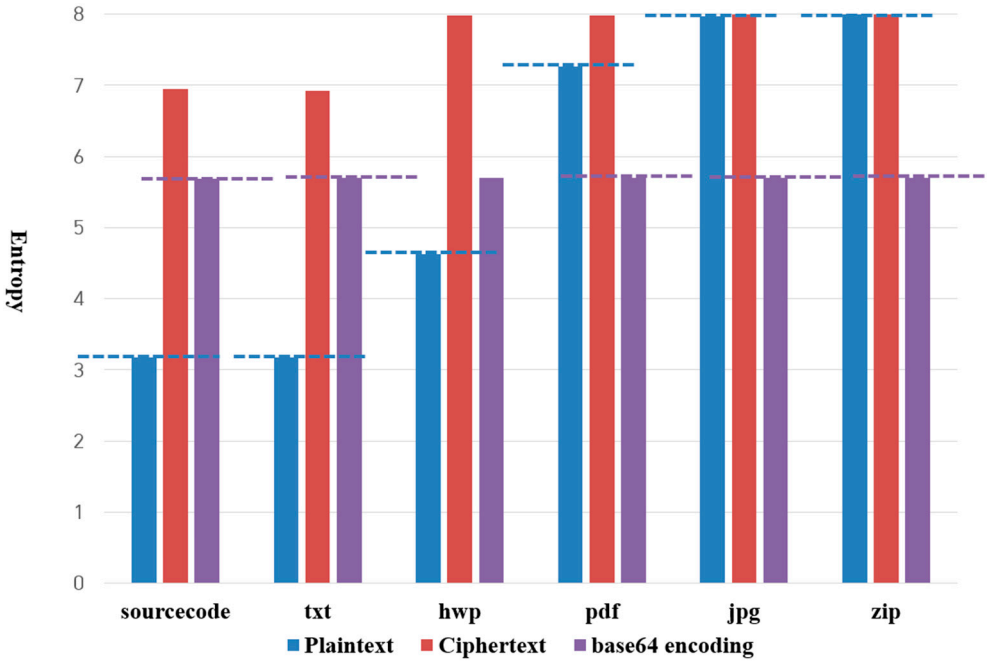

Figure 1. Result of entropy comparison with plaintext entropy, cryptogram entropy, and base64 encoding.

The figure shows that when this neutralization technique is used, the entropy of the base64-encoded file is lower than the entropy of the ciphertext. However, the entropy of

the base64-encoded file may be lower or higher than the entropy of the plaintext. This characteristic appears as a limitation that serves as the clue for recognizing that the attacker applied an encoding technique to neutralize ransomware detection from the defender's perspective. Ultimately, the technique is to neutralize ransomware detection using base64 encoding results in the entropy of the base64-encoded file lower than the entropy of the ciphertext from the attacker's perspective, whereas this technique has the flaw of providing a hint for neutralization attempts, due to this characteristic of the entropy of base64-encoded files being lower or higher than the entropy of plaintext. Moreover, although the entropy of the base64-encoded file is lower than the entropy of the ciphertext, it has a value between the entropy values of the plaintext and the ciphertext; consequently, there is also a drawback whereby the defender can clearly identify the encoded file and detecting ransomware.

2.2.2. Research on Neutralizing Ransomware Detection Using Optimal Encoding Methods for Each File Format

As mentioned in Section 2.2.1, neutralization technology using base64 encoding is used when the defender knows the entropy of plaintext files or knows the average value, and the entropy of specific files is higher or lower than their value. By identifying unusual features, this technology can identify attackers' attempts to neutralize ransomware detection. Therefore, in [6], to overcome the limitations of this previous study, a method was proposed to manipulate entropy more precisely using base32, ascii85, and URL encoding techniques, including base64. In order to verify the proposed method, an encoding technique closest to the entropy of the plaintext for each file format was selected, through experiments, to prevent the defender from identifying encoded files, and the optimal encoding method for each file format was derived. As a result of comparing the performance with Section 2.2.1 based on the derived results, if a file with an entropy difference from the plaintext entropy of 1.0 or more is detected as a ransomware-infected file, the performance in the CSV file format was improved by 84%.

3. Approaches to Ransomware Detection Using Machine Learning

3.1. System Configuration for Ransomware Detection

This section describes the configuration of the ransomware detection system applying the method proposed in this article, and the overall system configuration is shown in Figure 2.

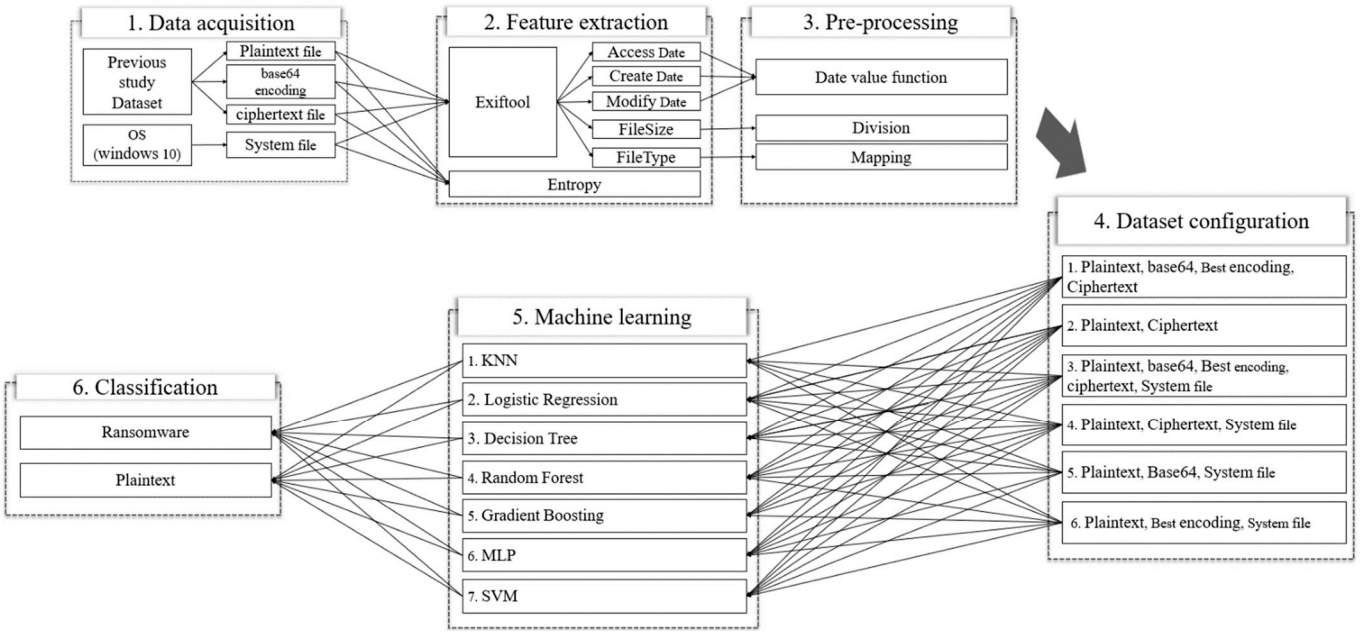

Figure 2. Comprehensive system configuration for ransomware detection.

The system for detecting ransomware consists of a total of six steps: "1. Data Collection, 2. Feature Definition, 3. Data Preprocessing, 4. Dataset Configuration, 5. Learning, and 6. Classification", and each step is explained in detail in the following sections.

3.1.1. Data Collection Stage

Firstly, data, such as plaintext, base64-encoded files, cryptographic files, and system files based on the same dataset as the previous study, are collected. This is intended to compare and evaluate performance with the ransomware neutralization study, which is a previous study. Additionally, to assume that the actual running system was infected with ransomware, Windows operating system files were included and collected as data. In previous research, datasets were built based on GovDocs1 data [5,6,16]. The data consisted of various types of files, including csv, doc, docx, pdf, jpg, txt, xls, and xlsx files. In this article, data were collected similarly to previous studies. Finally, the dataset was constructed based on the data collected at this stage.

3.1.2. Feature Extraction Stage

In the feature extraction step, features from data collected in Section 3.1.1 are extracted to be used as learning elements for data analysis and machine learning. The file scanning tool Exiftool (https://exiftool.org/) was used [17] to collect metadata from files, such as file creation, modification and access time, file size, and file type, which were defined as characteristics. In addition, in order to derive data on characteristics of entropy, which is the core information of the ransomware detection method proposed in this article, we analyzed entropy distribution according to the file format and number of files, targeting the plain text assumed as the original file, the cipher text assumed as the file infected with the ransomware, and the encoded file assumed as the file with entropy manipulated by the attacker to neutralize the ransomware detection technology. Figure 3 shows the obtained results.

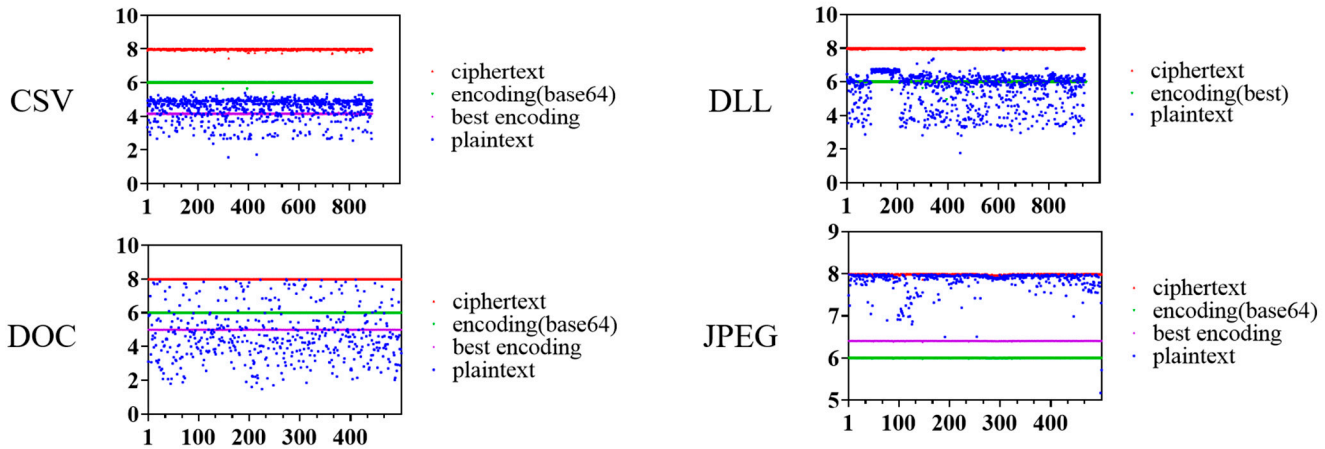

Figure 3. Entropy distribution based on file format and number of files.

The figure shows the entropy distribution of plaintext, ciphertext, base64 encoding, and optimal encoded files for CSV, DLL, DOC, and JPEG file formats. It was found that the ciphertext, which was considered to be a ransomware-infected file, had an entropy distribution close to eight in all file formats, while the plaintext considered to be the original file and the encoded file considered to be a manipulated file had different entropy distributions depending on the file format. Based on the results of this analysis, it was concluded that if entropy was defined as a learning characteristic, plain text, cipher text, and encoded files could be effectively classified. File creation, modification and access time, file size, and file type, including its entropy, were defined as features. Next, a new model which classifies ransomware-infected files through additional information other than entropy, was built.

Finally, a binary label was defined to classify each file. Ransomware-infected files, including files with encoding techniques, were defined as "0", while operating system files and general user files not infected by ransomware were defined as "1". The ransomware detection performance was assessed based on the final classification results.

3.1.3. Data Preprocessing Stage

In the data preprocessing step, data corresponding to the features defined in Step 2 are normalized to match the ranges of input data for the machine learning model and to improve classification performance. Among the features defined for learning in the proposed plan, entropy, file creation, modification and access time, file size, date, and file type are in text format. The file size appears to be significantly different from other data, which is likely to hinder machine learning classification performance [18]. Accordingly, the date data were converted to a serial number that can be recognized as a date using the DataValue() function. However, compared to entropy, which has a maximum value of eight, the file size feature varies greatly depending on the file; thus, data preprocessing through normalization is required. In order to meet these demands, the file size was converted into bytes and normalized by dividing it by 1,000,000. Like date data, file type features are basically composed of strings; as a result, normalization is required to train the data. To satisfy this requirement, the file type was converted from a string to an integer by assigning a zero-based number to each file type.

As a result of evaluating the performance of model learning according to data preprocessing, when the data were preprocessed through normalization, the performance improved by about 30%, on average. Through these results, we found that data preprocessing improves the learning performance of machine learning models. Therefore, data preprocessing must be applied for effective model learning, so the proposed method also normalizes the data based on the preprocessing process described above.

3.1.4. The Dataset Composition Stage

In this step, to compare and evaluate performance with ransomware detection and neutralization research as described in Section 2, datasets were configured, based on plaintext, base64 encoding, optimal encoding, and system files, with the following features defined: entropy, file creation, modification and access times, file size, and file type. The final dataset for learning and classifying the machine learning model was constructed by extending data characterization. In order to determine the model with the most effective performance in the dataset configuration process and prevent overfitting or underfitting, the training data and verification data were configured in a certain proportion. For more objective learning, experiments were conducted, including the process of randomly mixing data for each training. The datasets constructed in this article are shown in Table 1.

Table 1. Construction of datasets for experiments.

Dataset	Data Composition	Total Number of Files	Number of Files Infected with Ransomware	Number of Plaintext Files	Ratio
1	Plaintext, ciphertext	14,364	6983	7381	4.9:5.1
2	Plaintext, ciphertext, base64, optimal encoding	27,242	20,161	7081	7.5:2.5
3	Plaintext, base64, optimal encoding, ciphertext, system file	37,242	20,161	17,081	5.5:4.5
4	Plaintext, ciphertext, system file	24,634	6983	17,381	3:7
5	Plaintext, base64, system file	24,292	7146	17,146	3:7
6	Plaintext, optimal encoding, system file	22,868	5722	17,146	2.5:7.5

3.1.5. Learning Stage

In the learning step, each dataset is trained using a machine learning model to detect a ransomware-infected file based on six configured datasets. In this article, several models, such as K-Nearest Neighbors (KNN) [19], logistic regression [20], decision tree [21], random forest [22], gradient boosting [23], support vector machine (SVM) [23], and multi-layer perception (MLP) [24], are used to select the most efficient ransomware detecting machine learning model.

3.1.6. Classification Stage

The classification step evaluates the classification performance of ransomware-infected files based on the result of learning each dataset. In order to evaluate the results, account, precision, recall, F1-score, and AUC are used as indicators for each machine learning model.

3.2. Feature Extraction and Definition

In order to teach the machine learning model, which is a key technology for ransomware detection using machine learning proposed in this article, the process of extracting learning features, which greatly affect performance, should first be carried out. Therefore, this section describes the file scanning tools used for feature extraction. It also specifies the purpose defined by detailed explanations and features of entropy, file creation, modification and access time, file size, and file type defined in Section 3.1.2.

3.2.1. Feature Extraction Technique

Here, we describe the file scanning tool used in this article to extract features. The file scanning tool was developed to detect malicious codes attached to mail, overcoming the limitations of the network IDS (Intrusion Detection System), and is mainly used to quickly detect malicious files [25]. In this article, to prevent and minimize damage caused by ransomware, a technique is required to quickly extract features included in the metadata, and file scanning tools meet these requirements. To validate the suitability of the file scanning tool, a verification environment was configured by varying the number of files for sample files, and the performance and speed were analyzed for representative open-source file scanning tools such as LaikaBOSS [26], FSF [27], Strelka [28], and Exiftool [17]. The results are presented in Table 2.

Table 2. Performance evaluation results for file scanning scan speed.

File Scanning Tool	Number of files					
	10	**20**	**30**	**50**	**80**	**100**
Laika BOSS	1 s	3 s	4 s	8 s	12 s	15 s
FSF	11 s	38 s	42 s	75 s	126 s	157 s
Strelka	5 s	38 s	64 s	176 s	230 s	251 s
Exiftool	**0.18 s**	**0.2 s**	**0.23 s**	**0.3 s**	**0.35 s**	**0.4 s**

Blue emphasis: best performance results.

As shown as the file scan speed evaluation results, based on 100 sample files, the speed of Laika BOSS was 15 s, FSF's speed was 157 s, Strelka's was 251 s, and Exiftool's was 0.4 s. Compared to Strelka, which has the slowest scan rate, the efficiency increases by about 627 times using Exiftool. However, this verification showed that there are limitations in assessing the performance of the file scanning tool based on only 100 sample files because many files are stored in a system that is actually infected with ransomware. Therefore, to compare performance in a real-world environment, the performance of 100,000 system files and user files stored in the Ubuntu 20.04.4 operating system, rather than sample files, was evaluated. To appraise the performance, Exiftool and the Laika BOSS tool, which had the best performance in the previous experiment, were used. A total of 232,189 system files and 100,000 user files of the Ubuntu operating system were included, and, as a result, Laika BOSS took about 30 min and 23 s, and Exiftool took about 9 min and 47 s. Compared to

Laika BOSS, Exiftool achieves scanning file results approximately three times faster. Based on these results, Exiftool was selected as the file scanning tool for feature extraction.

3.2.2. Entropy Feature

As described in Section 3.1.2, entropy, which is the basic information of the plan proposed in this article, was defined as the main feature. Entropy consists of values from zero to eight, and was extracted through a source code that measures the uniformity of the data stored in the file [29]. By analyzing the entropy distribution results for plain text, cipher text, and encoded files, it was found that the entropy distribution was different for each file type. Based on this characteristic, if entropy is defined as a feature, it is speculated that ransomware-infected files can be classified.

3.2.3. Date Data (Modification, Access, Creation Time) Feature

Date data include file modification time, access time, and creation time among metadata extracted using Exiftool and are typically expressed as MAC (modify, access, create) data. Ransomware usually encrypts the majority of user files, except for system files; thus, if a system is infected with ransomware, the modification time and access time change. According to this characteristic, if the MAC information of normal files is compared and analyzed with the distribution of MAC information of files infected with ransomware, classifying files infected with ransomware is considered possible.

However, machine learning models used in the proposed scheme learn data via numerical information such as entropy; this produces a limitation: date data composed of strings cannot be used. To overcome this problem, data are preprocessed through a normalization process that converts date data composed of strings into numeric data using the DataValue () function, as mentioned in Section 3.1.3.

3.2.4. File Size Feature

Like date data, file size data are extracted through Exiftool. As mentioned above, most ransomware does not encrypt system files but encrypts user files. In general, the size of the system file is often relatively smaller than the size of the user file, and most user files are document files such as txt, doc, xls, and pdf; therefore, it is estimated that the file size can be generalized. Accordingly, if the file size is defined as a feature, it is considered possible to classify files infected with ransomware.

3.2.5. File Type Feature

The file type is the last feature of ransomware classification. It is extracted using the same Exiftool used for date data and file size data, and serves to distinguish the type of file; thus, it is generally referred to as a file type or file extension. Since the ultimate goal of the plan proposed in this article is to address the technology that prevents ransomware detection through entropy measurement, it aims to detect files infected with ransomware using a machine learning model. In other words, even if an attacker manipulates the entropy of a file infected with ransomware by applying an encoding method, the file is already infected with ransomware and should be classified as ransomware. Therefore, when an encoding method for neutralization is applied, the file type is defined as a feature for a more sophisticated ransomware classification in order to generalize the entropy criterion (threshold) that varies depending on the file type.

Many ransomware programs have the feature of maintaining the file type representing the information of the original file in the file name to seamlessly decode the encrypted file when the user pays for it [8]. Due to these characteristics, even if a file is infected with ransomware, the file type can be determined; therefore, it is believed that it is possible to learn a model using the file type characteristics and detect a file infected with ransomware. Therefore, to confirm whether a given file type can be exploited as a feature, the results of infection with the actual WannaCry ransomware for xls, pdf, and doc files are shown

in Figure 4. We assume that our target ransomware is one that preserves the original file extension, such as WannaCry ransomware.

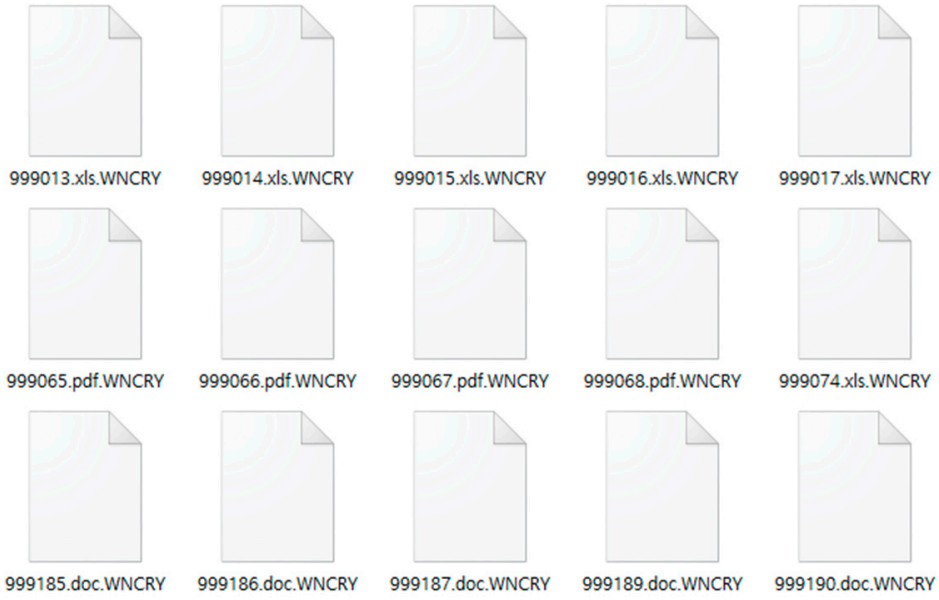

Figure 4. An example of a file infected with WannaCry ransomware.

3.3. Experimental Design and Validation According to the Dataset

This section presents the entire experimental configuration for generating a machine learning-based ransomware classification model with optimal performance based on the features defined in Section 3.1.2 and the dataset obtained in Section 3.1.4. First, in order to derive the optimal dataset for efficient training of a machine learning model, validation data and verification of each dataset and their results are described.

3.3.1. Experimental Configuration

Here, the experimental configuration is described to obtain a classification model with optimal performance based on the features and datasets defined in this article. The defined features consist of a total of four features: entropy, date data (modification, access, and creation time), file size, and file type. To compare and evaluate the performance of each feature based on the four features, the optimal features are derived by classifying them into a total of three features. Of all the defined characteristics, entropy is the most important feature of the proposed method; thus, it is considered in all experiments.

The first classification of features consisted of entropy and file type; the second classification consisted of entropy, file type, and file size; and the third classification consisted of entropy, file type, file size, and date data. Using a dataset of six based on three classifications, as shown in Table 3, an experiment was conducted to enable the comparison of a total of 18 features and dataset composition.

Table 3. Experimental design based on the derived 19 features and datasets.

Exp.	Features	Dataset
1	Entropy, file type	1. Plaintext, ciphertext 2. Plaintext, ciphertext, base64, optimal encoding 3. Plaintext, base64, optimal encoding, ciphertext, system file 4. Plaintext, ciphertext, system file 5. Plaintext, base64, system file 6. plaintext, optimal encoding, system file

Table 3. *Cont.*

Exp.	Features	Dataset
2	Entropy, file type, file size	1. Plaintext, ciphertext 2. Plaintext, ciphertext, base64, optimal encoding 3. Plaintext, base64, optimal encoding, ciphertext, system file 4. Plaintext, ciphertext, system file 5. Plaintext, base64, system file 6. plaintext, optimal encoding, system file
3	Entropy, file type, file size, date data	1. Plaintext, ciphertext 2. Plaintext, ciphertext, base64, optimal encoding 3. Plaintext, base64, optimal encoding, ciphertext, system file 4. Plaintext, ciphertext, system file 5. Plaintext, base64, system file 6. plaintext, optimal encoding, system file

3.3.2. Dataset Feature and Validation

In this article, a dataset was constructed using scikit-learn, a representative machine learning library, to efficiently learn data classification models and classify them into training datasets, verification datasets, and training test sets. In machine learning, dataset verification is essentially required to prevent overfitting and underfitting of the generated dataset. Therefore, each classified dataset was verified based on each score for the learning dataset, verification dataset, and test dataset. The verification results are shown in Figures 5 and 6.

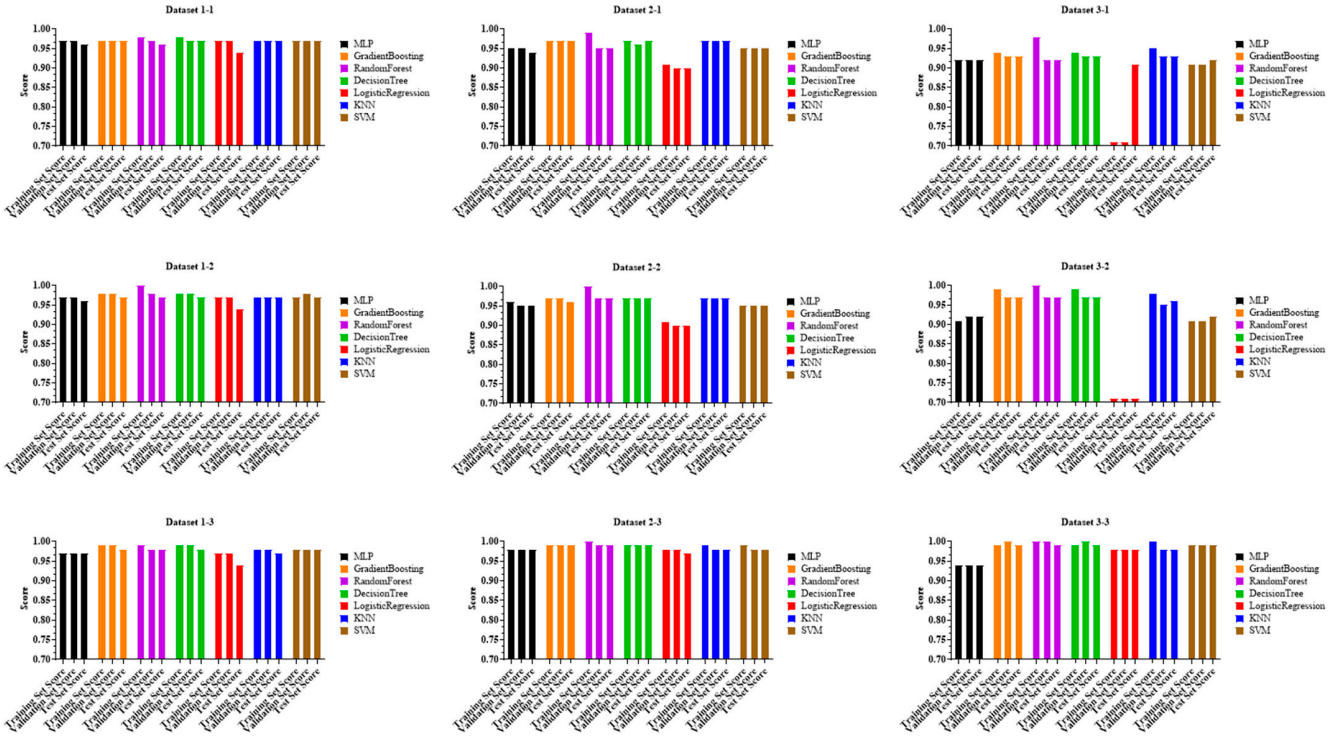

Figure 5. Validation results for Datasets 1 through 3.

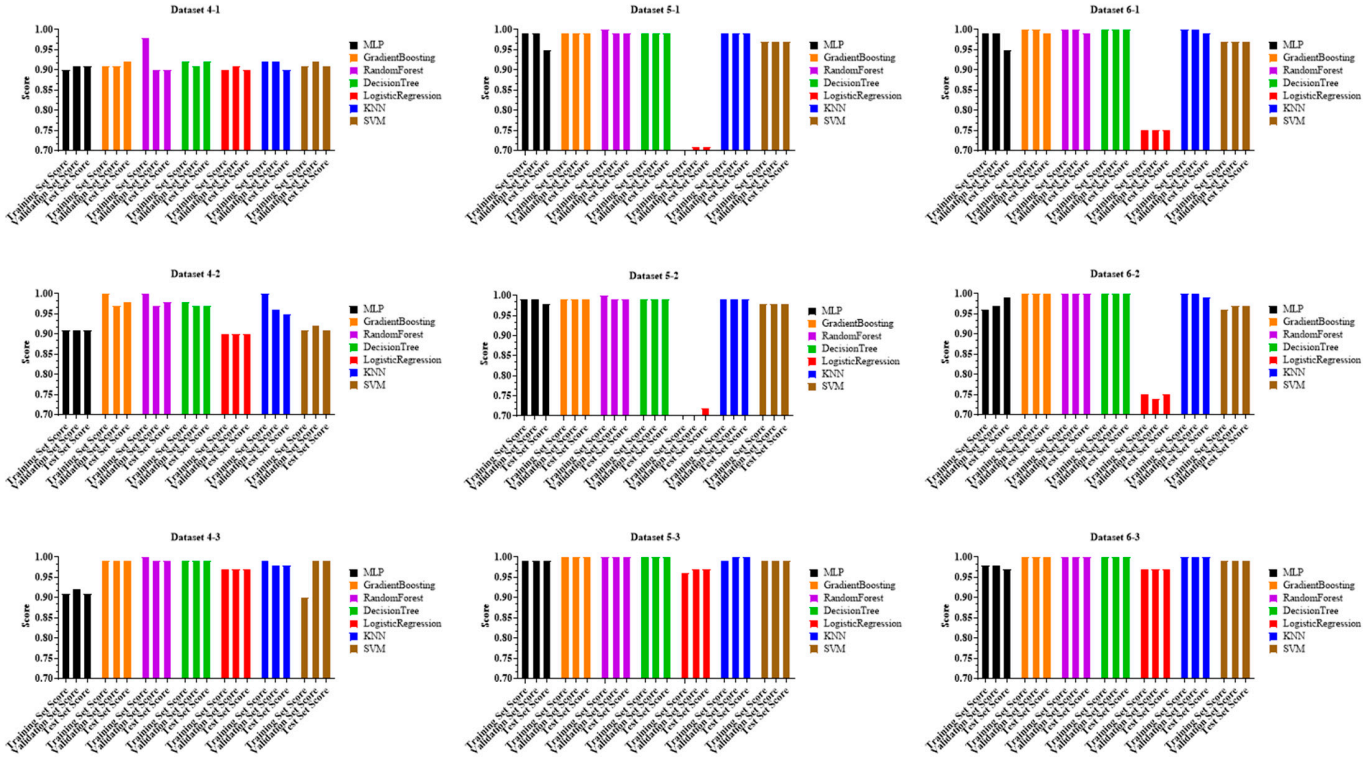

Figure 6. Validation results for Datasets 4 through 6.

Observing Figures 5 and 6, based on the six datasets we constructed and the three feature classifications defined in Section 3.3.1, the training data score, verification data score, and test data score for each machine learning model for 18 features and datasets are shown. The machine learning models used to verify the dataset are, in order, MLP, gradient boosting, random forest, decision tree, logistic regression, KNN, and SVM, and colors are expressed differently to classify performance by model.

Specifically, Dataset 1 demonstrated the lowest logistic regression model test score across all feature classifications compared to other models, and most of the remaining learning models showed similar overall performance. Dataset 2 showed the highest score in the 2-3 experiment in most models. Compared to all feature classifications, the second feature classification showed the lowest performance, and, in particular, the logistic regression showed the lowest. As for Dataset 3, the 3-3 experiment showed the highest performance, while the second feature classification showed the lowest performance. Logistic regression showed the lowest performance. Dataset 4 showed high performance in most models in experiment 4-3 compared to all feature classifications. On the other hand, the second feature classification showed the lowest performance, especially MLP and logistic regression. Dataset 5 showed the highest performance in most models compared to all feature classifications, while experiments 5-1 and 5-2 showed the lowest performance of logistic regression, at about 70% compared to all datasets. Finally, compared to all feature classifications, experiment 6-3 showed the highest performance on Dataset 6, while experiments 6-1 and 6-2 showed relatively low logistic regression.

To summarize the results, Dataset 1, Dataset 2, and Dataset 4 showed relatively high performance in most models. Putting together the results of the three feature classifications, it was found that for most datasets, the performance for the third feature classification, which included all features such as entropy, date data, file size, and file type was the highest. Although the performance varied slightly by dataset and feature classification, all datasets are considered suitable for machine learning for ransomware classification, except for the lowest-performing logistic regression model.

3.4. Optimal Parameters Derived According to the Model

In this article, to evaluate the performance of the machine learning model, hyperparameters for optimal machine learning were set based on a total of 18 experiments. If hyperparameters are described for each machine learning model, the KNN model is the number of neighbors (n_neighbors) to search, the logistic regression model is the C(class_weight) value, the decision tree is the maximum depth (max_step) value, the random forest is the number of decision trees (n_estimators) to generate, the GradientBoosting model is the maximum depth (max_step) value and learning rate (learning_rate), the MLP model is the maximum number of repetitions (max_iter) and alpha values, and the SVM model is the C(cost) value. To generate a learning model with optimal performance, the most efficient hyperparameter was derived by repeatedly substituting and applying the parameters of a certain section for each model, and the hyperparameters for the derived Dataset 1 are shown in Table 4.

Table 4. Optimal hyperparameters derived based on Dataset 1.

Dataset	Model	Hyper Parameters
DATASET 1-1	KNN	n_neighbors: 8
	LogisticRegression	C: 0.01, penalty: l2
	DecisionTree	max_depth: 7
	RandomForest	n_estimators: 2
	GradientBoosting	max_depth: 1, learning_rate: 1
	MLP	max_iter: 00, alpha: 0.0001
	SVM	C: 1
DATASET 1-2	KNN	n_neighbors: 10
	LogisticRegression	C: 0.01, penalty: l2
	DecisionTree	max_depth: 9
	RandomForest	n_estimators: 12
	GradientBoosting	max_depth:8, learning_rate: 0.001
	MLP	max_iter: 10, alpha: 1×10^{-5}
	SVM	C': 1,000,000
DATASET 1-3	KNN	n_neighbors: 2
	LogisticRegression	C: 0.01, penalty: l2
	DecisionTree	max_depth: 4
	RandomForest	n_estimators: 2
	GradientBoosting	max_depth: 1, learning_rate: 0.1
	MLP	max_iter: 10, alpha: 0.0001
	SVM	C': 1,000,000

The table shows the hyperparameter values obtained from Dataset 1. It was found that the optimal hyperparameter values were derived differently depending on the machine learning model and the characteristics of the dataset. In this article, optimal hyperparameters were derived for the entire dataset, including Dataset 1, and the performance was evaluated by applying the derived parameters to each model.

4. Experimental Results

This chapter describes the experimental performance results of ransomware detection using a machine learning model trained on the features defined in this article and the dataset constructed in this article for the proposed machine learning-based ransomware detection. To evaluate the performance from various perspectives, performance rating by machine learning models, performance rating by characteristics, and performance by datasets are evaluated and the results are described. In addition, the comparison and performance evaluation results of the proposed method are described to compare the accuracy of ransomware detection with the previous study, "Research on neutralizing ransomware detection using base64 encoding technique", which is the target of performance comparison in this article.

4.1. Experimental Results of Ransomware Detection Using the Proposed Method

As described above, this section presents the experimental results of the proposed machine learning-based ransomware detection techniques. Specifically, to explain the performance indicators used in performance evaluation and to compare and analyze performance according to the features defined in this article, we describe the performance evaluation results according to the features. Finally, considering the data characteristics in a detection situation after the system is actually infected with ransomware, the performance evaluation results according to the configured datasets are described for the purpose of deploying a ransomware classification model with the optimal detection rate for each dataset depending on the situation.

4.1.1. Performance Evaluation Metrics Using Machine Learning Models

The models used for machine learning of the proposed method are KNN, logistic regression, decision tree, random forest, gradient boosting, SVM, and MLP. In this section, to evaluate the ransomware detection performance according to each model, a confusion matrix was used to evaluate the model performance, and the classification performance according to the machine learning model was evaluated based on accuracy, precision, recall, F1-score, and AUC [25]. Table 5 shows the confusion matrix for evaluating the proposed classification model.

Table 5. Confusion matrix for evaluating the classification model.

Classification	Description
True positive (TP)	Applies encryption and accurately classifies files infected with ransomware
True negative (TN)	Accurate classification of plain text and system files
False positive (FP)	Incorrect classification of plain-text and system files as ransomware-infected files
False negative (FN)	Incorrect classification of encoded ransomware files as plaintext and system files

As mentioned above, this article uses performance indicators such as accuracy, precision, recall, F1-score, and AUC to evaluate classification performance. Accuracy is provided by Equation (1) and refers to the percentage of correct classification of true and false in the total data. Precision is provided by Equation (2) and refers to the ratio of files infected with ransomware among those classified as infected with ransomware. The reproduction rate is expressed as Equation (3) and refers to the ratio of files classified as infected with ransomware according to the classification model to files actually infected with ransomware. The F1-score is expressed as Equation (4) and is the harmonic mean of precision and recall. Finally, AUC is the area value under the ROC (receivers operating characteristic) curve, and the closer it is to one, the better the classification model.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$\text{F1} - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

4.1.2. Performance Evaluation Results by Feature

This section describes the performance evaluation results of three experiments constructed based on the four defined characteristics. In each experiment (Exp 1–3), the

classification performance was evaluated based on a dataset configured according to four characteristics. Based on the evaluation result, features with optimal performance can be derived. To evaluate the performance, accuracy, precision, recall, F1-score, and AUC were compared and evaluated for six datasets. The performance evaluation results for each experiment are shown in Figure 7 (Datasets 1–3) and Figure 8 (Datasets 4–6).

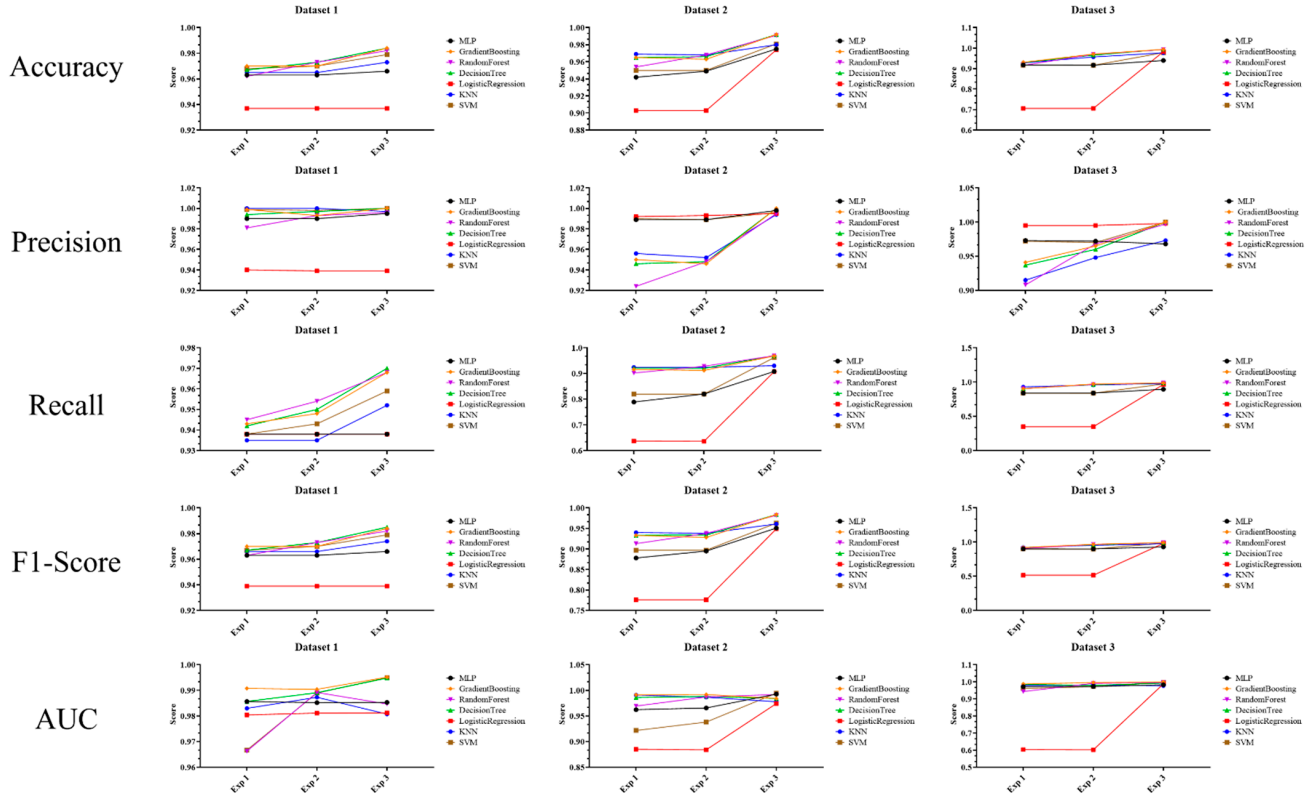

Figure 7. Performance evaluation results for each feature (Datasets 1–3).

According to the performance evaluation results by characteristics, Experiment 3 showed the highest performance in Dataset 1 in most models, whereas the logistic regression model showed the lowest performance in all performance indicators except the reproduction rate. In Dataset 2, Experiment 3 showed the highest performance in all models, whereas the logistic regression model showed the lowest performance in all performance indicators except precision. However, in Experiment 3, which showed the highest performance, the logistic regression model also showed relatively high performance. Regarding Dataset 3, Experiment 3 showed the highest performance in most models, whereas the logistic regression model showed the lowest performance in all performance indicators except precision. However, similar to Dataset 2, in Experiment 3, which showed the highest performance, the logistic regression model also showed relatively high performance. Dataset 4 showed the highest performance in Experiment 3 in most models, whereas the MLP and logistic regression models showed the lowest performance in performance indicators, excluding AUC and precision. In Dataset 5, Experiment 3 showed the highest performance in most models, while the logistic regression model showed the lowest performance in most performance indicators. In particular, the reproduction rate of the logistic regression model decreased sharply in Experiment 3 compared to Experiments 1 and 2. Finally, in Dataset 6, Experiment 3 showed the highest performance in most models, whereas the logistic regression model showed the lowest performance in all performance indicators except recall. In particular, the reproduction rate of the logistic regression was significantly lower in Experiment 3 than in Experiments 1 and 2.

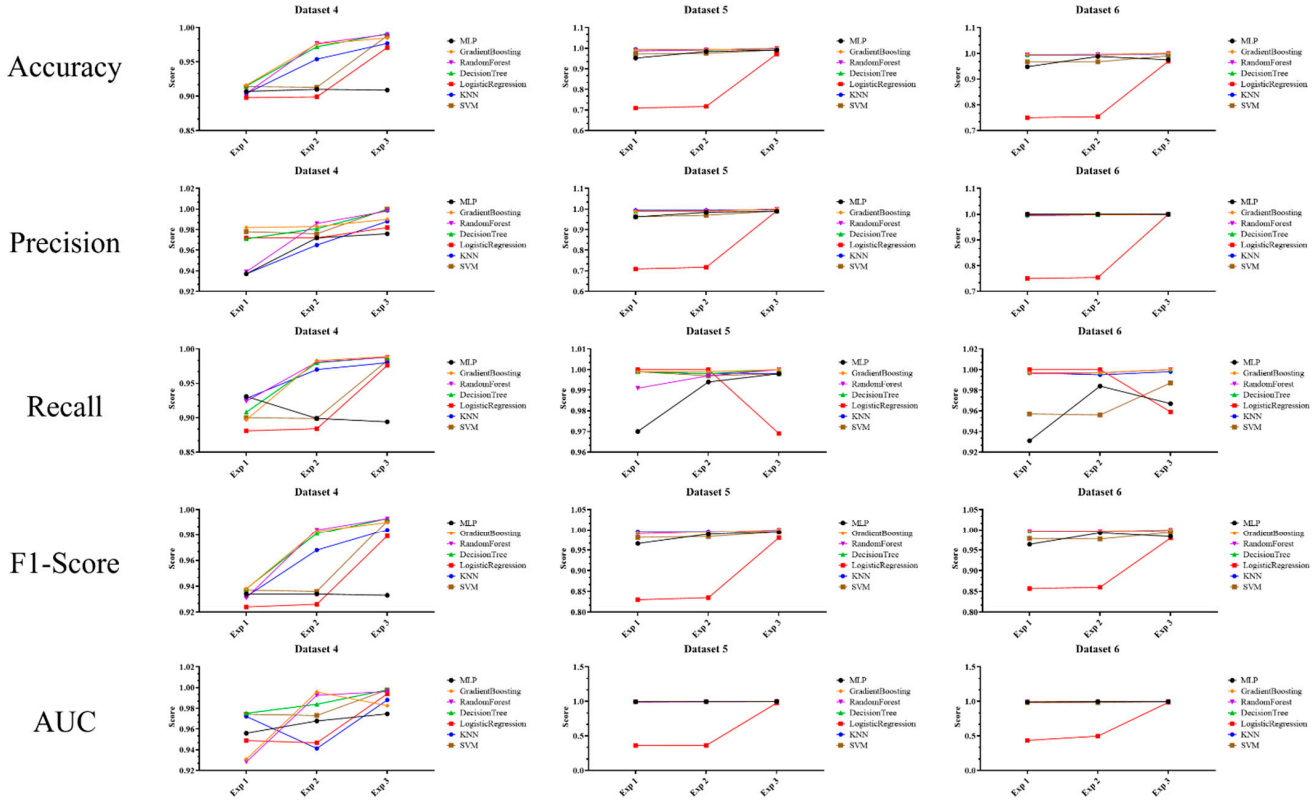

Figure 8. Performance evaluation results for each feature (Datasets 4–6).

By collating the experimental results in detail, it was found that Experiment 3 showed the highest performance in most learning models and performance indicators, followed by Experiment 2 and Experiment 1. This result is a feature of ransomware classification using machine learning proposed in this article, and when the defined features are entropy, file type, file size, and date data, it achieves the highest performance. Moreover, it is more important to define file size together rather than simply entropy and file type characteristics.

4.1.3. Performance Evaluation Results Based on the Dataset

This section, similar to Section 4.1.2, describes the performance evaluation results for each dataset, not the performance evaluation for the features. To evaluate the performance based on the dataset, in each experiment (Exp 1–3), the entire dataset (Datasets 1–6) was tested, and, as shown in Figure 9, the performance in terms of accuracy, precision, recall, F1-score, and AUC was derived.

According to the experimental results, Experiment 1 showed the highest performance in Dataset 4 and Dataset 5, while the logistic regression model showed the lowest performance, on average, in all performance indicators. Logistic regression showed a sharp drop in performance in Dataset 5 and Dataset 6 in all performance indicators. In Experiment 2, most models showed the highest performance in Dataset 4 and Dataset 5, while the logistic regression model showed the lowest performance, on average, across all performance indicators. The performance of Dataset 5 in precision was found to be approximately 20% lower compared to the other datasets. Finally, in Experiment 3, most models performed better on average than other datasets in Dataset 5, whereas the MLP model performed the lowest of all performance indicators in Dataset 4.

To summarize the experimental results, Experiments 1 and 2 showed the best performance in Datasets 4 and 5, and Experiment 3 showed the best performance in Dataset 5. These results were found to have relatively high performance compared to other datasets when plain text, cryptogram, and system files are used as datasets (Dataset 4) or when plain-text, base64 encoding, and system files are used as datasets (Dataset 5). However,

except in certain cases where performance decreases rapidly, the performance indicators across all datasets are rated as excellent, on average, and it is believed that a more efficient classification model will be created if it is used to learn ransomware classification using the appropriate dataset for the user's system situation. In other words, the machine learning-based ransomware detection method proposed in this article shows that most machine learning models detect ransomware-infected files with very high accuracy by evaluating the ransomware detection performance based on various datasets. In addition, ransomware detection is expected to be more effective if a machine learning model that ensures optimal performance for each dataset is seamlessly applied considering the data available when actual ransomware is detected.

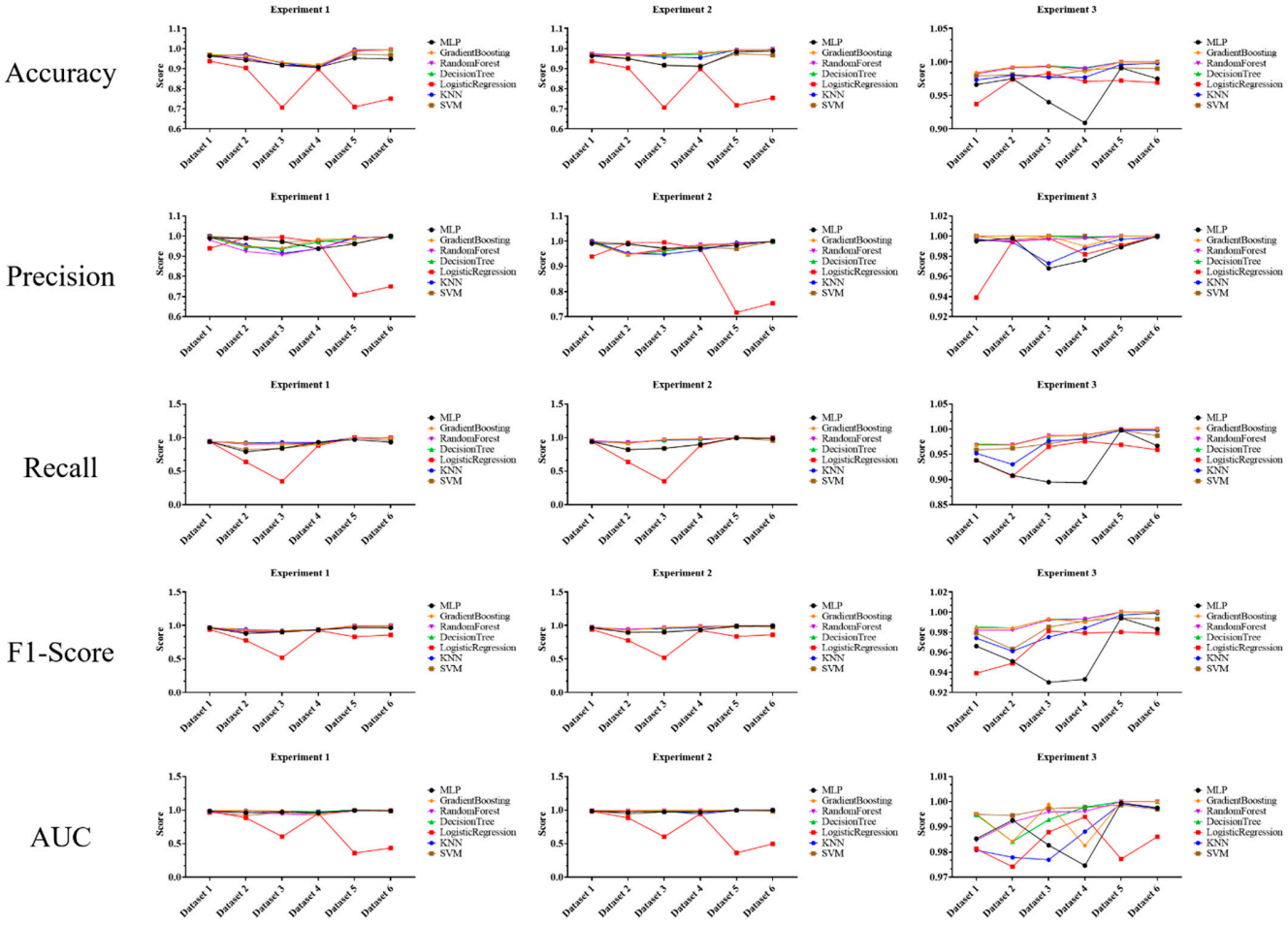

Figure 9. Performance evaluation results based on the dataset.

4.2. Performance Comparison Result with Ransomware Neutralization Techniques

In this article, we describe the effects of comparing the results of a previous study on the poor performance of ransomware detection technology based on the file entropy measurement detection method with the machine learning-based ransomware detection method proposed in this article. In the neutralization technology study, neutralization accuracy was obtained by calculating the difference between entropy and plain-text entropy when encoding techniques were applied to neutralization detection methods for various file formats. In other words, if the difference between the entropy of the plain text and the entropy after encoding is equal to or less than the set threshold, it means that it is possible to neutralize the ransomware detection method based on file entropy measurement.

In addition, in the previous studies that are the subject of performance comparison, the entropy thresholds were set to 1.0, 1.5, and 2.0, and in this study, the neutralization accuracy for each file format was derived according to the same threshold. However,

since the performance was compared based on the ransomware detection accuracy, the performance was compared by converting the neutralization accuracy derived from the study into detection accuracy. In addition, since the method proposed in this article does not determine the ransomware detection accuracy by file format, it was compared with the average detection accuracy by file format for performance comparison and evaluation, and the results are shown in Table 6.

Table 6. Comparison of ransomware detection accuracy between the proposed approach and prior research on neutralization techniques.

File Format	1.0		1.5		2.0		Proposed Method
	base64	Optimal Encoding	base64	Optimal Encoding	base64	Optimal Encoding	
CSV	90%	6%	34%	0%	19%	0%	98%
TXT	93%	12%	67%	2%	57%	0%	
DLL	34%	34%	25%	25%	15%	15%	
SYS	23%	32%	20%	21%	1%	2%	
DOC	76%	52%	59%	40%	36%	24%	
DOCX	100%	81%	72%	25%	0%	0%	
PDF	88%	100%	75%	97%	0%	94%	
PPT	79%	79%	53%	53%	7%	7%	
PPTX	97%	97%	97%	72%	0%	0%	
XLS	82%	31%	64%	19%	39%	7%	
XLSX	84%	68%	59%	30%	0%	0%	
HTML	20%	1%	4%	0%	1%	0%	
C	16%	0%	1%	0%	0%	0%	
CPP	40%	2%	6%	2%	2%	0%	
JPEG	98%	92%	91%	66%	0%	0%	
ZIP	100%	100%	100%	100%	80%	20%	
Average	70%	49%	52%	35%	16%	11%	

In the table, the average ransomware detection accuracy and detection accuracy of the proposed method are shown for the proposed method and previous studies on ransomware neutralization research. For previous studies, the main goal was to prevent detection rather than detect ransomware; thus, as mentioned above, the average ransomware detection accuracy of previous studies (by file format) was derived by conversion.

As a result of the performance evaluation, it can be observed that when the entropy threshold is 1.0, the base64 study has an average accuracy of 70%, and the optimal encoding study has an accuracy of 49%. When the threshold is 1.5, the base64 study has an average accuracy of 52%, and the optimal encoding study has an accuracy of 35%. Finally, when the threshold is 2.0, the base64 study has an average accuracy of 16%, and the optimal encoding study has an accuracy of 11%. On the other hand, the average accuracy of all machine learning models of all metrics datasets proposed in this article was 98%.

To summarize the results, previous studies that applied the neutralizing method showed that the lower the entropy threshold, the higher the ransomware detection accuracy, and for all thresholds, the base64 encoding study had relatively higher accuracy than the optimal encoding study. The higher the entropy threshold, the greater the probability of neutralizing ransomware detection from the attacker's point of view, and the optimal encoding method neutralizes it more effectively than base64 encoding. Therefore, the lower the entropy threshold (when base64 encoding is used), the higher the ransomware detection accuracy. Despite these results, it has been demonstrated that ransomware can be detected with a very high probability if the machine learning model proposed in this article is used.

5. Conclusions

To prevent the massive damage caused by the constant pre-emptive development of new and variant ransomware, this article proposed a way to effectively respond to

neutralization technology by analyzing prior studies that evaded ransomware detection methods based on file entropy measurement. To demonstrate the proposed method, a ransomware detection system was constructed in response to technology that neutralizes ransomware detection, and the detection system consists of "1. Data Collection, 2. Feature Definition, 3. Data Preprocessing, 4. Dataset Configuration, 5. Learning, and 6. Classification". In addition, by applying various machine learning models such as KNN, logical regression, decision tree, random forest, gradient boosting, SVM, and MLP, the detection system is expected to be able to respond more effectively to neutralizing ransomware detection technologies.

As a result of evaluating the ransomware detection performance of the proposed method, the average accuracy for six datasets was found to be approximately 98%, which means that files applying neutralizing technology or infected with ransomware were detected with a very high probability. In addition, a comparison of the ransomware detection accuracy of the proposed method with existing neutralization technologies showed that when the entropy threshold was 1.5, the efficiency was 46% higher than that in the base64 study.

In conclusion, it is believed that using the machine learning-based ransomware detection method proposed in this article will allow for the quick detection of ransomware-infected files, provide guidance for quick remediation through preliminary investigation, and provide results regarding prior research to determine ways to respond to additional neutralization technologies explored from attackers' perspectives.

Author Contributions: Conceptualization, J.L. and K.L.; methodology, J.L. and K.L.; software, J.L.; validation, J.L.; data curation, J.L. and K.L.; writing—original draft preparation, J.L. and K.L.; writing—review and editing, J.Y. and K.L.; supervision, K.L.; project administration, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was mainly (50%) supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2021R1A4A2001810). This work was also partly (50%) supported by "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) (2021RIS-002, 1345370809).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: Author Jaehyuk Lee was employed by the company Fescaro. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Everett, C. Ransomware: To pay or not to pay? *Comput. Fraud. Secur.* **2016**, *2016*, 8–12. [CrossRef]
2. Sakellariadis, J. *Behind the Rise of Ransomware*; Atlantic Council: Washington, DC, USA, 2022.
3. KISA. *Ransomware's Latest Trend Analysis and Implications*; Digital & Security Policy, KISA Insight: Naju, Republic of Korea, 2022; Volume 2. Available online: https://seed.kisa.or.kr/kisa/Board/142/detailView.do (accessed on 13 December 2023).
4. Lee, K.; Lee, S.-Y.; Yim, K. Machine learning based file entropy analysis for ransomware detection in backup systems. *IEEE Access* **2019**, *7*, 110205–110215. [CrossRef]
5. Mcintosh, T. The inadequacy of entropy-based ransomware detection. In Proceedings of the 26th Neural Information Processing, Sydney, NSW, Australia, 12–15 December 2019; pp. 181–189.
6. Lee, J.; Lee, K. A Method for Neutralizing Entropy Measurement-Based Ransomware Detection Technologies Using Encoding Algorithms. *Entropy* **2022**, *24*, 239. [CrossRef] [PubMed]
7. Lin, J. Divergence measures based on the Shannon entropy. *IEEE Trans. Inf. Theory* **1991**, *30*, 145–151. [CrossRef]
8. Davies, S.R.; Macfarlane, R.; Buchanan, W.J. Comparison of Entropy Calculation Methods for Ransomware Encrypted File Identification. *Entropy* **2022**, *24*, 1503. [CrossRef] [PubMed]
9. Lyda, R.; Hamrock, J. Using entropy analysis to find encrypted and packed malware. *IEEE Secur. Priv.* **2007**, *5*, 40–45. [CrossRef]
10. Guo, H.; Huang, S.; Huang, C.; Pan, Z.; Zhang, M.; Shi, F. File entropy signal analysis combined with wavelet decomposition for malware classification. *IEEE Access* **2020**, *8*, 158961–158971. [CrossRef]

11. Bhanot, R.; Hans, R. A Review and Comparative Analysis of Various Encryption Algorithms. *Int. J. Secur. Appl.* **2015**, *9*, 289–306. [CrossRef]
12. Jung, S.; Won, Y. Ransomware detection method based on context-aware entropy analysis. *Soft Comput.* **2018**, *22*, 6731–6740. [CrossRef]
13. The Base16, Base32, and Base64 Data Encodings. Available online: https://datatracker.ietf.org/doc/rfc4648/ (accessed on 13 December 2023).
14. Cooper, I. MPI-Style Web Services: An Investigation into the Potential of Using Web Services for MPI-Style Applications. Ph.D. Thesis, Cardiff University, Cardiff, UK, 2009.
15. Punycode: A Bootstring Encoding of Unicode for Internationalized Domain Names in Applications (IDNA). Available online: https://datatracker.ietf.org/doc/draft-ietf-idn-punycode/02/ (accessed on 13 December 2023).
16. Garfinkel, S.; Farrell, P.; Roussev, V.; Dinolt, G. Bringing science to digital forensics with standardized forensic corpora. *Digit. Investig.* **2009**, *6*, S2–S11. [CrossRef]
17. Suhardjono, S.; Handayani, P.; Sugiarto, H.; Aisyah, N.; Putra, A.S. Forensic Analysis Video Metadata Authenticity Detection Using ExifTool. *J. Innov. Res. Knowl.* **2022**, *1*, 1727–1734.
18. Gonzalez Zelaya, C.V. Towards Explaining the Effects of Data Preprocessing on Machine Learning. In Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE), Macao, China, 8–11 April 2019; pp. 2086–2090.
19. Zhang, M.-L.; Zhou, Z.-H. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognit.* **2007**, *40*, 2038–2048. [CrossRef]
20. Cheng, W.; Hüllermeier, E. Combining instance-based learning and logistic regression for multilabel classification. *Mach. Learn.* **2009**, *76*, 211–225. [CrossRef]
21. Sinclair, C.; Pierce, L.; Matzner, S. An application of machine learning to network intrusion detection. In Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99), Phoenix, AZ, USA, 6–10 December 1999; pp. 371–377.
22. Banfield, R.E.; Hall, L.O.; Bowyer, K.W.; Kegelmeyer, W.P. A Comparison of Decision Tree Ensemble Creation Techniques. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *29*, 173–180. [CrossRef] [PubMed]
23. Noble, W.S. What is a support vector machine? *Nat. Biotechnol.* **2006**, *24*, 1565–1567. [CrossRef] [PubMed]
24. Yin, C.; Zhu, Y.; Fei, J.; He, X. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access* **2017**, *5*, 21954–21961. [CrossRef]
25. Karim, A.; Azam, S.; Shanmugam, B.; Kannoorpatti, K.; Alazab, M. A Comprehensive Survey for Intelligent Spam Email Detection. *IEEE Access* **2019**, *7*, 168261–168295. [CrossRef]
26. Arnao, M.; Smutz, C.; Zollman, A.; Richardson, A.; Hutchins, E. Laika BOSS: Scalable File-Centric Malware Analysis and Intrusion Detection System. 2015. Available online: https://github.com/lmco/laikaboss (accessed on 13 December 2023).
27. File Scanning Framework. Available online: https://github.com/EmersonElectricCo/fsf (accessed on 13 December 2023).
28. Strelka. Available online: https://target.github.io/strelka/#/ (accessed on 13 December 2023).
29. Schneier, B. *Applied Cryptograph: Protocols, Algorithms and Source Code in C*, 2nd ed.; John Wiley & Sons, Inc.: Wiley, NJ, USA, 1996; p. 251, ISBN 9780471117094.