Identification of Key Factors that Influence False Positive Detection & Classification by Anti-Malware Program

MSc Internship
MSc in Cyber Security

Onyebuchi Aniekwena
Student ID: 15004058

School of Computing
National College of Ireland

Supervisor: Ben Fletcher

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	**Onyebuchi Aniekwena** ……..………………………………………………………………………………………..		
Student ID:	**15004058** ………………………………………………………………………………………………		
Programme:	**MSc in Cyber Security** …………………………………………………………	**Year:**	2020 ……………………..
Module:	**MSc Internship** ………………………………………………………………………………………………		
Supervisor:	**Ben Fletcher** ………………………………………………………………………………………………		
Submission Due Date:	**17th August 2020** ……………………………………………………………………………………………		
Project Title:	**Identification of Key Factors that Influence False Positive Detection & Classification by Anti-Malware Program** ……………………………………………………………………………………………		
Word Count:	7012 ………………………………………	**Page Count**	19 …………………………………..

I hereby certify that the information contained in the research work titled "**Identification of Key Factors that Influence False Positive Detection & Classification by Anti-Malware Program"** is information pertaining to the 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.
I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature: **Aniekwena Onyebuchi Chijioke**
……………………………………………………………………………………………………

Date: **17th August 2020**
……………………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

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

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

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Identification of Key Factors that Influence False Positive Detection & Classification by Anti-Malware Program

Onyebuchi Aniekwena
Student ID: x15004058

Abstract

False-Positive detection and classification by anti-malware product have been a cause of serious concern by sector leaders in the anti-virus program vendors and respective clients. Irrespective of incredible strides achieved by anti-malware program vendors, by proactively utilise machine learning and signatures heuristics pattern to detects and classify suspicious file and URLs links in real-time by anti-malware programs. The effect of constant mutation of recent malware sample, resulting in a high number of False-Positives seem to a portrayed severe question regarding the integrity of the vendor's product. This paper proposes an anti-malware detection method expected to improve accurate detection and classification accuracy; that reduces the likelihood of False-Positives detection using security scoring system in conjunction with traditional machine learning technologies. Where suspicious MD5 Hash samples submission from Microsoft Yara Hits through MalShare repository were analysed and matched with various hosts to determine Vendors Detection Rate (VDR) used as security score for this study. Experimental result features and metadata acquired are compared in conjunction with classification models to limit False-Positive dictation from suspicious samples, thereby creates a more efficient anti-malware program. The study experiments on two well-performing classifications, algorithm K-NN Classifier and Support Vector Machine (SVM) used for classification and regression to determine the accurate classification of malware sample, for effective False-Positive reduction. The findings from False-Positive results indicate that when the file types are compromised, it gives ample room for inaccurate performance in detection and classifications of anti-malware programs with the relative change in file size of the sample.

Keywords: Anti-malware, False-Positive, Obfuscation, Malware Detection, & Classification.

1 Introduction

False-Positive detection/classification by anti-malware product is a process whereby anti-malware software classifies a suspicious files or URLs link as malicious document or file, where it is rather benign. New anti-malware software utilises machine learning and signatures heuristics pattern to classify suspicious file or link as malicious document or file. No anti-malware vendor wants to allow high false-positive result in their anti-virus product due to the negative impact on the product reputation and multiplier effect on the vendor's profit. Malware attack in the order hand has proved to be serious security threats, encountered by all internet users, especially the enterprise clients that relied on the internet for the daily transaction of their services. The major cybersecurity anti-malware Vendors deploy a whole lot of resources in developing anti-malware software, host intrusion detection systems HIDs, Network Intrusion and Detection Systems NIDs. Due to some specific malware design techniques that utilise obfuscation, packing, and encryption seem to result in the increasing percentage of false-positive in the malware detection and classifications methods available. The shortfall in the expectation of the clients poses a serious concern to the software vendors, who sometimes claim to have real-time protection against malware attacks. Gandotra E. et al., [1] opined that significant reason that results in a high rate of False-Positive and False-Negative is partly as the strategy of deploying Intrusion Detection System (IDS), Intrusion Prevention System (IPS), and Anti-virus programs. These protocols cannot identify unknown malware due to methods of using signature-based decisions.

The anti-malware products customers, which includes enterprise and end-users, always expect stone proof protection to her respective IT architecture and devices. We often see vendors claiming real-time detection as a result of automation associated with improvement in artificial intelligence, machine learning and signatures heuristics pattern. However, there is a recent avalanche of malware researches that have near proven efficient in analysis for detection and classification of malware, due to tremendous improvement in the skill set of malware authors, who continuously improve their ability to attack IT architecture successfully. Malware authors devise the use of packers, obfuscator and Cryptor's to champion anti-malware programs evasion, malware. They employ the use polymorphic, metamorphic, and other deceptive techniques to change the malware behaviours quickly and to generate a large number of new malware behaviour that enable then to trick as well evade current methods used by anti-malware vendors [2]. The malware authors consistently utilise these polymorphic tools to obfuscate, encrypt and pack codes that mutate and constantly and modify malware behaviour during analysis entry point that usually subverts the automated process analysis and result in the high rate of False-Positives classifications.

Masoud Narouei et al. [3] proposed a heuristic technique based on static analysis which could detect malware with expected high accuracy; it is also robust against conventional evasion techniques such as junk injection and packing. The mixture of static analysis and dynamic carefully improves the investigation of suspicious malware through automation, that improve the detection rate through extracting more behavioural statistics of malware. However, an improvement on the metamorphic comportment of recently available malware has the potential to render signature and heuristics-based detection useless [4].

Despite the development by anti-malware program vendors, which using Endpoint and AMCore technology to mitigate False-Positive detection and classification of the suspicious malware sample. [1]**AMCore** is the next generation of anti-malware scanning technology developed by Mcafee that provides enhanced capabilities to counter the newest malware threats with speed and efficacy. It seems to be a consistent increase in the number of False-Positive which necessitated the need to explore possible determinant of the factors that influence False-Positive detection of the suspicious malware sample. It is not uncommon to see anti-malware vendors dedicate a significant channel for the request, discovery, revaluation and review of the suspicious sample due to consistent high rate of False-Positive detection and classification.

This research work proposes, protocol aim to accurately analyse the behaviour of a suspicious file that eludes correct classification, in order words, will mitigate the likelihood of high False-Positive. The result of this protocol will keep the effect of the False-Positive rate at least minimum.

According to Lungana-Niculescu, A. M. et al. [5], the act of creating white-list for an exception of clean files will tremendously reduce a false positive alarm. For example, if URLs or files from highly Alexa ranked site appears as a potentially malicious, it should not be automatically categorised as malicious. This type of file was classified as a suspicious file which requires further manual analysis before being classified as malicious; rather than categorised by a mere signature detection. The white-list data stand as control list included in the database of sanity list. This malicious file will be forward to reconsidered queue for further manual analysis [5] [6]. Further opined that in other to mitigate false positive, file versions extracted from hash properties and added to control repository, to facilitate the matching process to reduce false positive [7]. When False-Positive is detected in this process, measures including adding to reconsidered queue or further corrective actions shall be taken.

1

[1] https://www.mcafee.com/enterprise/en-us/assets/white-papers/wp-understanding-ep-security-10-module.pdf

1.1 Research Motivation and Research Question:

The major disbelieves faced by enterprise customers that paid heavily on these products expecting near full proof but instead receive a plethora of False-Positive that is quite challenging to muscle out; thereby always keep the anti-malware vendors on her toes. In most cases, we often find vendors making a childish response to her customers, stating that the sample was classified by automation due to sharing common characteristics with known malicious files. Manual review has determined that the sample poses no risk, and the rating has been changed.

The development in the skills set of malware authors, by continuing honing their skillset in the use of malware obfuscation and polymorphic mechanism to develop malware that mutates itself, to trick as well elude signature detections avoidance. Most of the recent studies identified that the result of reducing False-Positives impacts negatively on True-Positive, which suggest that False-positive reduction and False-Negative status is intricately interwoven. In the order hand, an attempt to have accurate elimination of False-Positive, with real-time automatic detection could negatively affect the efficiency of an anti-malware program.[8][2]. In order to find answers to this complex issue of False-Positive reduction necessitates rational of this current piece of research. "What factors are major determinant to False-Positive detection and classification by anti-malware programs?"

1.2 The Research Objectives/Specific Contribution includes:

In order to limit the high rate of False-Positive detection and classification, this study proposes an instance of security scoring method, that focus on the mutation in malware behaviour that muddled anti-malware programs to classify the suspicious sample incorrectly; when a suspicious file signature is not available in the Control list developed by anti-malware product vendors and updated continuously in her database. The control list code rules include considering the source of malware, especially those emanating from a high Alexa ranked site. The main contribution of this study is to subject all file with security score, which must hit a minimum of three different open sources automated vendors engines detections. Furthermore, such file will be sent to reconsidered queue for further machine learning reviews for proper classification after resolving conflicts, by utilising Support Vector Machine SVM Classifier and K-NN classifier that considers training and control set. This process is focused on the false positive population from the sample to analyse the details further.

1.3 Structure:

This research work, succinctly and for a specific reason is concise in the structure of its significant section's application organised by its vital content and original contents skillfully outlined. Structure of the paper boast of seven main sections briefly outlined below, referring page to the heading about to study in detail summarised as follows. Section I gives the introductory part of the study; it highlights the general context of the paper. These key areas include the research question, research objective and as well significant contribution of the study, with the detailed structure of the paper. Section II expounds on the review of related work, carried out till to date; as well logical present the gap in the area of study, also detailing data collection techniques utilised, data pre-processing, data balancing and automatic scan of the samples and pictorial representation of pre-processed output in the subsection. Section III presents the research methodology, with data evaluation and justifications. Section IV presents the design specification, which was depicted pictorially as a flowchart and contained the experimental setup and overview of the algorithms. Section V exclusively deals with implementation with elaborated design within chart format. Section VI presents the evaluation of the results obtained from the implementation, show the original experiments performed to improve outcomes. Furthermore, section VII provides the conclusion of the research and future work. The final part of the paper present bibliography, where all the references utilised in the study are listed in numerical order using the IEEE style referencing format.

[2]

2 Related Work

The two most crucial piece of work to this current research is Mitigating False Positives in Malware Detection, by Polyyakov Zhang, A. A. et al. [6] as well as Ma, X. et al. [7] in Using multi-features to reduce false positive in malware classification. For the fact that benign files create files and execute them likewise benign files, which are common behaviours performed by both, they are invariably performed in similar faction to anti-malware programs sometimes; thus, it causes false positives. [6] [7] [8] all suggest that corrective action taken by anti-malware programs when it incorrectly identifies a benign file as malicious; is to precautionary mitigation strategy. It advanced by either removing suspicious portions of the file, isolate the file to a safe location where it will not be executed or entirely delete the suspicious file. Zhang, Y. et al. [10] argued that two major tasks involved in malware analysis by anti-malware vendors include malware detection and malware classification. False-Positive detection/classification by anti-malware products arise from the complexity of utilising advanced techniques by malware authors; it has proved to negatively impact on the result of traditional protection techniques of anti-malware vendors. In the process of searching for ways to evade the ability of anti-malware software to detects, mitigate, or appropriately classify a suspicious files or URLs link as malicious document or file, polymorphism with addition malicious components were introduced by malware authors [11]. False-Positive is a process whereby anti-malware software classifies a suspicious files or URLs link as malicious document or file, where it is rather benign. No anti-malware vendor wants to allow a high number of a false-positive result in their anti-virus product due to the negative impact on the product reputation and multiplier effect on the vendor's profit.

The malware authors consistently utilise these polymorphic tools to obfuscate, encrypt and pack codes that continually mutate and modify malware behaviour during analysis entry point that usually subverts the automated process analysis and result in a high rate of False-Positives classifications [5] [6] [7][8]. Anti-malware vendors attempt to develop an automated real-time mechanism to detect and mitigate malware attacks tends to result in a high number of False-Positive from the real-time approach deployed by anti-malware vendors. To continually improve the efficiency of mitigation against the new version of malware or zero-day attack, the signature database of malware signatures must be periodically updated more often [7]. For instance, when there is imperfect heuristics, or the presence of a bug in the anti-malware product, the likelihood of imperfectly identify a benign file as positive is high.

Gandotra E. et al. [1] develop the concept of integrating both static and dynamic analysis of malware features using a machine learning process to detect zero-day malware with high accuracy. For the fact that static analyses encountered difficulties while dealing code obfuscation and polymorphic behaviour of packed malware [3]. It necessitates a more sophisticated approach requires to reduce the high rate of False-Positives

Dai, J. et al. [12] propose a technique of detecting unknown malware using dynamic instruction sequences mining approach to build a program monitor which can capture run-time instruction sequences of an arbitrary program. The host of other researchers adopt dynamic malware analysis protocol by proposing behavioural-based malware detection techniques.[15]. The method utilises the classification model to make an intelligent guess based on information derived from metadata. With the claim to an accurate result. According to [2], there is a deficiency of run-time overhead and weakness in monitoring Dynamic Library Links (DLL)

[2] https://malshare.com/

2.1 Obfuscation and Packing of Malware regarding Static Analysis

The execution of malware that comes in the packed condition is regarded as one of the significant reasons for False-Positive detection by the anti-malware program. Malware authors deployed sophisticated use packing or obfuscation techniques to make their files trickier to detect or analyse. Obfuscated programs are the method used by malware authors in an attempt to hide their maliciousness [2]. Packed programs are a subset of obfuscated programs in which the malicious program is compressed to make it difficult to examine without firstly unpacked the file. It is quite uncommon that anti-malware usually fails to correctly classify malware if a new packer comes with a unique technique. Narouei, M. et al. proposed the concept of grouping malware with known parkers such as UPX, ASPACK, Exe32pack, and Petit suggesting that the same level of accuracy is achieved when compared with their dependency tree with the original version.

2.2 Analyses of proposed techniques and shortfalls of recent research by various researchers

The proposed solution is expected to represent an average of over 20% improvement in performance on the reduction of False-Positive detection compared to existing solutions. Recent researchers and existing anti-malware products approach in reduction of False-Positive detection and classifications relied on the use of signature-based, heuristics and behavioural based technique as their primary decision engine detection and classification, [3] [4] [5], [6], [7], [8]. Despite the advantages of dynamic designed techniques, it has overhead during run-time and monitoring. The ingenuity of the malware authors strategy seems to continue to elude the accuracy in classification of suspicious samples by anti-malware products resulting in the high rate of False-Positive. This research strategy, upon utilising the static solution, also leverage open source repository of other vendors detection in developing security scoring point threshold. It as well incorporates automatic resolution from security scoring and channel unresolved security rate automatically to reconsidered queue for manual review demonstrated in research design for best human analysis classification in a short interval. This research utilises complete vulnerability findings that are automatically channelling unresolved classification to reconsidered queue for human review, which reduces the number of False-Positive samples. This strategy is a novel technique termed mixed-method that produces a more efficient result.

The modest existing resolution approach in reduction of the high rate of False-Positive choose precautionary mitigation strategy such as either removing suspicious portions of the file, isolate the file to a safe location where it will not be executed or entirely delete the suspicious file [10], by the anti-malware product in solving a continuing increase in False-Positive detection and alternatively provides a channel for clients and customers of their product to send suspected False-positive or wrong classified samples for review. The novel solution in this work is a mixed-method approach that offers automatic push notification on unresolved samples of security score to anti-malware product companies to resolve vague detection and classification. The standing malware analysis personnel within the anti-malware company will further do a human manual review as complete vulnerability findings. This approach stands a chance for further False-Positive reduction and customers satisfaction to achieve part of the objective of this research work.

2.3 Definitions

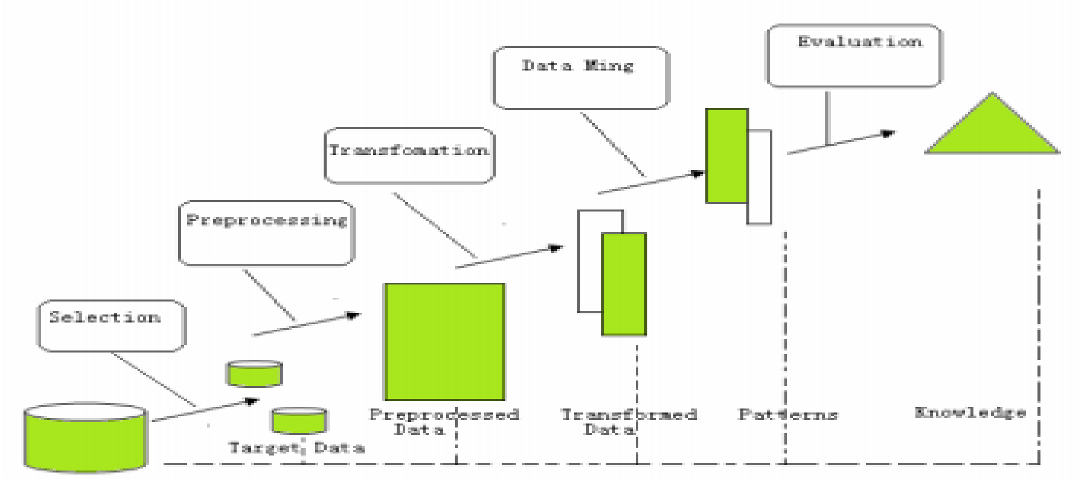
a. False-positive is incorrectly classified malware
b. True-Positive is correctly classified malware
c. The security rating is a categorisation of malware based on its family by anti-malware programs, such as Trojan Horse, Worms, Spyware, Rootkits, Phishing, etc.
d. Static analysis is the process of examinations of the malware, such that we do not actually execute the malware, but only try to figure out the malware is trying to do and the commands it is attempting to execute.
e. Dynamic analysis is the process of examinations of malware, such that we execute the malware. This method is only advisable to perform in a sandboxed environment as well as figure out the functionality of the malware.

3 Research Methodology

3

The research methodology process adopted in this research, followed the concept of Luo Q. [20] in Advancing knowledge discovery and data mining, (KDD), shown in general overview of the research methodology in Figure 1. Below. The research design is oriented towards the discovery of relationships which exists among meta-data in the sample collected, with the motive to analyse data from a different perspective to develop new detection and classification. This dependency could change the scale of False-Positive result by Anti-malware programs for a more accurate result. The concept is similar to the approach followed by Firdausi, I. et al. [13] and Zhang, F. et al. [14] and Ma, X. et al. [7] of data acquisition and analysis more relevant to this study.

Figure 2: Knowledge Discovery in Database (KDD) process. A general overview of the research methodology



Source: Q. Luo, “Advancing knowledge discovery and data mining” in *Proceedings - 1st International Workshop on Knowledge Discovery and Data Mining, WKDD* 3–5 (2008). doi:10.1109/WKDD.2008.153 [20]

3.1 Data Collection Method

The study was based on the Dataset collected from MalShare [2] MalShare repository is a free Malware repository platform that provides researchers with free access to potential, malicious feeds. The collected Dataset contains MD5 Hash user’s submission of Microsoft Yara Hits of respective customers captured over 30 days from June 2020 to July 2020. Worthy of mentioning that column with missing MD5 Hashes were deleted. All the MD5 Hashes samples captured in the Dataset were initially regarded as all suspicious files and potentially contained malicious and benign samples. The samples belong to mix of 24 unique formats of file types such as XML, PNG, RICH, PE32, PE32+, Rich, RAR, Ace, ELF, Composite, UTF-8, Java, ASCII, 7-ZIP, Zip, gzip, PHP, ISO, MS-DOS, DOS, CDFV2, UDF. MD5 Hash of each sample was submitted for static malware analyse of a total of 297 unique suspicious malware samples, which are recent publications of suspicious samples. The combination of malware monitor scan in [3] is used to extract the result of Engine Detection Rate (DER) to develop the Dataset for the malicious and benign samples used in this study. Another related [15]] study used a similar method to extract instructions for the development of Dataset used in the research.

[3] https://medium.com/bugbountywriteup/malware-analysis-101-basic-static-analysis-db59119bc00a

3.2 Data Sampling:

The concept of data sampling is fundamental to conduct academic research, and it justifies the interpretation of its result. The implement dynamic, is done in a sandbox environment created with an installed updated version of VirtualBox virtual machine with 32-bit Windows 7, were all the security services and firewall, including chrome security features were disabled. The rationale of considering the 32-bit Windows 7 is due to the lower memory size of the host machine when compared with Windows 10, and snapshot took after every execution of suspicious samples. The False-Positive or benign dataset samples identified after a dynamic analysis of suspicious samples from YARA feeds, making it a total of 297 suspicious samples. Including 64 False-Positive and 233 True-Positive making it 21.55% and 78.45% respectively. The dynamic analyses were performed in [3] [3]https://www.virustotal.com/gui/home open source security scanner.

3.3 Data Evaluation:

The proposed False-Positive rules for this research, consider the stipulated risk score of suspicious files or URLs. Before a suspicious file should be applied security rating category **vis**-à-**vis** malicious file, the risk score must be above a certain level. Support vector machines (SVM) K-NN were used as a supervised learning model to analyse data used for proper classification and regression analyses.

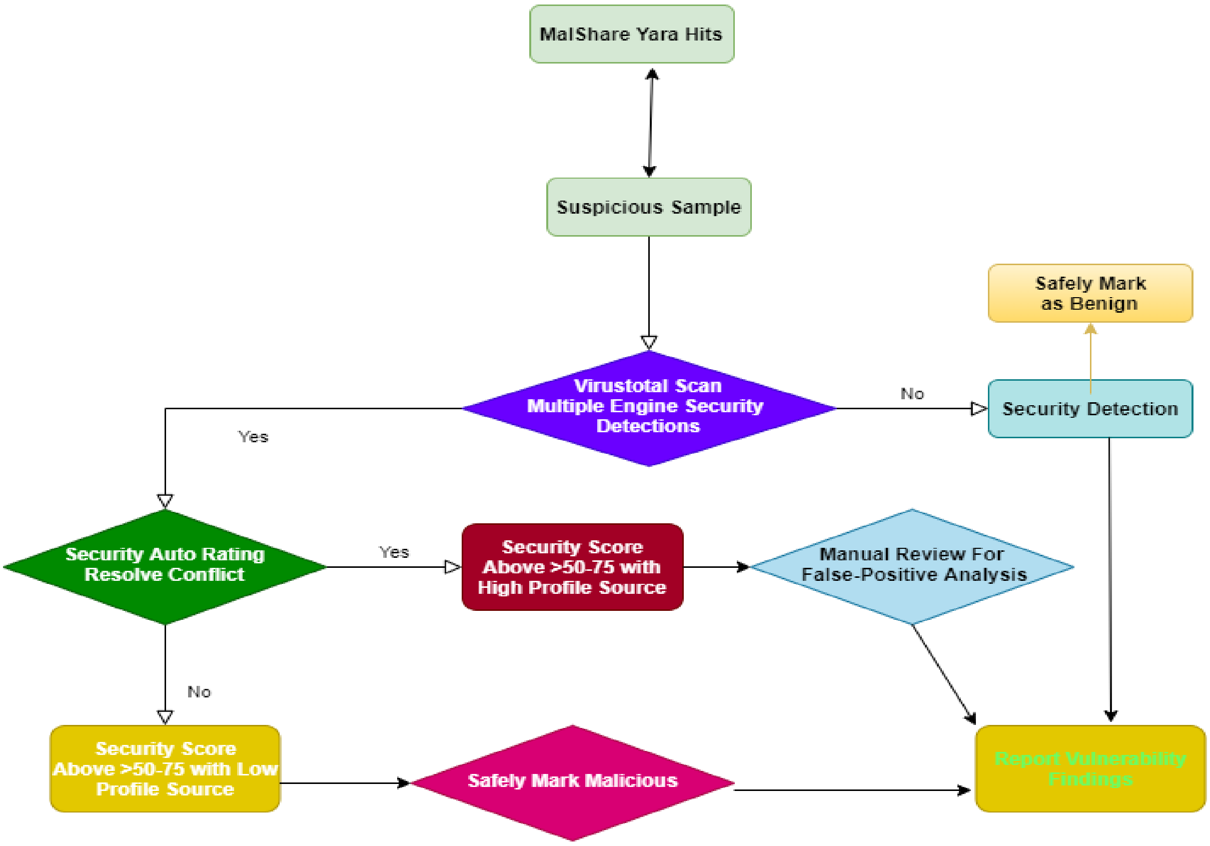
Threshold Scores / Comment	FP Security Score Range
Very High Chance that suspicious file or URL is Safe. Should never be applied Security Category, rather send to reconsider queue for Human analysis	**>= 50**
Medium Risk of FP. Can be sent to Security Auto rating to Resolve Conflicts.	**>= 50 < 75**
Can be safely marked malicious if rules detect a very low risk of escalations from the customer considering the IP of the source file	**>75**

Table 1: Experimental False-Positive Rules Evaluation

4 Design Specification

Figure 4 below shown proposed model design architecture which serves as a concatenation of two security protocols combined to achieve the objective of the proposed model and source of data collection. The MalShare Yara Hit at the top of the figure demonstrate the source of sample collection and initial assumption as only suspicious files queued for automated scan and analysis via [3] Virustotal. To achieve this, we collect various Vendors Detections used as security score, if there are security ratings, collected through static analyses of various vendors detections, considering the security score and source. The sample will be sent to auto-rating to resolve the IP address and identify the source. When the sample comes from high profile popular Alexa ranking with a higher threshold, the sample shall be sent to reconsidered queue for proper False-Positive analysis. However, if the threshold is low as demonstrated in False Positive evaluation in Table 1 above or less than 50, it can be safely marked as benign. However, if the security score is high as well the Alexa rank, it will send to manual review. Table 1 above demonstrated the adaptive process of recognising potential False-Positive detections.

Figure 4: The Research Proposed Design Specification


MalShare Yara Hits
Suspicious Sample
Safely Mark as Benign
Virustotal Scan Multiple Engine Security Detections
No
Security Detection
Yes
Security Auto Rating Resolve Conflict
Yes
Security Score Above >50-75 with High Profile Source
Manual Review For False-Positive Analysis
No
Security Score Above >50-75 with Low Profile Source
Safely Mark Malicious
Report Vulnerability Findings

[4]

[4] https://app.diagrams.net/

5 Implementation

The execution of the research design requires the use of centralised sandbox environments was set up for the purpose of experiments on the following architecture:

(a) Operating system: Windows 10, 64-bit, 750 GB Hard Drive. Host Windows Machine and used of Hypervisor, and Windows 10 Host and VirtualBox VM, having Windows 7 VM installed considering the memory available
(b) Microsoft Windows 10 Pro, Host version 10.0.18362 Build 18362, System Model VPCEL1E1E, 64-Bit base system type with AMD E-350 Dual-Core Processor. Installed Physical Memory (RAM) 8.00GB
(c) Microsoft Windows 7 VM

Date collected was pre-processed with excel and convert to CSV format to be ready for the modelling in R studio.

5.1 Data Pre-processing

The Dataset was capture manually into excel document cleaned up and converted into CSV format. The Preliminary investigation indicates that suspicious have both malicious and benign files, and XML file type shown to have the highest number of False-Positive with 16 hits out of 17, which account for 94.12% of the suspicious sample in False-Positive category. This XML statistics is very strategic since False-Positive detection is the focus of this study, which is impressive statistics from data pre-processing summarised in table 1 above
The study observed that this is an imbalanced dataset, such that the amount of the most majority file type is approximately 18.5 folds of the common file types taking cognisance of the number of PE32 vendors engine hits of 7493 in a total of 164 samples. A quite interesting fact is the Dataset is that the False-Positive and True-Positive is approximately a ratio of 1: 3.5

5.1.1 File Type Frequency Graph

The frequency graph of the suspicious sample, with outrageous occurrences of PE32 file type, with 164 as the highest in the Dataset out of 25 different file types

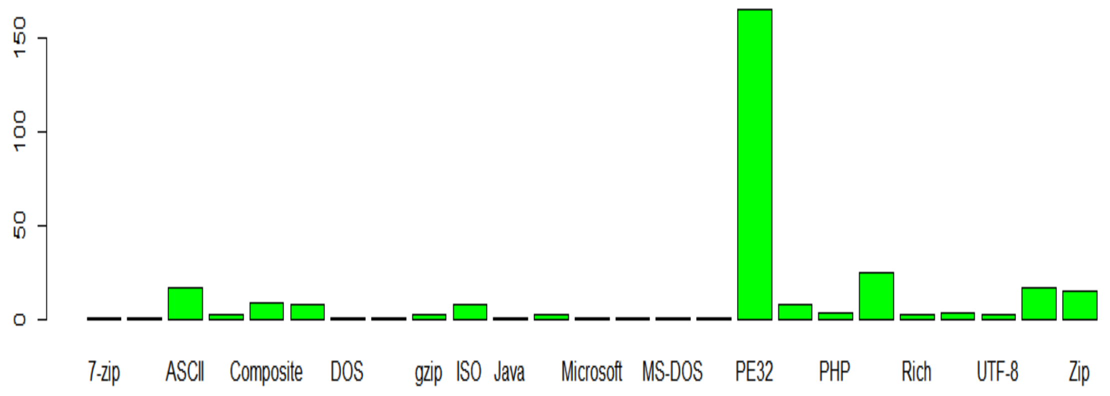


Figure 1: File Type Frequency of Suspicious Malware Sample in the Dataset

[5]

5.1.2 File Status Chart

The File status Chart shows the proportion of maliciousness and Benign of the suspicious sample.

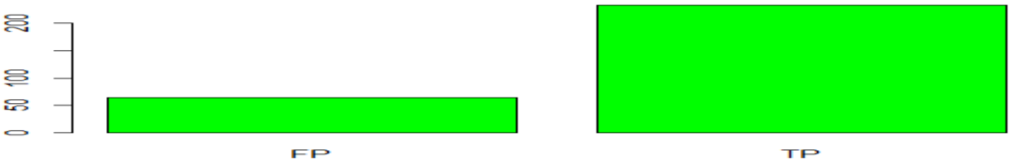


Figure 2: File Type Status (Benign and Malicious)

5.2 Automated Malware Scan

The step following data acquisition from [2] MalShare is to analyse MD5 Hash of all the 297 unique file types in [3] Virustotal. These analyses were performed within a sandbox environment, developed in VirtualBox VM to capture the number of vendor's detection rate used as security score and proof of false-positive when matched with other various vendors detection in the database to verify the status of the acquired MD5 Hashes. The population result with a higher percentage of False-Positives was further transformed after obtaining its active sample malware from free MalShare depository to obtain more meta-data.

Table 2: Dataset Summary

S/N	File Type	True-Positive	False-Positive (Benign)	Total File Type	Total Engine Detections
1	7-zip	1	0	1	9
2	ACE	1	0	1	33
3	ASCII	5	12	17	63
4	CDFV2	2	0	2	68
5	Composite	9	0	9	254
6	Data	2	6	8	58
7	DOS	0	1	1	0
8	ELF	1	0	1	13
9	GZIP	2	0	2	52
10	ISO	8	0	8	238
11	JAVA	1	1	2	24
12	JPEG	0	2	2	0
13	Microsoft	1	0	1	43
14	MS	1	0	1	34
15	MS-DOS	1	0	1	55
16	PDF	0	1	1	0
17	PE32	151	13	164	7493
18	PE32+	1	7	8	110
19	PHP	1	2	3	39
20	RAR	23	2	25	860
21	RICH	1	1	2	37
22	UDF	3	0	3	95
23	UTF-8	2	0	2	76
24	XML	1	16	17	22
25	ZIP	13	2	15	475
TOTAL		233 (78.45%)	64(21.55%)	297 (100%)	10151

[5] https://www.virustotal.com/gui/home.

On the second part of the study experiment, to check the performance classification in terms of time taken to train the models, in environment R studio, IDE used to write code and build machine learning classification model with support vector machine SVM and K-NN

6 Evaluation

6.1 Experiment / Case Study 1 Result of SVM Using Training Set Prediction

The study used SVN(f) function to build a support vector model. The hyper-Parameters of this function are C, kernel and gamma, while C defines the error penalty. The experiment SVN implementation using the training set to predict the accuracy of TP and FP classification of malware samples shown only 76% of File types analysed accurately by the model shown in figure 5 below where only six out of 25 different file types shown sign of False-Positive in the figure below. The result has shown a hyper-plane in the training set result, which means that the SVN model is implemented successfully.

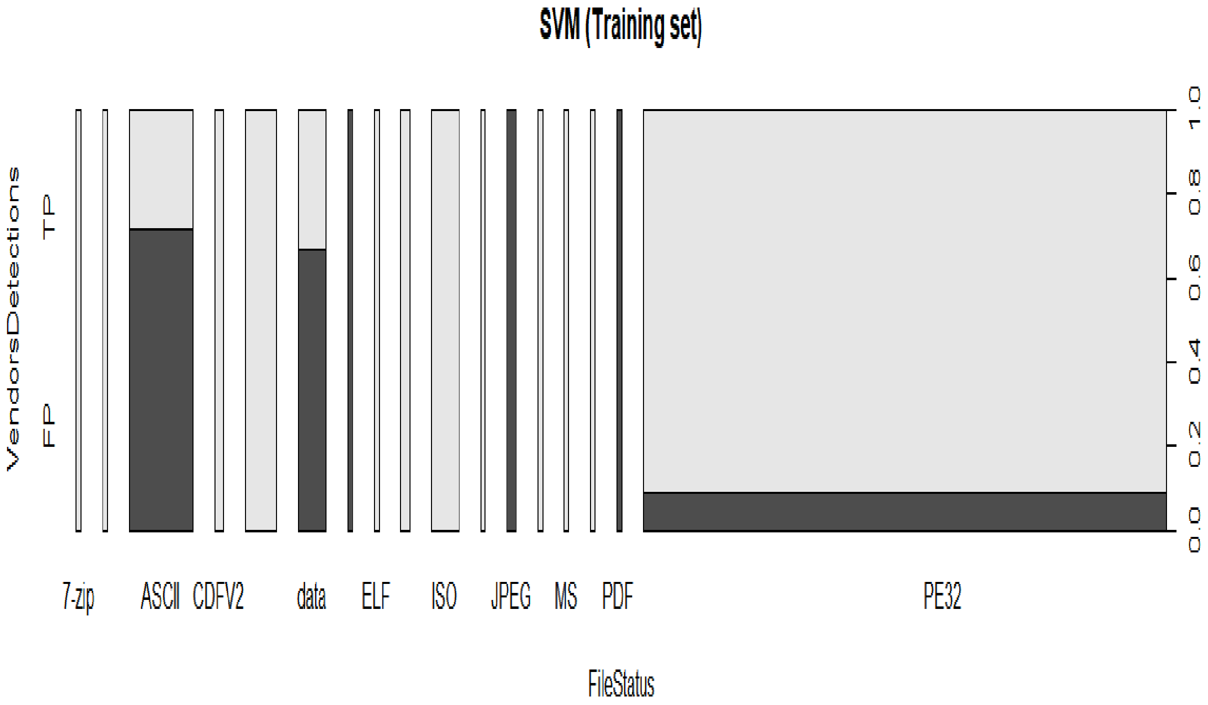
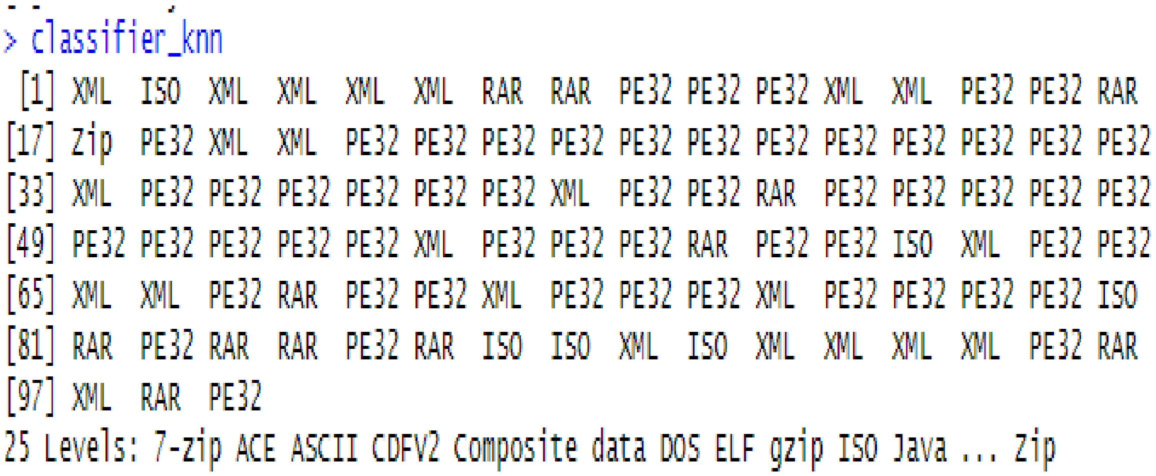


Figure 5: SVM File status Prediction Result

6.2 Experiment / Case Study 2 K-NN File Types of Accuracy Determination

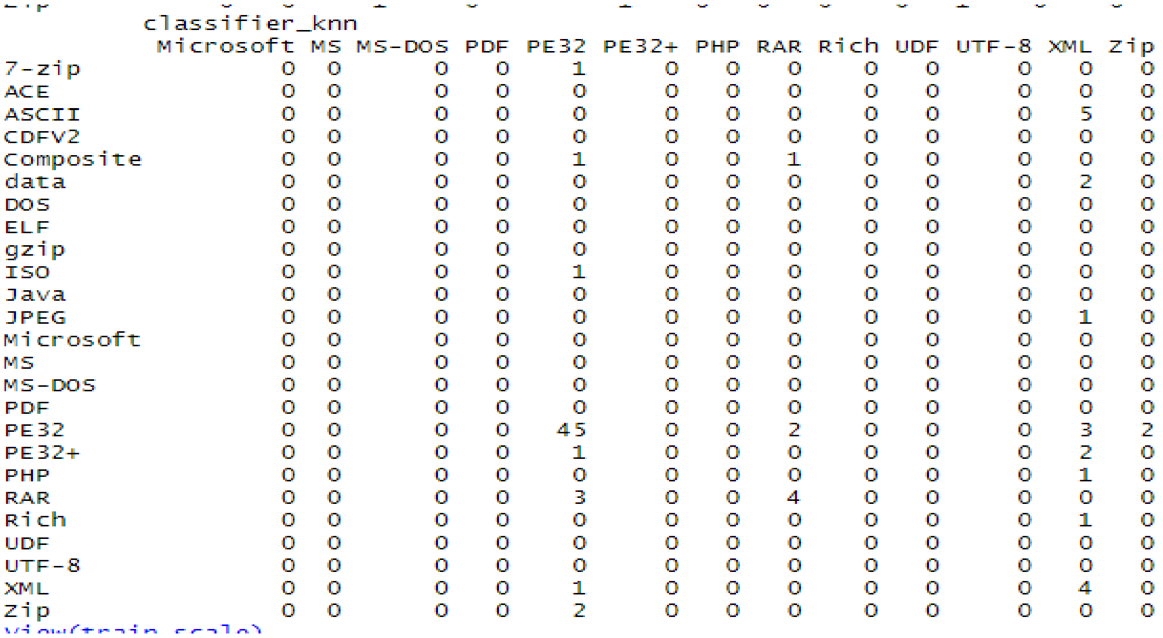
The K-NN algorithm is where K specifies the number of neighbours, such that number K is neighbour, which take K nearest neighbour to unknown data point according to distance. Count the datapoint to assign new datapoint category. The protocol considers the determinant factor of maliciousness and benign of different file types, containing False-Positive in the sample. For example, our Dataset contains two features of False-Positive and True-Positives from several file types.

Figure: 6 Model File Types Classifier_ K-NN



```
> classifier_knn
 [1] XML  ISO  XML  XML  XML  XML  RAR  RAR  PE32 PE32 PE32 XML  XML  PE32 PE32 RAR
[17] Zip  PE32 XML  XML  PE32 PE32 PE32 PE32 PE32 PE32 PE32 PE32 PE32 PE32 PE32 PE32
[33] XML  PE32 PE32 PE32 PE32 PE32 PE32 XML  PE32 PE32 RAR  PE32 PE32 PE32 PE32 PE32
[49] PE32 PE32 PE32 PE32 PE32 XML  PE32 PE32 PE32 RAR  PE32 PE32 ISO  XML  PE32 PE32
[65] XML  XML  PE32 RAR  PE32 PE32 XML  PE32 PE32 PE32 XML  PE32 PE32 PE32 PE32 ISO
[81] RAR  PE32 RAR  RAR  PE32 RAR  ISO  ISO  XML  ISO  XML  XML  XML  XML  PE32 RAR
[97] XML  RAR  PE32
25 Levels: 7-zip ACE ASCII CDFV2 Composite data DOS ELF gzip ISO Java ... Zip
```

The KNN model Classifier is fitted with a train, test and K value, and the classifier File Types feature fitted in the model



```
           classifier_knn
            Microsoft MS MS-DOS PDF PE32 PE32+ PHP RAR Rich UDF UTF-8 XML Zip
7-zip               0  0      0   0    1     0   0   0    0   0     0   0   0
ACE                 0  0      0   0    0     0   0   0    0   0     0   0   0
ASCII               0  0      0   0    0     0   0   0    0   0     0   5   0
CDFV2               0  0      0   0    0     0   0   0    0   0     0   0   0
Composite           0  0      0   0    1     0   0   1    0   0     0   0   0
data                0  0      0   0    0     0   0   0    0   0     0   2   0
DOS                 0  0      0   0    0     0   0   0    0   0     0   0   0
ELF                 0  0      0   0    0     0   0   0    0   0     0   0   0
gzip                0  0      0   0    0     0   0   0    0   0     0   0   0
ISO                 0  0      0   0    1     0   0   0    0   0     0   0   0
Java                0  0      0   0    0     0   0   0    0   0     0   0   0
JPEG                0  0      0   0    0     0   0   0    0   0     0   1   0
Microsoft           0  0      0   0    0     0   0   0    0   0     0   0   0
MS                  0  0      0   0    0     0   0   0    0   0     0   0   0
MS-DOS              0  0      0   0    0     0   0   0    0   0     0   0   0
PDF                 0  0      0   0    0     0   0   0    0   0     0   0   0
PE32                0  0      0   0   45     0   0   2    0   0     0   3   2
PE32+               0  0      0   0    1     0   0   0    0   0     0   2   0
PHP                 0  0      0   0    0     0   0   0    0   0     0   1   0
RAR                 0  0      0   0    3     0   0   4    0   0     0   0   0
Rich                0  0      0   0    0     0   0   0    0   0     0   1   0
UDF                 0  0      0   0    0     0   0   0    0   0     0   0   0
UTF-8               0  0      0   0    0     0   0   0    0   0     0   0   0
XML                 0  0      0   0    1     0   0   0    0   0     0   4   0
Zip                 0  0      0   0    2     0   0   0    0   0     0   0   0
```

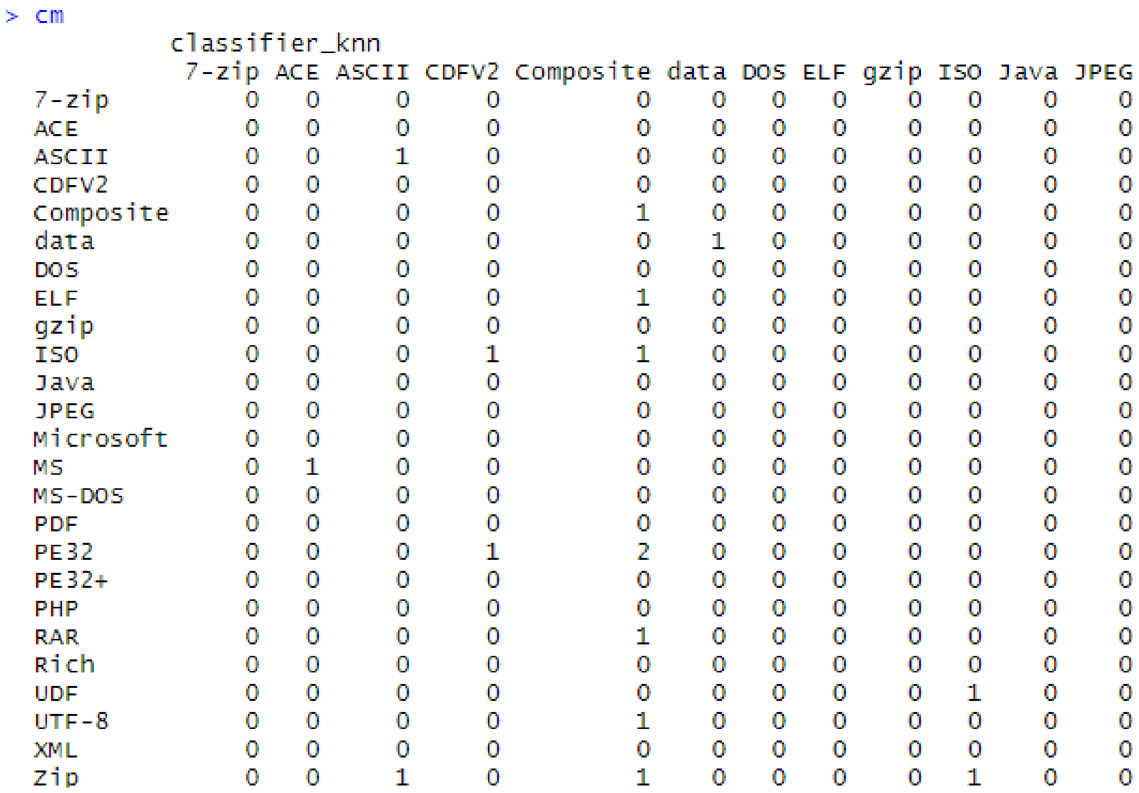
Figure: 7 K-NN Confusion Matrix

In the confusion matrix experimental result analyses in figure:7 above, the discussion is focusing only on the four file types with a significant number in the Dataset, the ASCII has a significant number. However, the file type is missing nearest neighbour from total samples correctly classified from the total of 17 samples, see Table 2: Dataset Summary above to confirm the total number of each sample at each unique test. The second file type considered in the analyses is ISO; it has only 1 out of 8 correctly classified. The most significant file type in the study is PE32 because of highest occurrence; it has 45 out of 164 samples correctly classified. The XML has only one out of 17 correctly classified.

In comparison, the ZIP file type has 2 out of 15correctly classified. The ISO has only one out of 17 correctly classified. The result from False-Positive analysed shown that misclassification arises as a result of unprecedented change or modification in the file name and its meta-data.

6.3 Experiment / Case Study 3 K-NN File Types of Accuracy Determination

In the confusion matrix, experimental result analyses in figure:8 below' the discussion are focusing only on the four file types with a significant number in the Dataset. The ASCII has 1 out of 17 samples is correctly classified out of one, see Table 2: Dataset Summary above to confirm the total number of each sample at test. The ISO has only one out of 8 correctly classified. The model in the case study 2 is missing in PE32 file type nearest neighbour as it seems to miss the significant target from the output samples correctly classified. Thus, maybe safe to say that the model is not the best fit. The XML has zero out of 17 correctly classified. The ZIP file type has 1 out of 15correctly classified.



```
> cm
            classifier_knn
             7-zip ACE ASCII CDFV2 Composite data DOS ELF gzip ISO Java JPEG
  7-zip          0   0     0     0         0    0   0   0    0   0    0    0
  ACE            0   0     0     0         0    0   0   0    0   0    0    0
  ASCII          0   0     1     0         0    0   0   0    0   0    0    0
  CDFV2          0   0     0     0         0    0   0   0    0   0    0    0
  Composite      0   0     0     0         1    0   0   0    0   0    0    0
  data           0   0     0     0         0    1   0   0    0   0    0    0
  DOS            0   0     0     0         0    0   0   0    0   0    0    0
  ELF            0   0     0     0         1    0   0   0    0   0    0    0
  gzip           0   0     0     0         0    0   0   0    0   0    0    0
  ISO            0   0     0     1         1    0   0   0    0   0    0    0
  Java           0   0     0     0         0    0   0   0    0   0    0    0
  JPEG           0   0     0     0         0    0   0   0    0   0    0    0
  Microsoft      0   0     0     0         0    0   0   0    0   0    0    0
  MS             0   1     0     0         0    0   0   0    0   0    0    0
  MS-DOS         0   0     0     0         0    0   0   0    0   0    0    0
  PDF            0   0     0     0         0    0   0   0    0   0    0    0
  PE32           0   0     0     1         2    0   0   0    0   0    0    0
  PE32+          0   0     0     0         0    0   0   0    0   0    0    0
  PHP            0   0     0     0         0    0   0   0    0   0    0    0
  RAR            0   0     0     0         1    0   0   0    0   0    0    0
  Rich           0   0     0     0         0    0   0   0    0   0    0    0
  UDF            0   0     0     0         0    0   0   0    0   1    0    0
  UTF-8          0   0     0     0         1    0   0   0    0   0    0    0
  XML            0   0     0     0         0    0   0   0    0   0    0    0
  Zip            0   0     1     0         1    0   0   0    0   1    0    0
```

Figure: 8 K-NN Confusion Matrix

6.4 Discussion of Findings

The initial pre-processing was done with open-source platform [3] Virustotal in order to normalise the Dataset appropriately. Experimental analysis and classification to achieve the main objective are demonstrated using K-NNN and SVM classifier algorithms in R. The performance of y_pred best fit was used for predictions to correctly classified malware TP against FP status from the sample and incorrectly identify False-Positive. The last phase is measuring the accuracy of the detection with K-NNN classifier to achieve the proposed objectives

The model evaluation achieved 56% accuracy with K=3. The model evaluation achieved 57% accuracy with K=5, which is higher than K=3. The model evaluation achieved 58% accuracy with K=5, which is higher than K=3, 5. The model evaluation achieved 58% accuracy with K=5, which is higher than K=3. The model evaluation achieved 58% accuracy with K=7, which is higher than K=3, 5 and is approximately the same as K=5, which means increasing K values does not affect the accuracy. The model evaluation achieved 59% accuracy with K=15, which is higher than K=7. Furthermore, finally, the model evaluation achieved 60% accuracy with K=19, which is higher than K=15, showing a steady increase as the K increase.

The K nearest neighbour is widely used in the industry, just like it shows consistency and robustness in predictions; however, future research may need to experiment with more machine learning classifiers for better compares with the result.

Figure 9: Model Evaluation KNN Classifier (K=….)

```
# Model Evaluation - Choosing K
> # Calculate out of Sample error
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.565656565656566"
> # K = 3
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$FileTypes,
+                       k = 3)
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.575757575757576"
> # K = 5
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$FileTypes,
+                       k = 5)
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.585858585858586"
> # K = 7
> classifier_knn <- knn(train = train_scale,
+                       test = test_scale,
+                       cl = train_cl$FileTypes,
+                       k = 7)
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.585858585858586"
> # K = 15
> classifier_knn <- knn(train = train_scale,
+                        test = test_scale,
+                        cl = train_cl$FileTypes,
+                        k = 15)
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.595959595959596"
> # K = 19
> classifier_knn <- knn(train = train_scale,
+                        test = test_scale,
+                        cl = train_cl$FileTypes,
+                        k = 19)
> misClassError <- mean(classifier_knn != test_cl$FileTypes)
> print(paste('Accuracy =', 1-misClassError))
[1] "Accuracy = 0.606060606060606"
```

7 Conclusion and Future Work

To answer the research questions, identifying the factors that contribute to False-Positive detection by anti-malware products and achieved research objectives. The study statistically analyses 297 suspicious malware samples after which 64 out of 297 samples are False-Positive, which means they are benign, which are identified through security scoring method introduced in the study. Moreover, the remaining 233 samples become True-Positives from suspicious malware samples obtained from [2] MalShare repository site. The samples contain mixed benign and malicious samples mainly contain computer viruses, trojan horses, worms, and bots. Our experimental results show that our proposal can detect about 56% to 60% of accurate classification of TP without any false positives; this shown a consistent, steady increase as the K value increases; it improves as the test of K value is higher; it proves consistency and robustness. Our proposal is straightforward; thus, it does not require dynamic analyses techniques. This protocol is justified because this study, like other recent studies, identified that some malware behaves in the same manner as benign programs such as downloading and proper installations. However, the experiment contains a combination of malicious and benign program samples, which the results might change if the model is run separately with the only malicious sample. The findings from False-Positive results indicate that when the file types are compromised, it gives ample room for inaccurate performance in detection and classifications of anti-malware programs, this indication is prevalent with there is a relative change in file size of the sample.

This study did not use dynamic analyses in generating the security scoring rate used for analyses, which we believe when incorporated in the model could generate a higher rate of accuracy. Despite achieving up to 60% detection and classification accuracy with K-NN evaluation, the model appears not up to expected True-positive classification and detection standard. Therefore require that future work build more robust experiments, which might need to consider increasing the number of suspicious samples, in order to appropriately evaluate our proposal by utilising the design specification used in this work, as well as try more different machine learning models that compared with current result, with more sophistry that have the ability to appropriate identifying the behaviour of malware such as running the sample with a combination of dynamic and static analyses.

8 References

[1] E. Gandotra, D. Bansal and S. Sofat, “Zero-day malware detection,” *2016 Sixth International Symposium on Embedded Computing and System Design (ISED), Patna,* 2016, pp. 171-175, doi: 10.1109/ISED.2016.7977076.

[2] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, S. Venkatraman, “Robust Intelligent Malware Detection Using Deep Learning” IEEE Access. 7, 46717–46738 (2019).

[3] M. Narouei, M. Ahmadi, G. Giacinto, H. Takabi, A. Sami, “DLLMiner: Structural mining for malware detection” *Security and Communication Networks.* 8, 3311–3322 (2015).

[4] T. Teller, A. Hayon, “Enhancing Automated Malware Analysis Machines with Memory Analysis Report” Black Hat USA, 1–5 (2014).

[5] A. M Lungana-Niculescu, A. Colesa, & C. Oprisa, “False positive mitigation in behavioral malware detection using deep learning” in *Proceedings - 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing, ICCP 2018* 197–203 (Institute of Electrical and Electronics Engineers Inc., 2018). doi:10.1109/ICCP.2018.8516611

[6] A. A. Polyyakov and R. Bikkula, “Mitigating False Positives in Malware Detection” United State Patent and Trademark Office US 8,719,935 B, 26th May 2014,

[7] X. Ma, Q. Biao, W. Yang, & J. Jiang, “Using multi-features to reduce false positive in malware classification” *in Proceedings of 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2016 361–365 (Institute of Electrical and Electronics Engineers Inc., 2016).* doi:10.1109/ITNEC.2016.7560382

[8] Y. Fukushima, A. Sakai, Y. Hori and K. Sakurai, “A behavior based malware detection scheme for avoiding false positive,” *2010 6th IEEE Workshop on Secure Network Protocols, Kyoto,* 2010, pp. 79-84, doi: 10.1109/NPSEC.2010.5634444.

[9] R. Kaur and M. Singh, “A Survey on Zero-Day Polymorphic Worm Detection Techniques,” *in IEEE Communications Surveys & Tutorials,* vol. 16, no. 3, pp. 1520-1549, Third Quarter, 2014, doi: 10.1109/SURV.2014.022714.00160.

[10] Y. Zhang, Q. Huang, X. Ma, Z. Yang, & J. Jiang, “Using multi-features and ensemble learning method for imbalanced Malware classification” *in Proceedings - 15th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 10th IEEE International Conference on Big Data Science and Engineering and 14th IEEE International Symposium on Parallel and Distributed Processing with Applications, IEEE TrustCom/BigDataSE/ISPA 2016 965–973* (Institute of Electrical and Electronics Engineers Inc., 2016). doi:10.1109/TrustCom.2016.0163

[11] J. Dai, R. Guha, J. Lee, Efficient virus detection using dynamic instruction sequences. *Journal of Computers*. **4**, 405–414 (2009).

[12] I. Firdausi, C. Lim, A. Erwin, & A. S. Nugroho, "Analysis of machine learning techniques used in behavior-based malware detection" in *Proceedings - 2010 2nd International Conference on Advances in Computing, Control and Telecommunication Technologies, ACT 2010* 201–203 (2010). doi:10.1109/ACT.2010.33

[13] F. Zhang & Y. Ma, "Integrated negative selection algorithm and positive selection algorithm for malware detection" in *PIC 2016 - Proceedings of the 2016 IEEE International Conference on Progress in Informatics and Computing* 605–609 (Institute of Electrical and Electronics Engineers Inc., 2017). doi:10.1109/PIC.2016.7949572

[14] J. Dai, R. Guha, J. Lee, Efficient virus detection using dynamic instruction sequences. *Journal of Computers*. **4**, 405–414 (2009).

[15] Choudhary, S. P. & Vidyarthi, M. D. A Simple Method for Detection of Metamorphic Malware using Dynamic Analysis and Text Mining. In *Procedia Computer Science* **54,** 265–270 (Elsevier, 2015).

[16] S. H. Moghaddam, M. Abbaspour, "Sensitivity analysis of static features for Android malware detection" *in 22nd Iranian Conference on Electrical Engineering, ICEE 2014 (Institute of Electrical and Electronics Engineers Inc., 2014),* pp. 920–924.

[17] R. Sihwail, K. Omar, K. A. Z. Ariffin, S. Al Afghani, "Malware detection approach based on artifacts in memory image and dynamic analysis" Applied Sciences (Switzerland). 9 (2019), doi:10.3390/app9183680.

[18] D. G. Gomes, R. N. Calheiros, R. Tolosana-Calasanz, "Introduction to the special issue on Cloud Computing: Recent Developments and Challenging Issues" *Computers and Electrical Engineering.* 42 (2015), pp. 31–32.

[19] X. Hu, "Large-Scale Malware Analysis, Detection, and Signature Generation, ProQuest" *Dissertations and Thesis, University of Michigan* (2011)

[20] Q. Luo, "Advancing knowledge discovery and data mining" in *Proceedings - 1st International Workshop on Knowledge Discovery and Data Mining, WKDD* 3–5 (2008). doi:10.1109/WKDD.2008.153

[21] B. J. Kwon, & T. Dumitras, "The Dropper Effect: Insights into Malware Distribution with Downloader Graph Analytics Categories and Subject Descriptors" *Ccs '15* 1118–1129 (2015)

[22] J. Devesa, I. Santos, X. Cantero, Y. K. Penya, & P. G. Bringas, "Automatic behaviour-based analysis and classification system for malware detection" in *ICEIS 2010 - Proceedings of the 12th International Conference on Enterprise Information Systems* 2 AIDSS, 395–399 (2010).

[23] G. Cabau, M. Buhu, & C. P. Oprisa, "Malware classification based on dynamic behavior" in *Proceedings - 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2016* 315–318 (Institute of Electrical and Electronics Engineers Inc., 2017). doi:10.1109/SYNASC.

[24] K. Rieck, P. Trinius, C. Willems, & T. Holz, "Automatic analysis of malware behavior using machine learning" *Journal of Computer Security* **19,** 639–668 (2011).