Egyptian Informatics Journal 24 (2023) 81–94

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Detection of malware in downloaded files using various machine learning models

Akshit Kamboj, Priyanshu Kumar, Amit Kumar Bairwa *, Sandeep Joshi

Manipal University Jaipur, Rajastham, India

ARTICLE INFO

Article history:
Received 13 August 2022
Revised 12 November 2022
Accepted 1 December 2022
Available online 19 December 2022

Keywords:
Cryptography
SHA 256
AES
LSB
Security

ABSTRACT

Malware has become an enormous risk in today's world. There are different kinds of malware or malicious programs found on the internet. Research shows that malware has grown exponentially over the last decade, causing substantial financial losses to various organizations. Malware is a malicious program or software that proves exceedingly harmful to the user's computer. The user's system can be affected in several ways. The proposed solution uses various machine learning techniques to detect whether a file downloaded from the internet contains malware or not. This research aims to use different machine learning algorithms to differentiate between malicious and benign files successfully. The main idea is to study different features of the downloaded file like MD5 hash, size of the Optional Header, and Load Configuration Size. Based on the analysis performed on these features, the files will be classified as malicious or non-malicious. The models are trained on these different features which enables them to learn how to classify files. The models after proper training will be compared among each other based on various criteria. This comparison is made with the help of the Validation and Test datasets. Finally, the model with the best accuracy will be selected. This process helps in identifying all those types of malware that can have a detrimental impact on the user's system after getting infected. The approach used here will be able to detect malware like Adware, Trojan, Backdoors, Unknown, Multidrop, Rbot, Spam, and Ransomware. After training and testing various machine learning models, the Random Forest Classifier was found to be the most accurate. It's accuracy went as high as 99.99% in the case of the test dataset. This was closely followed by the XGBoost model with an accuracy of 99.68%. The results of five different models have been compared with those obtained in the previous research. These include the Decision Tree Classifier (99.57% accuracy), Random Forest Classifier (99.99% accuracy), Gradient Boosting Model (99.09% accuracy), XGBoost Model (99.68% accuracy), and AdaBoost Model (98.87% accuracy). Four out of five of these models have been found to have accuracies greater than those obtained in previous research works.

1. Introduction

Over the last decade, there has been an 87% increase in malware infections and potentially unwanted programs. A significant contribution comes from downloaded files from the internet [1]. Many websites specialize in distributing vicious loads by offering them up as commodities or bundling the desired installer with new programs.

"Malware is a program that is malicious in intent. It is designed to harm the computer belonging to the victim". These so-called programs in the form of malware can cause harm to a victim's computer in many ways, including manipulating the computer's data, encrypting delicate data, or even monitoring the victim's activity without his/her consent.

The risk of computers that store most of our essential data getting infected by malware has increased exponentially [2]. The cause of these infections occurs due to files downloaded from the vast world of the internet. It takes just seconds for our systems to get compromised. Moreover, these hackers can intrude into

* Corresponding author.
E-mail address: amitbairwa@gmail.com (A.K. Bairwa).

☆ Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2022.12.002

our system to steal precious information and passwords. This impacts people of all communities, from middle-class men to private firms to government agencies [3]. All of this makes the detection of malware an essential requirement or need in the current scenario [4].

Over the past few years, the risk of financial fraud using technology has been rising and due to their large numbers some organizations and security-providing companies are forced to use automated or semi-automated analysis [5]. These are mainly analyzing the trends of fraud to develop an efficient detection system. But the time for any kind of defense system is very uncertain in the current situation which eventually harms the user.

Malware detection is scanning the system and its files to detect malware. It is effective at detecting malware because it involves advanced tools and approaches. It is pretty complex [6].

Habitually, swindlers on the internet get to know the Internet terms which are becoming fashionable as victims try to download some software, and these spreaders successfully make them land on their websites. These spreaders use many techniques for this, one of which is using SEO ways[7] to move up in the rank list of hunting results. They also spend hefty amounts on advertising their websites. These spreaders conceal malware in files that the victim will download or program malware so that if a user lands on the page, his computer will get infected [8]. These swindlers take advantage of these users' desperation to download illegal files or to get some software. Hence, by these techniques, the user gets convinced that the website is harmless, downloads files from these websites, and becomes a victim of malware attacks [9].

A classic case of a specific type of malware attack is called ransomware. As the name suggests this is a type of malware attack where these swindlers hold data hostage until a ransom amount is paid [10]. After the payment of the amount also it is unsure whether the data is going to be returned or not or whether the data has been sold to some other party.

Standard and signature-based ways to detect malware are getting more compound as all the modern malware are disposed to various covers to maintain their anonymity. This malware also improves itself periodically to escape from anti-malware systems. Over the last ten years, machine learning has seen a massive engagement in many areas, including cyber security [11] [12]. Cyber security experts firmly believe that using ML-driven anti-malware software will boost the detection of new-age malware. This will also help in the betterment of existing scanning engines. All this is evident from the past research papers on malware detection using machine learning techniques.[13] a simple solution in these kinds of situations in malware detection is to use machine learning wisely [14] [15]. This is a great way to determine whether a file contains malware or not.

In this work, the available dataset will be used to train different machine learning models. However, first, let us go through some of the most common machine learning algorithms [16]. These machine-learning algorithms can be applied to any given dataset.

1. **Linear Regression**: Linear Regression is based on supervised learning and performs regression tasks. Linear Regression models a target prediction value based on independent variables. It performs the task of predicting a variable value (y) supported by a given independent variable (x). This technique finds the relationship between input (x) and output (y) variables as per Eq. 1. The linear regression model can be represented through the linear equation: 1

$$y = a + b.x \tag{1}$$

where x: input training data y: label to data a: intercept b: coefficient of x

Linear regression aims to determine the best possible values of a and b, which would provide a best-fit line for data points and minimize the error between the predicted value and actual value. This technique is accurately depicted in Fig. 1.

2. **Logistic Regression**: Logistic Regression is one of the most common machine learning algorithms under supervised machine learning. It uses a set of independent variables for analyzing categorical dependent variables. It gives probabilistic values between 0 and 1 instead of discrete values, as shown in Fig. 2. Logistic regression is used for solving classification problems where we fit an "S" shaped logistic function and predict two maximum values between 0 and 1. Logistic regression can classify the observations using different data types and quickly determine the most influential variables for classification. The equation for Logistic Regression can be represented as shown in 2:

$$\log[\frac{y}{1-y}] = b_0 + b_1 \times x_1 + b_2 \times x_2 + \ldots + b_n \times x_n \tag{2}$$

3. **Decision Tree**: Decision Tree is a supervised machine learning technique that finds its use in classification and regression problems but is mainly preferred for solving classification problems. As clear from the name, this classifier has a tree structure, where internal nodes represent features of a dataset, the decision rules are represented by branches, and each outcome is represented by a leaf node (depicted in Fig. 3). Talking about nodes, there are two nodes Decision Node and Leaf Node. A decision node does decision-making and has multiple branches, whereas the leaf node depicts the decision output and has no further branches. The decision-making is done based on the features of a given dataset. So this algorithm is a graphical representation for getting all the possible solutions to a problem supporting given conditions. The algorithm used to make a decision tree is a CART(Classification and Regression Tree). Before diving into the CART algorithm, understanding impurity concepts, and their mathematics is essential. To measure impurity, we use Entropy and Ginni index. Entropy estimates the uncertainty in a given set. CART algorithm uses the Ginni index. Mathematically Entropy and Ginni index can be represented with the help of equations in 3 and 4 respectively.

$$-\sum_{i=1}^{\infty} p_i * log(p_i) \tag{3}$$

$$1 - \sum_{i=1}^{\infty} p_i^2 * log(p_i) \tag{4}$$

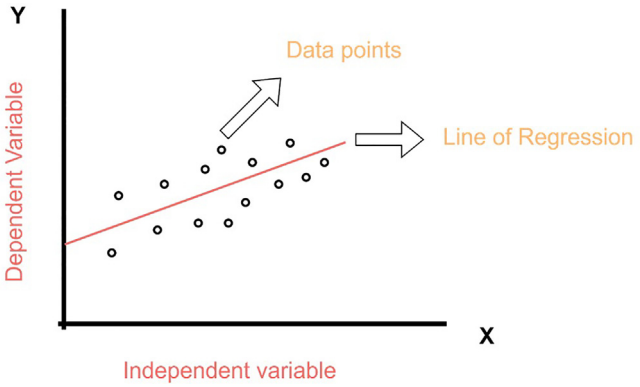

Fig. 1. Linear Regression.

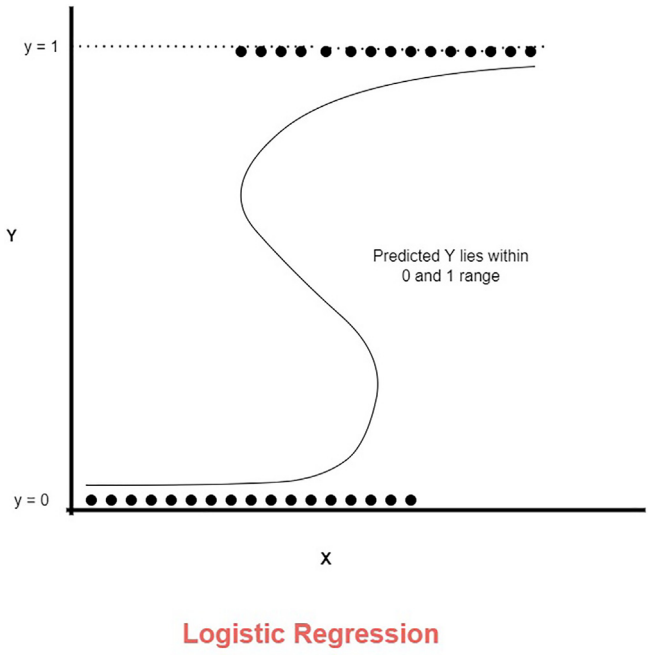

Fig. 2. Logistic Regression.

4. **Support Vector Machine (SVM)**: The algorithm is among the most popular supervised learning algorithms, which can be used for regression and classification problems. The primary use of SVM is to tackle problems related to classification. The main aim of the SVM algorithm is to create the best line or decision boundary. This line will divide an n-dimensional space into n different classes. This will ensure that new data points can be allotted the best or the closest possible category. The best decision boundary which is formed is called a hyperplane. SVM selects extreme points or vectors while creating a hyperplane. The dimension of the hyperplane also depends on the features present in the dataset. Hyperplane divides data points into two parts through a line written as shown by the two Eqs. 5 and 6.

$$y = a * x + b \tag{5}$$

$$a * x + b - y = 0 \tag{6}$$

Assuming vector X= (x,y) and W= (a,-1), so equation in hyperplane in vector form is written as in the Eq. 7.

$$W.X + b = 0 \tag{7}$$

Algorithm 1: Decision Tree Pseudocode

Input: I, where "I" is a set of classified instances.
Output: Decision Tree
Require: I is not empty, no_of_attributes is greater than 0.

```
1: Procedure Build the Tree
2:   Repeat
3:     maximum_gain = 0
4:     split_A = null
5:     e = Entropy (Attributes)
6:     for all Attributes a in S do
7:       gain = Information_Gain(a, e)
8:       if gain is greater than maximum_gain
9:       then
10:          maximum_gain = gain
11:          split_A = a
12:       end if
13:    end for
14:    Partition(I, split_A)
15:  until All partitions processed
16: End Procedure
```

5. **Random Forest**: A popular supervised learning technique in machine learning. Random Forests can be used for regression and classification problems in machine learning. This technique is based on the concept of ensemble learning. In Ensemble Learning [17] the results of multiple classifiers are used together to solve a complex problem and improve the overall accuracy of the model as defined in Fig. 4 [18]. Random Forest contains many decision trees depending on various subsets of the given dataset and then takes the average to improve predictive accuracy. More trees in the forest lead to higher accuracy and prevent overfitting. Random Forest has several advantages as it takes less training time than other algorithms. It also maintains accuracy when a large portion of data in the dataset is missing.

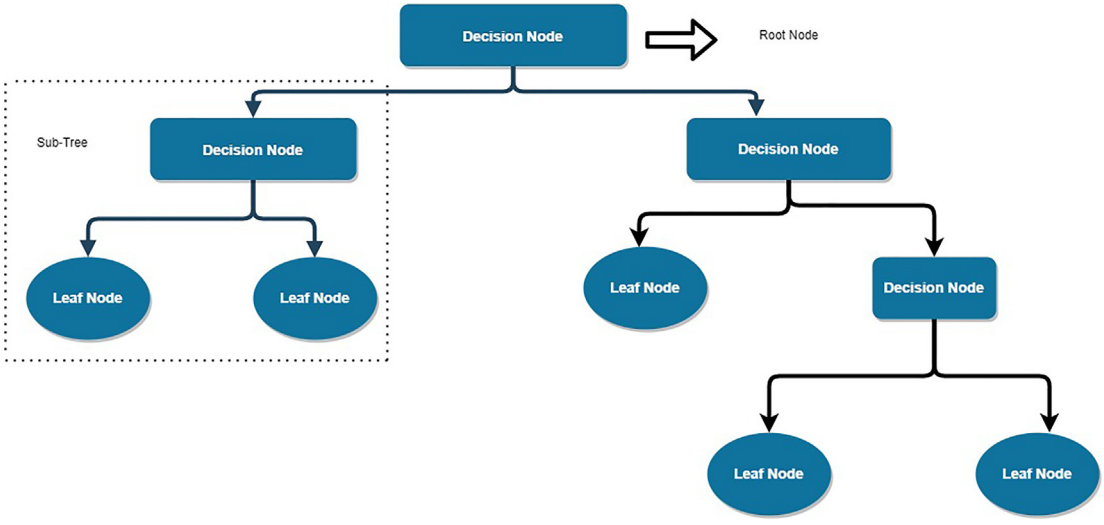

Fig. 3. Decision Tree.

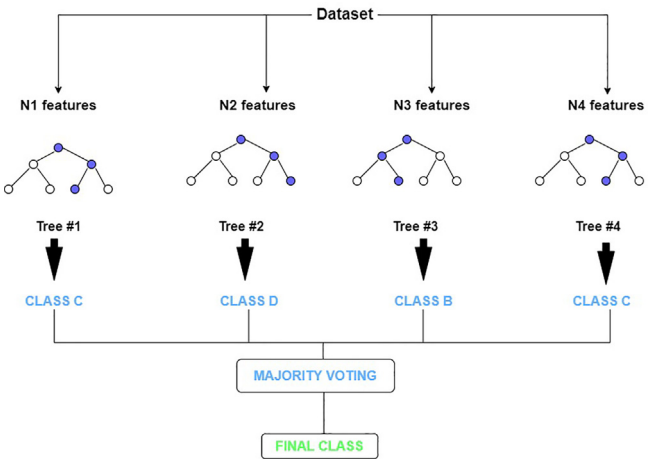

Fig. 4. Random Forest.

Algorithm 2: Random Forest Pseudocode

To generate c classifiers:
1. **for** i = 1 to c **do**
2. Randomly sample the training data TR with
3. replacement to produce TR_i. e
4. *Createarootnode*, $R_i$containing TR_i.
5. *CallBuildTree*(R_i)
6. **end for**

BuildTree Method
7. **BuildTree(R)**
8. **if** R contains instances of only one class **then**
9. **return**
10. **else**
11. Randomly select x% of the possible splitting
12. features in R.
13. Select the feature F with the highest information
14. gain to split on.
15. Create f child nodes of R, where F has f possible
16. values.
17. **for** i = 1 to f **do**
18. Set the contents of $R_i$to TR_i, where $TR_i$is all
19. instances in R that match F_i.
20. *CallBuildTree*(R_i).
21. **end for**
22. **end if**

6. **K-Means**: K-Means Clustering is a type of unsupervised machine learning algorithm used for solving clustering problems. The unlabelled dataset is grouped into several different clusters. K denotes the number of clusters required to be formed during the process. Each data point will belong to a single group [19]. All the data points within a group will have similar properties. This algorithm is centroid based, and each cluster is associated with a centroid. Minimizing the sum of distances between the data point and their corresponding clusters is the main aim of this algorithm. The best value for K-centre points or centroid is calculated with the help of an iterative process. After this, each data point is assigned to the closest K-center. An accurate representation of this technique is shown in Fig. 5.
K-means clustering targets minimizing an objective function such as squared error function 8 (See Figs. 6–16).

$$\sum_{i=1}^{k}\sum_{j=1}^{n}(||x_i - v_j||)^2 \quad (8)$$

where,

$||x_i - vj||$

represents Euclidean distance from

x_i

to

vj

which happens to be repeated over all k points among the "I" clusters, for n clusters.

Algorithm 3: K-Means Pseudocode

1. Specify the number K of clusters to assign.
2. Randomly initialize K centroids.
3. **repeat**
4. **expectation:** Assign each point to its closest
5. centroid.
6. **maximization:** Compute the new centroid (mean)
7. of each cluster.
8. **until** The centroid positions do not change.

1.1. Problem statement

"Malware Detection in downloaded files using several different Machine Learning algorithms. Comparing these models based on parameters such as accuracy, efficiency, and F-1 Score. Finally, after proper comparison, determining the best model."

1.2. Contribution of the study

The main objective of this research work is to gain valuable insights from several different research papers available at various conferences and journals. These papers will be closely related to the kind of work defined in our problem statement. This will enable us to learn about various kinds of techniques that can be used to tackle the given problem. In today's world, the threat posed by malware has increased by a tremendous amount. Malware can be used for several ill intentions, such as stealing passwords or files, rendering computers inoperable, etc. This makes it very important to minimize the occurrence of malware. Through this study, efficient Machine Learning models can be built for detecting and classifying malware.

2. Motivation

Today, most of the critical data is stored on different types of electronic devices. The risk of these devices getting infected by malware has increased significantly. To be specific there has been a 74% increase in the spread of malware in the year 2022. The most common cause of these intrusions is through the files downloaded from the internet. Detecting malware on a system can be difficult, especially when downloaded along with a file from the internet. Researchers have highlighted that these malware attacks can be associated with social, economic, cultural, or political conflicts. Malware can wreak havoc on a system. Hackers can use it to steal passwords and files and render computers inoperable. A large number of breaches are reported every year. These breaches have affected almost all industries, from government operations to small and large businesses. For instance, in the year 2021, around

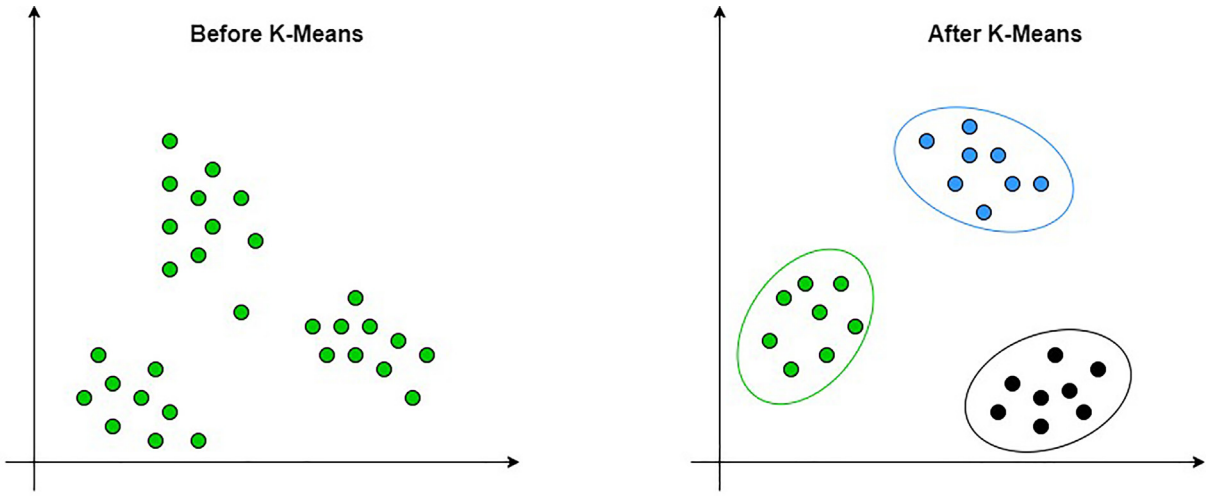

Fig. 5. K-Means.

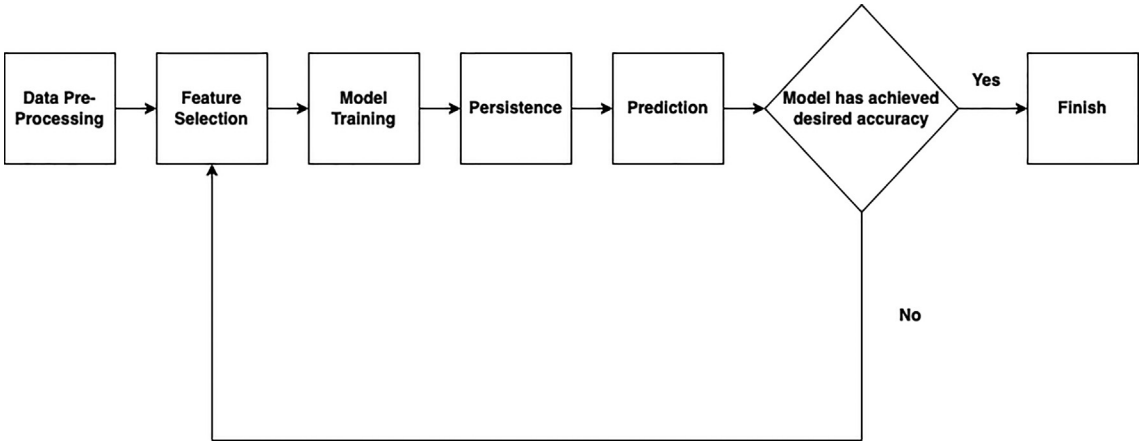

Fig. 6. Methodology Overview.

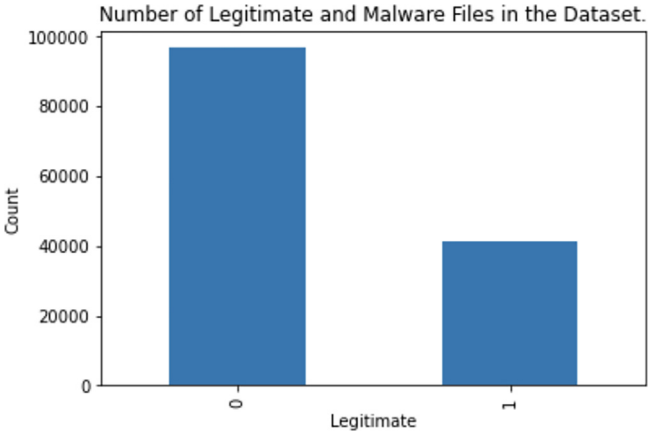

Fig. 7. Distribution of Legitimate and Malware Files.

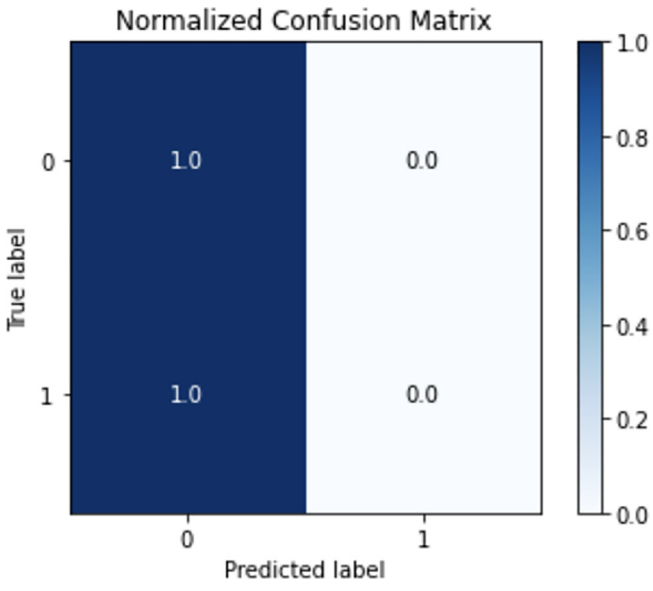

Fig. 8. Confusion Matrix: Logistic Regression.

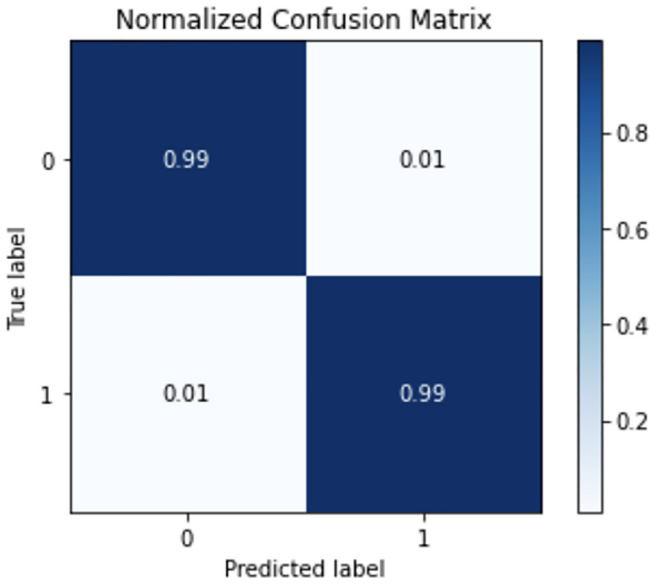

Fig. 9. Confusion Matrix: Decision Tree.

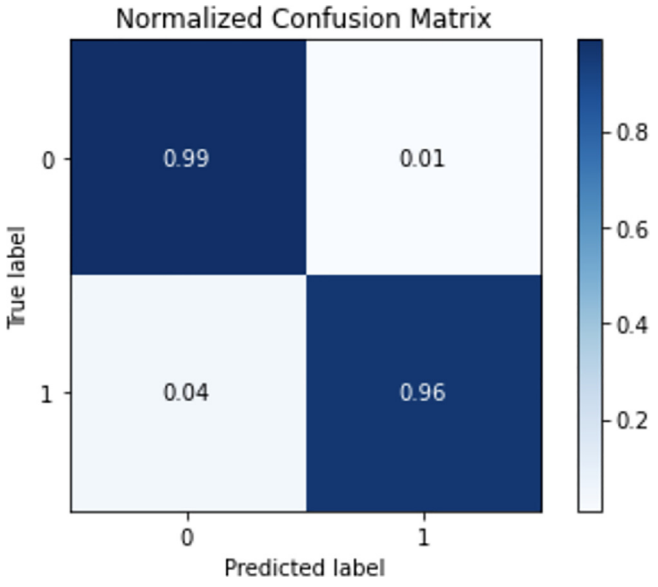

Fig. 10. Confusion Matrix: Random Forest.

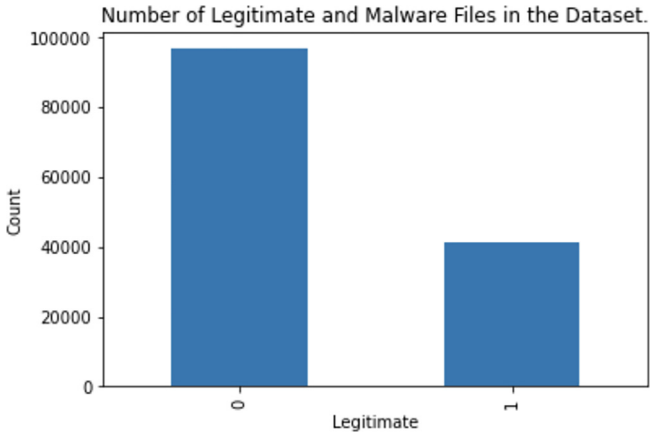

Fig. 11. Imbalanced Data.

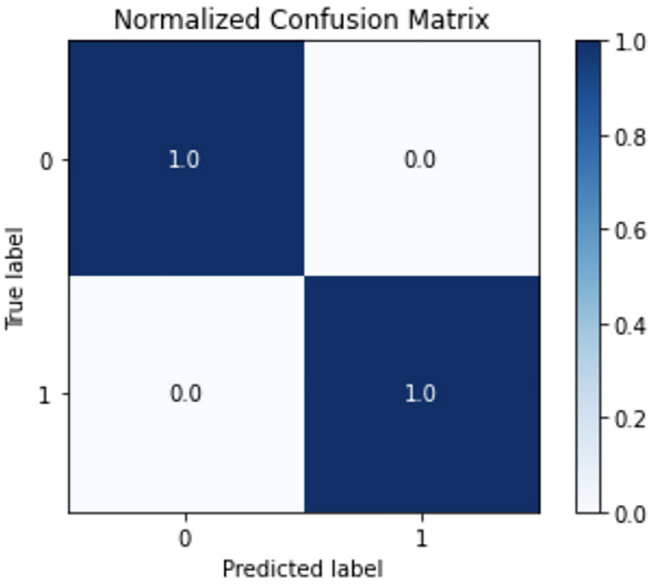

Fig. 12. Confusion Matrix: XGBoost Model.

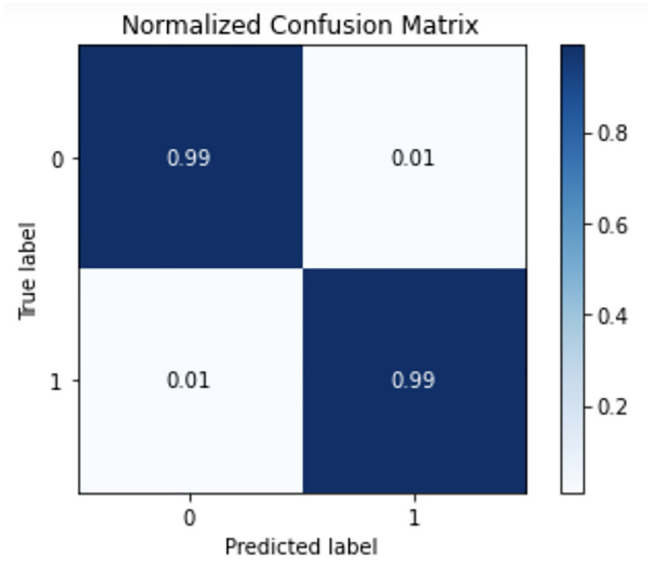

Fig. 13. Confusion Matrix: Gradient Boosting Model.

61% of organizations experienced a ransomware attack. Once malware is in action, it consumes large chunks of a system's memory. This memory loss can slow down the system, which will cause much trouble for the user. After infecting a system, malware can also spread throughout the network in which the infected system is present. It has also been observed that malware has become 57 times more destructive in 2021 as compared that in 2015. This means that malware can damage not just one employee's computer but the entire organization. It is estimated that by the year 2025, malware attacks can cost around $10.5 trillion annually.

All this highlights an urgent need for some proper methods to be developed to prevent these kinds of events or minimize their occurrence. Efficient machine learning algorithms for detecting malware can be a simple solution in these situations. For this different machine learning models can be trained on different types

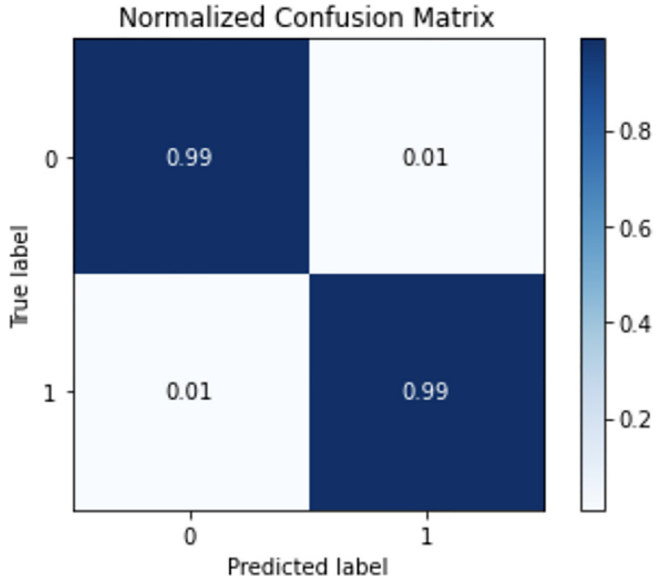

Fig. 14. Confusion Matrix: AdaBoost Classifier.

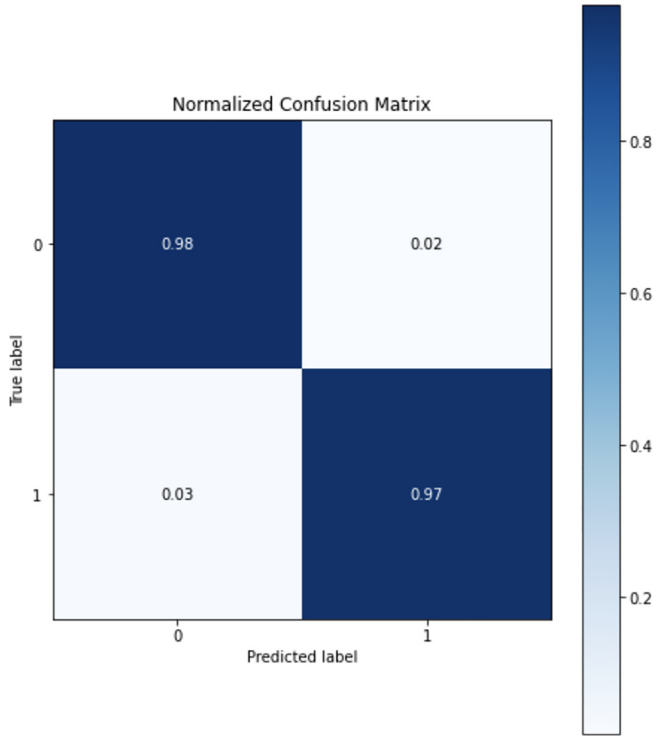

Fig. 15. Confusion Matrix: Principle Component Analysis.

of characteristics that a file possesses. This will enable the models to determine whether a file is containing malware or not.

3. Brief literature review

The main aim of the research by Jatin Acharya et al. [20] is to detect malicious URLs and programs and act upon them. Various signature matching and machine learning algorithms have been used for identification. A Random Forest Model has been trained based on the features obtained from the Executable headers of various files. The model obtained is then used for checking a given file and determining whether the file is malicious or not. In the case of URLs, a Logistic Regression Model has been trained on appropriate datasets. To perform feature extraction on the URLs, a tokenizer has been used by the regression model. The regression model is then trained on these values. After training, the model predicts whether the URL is shielded to visit or not. All these modules are combined to obtain the final application. This application will be used to protect the entire system against malicious software.

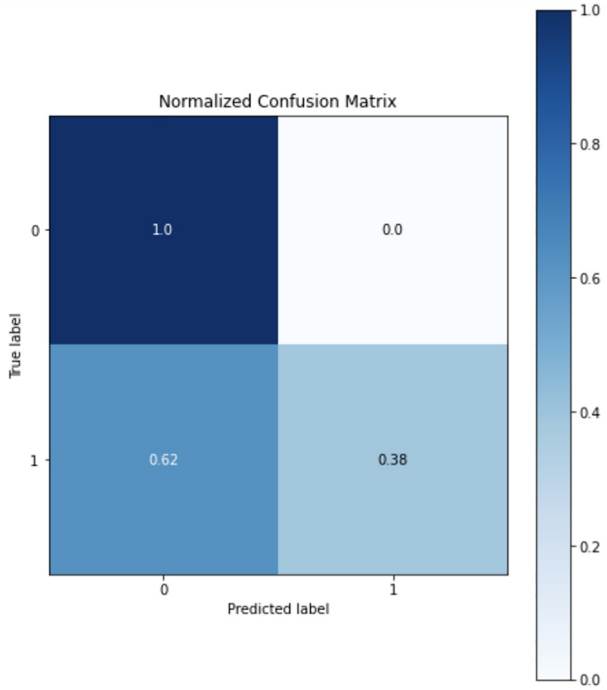

Fig. 16. Confusion Matrix: K-Means Clustering.

The research by N Udayakumar et al. [21] aims to understand different types of malware and how Machine Learning can detect and classify them. Several parameters have been used for the collection of data about various malware. These parameters are Debug Size, Export Size, Image Version, Resource Size, and Number of Sections. For the classification of malware, three Machine Learning Algorithms have been used. These are Decision Trees, Multilayer Perceptron, and Multi SVM (Support Vector Machine). In the case of the Multi SVM Algorithm, several binary SVMs have been used to ensure proper multi-class classification. Finally, the experiment has concluded that the neural networks model has shown the best performance. This model has an accuracy of about 98% on the training dataset and 99% on the test dataset. Another interpretation that has been made is that Debug Size is the essential feature and is also found to have the highest correlation.

The primary purpose of the research by Ajay Kumar et al. [22] is to apply efficient decision-making in the field of security. This decision-making will be implemented using various Machine Learning algorithms. A machine-learning-based technique is used to classify Windows PE files as malicious or non-malicious. A total of seven models were trained during the research. The models have been trained on the Brazilian Malware dataset. This dataset contains about 1,00,000 rows, with each row representing a file. There are about 57 features for each file in the dataset. Only 15 features have been selected by applying the Extra Trees Classifier to the pre-processed dataset. The models have been persistent, so they are not required to be trained repeatedly before making predictions. The Random Forest Model shows the most remarkable accu-

racy in the given dataset. Other models that have been used in the analysis are Logistic Regression, SVM (Support Vector Machine), Adaboost, Gradient Boost, XGBoost, and Decision Tree.

The main aim of the research by Sunita Choudhary et al.[23] is to detect and then classify polymeric malware. This is a type of malware that keeps on updating its key recognizable features. This makes it difficult to detect the malware using typical signature-based methods. Firstly, the behavior-based pattern of different types of malware is obtained with the help of static or dynamic analysis. After this, various types of Machine Learning techniques have been used to develop a social-based malware detection and classification model [24]. These techniques include SVM (Support Vector Machine), KNN (K-Nearest Neighbors), Naïve Bayes, J48 Decision Tree, and MLP (Multi-Layer Perceptron). The Naïve Bayes technique distinguishes or categorizes malware based on conditional probability. The Multi-Naïve Bayes technique that has utilized the 'Bytes' attribute is found to have the highest Detection Rate, whereas the Least False Positive Rate has been found in the case of the Signature Method with strings. Integrating all these techniques has led to the creation of a system that is exceptionally capable of detecting and classifying malware.

The research by Dragoş Gavriluţ et al. [25] aims to employ different machine learning algorithms and combine them to develop a versatile framework that can distinguish between malware files and clean files. In addition to classifying files, the experiment also aims to minimize the number of false positives. At first, the models were trained and tested successfully on medium-size datasets of malware and clean files. All the information obtained from this has been used to work with much larger datasets. Hence, a total of three datasets have been used, namely: training dataset, test dataset, and 'scale-up' dataset. The perceptron algorithm has been modified five times. This is done to correctly recognize malware files, setting a target of 100% detection rate for each category. Although this goal could not be achieved, an efficient machine-learning framework has been obtained to recognize most of the malware samples.

The primary purpose of the research by Sanket Agarkar et al. [26] is to develop efficient Machine Learning Models that can be used to identify malware and its family. These models have been used to develop behavior-based recognition and classification methods. Behavior-based identification of ransomware takes into consideration not only the identity of the file but also the different types of functions it will try to perform after a particular interval of time. For feature extraction from malware binary, two approaches have been used. These are Static and Dynamic Analyses. In the case of static analysis, the inspection is done without executing the malware file. The feature extraction is mainly from the PE header or dissecting executable file. In the case of dynamic analysis, the behavior of the executable file is monitored by running the file in a controlled environment. The experiment uses Machine Learning Methods: Light GBM, Decision Tree, and Random Forest. An accuracy of 99.5% has been achieved in the case of the Light GBM Classifier.

The paper by Harith Ghanim Ayoub et al. [27] aims to exhibit several works in detecting encrypted viruses. Machine learning algorithms have implemented an accurate and efficient malware detector. The researchers advanced and tested a classifier based on the Hidden Markov Model (HMM) to detect viruses and their families. This method showed high performance and accuracy, close to 90.86%. The second method included structural entropy to detect viruses which gave 74.7% and could detect unknown ransomware. Other methods were malware detection based on Opcode Frequency [28] which was implemented using different machine learning algorithms, namely Decision Tree, Boost, Random Forest, Support Vector Machine, which gave an accuracy of 80%, 86.67%, 96.67%, and 93.33% respectively.

The study aim by Sanjay Sharma et al. [29] is to study the opcode occurrence frequency for detecting unknown malware using different machine learning techniques. To fulfill this purpose, the researchers have used a dataset from the Kaggle Microsoft malware classification challenge. The dataset used for this purpose contains 21653 assembly codes representing malware. This is a combination of 9 different families. These files weighed 4000, so to maintain the proportion of malware and benign, all files whose opcode weight was under were selected. Top features are selected and compared with the help of the fisher score, symmetric uncertainty, information gain, chi-square, and gain ratio. Each feature is run through a different classifier for selecting classifiers to determine which classifier gives the best accuracy. Five classifiers, namely: Random Forest, NBT, LMT, J48 Graft, and REPTree, have also been used. After training and testing the dataset Random Forest, NBT, LMT, J48 Graft, and REPTree could detect malware with 100% accuracy.

The aim of the study by Hao Yang et al. [30] is to detect malware based on the TF-IDF model. The benefit of using NLP over other machine learning algorithms is that after processing the data, NLP gives better accuracy. In the construction of later models, the Random Forest classifier, Gradient boosting classifier, AdaBoost classifier, and ensemble models are used. Simultaneously, Convolutional Neural Network is also used for training because of its efficiency in extracting data information. The TF-IDF model is used to determine the TF-IDF value of each keyword in the traffic packet and does not carry out the splitting process on the traffic packet. The dataset has a total of 3000 data packets, including 1500 black-and-white data. Different classifiers are trained after keyword extraction and data set reconstruction from the TF-IDF model. The encrypted traffic data uses One-Hot Encoding, input to the above classifier, and CNN network training and detection. Machine learning with deep learning is a feasible solution to detect encrypted malware traffic data. With TF-IDF-based model accuracy of detection of encrypted traffic, malware has been 93%, which is bound to improve with further research and development in this area.

The research by Qing Wu et al. [31] is related to the Android Domain. Android device users and application growth has seen rapid growth in the past years; hence, the android environment faces more malware security threats [32]. To detect Malware in Android devices and applications, researchers have proposed various detection techniques, including machine learning-based methods with static features of apps as input vectors. Using complex machine learning models has been proven to be highly accurate in detecting Malware in Android applications, but their efficiency is not desirable. Some machine learning models like Logistic Regression, Naive Bayes, Support Vector Machine, Decision Tree, and Random Forest have proved their efficiency but have low accuracy. Machine learning with a Deep neural network has been proven to be more suitable for detecting Android Malware which mainly includes Deep Belief Networks, CNN neural networks, RNN neural networks, and Generative Adverbial Networks, and has achieved accuracy up to 99%.

The paper by Hemant Rathore et al. [33] aims to present the work on malware detection with various machine learning algorithms and deep learning models. For this purpose, opcode frequency has been used as a feature vector, and supervised and unsupervised learning has been used for malware classification. The researchers have worked on the problem of analysis and detection of malware as a binary classification problem where the two classes are malware and benign. The dataset for this experiment has been collected from various sources. One of them is an open-source repository known as Malaca Project for malware samples. The collection of benign executable samples has been from different files in various Windows operating systems.

Anti-virus aggregators have been used to check for malware or benign in the executable file in the operating system. The dataset contained 2,819 benign and 11,308 malware executable files to generate feature vectors. Results show that in malware detection, random forest outperforms all three deep neural network models with the highest accuracy of 99.78% with random forest and variance threshold.

The paper by Jinpei Yan et al. [34] aims to propose a unique type of malware detection method that can automatically learn the features of raw data. The name of this technique is MalNet. Malet learns about grayscale image and opcode sequences with the help of Convolutional Neural Networks (CNN) and Long-Short Term Memory (LSTM). This forms a basis for the classification of malware. Extraction of opcode sequences is done by a decompilation tool, whereas LSTM is capable of learning features about malicious code patterns and sequences. The dataset used contains samples of over 40,000, which contains 21,736 malware files and 20,650 benign files. This malware detection model goes through two stages. Finally, it achieves 99.88% accuracy, and rates for True Positive and False Positive are 99.14% and 0.1%, respectively.

3.1. Research gap

This paper also aims at overcoming some of the limitations that are being faced by the previous research works. These are as follows:

1. Only a limited number of machine learning models have been used in the previous works. In this research, a large number of machine-learning models have been trained and tested. This enables increasing the number of options that are available for classifying the downloaded files.
2. The accuracy of the classification of files has also been improved by a very large amount as compared to the accuracies obtained in the related works.
3. A much larger dataset has been used in comparison to the previous related works. This helps in representing a much broader spectrum of malicious software. This also helps in making the machine learning models more robust and efficient.
4. In the previous works, the types of malware that are being detected are limited only to a very small amount. The machine learning models in this paper can detect a much larger number of malware. These include Adware, Trojan, Backdoors, Unknown, Multidrop, Rbot, Spam, and Ransomware.

4. Methodology

4.1. System architecture

1. RAM: Minimum 8–16 Gb
2. CPU: Intel Corei5 7^{th} Generation or more is preferred.
3. Storage: Minimum 128 GB or more.
4. OS: Linux/Windows/macOS
5. Internet: High-speed internet connectivity required.

4.2. Technology used

1. Python
2. Sckit-Learn
3. Pandas
4. Matplotlib
5. Jupyter Notebook Or Google Colab
6. Basic Knowledge Of Machine Learning And Data Science

4.3. Algorithms/techniques

There are several machine learning models that can be used for classification. These models can be broadly divided into two types: Supervised and Unsupervised. Some of the supervised machine-learning models include Logistic Regression, Decision Trees, Random Forest, and Support Vector Machine (SVM). On the other hand, some of the unsupervised machine-learning models include K-Means Clustering and Principal Component Analysis.

4.3.1. Logistic regression

Logistic regression is a type of Supervised Machine Learning Algorithm. This is a statistical model that uses a logistic function to model a binary dependent variable. This is one of the most commonly used methods in cases when the target or the dependent variable is categorical.

4.3.2. Decision tree

Decision Tree is a Supervised Machine Learning Technique. This algorithm can be used for Regression as well as Classification Problems. The overall structure of this classifier is in the form of a tree. It has the following parts:

1. Internal Nodes: Features of a Dataset.
2. Branches: Decision Rules.
3. Leaf Nodes: Outcomes

4.3.3. Random forest

Random Forest is a Supervised Machine Learning Algorithm. This classifier contains a specific number of Decision Trees on different subsets of the provided dataset. The overall average of the results of the individual decision trees is used to improve the accuracy of the prediction. This concept is known as Ensemble Learning.

4.3.4. Support Vector Machine (SVM)

Support Vector Machine is a very popular Supervised Machine Learning Technique. This algorithm can be used for classification as well as regression. The main aim of this method is to form the best decision boundary or line. This line is used to divide n-dimensional space into different classes. This allows for assigning the new data point to the correct category.

4.3.5. K-Means Clustering

K-Means is a type of Unsupervised Machine Learning Technique. This algorithm is used in the case of Clustering Problems. The unlabeled dataset is grouped into several different clusters. The K represents the number of clusters required to be created during the process. Each cluster is related to a centroid. This algorithm aims at minimizing the sum of distances between data points and their corresponding clusters.

4.3.6. One Hot Encoding

One Hot Encoding is an encoding technique used to convert categorical parameters into numeric values (in the form of 0s and 1s usually). This allows categorical information about the dataset in a form compatible with the various Machine Learning Algorithms. One of the possible examples of One Hot Encoding is shown below:

The following table displays a list of colors and their respective prices:

After implementing One Hot Encoding, the following table has been obtained:

5. Implementation and result

5.1. Data preprocessing

5.1.1. Preparing the dataset

For this work, a large number of different types of datasets that are related to malicious and benign files have been explored from different parts of the internet. These datasets from various sources have been combined to obtain the final dataset. This final dataset consists only of the most relevant or essential features related to the nature of a file. The dataset has about 130000 rows (data points) and 57 columns (features).

5.1.2. Exploratory data analysis

This step involved getting utterly familiar with the dataset. Various aspects of the dataset have been studied. During this process, all the essential features or attributes present in the dataset were identified [36]. At the same time, all the irrelevant features were dropped from the dataset. Further exploration of the selected features was conducted to identify trends or patterns.

It is essential to understand the given dataset and gather as many insights as possible. This is crucial before sending the data to work with the model [37].

Two main events that took place during this process:

1. Dropping of the Name and MD5 columns (features) of the Dataset.
2. Performing Train-Test Split on the Orignal Dataset.

5.2. Result

The results that have been observed in the case of supervised machine learning models are much better as compared to those obtained in the case of unsupervised machine learning models. Among supervised machine learning models, the Random Forest Model is the most accurate. On the other hand, in the case of unsupervised machine learning models, Principal Component Analysis has been observed to achieve the highest accuracy.

5.2.1. Logistic regression

An accuracy of 70.11% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 69.85% has been achieved on the Test Dataset. The F-1 Score of the Test Dataset is 0.0.

5.2.2. Decision tree

An accuracy of 99.99% has been achieved on the Training Dataset. On the other hand, an accuracy of 99.14% has been achieved on the Test Dataset. The F-1 Score of the Test Dataset is 0.9859.

5.2.3. Random forest

An accuracy of 98.24% has been achieved so far on the Training Dataset. On the other hand, 98.28% has been achieved on the Test Dataset. The F-1 Score of the Test Dataset is 0.9713.

5.2.4. Imbalanced data

The figures that have been obtained for the Decision Tree Classifier and Random Forest Classifier are very high. There are very high chances for this to indicate some kind of anomaly like Model Overfitting. After careful examination of both the dataset and the models, it was found that the reason for these high values was the data imbalance between the two output classes. In the given dataset, the number of Benign files is almost double that of Malware files.

There are several ways to handle problems related to Imbalanced Data.

1. **Oversampling:** In this technique, we increase the number of instances related to the Minority Class using replacement. This way, the number of instances of majority and minority classes becomes almost equal. After implementing this method, an accuracy of 99.99% was achieved with the Decision Tree Classifier, in the case of the Training Dataset. In the case of the Test Dataset, an accuracy of 99.59% was achieved with the Decision Tree Classifier. Whereas with Random Forest Classifier, accuracies of 98.00% and 97.97% were observed for the Training and Test Dataset respectively. These accuracies improved after scaling the Dataset. With the Decision Tree Classifier, the accuracies were 99.99% (Training) and 99.57% (Testing). In the case of the Random Forest Classifier, the accuracies were 99.99% and 99.73% for the Training and Test Dataset respectively.
2. **Undersampling:** In this technique, we randomly delete several rows related to Majority Class. The result will be a dataset where the number of instances related to the two output classes is almost equal. After implementing this method, an accuracy of 100.00% was achieved with the Decision Tree Classifier, in the case of the Training Dataset. In the case of the Test Dataset, an accuracy of 99.23% was achieved with the Decision Tree Classifier. Whereas with Random Forest Classifier, accuracies of 98.00% and 97.90% were observed for the Training and Test Dataset respectively. These accuracies improved after scaling the Dataset. With the Decision Tree Classifier, the accuracies were 100.00% (Training) and 99.12% (Testing). In the case of the Random Forest Classifier, the accuracies were 100.00% and 99.44% for the Training and Test Dataset respectively.
3. **Synthetic Minority Oversampling Technique (SMOTE):** This technique also involves oversampling of the Minority Class. This method synthesizes new records or instances with the help of existing data. This synthesis involves the use of the K Nearest Neighbour Method. After implementing this method, an accuracy of 99.99% was achieved with the Decision Tree Classifier, in the case of the Training Dataset. In the case of the Test Dataset, an accuracy of 99.35% was achieved with the Decision Tree Classifier. Whereas with Random Forest Classifier, accuracies of 98.03% and 97.87% were observed for the Training and Test Dataset respectively. These accuracies improved after scaling the Dataset. With the Decision Tree Classifier, the accuracies were 99.99% (Training) and 99.29% (Testing). In the case of the Random Forest Classifier, the accuracies were 99.99% and 99.61% for the Training and Test Dataset respectively.
4. **Balanced Bagging Classifier:** This classifier is the same as any standard classifier, but it supports additional balancing. This technique is another step for balancing the training set during the fitting data phase. After implementing this method, an accuracy of 99.96% was achieved with the Decision Tree Classifier, in the case of the Training Dataset. In the case of the Test Dataset, an accuracy of 99.44% was achieved with the Decision Tree Classifier. Whereas with Random Forest Classifier, accuracies of 99.91% and 99.50% were observed for the Training and Test Dataset respectively.

5.2.5. Other models

1. **XGBoost Model:** An accuracy of 99.96% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 99.68% has been achieved on the Test Dataset.
2. **Gradient Boosting Model:** An accuracy of 99.10% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 99.09% has been achieved on the Test Dataset.

3. **AdaBoost Classifier:** An accuracy of 98.89% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 98.87% has been achieved on the Test Dataset.

5.2.6. Unsupervised machine learning models

1. **Principal Component Analysis:** An accuracy of 97.21% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 97.12% has been achieved on the Test Dataset.
2. **K-Means Clustering:** An accuracy of 69.03% has been achieved so far on the Training Dataset. On the other hand, an accuracy of 68.96% has been achieved on the Test Dataset.

5.2.7. Comparison with previous work [22]

The results obtained for various models have been compared with those obtained in the research by Ajay Kumar et al. [22]. Five models are common between the two works: Decision Tree Classifier, Random Forest Classifier, Gradient Boost, XGBoost, and AdaBoost. An overall improvement in the accuracies (as compared to the previous work) can be observed in the case of Gradient Boost, XGBoost, and AdaBoost Models. These comparisons can be observed from Fig. 17.

5.3. Discussion

Three different machine learning models (Logistic Regression, Decision Tree Classifier, and Random Forest Classifier) have been trained on the initially obtained dataset. The Decision Tree Classifier and Random Forest Classifier were found to have accuracies of 99.14% and 98.28% respectively. However, the initial dataset was found to be highly imbalanced. In the given dataset, the number of Benign files is almost double that of Malware files. Several different techniques have been used to obtain a balanced dataset. Some of these include Oversampling, Undersampling, Synthetic Minority Oversampling Technique (SMOTE), and Balanced Bagging Classifier. Around 5 supervised and 2 unsupervised machine learning models have been trained on this balanced dataset.

The results of five of these models have been compared with those obtained in the previous research. These include the Decision Tree Classifier (99.57% accuracy), Random Forest Classifier (99.99% accuracy), Gradient Boosting Model (99.09% accuracy), XGBoost Model (99.68% accuracy), and AdaBoost Model (98.87% accuracy). Four out of five of these models have been found to have accuracies greater than those obtained in previous research works.

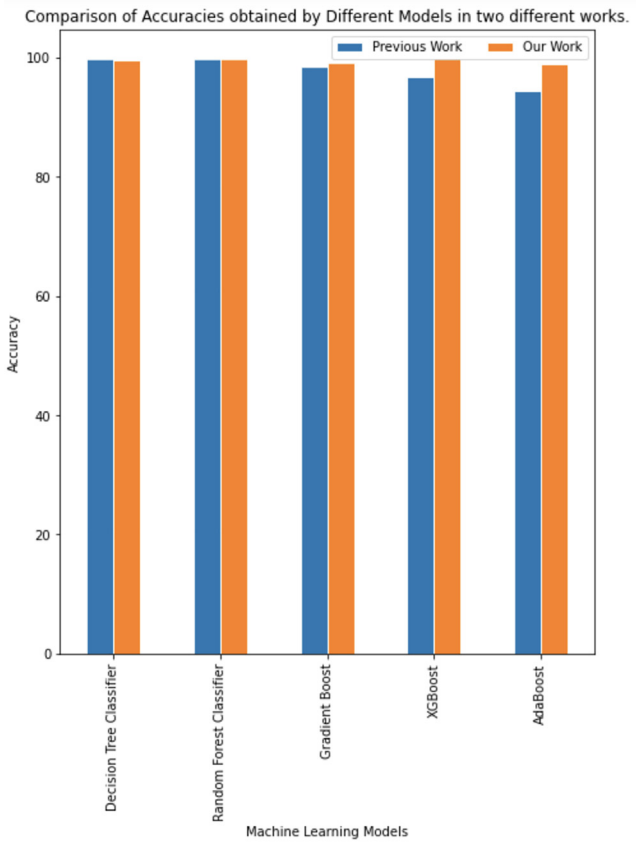

Fig. 17. Comparison Graph.

5.4. Challenges faced

The following are some of the major challenges faced during the implementation:

1. The obtained dataset had a lot of issues associated with it like a lack of uniform data formatting across a specific column, and rows with null values for various attributes. These were resolved by using several Data Preprocessing Techniques mentioned in the Section 5.1.
2. In the beginning, the predictions made by the trained models were not very accurate. This problem was tackled by making several changes in the dataset and hyperparameter tuning of the models. All of this enabled us in improving the accuracies of the models by a very large amount.
3. There were a lot of attributes (columns) in the obtained dataset which were present in the categorical form. Efficient data encoding techniques were employed to convert the entries under these attributes into numerical values so that they can be used for training the models.
4. The dataset was found to be imbalanced right in the middle of the project. As a result of this, the trained models were making predictions in favor of one class as compared to the other. This was resolved by using various dataset-balancing techniques like oversampling, undersampling, Synthetic Minority Oversampling Technique (SMOTE), and Balanced Bagging Classifier. The outcomes of these techniques have been mentioned in Section 5.2.4.

6. Tradeoffs

There are several tradeoffs for using machine learning models for malware detection as compared to standard techniques:

1. The entire process of detecting any kind of malware using machine learning is very time-consuming when compared to standard methods. This is due to the amount of time required for the machine-learning models to learn from the highly complicated malware dataset.
2. To work efficiently, machine learning models require a very large amount of data containing information about all the possible malware that might be there in the world. If these models are introduced to some new kind of malware that was not present in the training dataset, they might fail to detect that.
3. The dataset related to malware tends to be very complicated as it contains information regarding the various parameters of malware. This requires the use of a large number of resources.
4. In the case of some kind of malware, it might be possible that the standard methods can perform better in terms of accuracy as compared to machine learning techniques.

7. Conclusion

For this project, the obtained dataset was used for training several different supervised and unsupervised machine learning models. Some of the main supervised machine learning models that were used include Logistic Regression, Decision Tree, and Random Forest Model. On the other hand, Principle Component Analysis and K-Means clustering are some of the unsupervised machine learning models that were used for this project. Several dataset balancing operations were also performed on the obtained dataset. Some of these techniques include Oversampling, Undersampling, SMOTE, and Balanced Bagging Classifier.

Out of all these models, the Random Forest Model was found to be the most accurate. Its accuracy went as high as 99.99% for the test dataset. This was followed by XGBoost (99.68%), Decision Tree (99.57%), Gradient Boosting (99.09%), and AdaBoost Classifiers (98.87%). Finally, a comparison was made between the results obtained in this research and those of previous research. Four of five standard models had accuracies greater than those obtained in the previous research.

8. Future work

In the case of our project, a large number of ideas can be implemented both soon and over the long term. In the immediate future, the primary goal will be to keep trying different Machine Learning Models and determine the best model per the needs. At the same time, there is also a need to improve the prediction accuracy of each model tried. There is also a need to take care of several things like avoiding multicollinearity and

Table 1
Literature Review: Summary Table.

Author and Year	Contribution	Scope	Limitations
Jatin Acharya et al. [20] – 2021	Using Machine Learning and Signature Matching to detect Malware, Malicious URLs, and Viruses.	The application obtained from this research can be used to keep users safe from various cyber-attacks. The application also allows the detection of mutated malware or viruses that the attackers have refined.	The number of Machine Learning Models is limited to one for each main task. Also, this will not be able to handle new types of malware.
Harith Ghanim Ayoub et al. [27] – 2021	Analysing various methods for detecting the encrypted virus.	This paper demonstrates various machine learning algorithms and methods to detect encrypted malware.	Malware detection accuracy can be further improved by improving feature selection and efficiency.
Hao Yang et al. [30] – 2021	Detecting encrypted malicious traffic based on Natural Language Processing methods.	this research paper aims to detect malware based on TF-IDF (Term Frequency-Inverse Document Frequency) model.	More models could be used for detection and to improve accuracy.
Qing Wu et al. [31] – 2021	Investigating different Android Malware Static Detection Technology based on Machine Learning.	Various machine-learning models have been proposed to detect Malware in Android devices and applications.	Lack of standard benchmark datasets. More models should be implemented to improve the accuracy.
Sanket Agarkar et al. [26] – 2021	Using efficient Machine Learning Models to identify the malware and the family it belongs to.	The static features extracted from legitimate and malicious executable files can be used to train different machine learning algorithms. As a result of this, the prediction of whether a given file is malicious or not can be performed very fast and efficiently.	Several improvements must be made for this work to be applied in practical reality. A few more Machine Learning models could have been explored for detection and classification. Also, some advanced forms of polymeric malware may not be detected.
Ajay Kumar et al. [22] – 2020	Using Machine Learning Algorithms to apply efficient Decision Making in the security field.	The latest malware issues cannot be tackled with the help of traditional protection methods used by anti-virus. Machine Learning methods can be used efficiently in cybersecurity to protect against advanced malicious software.	Limited to just the detection of Malware in Windows PE files. This cannot be used to predict whether it is safe to visit a particular website or not.
Sunita Choudhary et al. [23] – 2020	To detect and classify Polymeric Malware based on the behavior-based pattern.	Polymeric malware is a type of malware that is difficult to detect with the help of typical signature-based methods. Behavior-based patterns of these kinds of malware can be obtained using static or dynamic analysis.	Several other Machine Learning models could also have been tried for classification. Some advanced forms of polymeric malware may not be detected.
Sanjay Sharma et al. [29] – 2019	Detecting Advanced Malware Using different Machine Learning Algorithms.	This paper aims to study the opcode occurrence frequency for detecting unknown malware by using different types of Machine Learning algorithms.	Maximum size of the assembly code has been set to 147.0 MB. So it will not be able to detect malware in large files.
N Udayakumar et al. [21] – 2018	Detection and Classification of Malware using different Machine Learning Techniques.	Various Machine Learning Algorithms can be used to detect and classify malware. This can act as a step toward resolving the Malware Crisis caused by different kinds of malware or malware programs.	More Machine Learning Models could have been tried for classification. Also, this will find difficult to classify unknown malware types.
Hemant Rathore et al. [33] – 2018	Presenting the work on malware detection with various machine learning algorithms and deep learning models.	Same models can also be used to detect more complex malware (polymorphic and metamorphic) in the future. Other deep learning techniques can also prove to be effective in the case of Malware Detection.	Deep learning models used were compelling enough, and there is scope for improving their accuracy.
Jinpei Yan et al. [34] – 2018	Proposal of MalNet, a malware detection system that learns features automatically from raw data using CNN and LSTM.	MalNet also solves practical problems related to the generation of Grayscale Image Generation, lengthy sequence learning, parallel computation of LSTM, and noise data processing.	Works only in case of Windows executable file.
Dragoş Gavriluţ et al. [25] – 2009	Developing a versatile framework by combining different Machine Learning Techniques to differentiate malicious and benign files.	A Machine Learning model should be such that it can detect as many malware samples as possible with a difficult-to-achieve constraint of zero false-positive rates. This can add to the standard methods of detection that anti-virus vendors use.	The methods used for malware detection are limited to perceptron. Many other Machine Learning techniques could also be used for classification.

Table 2
Pros & Cons: Logistic Regression.

PROS	CONS
One of the most straightforward methods to implement.	Poor performance in the case of Unstructured Data (such as images, audio, videos)
No need for hyperparameter tuning.	Does not work well with features that are highly correlated with each other.
Usually effective in the case of Structured Data (such as tables and lists).	Not the best algorithm in terms of power. Several better algorithms are available.
Can work with Scaled as well as Unscaled Data. Feature Scaling is not necessary.	This algorithm assumes linearity between dependent and independent variables.

Table 3
Pros & Cons: Decision Tree.

Pros	Cons
Fewer efforts are required for Data Preparation during the Data Preprocessing Stage.	Very sensitive to even the slightest changes.
Normalization of data is not required.	Can involve very complex calculations when compared to other algorithms.
Data Scaling is not necessary.	Takes more amount of time to train the model.

Table 4
Pros & Cons: Random Forest.

Pros	Cons
Can work with both Categorical and Continuous Data.	Cannot work without implementing pruning.
Low sensitivity to outliers.	Offers less accuracy for predictions as compared to other Machine Learning Techniques.
Easy to Interpret.	Susceptible to Overfitting. Several better algorithms are available.
Nature is Non-Parametric.	Involves very complex calculations, especially in the presence of many class variables.

Table 5
Pros & Cons: SVM.

PROS	CONS
Memory Efficient.	No probabilistic explanation for classification.
Very efficient in high-dimensional spaces.	Does not work well with noisy data.
It works well when a clear margin separates different classes.	Scaling with number of dimensions
Can warm-start the positions of centroids.	Unsuitable for large datasets.
It works well when the number of dimensions exceeds the number of samples.	Will does not perform well when the number of training data samples is less than the number of features for each data point.

Table 6
Pros & Cons: K-Means Clustering.

PROS	CONS
Suitable for Large Datasets.	Clusters with uniform sizes are produced even for different sizes of the input data.
Relatively simple in the case of implementation.	Not possible to find the optimal set of clusters for any given Problem [35].
Very flexible to different kinds of changes.	Much inconsistency in the results when used multiple times.
Improved Clustering Accuracy.	Need to manually specify the value of K (number of clusters).

Table 7
One Hot Encoding: Input Table.

Color	Price (in USD)
Gold	50
Red	20
Blue	25
Silver	40
Gold	50

Table 8
One Hot Encoding: Output Table.

Gold	Red	Blue	Silver	Price ($)
1	0	0	0	50
0	1	0	0	20
0	0	1	0	25
0	0	0	1	40
1	0	0	0	50

avoiding model overfitting. This is especially true in the case of Decision Tree and Random Forest Models. During these different processes, the main priority will always be to achieve the highest accuracy while ensuring that problems like multicollinearity

Table 9
Studying the Dataset.

Param.	Machine	Size Of Optional Header	Characteristics	Major Linker Version
count	138047	138047	138047	138047
mean	4259.069274	225.845632	4444.145994	8.619774
std	10880.34725	5.121399	8186.782524	4.088757
min	332	224	2	0
25%	332	224	258	8
50%	332	224	258	9
75%	332	224	8226	10
max	34404	352	49551	255

and overfitting do not occur. Several things can be implemented on a long-term basis. For instance, the ability to detect whether a given file contains malware or not can be combined with the ability to detect what kind of malware the given file contains. Similarly, the project can also implement the idea of determining the primary source of virus or malware from where the file got infected (See Tables 1–9).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Han W, Xue J, Wang Y, Huang L, Kong Z, Mao L. Maldae: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics. Comput Secur 2019;83:208–33. doi: https://doi.org/10.1016/j.cose.2019.02.007.

[2] Burnap P, French R, Turner F, Jones K. Malware classification using self organising feature maps and machine activity data. Comput Secur 2018;73:399–410. doi: https://doi.org/10.1016/j.cose.2017.11.016.

[3] Bushby A. How deception can change cyber security defences. Comput Fraud Secur 2019;2019(1):12–4. doi: https://doi.org/10.1016/S1361-3723(19)30008-9.

[4] Bairwa AK, Joshi S. Una metodolog a de b squeda de enrutamiento basada en agentes para mejorar la qos en manet, Ingeniare. Revista chilena de ingenier a 28 no.4 Arica dic. doi: 10.4067/S0718-33052020000400558. URL: https://www.ingeniare.cl/index.php?option=com_ingeniare&view=va&aid=813&vid=107&lang=es.

[5] Sharma A, Sahay SK. Evolution and detection of polymorphic and metamorphic malwares: A survey. CoRR abs/1406.7061.

[6] Royi Ronen CFEYT, Marian Radu, Ahmadi M. Microsoft malware classification challenge, CoRR abs/1802.10135.

[7] Ankalkoti P. Survey on search engine optimization tools and techniques. Imperial J Interdiscip Res 2017;3:40–3.

[8] Singh J, Singh J. A survey on machine learning-based malware detection in executable files. J Syst Architect 2021;112:. doi: https://doi.org/10.1016/j.sysarc.2020.101861101861.

[9] Elovici Y, Shabtai A, Moskovitch R, Tahan G, Glezer C. Applying machine learning techniques for detection of malicious code in network traffic. In: Hertzberg J, Beetz M, Englert R, editors. KI 2007: Advances in Artificial Intelligence. Berlin, Heidelberg: Springer, Berlin Heidelberg; 2007. p. 44–50.

[10] Akhtar Z. Malware detection and analysis: Challenges and research opportunities. ArXiv abs/2101.08429.

[11] Tchakounté F, Hayata F. Chapter 6 – supervised learning based detection of malware on android. In: Au MH, Choo K-KR, editors. Mobile Security and Privacy. Syngress, Boston; 2017. p. 101–54. doi: https://doi.org/10.1016/B978-0-12-804629-6.00006-7.

[12] Sharma A, Sahay SK. An effective approach for classification of advanced malware with high accuracy. CoRR abs/1606.06897.

[13] Schultz M, Eskin E, Zadok F, Stolfo S. Data mining methods for detection of new malicious executables. In: Proceedings 2001 IEEE Symposium on Security and Privacy. S P 2001. p. 38–49. doi: https://doi.org/10.1109/SECPRI.2001.924286.

[14] Gibert D, Mateu C, Planes J. The rise of machine learning for detection and classification of malware: research developments, trends and challenges. J Network Comput Appl 2020;153:. doi: https://doi.org/10.1016/j.jnca.2019.102526102526.

[15] Dietterich TG. Machine learning in ecosystem informatics and sustainability. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 2009. p. 8–13.

[16] Bairwa AK, Joshi S. Mutual authentication of nodes using session token with fingerprint and mac address validation. Egypt Inf J 2021;22(4):479–91. doi: https://doi.org/10.1016/j.eij.2021.03.003.

[17] Sayadi H, Patel N, SMPD, Sasan A, Rafatirad S, Homayoun H. Ensemble learning for effective run-time hardware-based malware detection: A comprehensive analysis and classification. In 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC); 2018. pp. 1–6. doi:10.1109/DAC.2018.8465828.

[18] Sayadi H, Mohammadi Makrani H, Randive O, SMPD, Rafatirad S, Homayoun H. Customized machine learning-based hardware-assisted malware detection in embedded devices. In 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE); 2018. pp. 1685–1688. doi:10.1109/TrustCom/BigDataSE.2018.00251.

[19] Gorment NZ, Selamat A, Krejcar O. A recent research on malware detection using machine learning algorithm: Current challenges and future works. In H. Badioze Zaman, A.F. Smeaton, T.K. Shih, S. Velastin, T. Terutoshi, B.N. Jørgensen, H. Aris, N. Ibrahim (Eds.), Advances in Visual Informatics. Springer International Publishing: Cham; 2021. pp. 469–481.

[20] Acharya J, Chuadhary A, Chhabria A, Jangale S. Detecting malware, malicious urls and virus using machine learning and signature matching. In: 2021 2nd International Conference for Emerging Technology (INCET). p. 1–5. doi: https://doi.org/10.1109/INCET51464.2021.9456440.

[21] Udayakumar N, Saglani VJ, Cupta AV, Subbulakshmi T. Malware classification using machine learning algorithms. In: 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI). p. 1–9. doi: https://doi.org/10.1109/ICOEI.2018.8553780.

[22] Kumar A, Abhishek K, Shah K, Patel D, Jain Y, Chheda H, Nerurkar P. Malware detection using machine learning. In: Villazón-Terrazas B, Ortiz-Rodríguez F, Tiwari SM, Shandilya SK, editors. Knowledge Graphs and Semantic Web. Cham: Springer International Publishing; 2020. p. 61–71.

[23] Choudhary S, Sharma A. Malware detection amp; classification using machine learning. In 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3); 2020. pp. 1–4. doi:10.1109/ICONC345789.2020.9117547.

[24] Naseer M, Rusdi JF, Shanono NM, Salam S, Muslim ZB, Abu NA, Abadi I. Malware detection: Issues and challenges. J Phys: Conf Ser 2021;1807(1):. doi: https://doi.org/10.1088/1742-6596/1807/1/012011012011.

[25] Gavriluț D, Cimpoeşu M, Anton D, Ciortuz L. Malware detection using machine learning. 2009 International Multiconference on Computer Science and Information Technology 2009:735–41. doi: https://doi.org/10.1109/IMCSIT.2009.5352759.

[26] Agarkar S, Ghosh S. Malware detection amp; classification using machine learning. 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC) 2020:1–6. doi: https://doi.org/10.1109/iSSSC50941.2020.9358835.

[27] Ayoub HG, Suhail AT. Review of encrypted virus: Detection analyses methods. In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), vol. 1; 2021. pp. 1946–1952. doi:10.1109/ICACCS51430.2021.9441915.

[28] Shabtai A, Moskovitch R, Feher C, Dolev S, Elovici Y. Detecting unknown malicious code by applying classification techniques on opcode patterns. Secur Inf 1(1). doi:10.1186/2190-8532-1-1.

[29] Sharma S, Rama Krishna C, Sahay SK. Detection of advanced malware by machine learning techniques. In: Ray K, Sharma TK, Rawat S, Saini RK, Bandyopadhyay A, editors. Soft Computing: Theories and Applications. Singapore: Springer Singapore; 2019. p. 333–42.

[30] Yang H, He Q, Liu Z, Zhang Q. Malicious encryption traffic detection based on nlp. In: Security and Communication Networks, vol. 2021, Article ID 9960822, 10 pages, 2021, 2021. doi: 10.1155/2021/9960822.

[31] Wu Q, Zhu X, Liu B. A survey of android malware static detection technology based on machine learning. In Mobile Information Systems, vol. 2021, Article ID 8896013, 18 pages, 2021. doi: 10.1155/2021/8896013.

[32] Joshi R, Bairwa AK, Soni V, Joshi S. Data security using multiple image steganography and hybrid data encryption techniques. In 2022 International Conference for Advancement in Technology (ICONAT), IEEE; 2022. pp. 1–7.

[33] Rathore H, Agarwal S, Sahay SK, Sewak M. Malware detection using machine learning and deep learning. In: Mondal A, Gupta H, Srivastava J, Reddy PK, Somayajulu D, editors. Big Data Analytics. Cham: Springer International Publishing; 2018. p. 402–11.

[34] Yan J, Qi Y, Rao Q. Detecting malware with an ensemble method based on deep neural network. Secur Commun Networks 2018:16. 7247095:1–7247095:16.

[35] Amit Kumar Bairwa DS, Joshi Sandeep. Dingo optimizer: A nature-inspired metaheuristic approach for engineering problems. Math Probl Eng 2021:12. doi: https://doi.org/10.1155/2021/2571863. URL: https://www.hindawi.com/journals/mpe/2021/2571863/.

[36] Ahmadi M, Ulyanov D, Semenov S, Trofimov M, Giacinto G. Novel feature extraction, selection and fusion for effective malware family classification. In: Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy. New York, NY, USA: Association for Computing Machinery; 2016. p. 183–94. doi: https://doi.org/10.1145/2857705.2857713.

[37] Blum AL, Langley P. Selection of relevant features and examples in machine learning. Artif Intell 1997;97(1):245–271, relevance. doi: 10.1016/S0004-3702(97)00063-5.