

# Malware Detection Using a Novel Ensemble Machine Learning Technique

MSc Research Project

MSc in Cyber Security (MSCCYB1)

Shivam Thakur

Student ID: x21220891

School of Computing

National College of Ireland

Supervisor: Vikas Sahni

**National College of Ireland**  
**MSc Project Submission Sheet**



**School of Computing**

**Student Name:** Shivam Thakur  
**Student ID:** X21220891  
**Programme:** MSc in Cyber Security (MSCCYB1)      **Year:** 2022-2023  
**Module:** MSc Research Project  
**Supervisor:** Vikas Sahni  
**Submission Due Date:** 14<sup>th</sup> August 2023  
**Project Title:** Malware Detection Using a Novel Ensemble Machine Learning Technique  
**Word Count:**      **6,769**      **Page Count 20**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:**                      SHIVAM THAKUR

**Date:**                              8<sup>th</sup> August 2023

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
<b>Attach a Moodle submission receipt of the online project submission,</b> to each project (including multiple copies).	<input type="checkbox"/>
<b>You must ensure that you retain a HARD COPY of the project,</b> both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

<b>Office Use Only</b>	
Signature:	
Date:	
Penalty Applied (if applicable):	

# Malware Detection Using a Novel Ensemble Machine Learning Technique

Shivam Thakur  
X21220891

## Abstract

Malware is costing billions of dollars to organizations worldwide. The first line of defence against malware is Signature-Based Malware detection. This technique while great for initial detection has limitations that it works only against those malware whose signature is in the database. Therefore, there is a need for an Artificial Intelligence (AI) and Machine Learning (ML) model that can be trained on signatures and then can predict quickly and accurately whether a new signature can be classified as a safe file or malware.

In this research a novel ensemble model is presented that uses three AI models, Naïve Bayes, K-Nearest Neighbour and Logistic Regression. The accuracy of the model in predicting malware over the used dataset was 92.69%. and a run time of 91 seconds. However, this work concludes that KNN alone is the most suitable technique.

## 1 Introduction

### 1.1 What is Malware?

Malicious software, popularly known as Malware is any kind of software that has been developed by cybercriminals or hackers in order to carry out data theft or system infect, damage, and destroy systems and networks. Malware is used in many cases for access to private data and then leverage it for monetary gain (Priya & Sofia, 2023).

Malware has kept evolving over the years and there are several different types that it can be classified into for example – Viruses, Ransomware, Spyware, Adware, Trojans, Worms, Rootkits, Keyloggers, Botnets and more.<sup>1</sup>

An example of the amount of damage malware can cause can be given with the example of the WannaCry ransomware attack. Ransomware is a type of malware that involves, encrypting the users file with the help of a strong algorithm and the only way for the user to reverse the process is by paying the attacker with money usually in the form of a cryptocurrency. WannaCry is one of the largest known ransomware attacks that occurred in 2017.

According to (Mohurle & Patil, 2017), India was one of the worst affected countries. In terms of share of damage, Madhya Pradesh was the worst affected state at 32.63% of the total share of attacks, followed by Maharashtra at 18.84 % and then Delhi at 8.76%. Additionally, WannaCry also affected companies like FedEx, Nissan, railway companies in Germany, Russian Railways, telecommunications company like Megafor Telefonica in Spain. At least 16 NHS organizations in the United Kingdom were also affected. Even in China, a lot of college systems were affected.

---

<sup>1</sup> <https://www.crowdstrike.com/cybersecurity-101/malware/types-of-malware/>

## 1.2 What are the Different Ways of Malware Detection?

Some popular techniques for malware detection today are as follows:

- **Signature-Based Malware Detection**  
Signature-Based Malware Detection makes use of the unique digital footprint or the signature of the file. The signature is compared with the file signatures stored in a large database. If the signature matches with a previously obtained malware signature, then the file is either quarantined or deleted. This is used by various antiviruses and is often the first line of defence. This technique's disadvantage is it cannot protect against newer threats unless they are added to the signature database.
- **Checksumming**  
Checksumming is like a type of signature-based malware detection technique which makes use of calculating cyclic redundancy check (CRC) checksums. Through checksumming, one can verify if a file is corrupted or not. Through checksumming, a drawback of false positives generated due to the large size of signature-based detection database is also addressed.
- **Application Whitelisting**  
Application Whitelisting is a technique opposite to blacklisting. Instead of marking applications from which requests are not to be accepted, one should whitelist those applications from which they want requests and stop every other request from coming in.
- **Machine Learning Behavioural Analysis**  
The above techniques fall under the bucket of static analysis. As attacks grow, malware evolves, the need for dynamic analysis that can predict with reasonable accuracy whether or not a given file is a malware or not only grows. This is where Machine Learning Behavioural Analysis steps in. By predicting patterns of network requests and analysing suspicious file behaviour, AI and ML can help in predicting malware. This acts as a powerful tool in malware detection however, there is a chance of false positives being obtained in many cases. <sup>2</sup>

## 1.3 Research Question

On the basis of the background study, the research question is as follows:

- How accurately can an ensemble of Artificial Intelligence and Machine Learning models detect malware?

## 1.4 Contribution and Benefits

Through the research project the aim is to find out how accurately 3 machine learning models can detect malware. This will be achieved by conducting Artificial Intelligence model training and testing over a single comprehensive dataset of malware <sup>3</sup>. Basis the accuracies obtained a final accuracy combining the results of the models will be mentioned.

Through this research the field of malware detection and AI will evolve. Instead of using a singular model to test accuracy, 3 models were used, and the final accuracy will help to identify a correct result more often than not. The aim is to also do this in a short time so that even if

---

<sup>2</sup> <https://www.cynet.com/malware/4-malware-detection-techniques-and-their-use-in-epp-and-edr/>

<sup>3</sup> <https://www.unb.ca/cic/datasets/nsl.html>

there are false positives, the process is quick. It is important that malware is detected or even a false positive is detected in a short time than the process be slow and inaccurate as well.

## 1.5 Document Structure

In the rest of the document, the related work which includes a comprehensive literature review of malware detection techniques used with AI and ML has been discussed in section two. Section three then covers the research methodology adopted for the project. This will include dataset information, the AI models and more. Section 4 will provide the design specifications that were used. After design specification, section 5 shows the implementation of the model in detail. Following the implementation, the report shows the results of the model in section 6. The last sections include the conclusions derived from the research and future scope along with the relevant references to the various other research papers that help establish this research itself.

## 2 Related Work

In this section we have covered several different literature works which cover the domain of AI and ML based Malware Detection. The objective was to understand in detail the prior works conducted and what contributions can be made through this work.

### 2.1 Malware Detection Using Machine Learning Algorithms

In the research conducted by (V & Devi K.A., 2019), it has been shown how AI and ML can be used to detect malware. Different types of malware are described like trojans, worms, viruses and spyware. Additionally, difference in different types of analysis like static, dynamic and deep learning are also introduced. The paper focusses on spyware detection as its main goal and uses the technique of extra tree classifier in tandem with XGBoost on a dataset to accurately detect spyware. Basis the specifications stated by the author and the testing conducted this algorithm helped them achieve an accuracy of malware detection of 99%.

Research done by (Naser & Abu Al-Haija, 2023), covers the effects a spyware could have on various users and devices has been conducted. Exploration of how AI and ML can be used in order to conduct behavioural analysis for detection and prevention of spyware has also been conducted. After exploring several different machine learning algorithms they performed a comparative study of Fine Decision Trees (FDT), Support Vector Machines (SVM) and Naïve Bayes. The train to test dataset ratio was kept at 4:1. The final results of their research were as follows:

**Table 1 : Summary of the Results**

<b>Algorithm</b>	<b>Accuracy</b>	<b>Precision</b>	<b>Sensitivity</b>
Fine Decision Trees	98.2%	98.3%	98.1%
Support Vector Machines	97.7%	97.5%	97.3%
Naïve Bayes	93.9%	94.1%	91.1%

Therefore, FDT had the best result for them.

In the research conducted by (Conti, et al., 2020), they focus on spyware and develop a new technique for its detection and prevention that they call ASAIN. ASAIN essentially stands for the title of their paper, A Spy App Identification System based on Network Traffic. Through this system they have compared 3 algorithms in Random Forest, K-nearest neighbours and Logistic Regression. Basis their studies they obtained the following results on F1 Scores:

**Table 2: Summary of F1 Score.**

<b>Algorithm</b>	<b>F1 Score</b>
Random Forest	0.852
K-Nearest Neighbours	0.658
Logistic Regression	0.476

Therefore, from their research Random Forest had the best result with 0.852.

In the research conducted by (Malik & Kaushal, 2016), they explore malware detection in Android operating systems by using network traffic analysis with the help of their own system called Credroid. The main algorithm that they used in their analysis was Random Forest. Their analysis involved a lot of phases in tandem with virus total for APK analysis. However, focussing on the dynamic and AI and ML aspect of it, the accuracy they obtained was 63%.

(Majdi K., et al., 2022) in their work, explore spyware detection on Android Operating Systems using a novel dataset. They used the random forest algorithm for their research purposes to come up with 2 classification methods, binary and multi class. The average accuracy obtained over multiple datasets for both methods were 79% for binary class and 77% for multi class.

(Mohapatra, et al., 2022) conduct a research where, they describe what malware is and the effects it can have on every device. They also give an overview of the different techniques available for Malware detection like static detection and dynamic detection. The scope and motive of their research was to compare a wide variety of machine learning algorithms and test which of them performs the best. The algorithms used in their research were Random Forest, XGBoost, CatBoost, LightGBM, Decision Tree, K-Nearest Neighbours, Logistic Regression and Naïve Bayes. The metric used in order to rank the algorithms was F1 score. The following table shows the results from highest F1 Score to the lowest.

**Table 3: Summary of F1 Score.**

<b>Algorithm</b>	<b>F1 Score</b>
Random Forest	0.98567
XGBoost	0.98554
CatBoost	0.98485
LightGBM	0.98134
Decision Tree	0.97546
K-Nearest Neighbours	0.96504
Logistic Regression	0.51822
Naïve Bayes	0.50025

Therefore, the result was Random Forest performed the best in this particular scenario.

In the work done by (Abdullahi, et al., 2022), they perform a comprehensive literature review comparing different works catering to malware detection using AI. They have not suggested an own model but have explored several different algorithms including but not limited to Artificial Neural Networks, Deep learning, Fuzzy C means, Fuzzy Pattern Tree, K nearest Neighbour and SVM. Through this paper the authors have conveyed that among all algorithms Boosting algorithms like XGBoost and Neural Networks have the best accuracy and may have a higher promise than other algorithms if they are trained properly. The authors go further into

depth about what algorithm can be used better for different attacks like which algorithm is most suited for DoS attacks and so on. They have not ranked any algorithm or created an own model but supports the idea that AI based malware detection will grow in the future because of its scope.

In the research conducted by (V & Devi K.A., 2019), they test their own unique algorithm that is based on Artificial Neural Networks. Once again, this paper deals with Spyware detection. The authors have not mentioned any accuracy because the algorithm is just a proposal, however they designed a truth table of 8 scenarios which can be used to check if a file is clean or a malware. The researchers claim the algorithm is viable and can be used to detect whether a file is malware or clean.

In the work done by (Fatima & Quadri, 2022), the different types of malware like viruses, trojans, worms, spyware, adware, rootkits, backdoors and ransomware are discussed. They further describe static and dynamic analysis for malware detection. In their research, they come up with a technique that makes use of Cuckoo sandbox <sup>4</sup> and Machine Learning techniques. They used the SVM algorithm for their algorithm. The process followed by their algorithm was scanning the system, start cuckoo sandbox, reading a malware dataset, data pre processing to filter out missing and null values and finally applying the SVM algorithm for prediction. The final result obtained by their algorithm was an accuracy of 95.1%.

(Mustafa Majid, et al., 2021) introduce what malware is and through it, the need of better means of malware detection with the help of AI and ML. They used the 3 algorithms of Convolutional Neural Networks (CNN), Long Short-term Memory (LSTM) and auto encoders. Their research covers malware detection on Android OS and they review how these 3 algorithms can be used in order to detect malware. Their future scope claimed that auto encoders could have the most promise as time goes by.

In the research conducted by (Rathore, et al., 2018), they describe what malware is and the need for malware detection and prevention. For the purposes of malware detection using AI, they conduct a comparative study between the models of Random Forest and Neural Networks with hidden layers. So, a total of 4 algorithms were used. Random Forest, Neural Networks with 2 hidden layer, with 4 hidden layers and 7 hidden layers. The conclusion of this paper was that the Random Forest Algorithm performs the best at 99.78%.

In the research conducted by (Al-Haija, et al., 2021), machine learning models have been compared in order to prevent against port scanning attacks. The algorithms evaluated were logistic regression, decision trees, linear/quadratic discriminant, Naïve Bayes, and ensemble boosted trees. On the basis of their evaluation the best performing algorithm was Logistic Regression was 99.4%. Additionally, the claim is this model performs than other research works.

(Aksu & Aydin, 2018), in their work write about cybercrimes and the need to protect information systems through intrusion detection systems. With the evolution in technology, they suggest using AI models for intrusion detection and protecting against port scanning attacks, For the purposes of their testing they made use of the CICIDS2017 dataset. The Machine learning models used were deep learning models like Convolutional Neural Networks (CNN) and Support Vector Machines (SVM). The conclusion was that CNN worked better with an accuracy of 97.8% and SVM had an accuracy of 69.79%.

---

<sup>4</sup> <https://cuckoosandbox.org/>

According to (Kumar & Lim, 2019), due to the widespread evolution and use of IoT (internet of Things) devices, there have been many security issues. After the Mirai botnet-based DDoS attack there have been several more malware that target IoT devices like Satori, Reaper, etc. In their research, they have suggested the use of EDIMA, a machine learning solution that could help in detection of IoT malware activity in a network. 3 machine learning models have been tested in this paper, Random Forest, KNN and Gaussian Naïve Bayes. The accuracy for each was 88.8%, 94.44% and 77.78% respectively.

In the research conducted by (Aygun & Yavuz, 2017), they state that intrusion detection systems do not perform well against zero-day attacks and that there is a need to improve their performance in this subject. They have used auto encoders and denoising encoders in their proposal for deep learning anomaly-based detection. For the evaluation of their tests the KDDCUP'99 dataset was used. According to the evaluation and results of their experiments auto-encoders had an accuracy of 88.28% and denoising auto-encoders had an accuracy of 88.65%.

(Liu, et al., 2020), in their research suggest the increase in usage of IoT devices and the amount of application security risks associated with them. They talk about several attacks that affect IoT devices. For the purposes of their evaluation, they chose 8 machine learning algorithms. The table below shows the accuracy they obtained for each algorithm.

**Table 4 : Summary of Accuracy**

<b>Algorithm</b>	<b>Accuracy</b>
Random Forest	96.6%
Decision Tree	96.6%
Bagging	96.7%
Support Vector Machines	95.7%
Naïve Bayes	45.2%
Bayes Network	88.2%
Adaboost	74.0%
XGBoost	97.0%

Therefore, as per their work XGBoost performed the best.

(Al-Haija & Al-Dala'ien, 2022), in their proposed work talk about the growth of IoT devices and the risk of the botnet attacks they face. Through this paper, they have proposed an ensemble model called as Elba-IoT that profiles the behaviour of IoT networks and uses machine learning to identify anomalies in the network traffic. The dataset used was the N-BaIoT-2021. From their experimental results they concluded that Elba-IoT can detect botnet attacks with a detection accuracy of 99.6% and a low inference overhead of 40 microseconds.

In the research conducted by (Alkahtani & Aldhyani, 2022), they speak about the rapid evolution of Android Operating System and the need to secure it. They have done experimental analysis of various AI models over 2 datasets. The algorithms used were Support Vector Machines, K-Nearest Neighbour, Linear Discriminant Analysis, Long Short-Term Memory (LSTM), Convolution Neural Network Long Short-Term Memory and auto encoders. The conclusion of these experiments was SVM had 100% accuracy on the CICAndMal2017 dataset and LSTM was the best performing algorithm on the Drebin dataset of 99.40%.



(Yan, et al., 2018) in their proposal speak about how crucial malware detection is. In their paper they have stated a new technique of MalNet that uses Convoluted Neural Networks and Long Short-Term Memory algorithms. For the purposes of their experiments, they use 40,000 samples, 20,650 benign files and 21,736 malware provided by Microsoft. Through their proposed work they conclude that MalNet has an accuracy of 99.88% in malware detection.

(Goeschel, 2016), in their work have spoken about how false positives can affect the efficiency of Intrusion Detection Systems. In order to combat this, they have come up with an ensemble of SVM, Naïve Bayes and Decision Tree algorithms. The model was run thrice to average out the overall accuracy at 99.62%.

In the research done by (Syarif & Gata, 2017), they have proposed an intrusion detection system that is based on the binary particle swarm optimization in tandem with the K-Nearest Neighbour algorithms. The final results of this experiment were that KNN and Binary PSO have the best accuracy of 99.62% when K is set to 10. KNN accuracy alone is best at 97.92% when K is set to 5.

In the work done by (Stiawan, et al., 2021), the damage that can be caused by ransomware has been stated. In their work, they have suggested a control flow graph to extract opcode and then perform a trojan ransomware detection method on it using K-Nearest Neighbour. The results of their experiment showed that the best accuracy obtained through this method was 98.86%.

(Kumar, et al., 2017), in their work reiterate the need for cybersecurity and protection against malware. They explain signature-based malware detection and how they fail against polymorphic malware. In their work, they have suggested using Logistic Regression with Anova F-Test and Snort for detecting polymorphic malware. This model achieved 97.7% accuracy.

**Table 5 : Literature Review**

<b>Sr No</b>	<b>Authors</b>	<b>Best Algorithm</b>	<b>Accuracy</b>
1	(V & Devi K.A., 2019)	Extra Tree Classifier + XGBoost	99%
2	(Naser & Abu Al-Haija, 2023)	Fine Decision Trees	98.2%
3	(Conti, et al., 2020)	Random Forest	85.2%
4	(Malik & Kaushal, 2016)	Random Forest	63%
5	(Majdi K., et al., 2022)	Random Forest	79%
6	(Mohapatra, et al., 2022)	Random Forest	98.57%
7	(Abdullahi, et al., 2022)	Multiple Algorithm Survey	N/A
8	(V & Devi K.A., 2019)	Algorithm Proposal	N/A
9	(Fatima & Quadri, 2022)	Cuckoo + SVM	95.1%
10	(Mustafa Majid, et al., 2021)	Auto Encoders	N/A
11	(Rathore, et al., 2018)	Random Forest	99.78%
12	(Al-Haija, et al., 2021)	Logistic Regression	99.4%
13	(Aksu & Aydin, 2018)	Convoluted Neural Networks	97.8%
14	(Kumar & Lim, 2019)	K-Nearest Neighbours (KNN)	94.44%
15	(Aygun & Yavuz, 2017)	Denoising Auto Encoders	88.65%
16	(Liu, et al., 2020)	XGBoost	97.0%
17	(Al-Haija & Al-Dala'ien, 2022)	Elba-IoT	99.6%
18	(Alkahtani & Aldhyani, 2022)	SVM	100%
19	(Yan, et al., 2018)	CNN + LSTM	99.88%
20	(Goeschel, 2016)	SVM, Naïve Bayes, Decision Tree	99.62%

21	(Syarif & Gata, 2017)	KNN + PSO	99.62%
22	(Stiawan, et al., 2021)	KNN	98.86%
23	(Kumar, et al., 2017)	Logistic Regression	97.7%

Therefore, on the basis of the evaluation of the above works, the models selected for the ensemble implemented in this work were Naïve Bayes, Logistic Regression and K-Nearest Neighbour.

### 3 Research Methodology

This section describes the various parts of how the research has been conducted.

#### 3.1 Dataset Identification

For the purposes of this experiment, the publicly available NSL-KDD dataset, made by the University of New Brunswick has been used. This is a 2009 version of the dataset, which acts as a great benchmark dataset in order to detect and compare different intrusion detection systems. Another benefit of using this dataset is that the evaluation results can be compared to other literature works that have been published.

#### 3.2 Dataset Processing

Data Processing means analyzing and processing made on the raw data before it is used as an input in machine learning training models or data mining models. Essentially, data processing steps help in transforming the raw data into a format that is easier to read and more efficient to perform machine learning and data mining tasks over. Sampling, transformation, normalization, denoising, imputation are some examples of techniques that are used in Dataset Processing.<sup>5</sup>

#### 3.3 Data Split into Training and Testing

Before starting the phase of Training and Testing, the dataset will need to be split into the training as well as the testing part. For the purposes of this work, 80% of the dataset has been made the training dataset and 20% of the dataset has been made the test dataset.

#### 3.4 Training Data

The machine learning models that are developed rely on sufficient data training in order to be efficient. This phase aims to achieve that. Without an efficient training data, the algorithm will fail. Training data is also called as the learning set. To put it simply, training data is what helps in building the machine learning model. It makes the model understand what the expected output is and how it looks. The model will deeply analyze the dataset and keep iterating over it a number of times to fully grasp the characteristics of the data and adjust for better performance.<sup>6</sup>

#### 3.5 Testing Data

This phase includes providing an input to the now trained model. Here, the model will now put into effect what it has learnt from the training phase and try to predict the outcome on new information. To put it simply, testing phase involves data that has not been provided to the model before so that the preparation quality can be calculated in the results.

<sup>5</sup> <https://www.techtarget.com/searchdatamanagement/definition/data-preprocessing#:~:text=What%20is%20data%20preprocessing%3F,for%20the%20data%20mining%20process.>

<sup>6</sup> <https://learn.g2.com/training-data>

### 3.6 Evaluation Parameters

For the purposes of this research, the standard parameters that are used to evaluate a machine learning model like accuracy, precision, recall and F1 score. The mathematical formulas for these parameters are as follows:

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

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

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

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where,

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

### 3.7 Summary of the model

Once all the models are ready the final step involves clubbing the results and averaging it out to obtain the accuracy, precision, recall and f1 score of the ensemble consisting of the selected machine learning models.

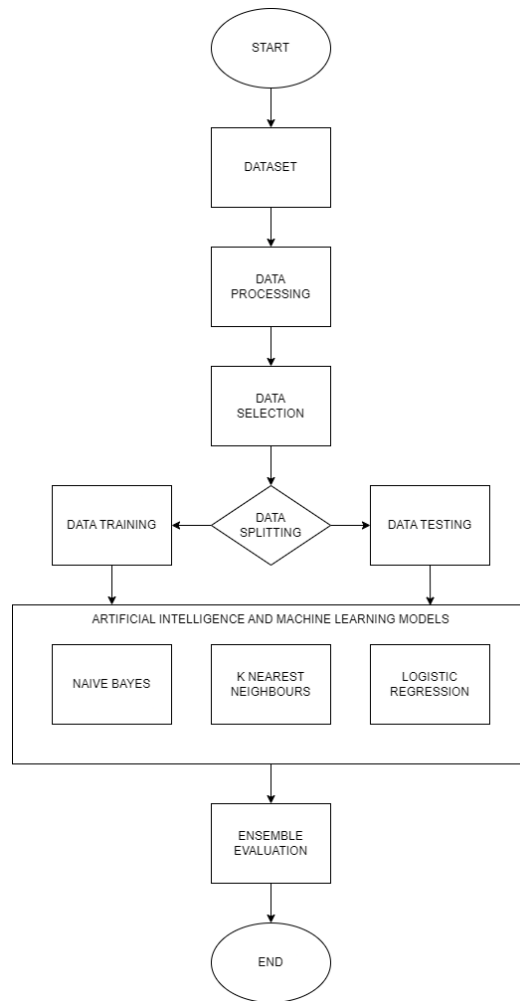
## 4 Design Specification

### 4.1 Scope

The 3 AI and ML models that are being used for the purpose of this research have been selected to verify the feasibility of using AI and ML in malware detection and help in the evolution of malware detection systems. The aim is to create an ensemble that utilizes multiple AI models to accurately and efficiently predict whether a file is a malware or not. The intention is to develop the ensemble and assess its accuracy and runtime.

### 4.2 Data Model

Figure 1 below shows a process flow diagram to summarize the recommended system. First the dataset will be processed, and final data selection process will occur. This is to train the model efficiently without any missing data in the dataset. Next the data will be split into training and test datasets. The 3 models of Naïve Bayes, K-Nearest Neighbours and Logistic regression will be trained and tested and then the final ensemble of the 3 will be evaluated.



**Figure 1 : Process flow of the implemented Malware Detection System**

## 5 Implementation

In this section, the implementation of the proposed malware detection system has been explained. Preparation/processing of the data followed by model training, testing and final ensemble evaluation is the process followed. The final results of the model along with the runtime of the experiment has been considered as the result of this experiment.

### 5.1 Naïve Bayes

Naïve Bayes is a method of classification that relies on the independence of predictors in Bayes' theorem. To put it simply, a Naïve Bayes classifier believes that the presence of one feature in a class has nothing to do with the presence of another feature.

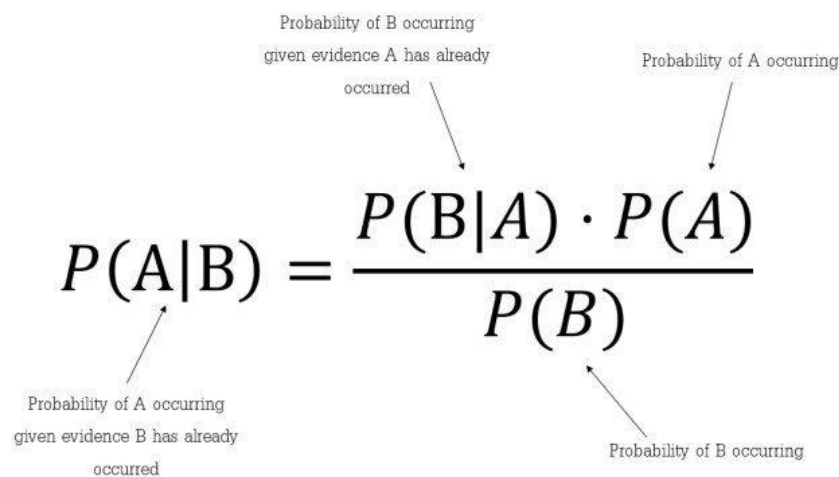
The Naïve Bayes Classifier is a well-liked supervised machine learning model. It can be used for data classification tasks like text classification. Since the Naïve Bayes classifier is capable of modelling the distribution of inputs for a given class or category, it belongs to the family of generative learning algorithms. The predictions made by the classifier are fast and efficient. This is because, its approach is based on the assumption that the features of the input data are conditionally independent of the class.

The reason this classifier is called Naïve is because it is quite simplistic. This is better explained with a following example. Naïve Bayes will classify a fruit as an apple if it is red, round and

has a diameter of about 3 inches. To the classifier it does not matter these features are dependent on each other or if it depends on other external features, all the properties independently comprise a probability that the fruit is an apple, hence it classifies it as an apple.

Naïve Bayes model is easy to build and is useful for large datasets. Not only is it simple but Naïve Bayes can sometimes also outperform other sophisticated algorithms. This is to say that if the assumption of independence holds, then the classifier requires less training data and can outperform logistic regression or decision tree algorithms.<sup>7</sup>

The equation for Naïve Bayes Classifier is as follows:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$


**Figure 2 - Naive Bayes Classifier**<sup>8</sup>

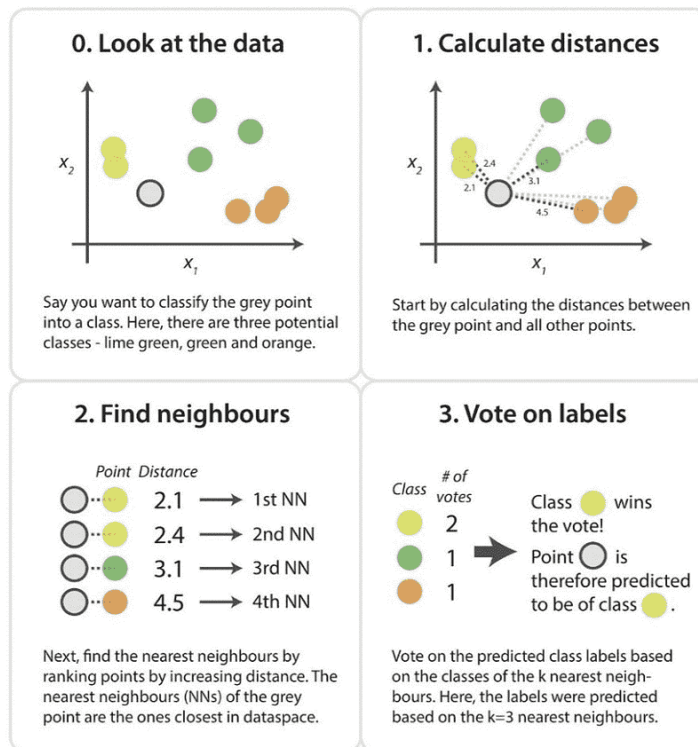
## 5.2 K-Nearest Neighbour

K-Nearest Neighbour (KNN) is a type of supervised learning algorithm that has applications in both classification as well as regression. This technique tries to predict a correct category for the test data by computing the shortest distance between the test data point and the training data points. It considers “k” number of data points that the test data is closest to. The algorithm calculates the probability of the test data belonging to a class of the ‘k’ training points. The class that has the highest probability is the class this algorithm assigns the test datapoint to. If the application is for regression, the value is the average of the ‘k’ selected training points.

While the above can be a bit confusing, consider an example and refer to Figure 3 below in order to better understand this algorithm.

<sup>7</sup> <https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/>

<sup>8</sup> <https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html>



**Figure 3 - K-Nearest Neighbour Example**

Suppose there are 3 classes of lime, green and orange colours. Now a new point appears which is grey and needs to be assigned to one of these 3 colours. The algorithm will first calculate the distance between the grey point and the neighbours nearest to it. Next, the algorithm will rank the distances from lowest to highest. Next, the algorithm will poll as to what colours were the closest, in this case 2 lime, 1 orange, 1 green. Since lime had the least distance and more occurrences, the algorithm will classify the grey dot as lime.

KNN is therefore, useful in such scenarios where the need is to classify an unknown point into known classes.<sup>9</sup>

### 5.3 Logistic Regression

Logistic Regression is a statistic model also referred to as the logit model. The main use cases / purpose of this model is predictive analysis and classification. By using logistic regression, one can estimate the probability that an event will occur in a dataset of independent variables. An illustration of such a situation would be whether or not a person voted, survived, etc. A probability results from this; the dependent variable has a value of either 0 or 1. "In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure." The natural logarithm of odds or the log odds are common names for this. The formulas below represent the function and provide a detailed explanation.

$$\text{logit}(p_i) = \frac{1}{1 + \exp(-p_i)}$$

$$\ln\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p$$

<sup>9</sup> <https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4>

Pi represents the dependent variable and X represents the independent variables in the aforementioned equations for logistic regression. The Maximum Likelihood Estimation (MLE) method is frequently used to estimate the Beta in mathematical equations. This approach iterates Beta across a range of values to determine the optimal log odds. The log likelihood function is created by these iterations, and logistic regression maximises this function to determine the optimal parameter estimate. An ideal coefficient may be identified, and the conditional probability for each observation can be calculated and recorded in order to provide a forecast probability. In a binary classification, the prediction is 0 if the probability is less than 0.5 and 1 if the probability is greater than 0.5. The goodness of fit test is the most effective method for assessing how well a model predicts. The Hosmer-Lemeshow test is one of the most widely used techniques to assess goodness of fit.<sup>10</sup>

## 5.4 Ensemble

After implementing the above algorithms, the last part of the model includes grouping the results obtained from these 3 algorithms and computing the final evaluation parameters of Accuracy, Precision, Recall and F1Score over both the training set as well as the testing split. This has been done by calculating the simple average of the parameters obtained from the 2 algorithms.

## 6 Evaluation

In this section the results of the implemented model, the parameters for each model and the final ensemble have been shown. The confusion matrices have been shown wherever necessary and a bar graph plot for the evaluation parameters.

### 6.1 Naïve Bayes

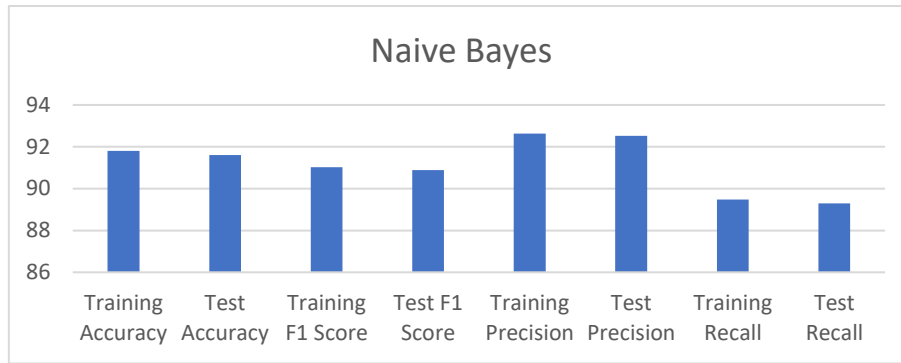
The first model used to evaluate the ensemble was Naïve Bayes. Table 4 shows the results obtained. The Testing accuracy was 91.61% with an F1 Score, Precision and Recall of 90.89%, 92.53% and 89.30% respectively for the testing scenarios.

**Table 6 : Naive Bayes Results**

<b>Training Accuracy</b>	<b>Test Accuracy</b>	<b>Training F1 Score</b>	<b>Test F1 Score</b>	<b>Training Precision</b>	<b>Test Precision</b>	<b>Training Recall</b>	<b>Test Recall</b>
91.80	91.61	91.03	90.89	92.63	92.53	89.48	89.30

---

<sup>10</sup> <https://www.ibm.com/topics/logistic-regression>



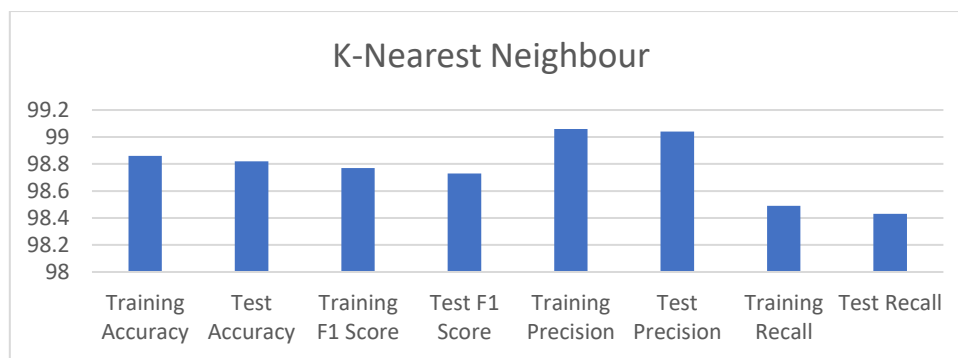
**Figure 4 : Naive Bayes Evaluation**

## 6.2 K-Nearest Neighbour

K-Nearest Neighbour was the next model that was used. This model gave the best results in all parameters among the 3 models. In the respective order of accuracy, F1 Score, Precision and Recall for testing, it gave the results of 98.82%, 98.73%, 99.04% and 98.43%

**Table 7 : K-Nearest Neighbour Results.**

Training Accuracy	Test Accuracy	Training F1 Score	Test F1 Score	Training Precision	Test Precision	Training Recall	Test Recall
98.86	98.82	98.77	98.73	99.06	99.04	98.49	98.43



**Figure 5 : K-Nearest Neighbour Evaluation**

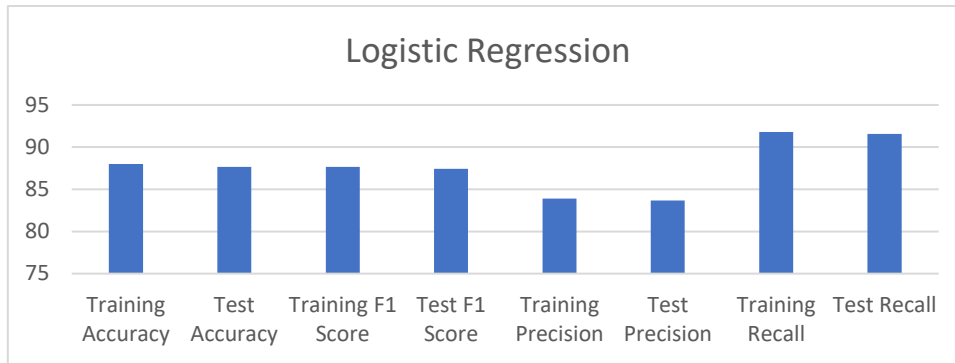
## 6.3 Logistic Regression

The last model trained and tested for the ensemble was the Logistic Regression Model. This model performed the worst among the 3 models. It resulted in a test accuracy of 87.66% and 87.43%, 83.66% and 91.56% on F1 Score, Precision and Recall for testing respectively.

**Table 8 : Logistic Regression Results**

Training Accuracy	Test Accuracy	Training F1 Score	Test F1 Score	Training Precision	Test Precision	Training Recall	Test Recall
88.00	87.66	87.67	87.43	83.89	83.66	91.81	91.56





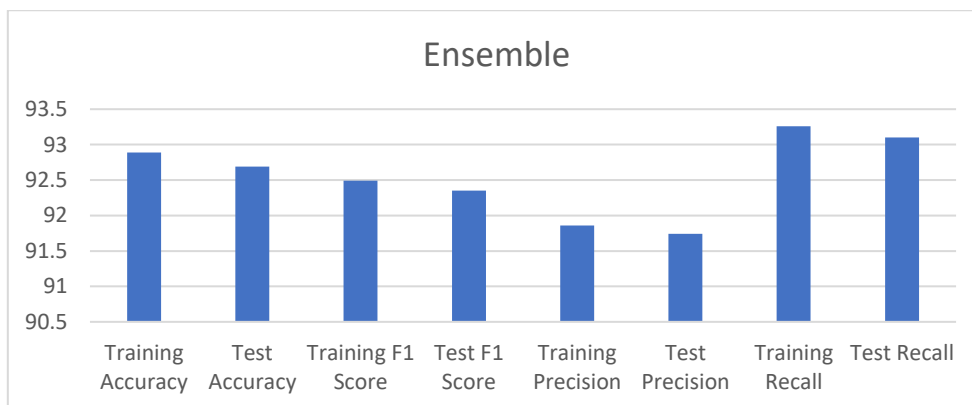
**Figure 6 : Logistic Regression Evaluation**

### 6.4 Ensemble Evaluation

Finally, the results of the ensemble have been given below where the accuracy is 92.69%, F1 score is 92.35%, precision is 91.74% and Recall is 93.10%.

**Table 9 : Ensemble Evaluation**

Training Accuracy	Test Accuracy	Training F1 Score	Test F1 Score	Training Precision	Test Precision	Training Recall	Test Recall
92.89	92.69	92.49	92.35	91.86	91.74	93.26	93.10



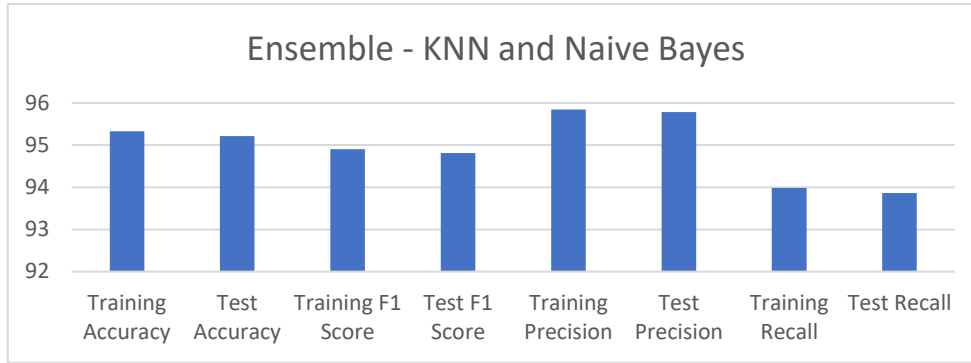
**Figure 7 : Ensemble Evaluation**

### 6.5 Discussion

On the basis of the above findings and experiments, the ensemble has a reasonable accuracy, but the accuracy of the K-Nearest Neighbour model is higher than the ensemble. It can also be seen that because Logistic Regression has performed so poorly, it should not be used in this ensemble. For verification purposes, a new ensemble of just the KNN and Naïve Bayes model was also created. The results obtained for the below are as follows:

**Table 10 : Ensemble with KNN and Naive Bayes Results.**

Training Accuracy	Test Accuracy	Training F1 Score	Test F1 Score	Training Precision	Test Precision	Training Recall	Test Recall
95.33	95.215	94.9	94.81	95.845	95.785	93.985	93.865



**Figure 8 : Ensemble with KNN and Naive Bayes Evaluation**

This ensemble has a relatively better accuracy and parameters than the previous one comprised of KNN, Naïve Bayes and Logistic Regression, however, as shown in Table 11 this ensemble is still outperformed by K-Nearest Neighbour model alone.

**Table 11 : Results Comparison**

Model	Training Accuracy	Test Accuracy	Training F1 Score	Test F1 Score	Training Precision	Test Precision	Training Recall	Test Recall
Ensemble with three models	92.89	92.69	92.49	92.35	91.86	91.74	93.26	93.10
Ensemble with two models	95.33	95.215	94.9	94.81	95.845	95.785	93.985	93.865
K-Nearest Neighbour	98.86	98.82	98.77	98.73	99.06	99.04	98.49	98.43

Therefore, based on this work, it is recommended to use the K-Nearest Neighbour algorithm instead of making an ensemble with other algorithms.

## 7 Conclusion and Future Work

The objective of this work was to find out how accurately an ensemble of Machine Learning algorithms can detect malware. The 3 algorithms used for this purpose were K-Nearest Neighbour, Naïve Bayes, and Logistic Regression. After performing experiments, the conclusion was the ensemble provided an accuracy of 92.69%. This accuracy is reasonable compared to many literary works that were researched. In the discussions section, it has been researched how the 2 models of KNN and Naïve Bayes would fair if we disregard the Logistic Regression model that performed the worst individually. This helped to obtain an accuracy of 95.22%. This accuracy is significantly better but still not over the accuracy given by K-Nearest Neighbour model alone which was 98.8%. Therefore, the conclusion of this work is while an ensemble can work and provide reasonable accuracy, in this case it is better to not use an ensemble but use just one ML model i.e., K-Nearest Neighbour.

### Limitations of the work:

The models have been tested over an already created dataset. Ideally, given more time, it would have been better to create an original dataset to do analysis on instead of using one that was already available. In this work, an ensemble with 3 models has been implemented. Given more

time, more ensembles using more different combinations of algorithms could have been made and explored.

### **Future Scope:**

For the future prospects of this work, one could explore different ML algorithms and techniques to either create different ensembles. Additionally, the models can be tested over a different dataset as well.

## **8 References**

Abdullahi, M. et al., 2022. Detecting Cybersecurity Attacks in Internet of Things Using Artificial Intelligence Methods: A Systematic Literature Review. *Electronics*, 11(2), p. 198.

Aksu, D. & Aydin, M. A., 2018. Detecting Port Scan Attempts with Comparative Analysis of Deep Learning and Support Vector Machine Algorithms.. *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pp. 77-80.

Al-Haija, Q. A. & Al-Dala'ien, M., 2022. ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks. *J. Sens. Actuator Netw.*, 11(1), pp. 1-18.

Al-Haija, Q. A., Saleh, E. & Alnabhan, M., 2021. Detecting Port Scan Attacks Using Logistic Regression.. *4th International Symposium on Advanced Electrical and Communication Technologies*, pp. 1-5.

Alkahtani, H. & Aldhyani, T. H. H., 2022. Artificial Intelligence Algorithms for Malware Detection in Android-Operated Mobile Devices. *Sensors*, 22(6), p. 2268.

Aygun, R. C. & Yavuz, A. G., 2017. Network anomaly detection with stochastically improved autoencoder based models. *4th International Conference on Cyber Security and Cloud Computing*, pp. 193-198.

Conti, M., Rigoni, G. & Toffalini, F., 2020. *ASAINTE: a Spy App Identification System Based on Network Traffic*. New York, ACM Conference (Conference'17).

Fatima, S. & Quadri, S., 2022. Malware Detection Using Cuckoo and ML Techniques. *Science & Technology Development Journal - Engineering and Technology*, 11(7), pp. 165-170.

Goeschel, K., 2016. REDUCING FALSE POSITIVES IN INTRUSION DETECTION SYSTEMS USING DATA-MINING TECHNIQUES UTILIZING SUPPORT VECTOR MACHINES, DECISION TREES, AND NAIVE BAYES FOR OFF-LINE ANALYSIS.. *SoutheastCon 2016*, pp. 1-6.

Kumar, A. & Lim, T. J., 2019. EDIMA: Early Detection of IoT Malware Network Activity Using Machine Learning Techniques. *IEEE 5th World Forum on Internet of Things (WF-IoT)*, pp. 289-294.

Kumar, B. J. et al., 2017. Logistic regression for polymorphic malware detection using ANOVA F-test. *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, pp. 1-5.

- Liu, J., Kantarci, B. & Adams, C., 2020. Machine learning-driven intrusion detection for Contiki-NG-based IoT networks exposed to NSL-KDD dataset. *WiseML'20: Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*, pp. 25-30.
- Majdi K., Q., Naser, M. & Alkasassbeh, M., 2022. Android Spyware Detection Using Machine Learning: a Novel Dataset. *Sensors*, 22(15), p. 5765.
- Malik, J. & Kaushal, R., 2016. *CREDROID: Android Malware Detection by Network Traffic*. Paderborn, s.n.
- Mohapatra, N., Satapathy, B., Mohapatra, B. & Mohanta, B. K., 2022. *Malware Detection Using Artificial Intelligence*. s.l., IEEE, pp. 1-6.
- Mohurle, S. & Patil, M., 2017. A brief study of Wannacry Threat: Ransomware Attack 2017. *International Journal of Advanced Research in Computer Science*, 8(5), pp. 1938-1940.
- Mustafa Majid, A.-A., Alshaibi, A. J., Kostyuchenko, E. & Shelupanov, A., 2021. *A review of artificial intelligence based malware detection using deep learning*. s.l., s.n.
- Naser, M. & Abu Al-Haija, Q., 2023. Spyware Identification for Android Systems Using Fine Trees. *Information*, 14(2), p. 102.
- Priya, V. & Sofia, A. S., 2023. Review on Malware Classification and Malware Detection Using Transfer Learning Approach. *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 1042-1049.
- Rathore, H., Agarwal, S., Sahay, S. K. & Sewak, M., 2018. *Malware Detection Using Machine Learning and Deep Learning*. s.l., Springer, pp. 402-411.
- Stiawan, D. et al., 2021. Ransomware Detection Based On Opcode Behavior Using K-Nearest Neighbors Algorithm. *Information Technology and Control*, 50(3), pp. 495-506.
- Syarif, A. R. & Gata, W., 2017. Intrusion detection system using hybrid binary PSO and K-nearest neighborhood algorithm. *2017 11th International Conference on Information & Communication Technology and System (ICTS)*, pp. 181-186.
- V, D. M. & Devi K.A., S., 2019. *Prevention from Security Risks of Spyware by the Use of AI*. Bangalore, IEEE, pp. 131-135.
- V, D. M. & Devi K.A., S., 2019. Spyware Detection and Prevention Using Deep Learning AI for User Applications. *International Journal of Recent Technology and Engineering (IJRTE)*, 7(5), pp. 345-349.
- Yan, J., Qi, Y. & Rao, Q., 2018. Detecting Malware with an Ensemble Method Based on Deep Neural Network. *Security and Communication Networks*, pp. 1-16.