*George Gilbert*

*University of Salford | Cyber Threat Intelligence*

**@00476947**

***Classification of ransomware family locky using machine learning with weka***

Keywords: Ransomware, Malware, WEKA.

# Abstract

Ransomware has been used more frequently by cyber attackers than any other malicious attack, just a few months back we heard about a ransomware cyber-attack called “wannacry” which attacked the National Health Service in the UK and among others. Ransomwares mostly use features on Microsoft systems to initialize and spread to other machines such example of a feature is the Server massage book “SMB” which was developed with the intent of providing shared access to files, printer and ports. These cyber criminals now use pivoting techniques to stage sophisticated almost untraceable and undetectable attacks. There attacks are carried via many ways but must recent done by Phishing which is a technique used by cyber attackers to exploit systems by send a malicious email to un suspecting victims encoded malicious codes in attachments. The Victims of such attacks opens these file attachments by that triggering the malicious codes. Once the malicious codes are executed, it encrypts the victim’s files and request for payment in bitcoins.

Ransomwares have now become worrisome to both governmental organizations and private sectors, and the rapid time frame it takes to spread from one geographical area to another.

Due to the constant evolution of these malwares, it is almost impossible for traditional signatures based system to detect these malwares, malwares are more flexible and can be modified easily, but the malware detection systems a rigid and these limits their detection capacities and criteria. These is where this research come in, it is aimed at developing a more efficient way to detect these malwares by the use of machine learning techniques.

It is worrisome to face such a threat and not have a high assurance of detection. Traditional signature based approaches have been limited in their detection capabilities due to the limitations in their detection criteria. That is where this research comes in. It is aimed at increasing the detection ratio through the application of machine learning techniques. This research made use of some features used by most anti malware engines and passed them to four machine learning classifier algorithms on WEKA, which are Radom Tree, ZeroR, Naïve Bayes and Decision Table classifier, which resulted in 80 % detection accuracy.

# Introduction

The first known ransomware attack was recorded to be created by Joseph Popp in 1986[1], it was distributed via a floppy disk which was handed to participants in a conference on the AIDS disease, the malicious codes encrypted just the filenames and demanded for a ransom to be paid to an account in Panama, but the was thwarted by a quick fix thereby rendering the attack unsuccessful. But with time the malware programs evolved and found ways to carry out zero day exploits. The Idea of a file encryption ransomware was developed by Moti Yung and A.Young and was presented at the 1996 IEEE security and Privacy conference[2], it was called “crytoviral extortion”. The key feature of ransomwares is the ability to retrieve the ransom successfully and leaving no trace for forensic investigation. Young and Yung introduced a notion of using public key cryptography for encrypting ransomed data, they criticized the failed AIDS ransomware attacked and suggested the of failure was its use of symmetric cryptography alone.

This idea brought an evolution process in which malware developers develop ransomware.

Most of the ransomwares are distributed via emails, by embedding the malicious codes in the authentic files [3], the unsuspecting victim opens the malicious files, there by triggering the autoexec function [3], the malicious codes then generate a random symmetric key and encrypts the victim Files with, it uses a public key imbedded in the malware to encrypt the symmetric key,[2] this results to asymmetric and symmetric cipher text of the victim’s files. It the zeroes the symmetric key value to avoid data recovery. it sets up a message to the user with the asymmetric cipher text and how to pay the money. If successful the victim has no choice but to send the E-money by sending the asymmetric cipher text alongside the e-money. The attacker receives the payment and the asymmetric cipher text and then deciphers it with a private key and send the symmetric key to the victim and uses it to decrypt the files. Many governmental and private organizations have been attacked by the use of ransomware, most recent was the “wannacry” ransomware that hit over 20 countries in May 2017[4] and just when the world thought it was over; then came “Petya” in June 27 2017 which hit Russian oil companies and Ukrainian banks and has spread to other countries [5], Patches to counteract this malicious programs are not adequately deployed leaving a lot of room for damages [6].

It is clear that the current detection solutions aren’t sufficient to tackle the arising security threats and challenges[6]. This brings up the big question; how do we develop or improve the current systems to detect these malicious files and prevent them for being executed in the first place?

The approach taken to solve the problem is based on a previous application of machine learning on malware features to improve malware detection [7]-[8]. The aim of these research paper is to detect the threat before they are able to get executed. This will be done by using specific machine learning algorithms to decide based on the data passed to them.

In the chapters 2, I will be discussing the current ways of detection and analysis of related research works. In chapter 3, I will go ahead to discuss the methodology used in this research paper, in chapter 4, I discuss the results gotten from the previous chapter and last but not lest chapter 5 will contain the conclusion and recommendation to the research and any future work.

# Related works

In this section, enlightenment is provided to the reader about different ways of malware analysis, the advancements and limitations done in the area of study [8], explanations will be given for different existing type of malware analysis and point out their limitations. I will the go further to explain the need my research to the field.

## Static analysis

### What is static analysis?

Static analysis is the use of certain features extracted from malicious software, these features include program header information, binary strings and source codes extracted from the malware[8]-[10].A particular software is able to extract the features, it decompiles the application back to codes. These can be done without executing the codes [10]. When the features are extracted, they will have to be examined by forensic investigators that will keep an eye for abnormalities in the codes[11]. This analysis is effective due the careful examination done[11], however it is time consuming [11], obfuscation of the malicious programs by malware developers make it really hard to detect abnormality [12], [13] and leaves room for human error. Because of these problems mentioned above, static analysis is rarely used for analysis.

## Dynamic analysis

Dynamic analysis involves the investigation of malware performed by running the malicious code in a sandbox [8] (virtual environment). This process allows the examiners to monitor the API calls made by the malicious application to identify the features and behaviours of the malware. During these analysis procedures are put into place to study the behaviour of executed malware codes containing the binary occurrences and function calls [7].

The limitation to dynamic analysis is that most times malware programmers sometimes modify codes to detect sand boxes and this will result in an inaccurate analysis of the malware characteristics [14]. The features extraction can be a whole lot wearisome and tiring and a large amount of time can be spent on analysing just one malware, which makes it a very unsuitable method for analysing a large data set of malware[11].

## Machine Learning Practices

There have been several similar works done using machine language to detect malware. The first of this goes back to 2001, when a group of academic investigators [15], proposed that is a better method for malware detection, rather than the old signature based detection techniques. In the effort to prove it, 4301 applications were examined,3301 of these applications were deemed malicious and a 1000 of them were benign files. Binary profile files, string sequences and hex dumps were used in the experiments. Many more works have followed after this, such as a group of researchers went on to use the n-gram attributes obtained from the binary files were added to the classifier, which yielded a high success ratio of 98% [16]. Over time different features have been used by these classifiers to yield a greater accuracy [16]. The classifiers require numerous number of samples to yield higher accuracy [16]. To generate all the data is time consuming. Within the machine learning techniques, several suggestions have been brought forward to fix the issue, which involves the use of multi classifiers approach [17]. This aimed at getting the best true positive ratio.

# Methodology

## 

The table shown in table 1.0 shows a breakdown of the dataset used in this research work. It also shows the processes taken to obtain the results. I used a total of 50 malwares and 50 benign ware samples. These samples were obtained from [www.ransomtracker.com](http://www.ransomtracker.com) and portableapps.com. The malwares were obtained by the collection of the latest 50 hashes of the bonet C&C, Locky family to develop the dataset due to the need to use the latest samples for this analysis. I then ran the malware samples against the virus total intelligence platform to retrieve relevant data of the behaviors exhibited by these malwares.

The benign hashes which I got from [portableapps.com](http://www.portableapps.com) ,were passed through the virus total intelligence frame work which yielded no detection whatsoever from any antivirus software hosted on virus total framework.

The machine will learn with our data-set using WEKA[18] on an OSX operating system, 8.00GB RAM and Dual Core i5 2.67GHz processor.

|  |  |  |
| --- | --- | --- |
| Type | Family | Number of Experimental Samples |
| Win 32  Win 32 | Locky  Benign ware | 50 - split into 25(s) for training and test datasets.  50 - split into 25(s) for training and test datasets. |
| Total |  | 100 |

*Table 1: Details of Malware Dataset*

Figure 1 malware analysis steps

## The Dataset

The table 1 shows the information on the dataset that contains the malware hashes used in this research. In this section, I will discuss the details in the malware selection and the criteria’s used to select such samples. The malware hashes were collected based on the dates submitted on virus total and published on ransomware tracker website, most of them are occurrences that happened within the year 2017, the families of ransomware belong to the locky family, the file types were all windows 32 bit portable executables. The malware hashes were running individually on the virus total intelligence platform and features extracted.

## Static features extractions.

As mention above the malicious hashes were ran individually on the virus total intelligence platform and analysis reports gotten from the site was used to determine the relevant features needed in this research work. The classification of the features was based on the results obtained from static analysis, such as the Stings found in the file binaries codes. For all the samples in the dataset, I examined the extracted strings and identified the recurring strings in majority of the hashes both in malware and benign ware but there was a string only found in malware hashes which was the string “.rdata” while in the benign ware the “.ndata”. The information gotten from this was passed to the machine learning classifiers for automated classification.

|  |  |
| --- | --- |
| .reloc | True |
| .rdata  .data  ndata  .rsrc  .Text  .Tex2t | True  True  false  True  True  True |

## Dynamic Feature Extraction

There were also some dynamic features extracted from the malware hashes and used for the classification process, they include specific (Dynamic Link Library) dll files called by the ransomware in a live environment.

Most of all the hashes examined had the following dll files called and sometimes executed.

|  |  |
| --- | --- |
| .DLL | Number of appearance |
| ADVAPI32.dll  GDI32.dll  KERNRL32.dll  PSAPI.dll  SHELL32.dll  SHLWAPI.dll  USER32.dll | 45/50  6/50  18/50  5/50  40/50  4/50  16/50 |
| VERSION.dll | 11/50 |
| WTSAPI32.dll  msvcrt.dll  ole32.dll | 2/50  6/50  41/50 |

## Integrated Features

This section involves combining the features from both dynamic and static analysis[8], it provides explanations behind the combination of the features selected for the research.

Why the combination of both static and dynamic features is because, malware programmers have developed complex and sophisticate methods to evade detection by antimalware engines[8]. Using the unique strings in both malware and benign hashes, I was able to train the machine to differentiate between malware and benign.

# Result set and Discussion

The pre-processed data is a csv file, containing the 50 ransomware of family locky and 50 benign wares, were equally divided into a set of 25 of each, for both malware and benign ware, used for training set, to be passed through WEKA using the specified machining learning classifier algorithm, Naïve Bayes, ZeroR, Decision Tree and Random Tree. The training set was passed to the machine learning classifiers and results were obtained.

With the results gotten from the training set, the remaining 25 malware and benign ware were re-evaluated as the test dataset, this is shown in detail below.

## Training set results

To obtain the training set results below, I manually separated my datasets into two sets of equal number of malware and benign (25 of each) making up half of the dataset, using Naïve Bayes, Random Tree, Decision Table and ZeroR classifiers and cross validation option set to 10-folds (which is the standard value of folds).

Using random tree classifier, I got an accuracy of 88% with cross validation of 10 folds, the classifier classifies all 24-benign and 20 malwares correctly, having a false positive of 0.120 and a true positive of 0.880.

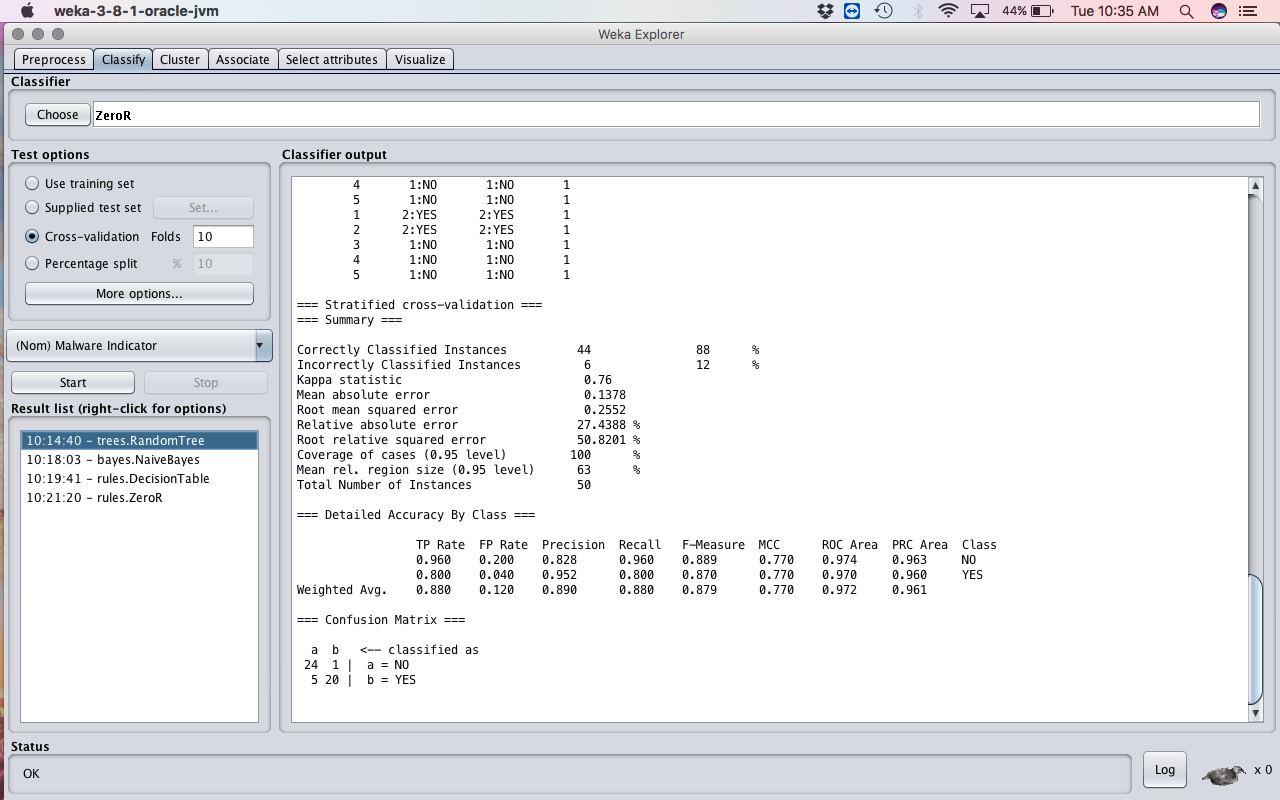


Figure 2 screenshot showing the use of random tree algorithm on triaining set

When Naïve Bayes was used it gave a much better result than that of the random tree classifier with an accuracy of 100% and no false positive, and a true positive of 1.000, these from this experiment proves to be the best result generated by any classifier.

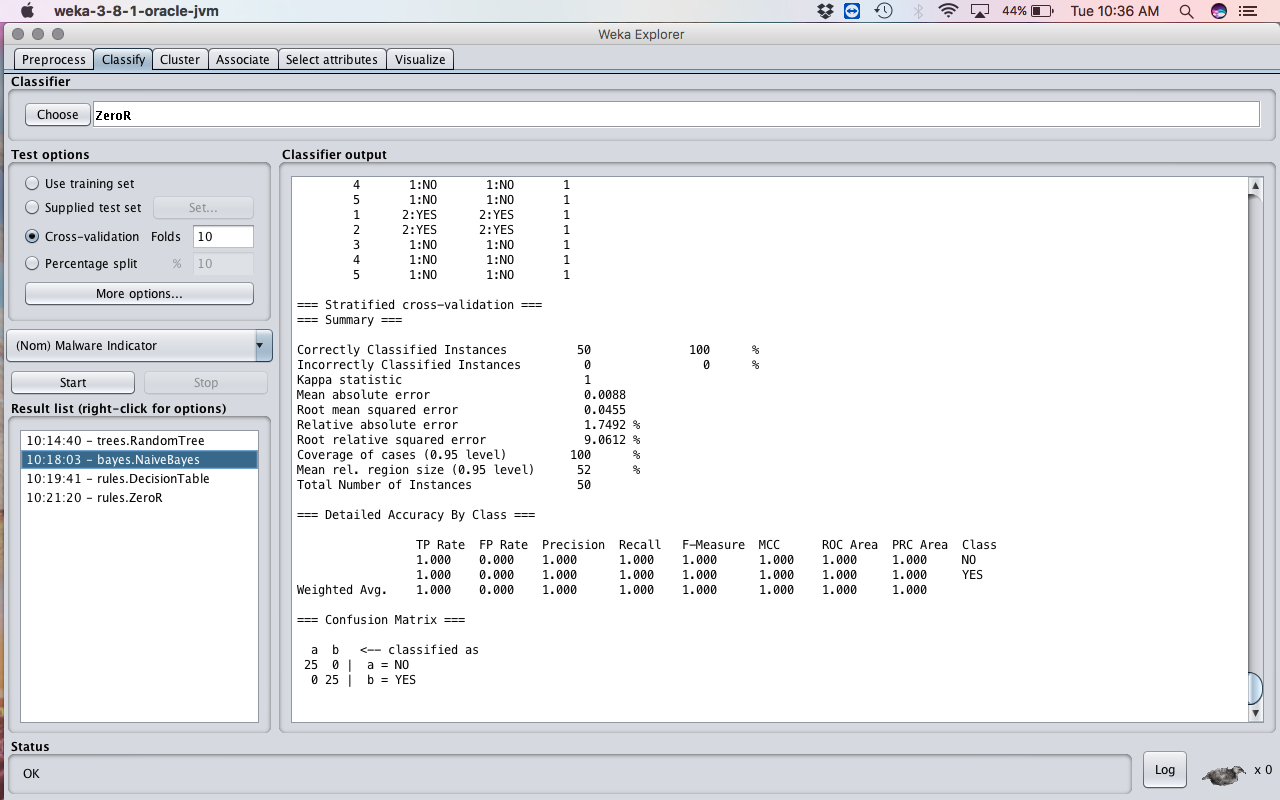


Figure 3 screenshot of the use of bayes.NaiveBayes on training set

Using decision tree classifier, I got an accuracy of 98%, false positive of 0.020 and a true positive of 0.900, which shows an improvement to the result gotten using random tree algorithm.

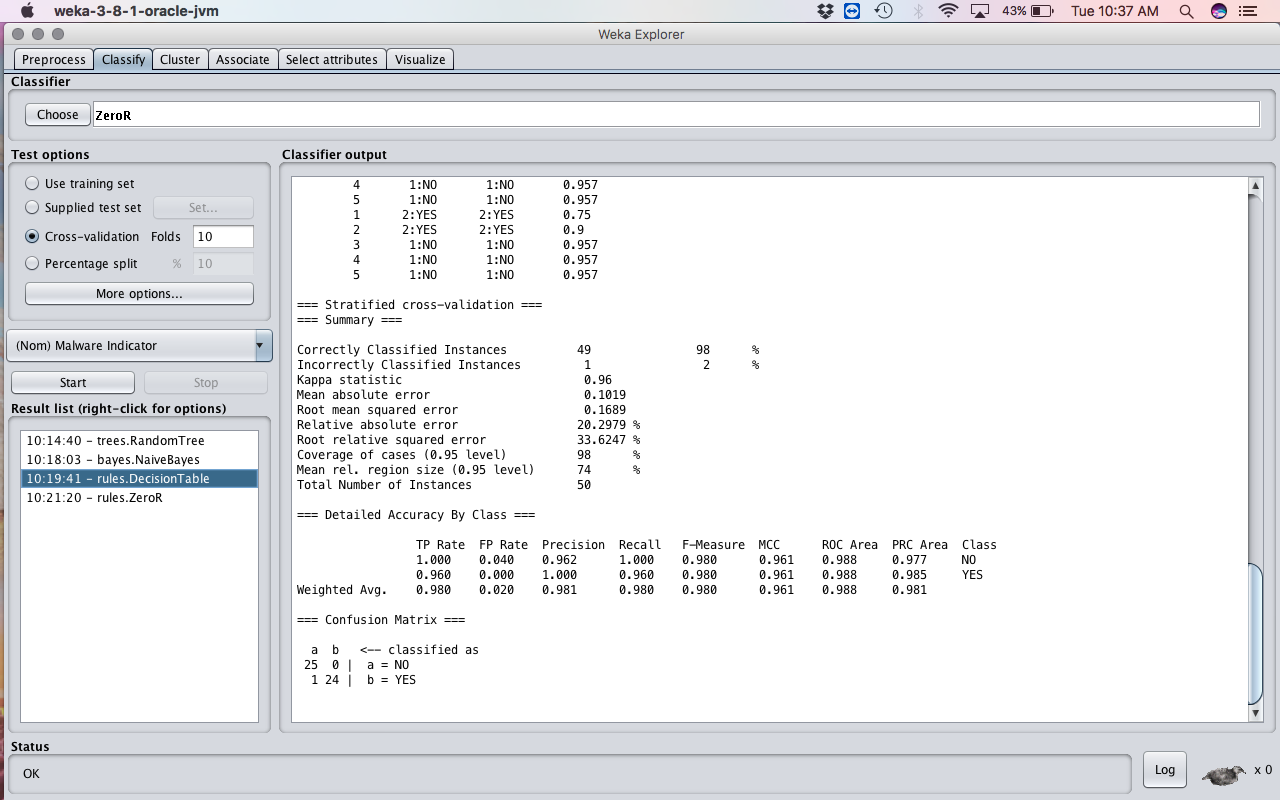


Figure 4 screenshot showing the decision tree classifier on training set

When I classified the training, data set with zeroR classifier, it yelled a poor result of 40%, true positive of 0.400 and false positive of 0. 600.This algorithm gave the worst result.

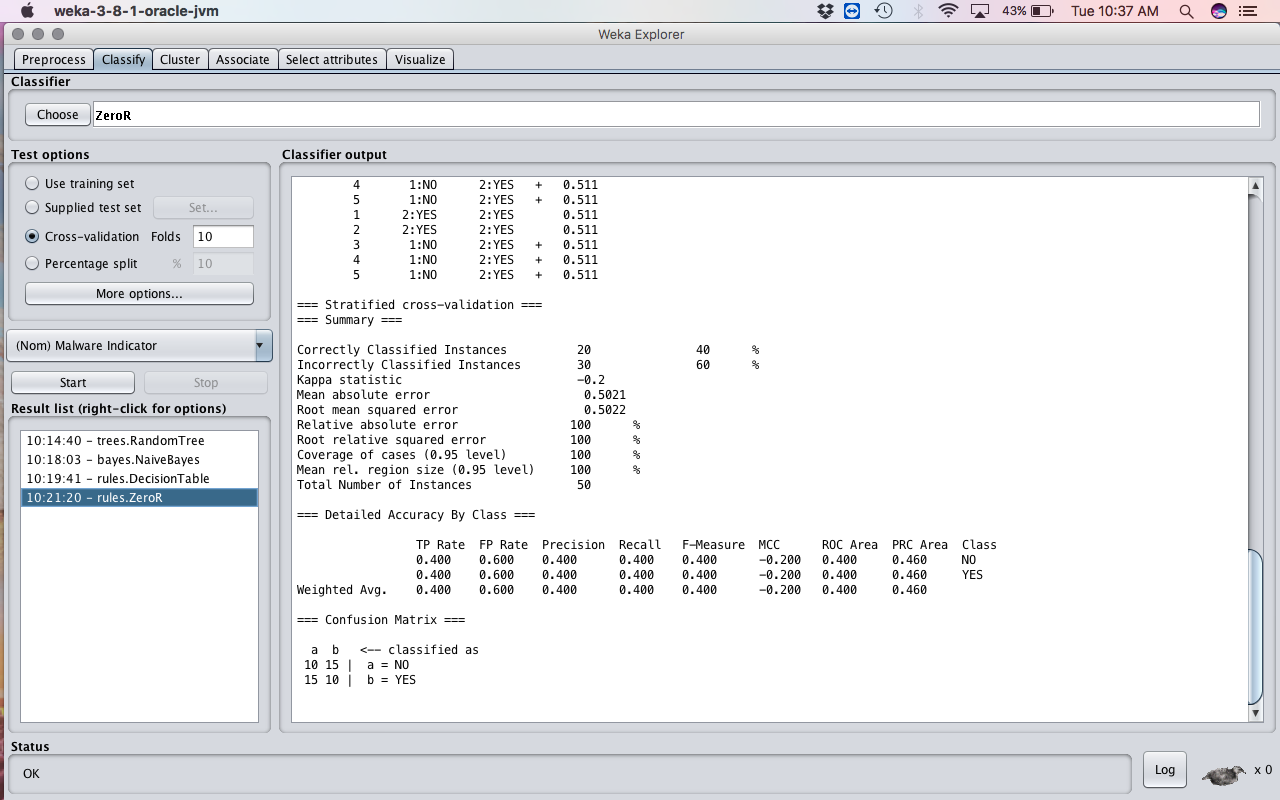


Figure 5 screenshot showing the use of zeroR classifier on training dataset

Table 0‑1 showing the result of using the classifiers on the training set

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Training Set  Meta-Classifier |  |  |
| Classifiers | TP | FP | Precision | Acc(%) |
| RT | 0.880 | 0.120 | 0.960 | 88 |
| NB | 1.000 | 0.000 | 1.000 | 100 |
| DT | 0.900 | 0.020 | 0.980 | 98 |
| ZeroR | 0.400 | 0.600 | 0.400 | 40 |

## Test set results

Using the results gotten from training the machine with the training dataset, I provided WEKA with the test dataset and used the re-evaluate option to generate the results for the test set.

Using the random tree trained results to re-evaluate the test set, I got an accuracy of 80%, true positive of 0.800 and false positive of 0.200. it classified all 25 of the benign ware correctly and 15 of the malware.

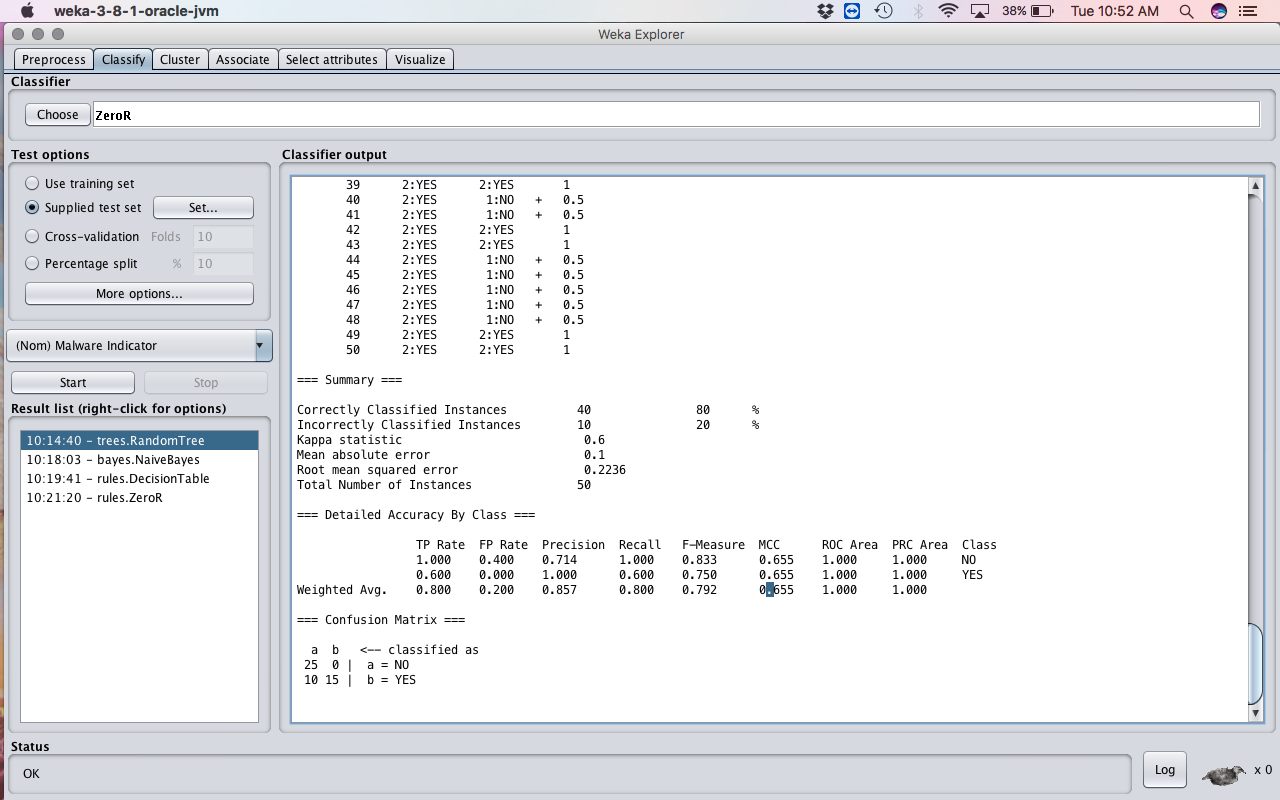


Figure 6 showing the result of reevaluating of the test data set using random tree classifier

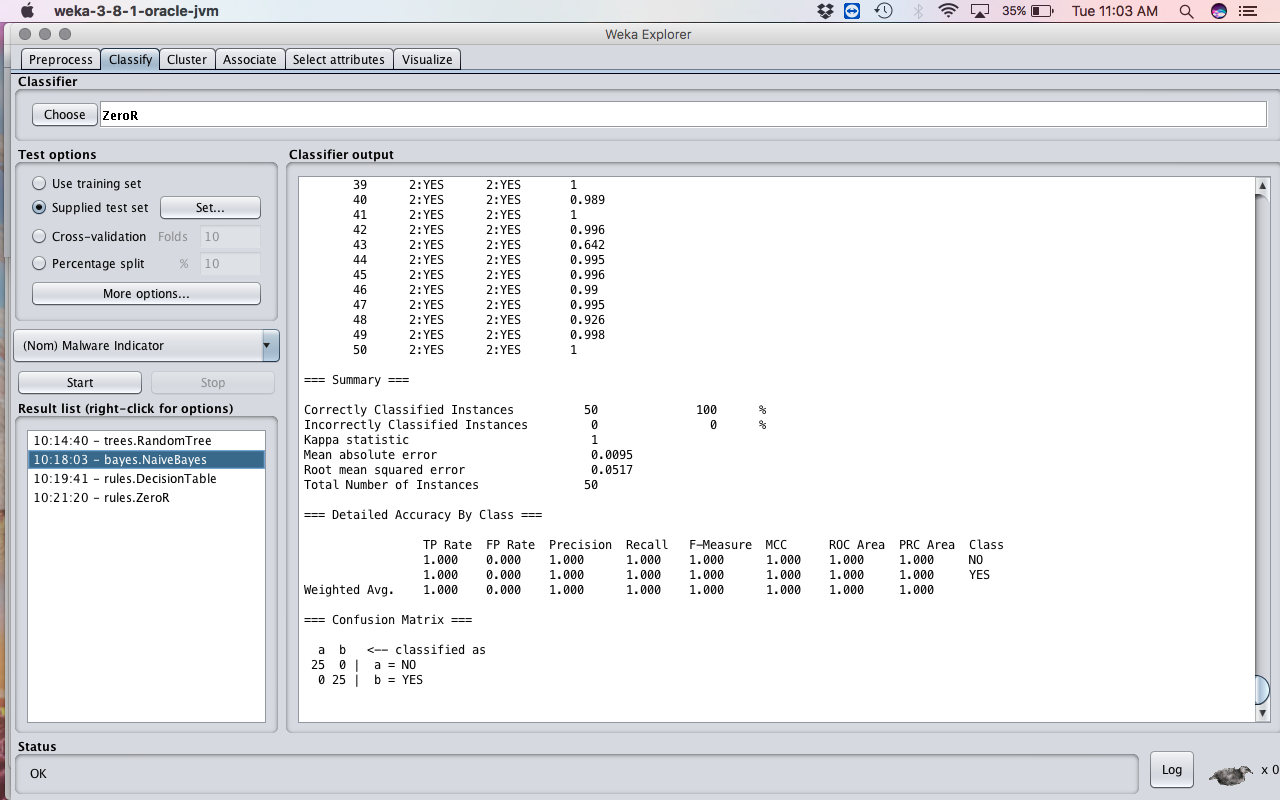
Naïve Bayes classifier when used, I got an accuracy of 100%, true positive of 1.000 and false positive of 0.000, using these algorithm has proved to be the best when it comes to classification in this experiment. 

Figure 7 showing the result of reevaluating of test dataset using naive bayes

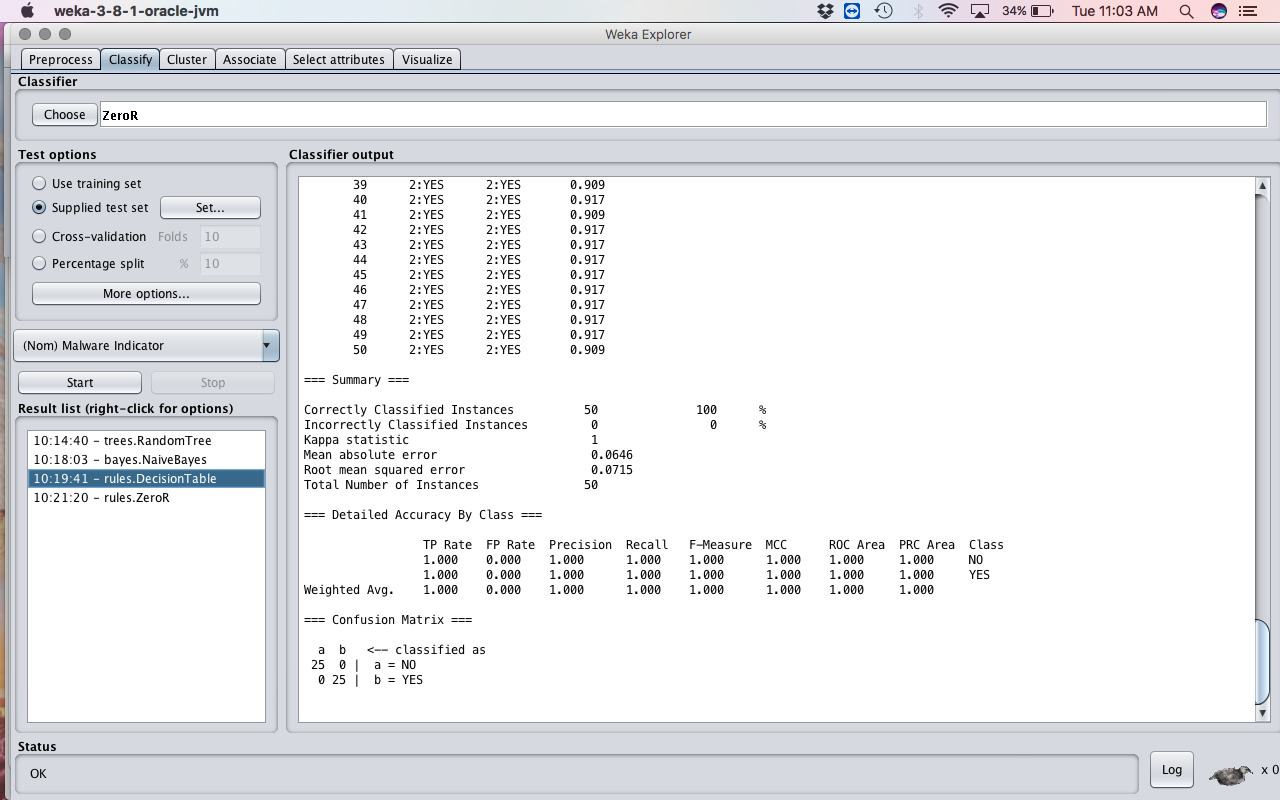
Using Decision tree classifier, I also had an accuracy of 100%, with true positive of 1.000 and false of 0.000, also using these algorithm is also recommended. 

Figure 8 showing the result of revaluating test dataset using decision tree

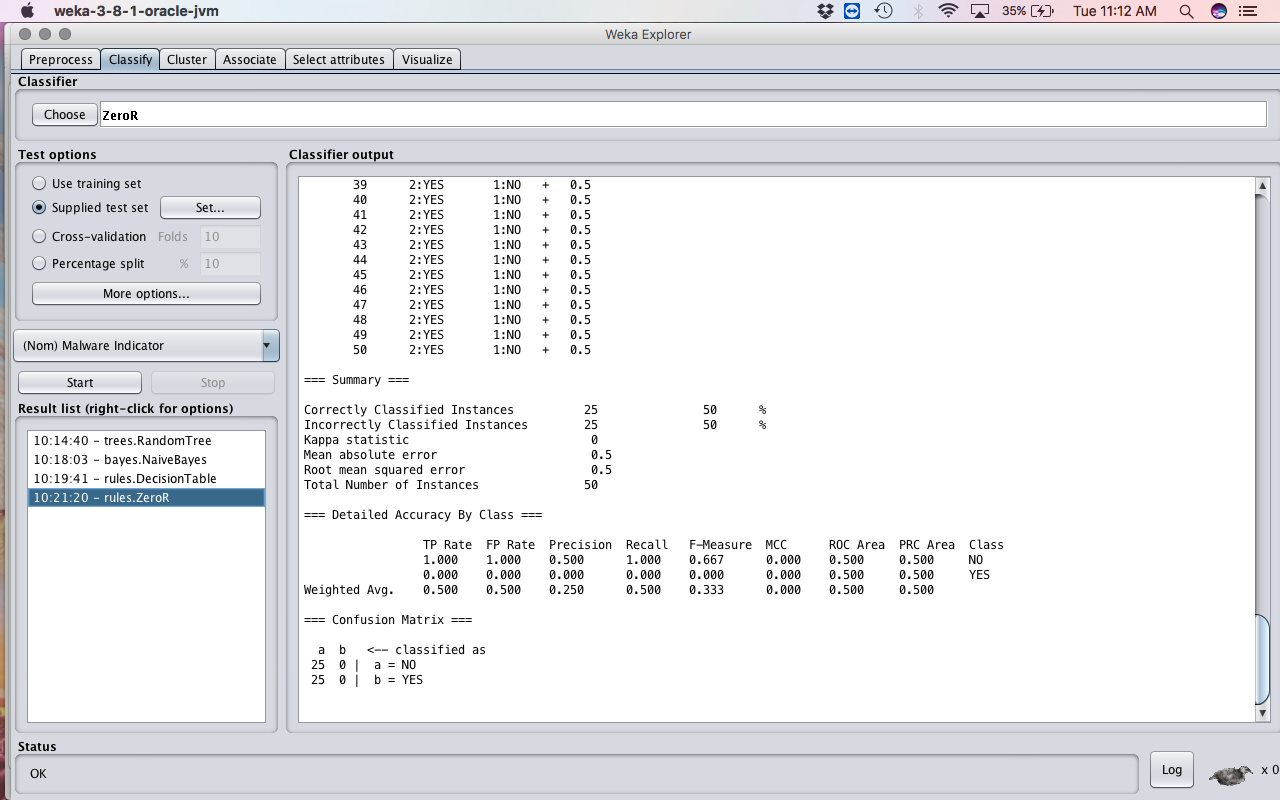
Re-evaluating with zeroR algorithm, I obtained an accuracy of 50% with a true positive of 0.500 and false positive of 0.500, it classified all the malware as benign ware, given this result shows among the four algorithms used these is the most unreliable. 

Figure 9 showing the result of reevaluating of test dataset using zeroR classifier

Table 0‑2 re evaluation of test dataset with trained classified results

Table 0‑3 Showing the test set reevaluated results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Test Set  Meta-Classifier |  |  |
| Classifiers | TP | FP | Precision | Acc(%) |
| RT | 1.000 | 0.400 | 0.714 | 80 |
| NB | 1.000 | 0.000 | 1.000 | 100 |
| DT | 1.000 | 0.000 | 1.000 | 100 |
| ZeroR | 0.500 | 0.500 | 0.250 | 50 |

Figure 10 showing a bar chart of the classification result

## Counter Measures

I have provided some recommendation on how systems would be able to prevent such malware attacks.

* Systems should make sure they only are able to run signed programs and avoid running on signed, (genuinely signed) programs
* System should not allow auto saving options of files.
* The System should avoid auto-run of programs as much as possible.
* Updating antivirus engines on the system may help in detecting such malicious programs.
* Control or confined applications that runs on the system, using for AppLocker Policy.

# Conclusion and Future work

The Random Forest, Naïve Bayes and Decision table classifiers have proved to be competent classifiers for the selected feature. And when used on the test set provide an improved result in most cases improving the true positive, increased precision and reduced false positive. The results gotten from this experiment proves the aims and objectives of this research work. The use of ransom ware attacks has doubled and is now the most predominant means of attack used in this century. Therefore, I assume that the investigative results on recent locky ransomware will as well be effective on new or future ones, but as malware developers find ways to modify and create advance malware programs, the research work done here might not be sufficient.

I propose to a continuous research on the use of machine learning techniques to detect advance malicious features as they evolve by searching regularly for specifically ransomware malware data and constant monitoring of these malicious codes for improved results.

# Reference:

[1] Michael Kassner, “Ransomware: Extortion via the Internet - TechRepublic.” [Online]. Available: http://www.techrepublic.com/blog/it-security/ransomware-extortion-via-the-internet/. [Accessed: 20-Jul-2017].

[2] A. Young and Moti Yung, “Cryptovirology: extortion-based security threats and countermeasures,” in *Proceedings 1996 IEEE Symposium on Security and Privacy*, pp. 129–140.

[3] N. Nissim, A. Cohen, and Y. Elovici, “ALDOCX: Detection of Unknown Malicious Microsoft Office Documents Using Designated Active Learning Methods Based on New Structural Feature Extraction Methodology,” *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 3, pp. 631–646, Mar. 2017.

[4] BBC, “Cyber-attack: Europol says it was unprecedented in scale - BBC News.” [Online]. Available: http://www.bbc.co.uk/news/world-europe-39907965. [Accessed: 20-Jul-2017].

[5] Paul Wagenseil, “New ‘Petya’ Ransomware Attack Spreads: What to Do.” [Online]. Available: https://www.tomsguide.com/us/petya-ransomware-attack,news-25389.html. [Accessed: 20-Jul-2017].

[6] ProofPoint, “August in November: New Information Stealer Hits the Scene | Proofpoint.” [Online]. Available: https://www.proofpoint.com/us/threat-insight/post/august-in-december-new-information-stealer-hits-the-scene. [Accessed: 20-Jul-2017].

[7] S. K. Sahay and A. Sharma, “Grouping the Executables to Detect Malwares with High Accuracy,” *Procedia Comput. Sci.*, vol. 78, pp. 667–674, 2016.

[8] P. V. Shijo and A. Salim, “Integrated Static and Dynamic Analysis for Malware Detection,” *Procedia Comput. Sci.*, vol. 46, pp. 804–811, 2015.

[9] S. M. Mohamed Belaoued, “An MCA Based Method for API Association Extraction for PE Malware Categorization,” *Int. J. Inf. Electron. Eng.*, vol. Vol. 5, No.

[10] H. V. Nath and B. M. Mehtre, “Static Malware Analysis Using Machine Learning Methods,” 2014, pp. 440–450.

[11] N. Milosevic, A. Dehghantanha, and K.-K. R. Choo, “Machine learning aided Android malware classification,” *Comput. Electr. Eng.*, Feb. 2017.

[12] A. Sharma and S. K. Sahay, “An Effective Approach for Classification of Advanced Malware with High Accuracy,” *Int. J. Secur. Its Appl.*, vol. 10, no. 4, pp. 249–266, 2016.

[13] R. Chen, W. Niu, X. Zhang, Z. Zhuo, and F. Lv, “An Effective Conversation-Based Botnet Detection Method,” *Math. Probl. Eng.*, vol. 2017, pp. 1–9, Apr. 2017.

[14] V. M. Afonso, M. Favero De Amorim, · André, R. Abed, G. · Glauco, B. Junquera, P. Lício De Geus, V. M. Afonso, M. F. De Amorim, A. R. A. Grégio, P. L. De Geus, and G. B. Junquera, “Identifying Android malware using dynamically obtained features,” *J Comput Virol Hack Tech*, vol. 11, pp. 9–17, 2015.

[15] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, “Data mining methods for detection of new malicious executables,” in *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001*, pp. 38–49.

[16] J. Gardiner and S. Nagaraja, “On the Security of Machine Learning in Malware C8C Detection,” *ACM Comput. Surv.*, vol. 49, no. 3, pp. 1–39, Dec. 2016.

[17] H. Sajedi and F. Allameh, “Detection of malicious web pages by evolutionary ensemble learning,” *Int. J. Hybrid Intell. Syst.*, vol. 13, no. 3–4, pp. 151–159, Feb. 2017.

[18] M. Norouzi, A. Souri, and M. Samad Zamini, “A Data Mining Classification Approach for Behavioral Malware Detection,” *J. Comput. Networks Commun.*, vol. 2016, pp. 1–9, Jul. 2016.