International Journal of Computer Applications Technology and Research
Volume 13–Issue 04, 07 - 17, 2024, ISSN:-2319–8656
DOI:10.7753/IJCATR1304.1002

Malware Attacks Classification Model

Ogwuche I.T.
Joseph Sarwuan Uni
Makurdi Benue State

Dr. Gbaden T.
Joseph Sarwuan Uni
Makurdi Benue State

Dr Ogala E
Joseph Sarwuan Uni
Makurdi Benue State

Yugh M.S
Joseph Sarwuan Uni.
Makurdi Benue State

Ekoja P
Ben State Poly
Benue State Nigeria

Abstract: This study developed a model to classify attacks in a digital economy system using Random Forest and support vector machine. The Rational Unified process research methodology was used to develop the model. It was implemented using the spyder notebook development environment and using Python programming language version 3.7. In this research, experiments were conducted to check the performance of the models based on the accuracy, precision, recall rate, and F1 – Score. From the results achieved, the classification metrics shows that the Random **Forest** Classifier scored 98.9% in accuracy, precision, recall rate, and F1 – Score. The classification metrics show that the Support Vector Machine scored 98.9% in accuracy, precision, recall rate, and F1 – Score. The experimental result implies that Random Forest Classifier and Support Vector Machine Classifier scored the same in performance when compared. This research contributed to the enhancement of threat classification and made a proper decision(s) as to the rate of occurrence of specific types of threats using Random Forest and Support Vector Machine.

Keywords: Threats, attacks, Digital Economy, Model

1. INTRODUCTION

The rapid spread of the use of technology aided by the internet has brought about so many transformations to our economic and social lives. This transformation is now termed as digital economy. Digital Economy allows and enhances trade of goods and services to be executed via electronic commerce on the Internet (Bukht and Heeks, 2017). Nigeria's cashless policies have accelerated internet use, enabling a digital economy. However, the digital economy faces security threats like smart threats, ransomware, and malware attacks, with malware being the most prevalent threat (Erdal and Milad, 2019).

Supervised machine learning is self-learning that depends upon labels from the data. A supervised learning algorithm identifies abnormal indications of deep threats concerning labeled outcomes as opposed to unsupervised learning (Hamad *et al.*, 2019).

Cyber security involves policies, processes, technologies, and techniques to protect computing resources, networks, software programs, and data from attacks. Tools like firewalls, antivirus software, intrusion detection systems, and IPS prevent attacks. However, the increasing number of internet-connected systems increases the risk of attacks. As attackers become more sophisticated, they develop zero-day exploits and malware that evade security measures

These vulnerabilities if not addressed will cause huge financial losses in the digital economy, therefore, against this backdrop, the research has developed a model that classifies attacks arising from vulnerabilities in systems and trains the model to detect these attacks.

This research classifies the topologies of attacks that threaten the security of digital services and recommends solutions for the protection of the digital economy and its services. The research used a secondary dataset from an online data source to train the model

2. LITERATURE REVIEW

Hansen, (2016) implemented a novel Monte Carlo tree search algorithm to forecast the presence of active malware in the training dataset. To forecast the new form of malware as the test data, a new search model is created in this research and applied to the active malware attacks. Using the search technique, it is difficult to forecast the high number of active malware.

Aghaeikheirabady,(2014) implemented different malware attack detection models on the training dataset using a comparison of the information in the user space memory data structures to expedite information extraction and ensure accuracy. In this method, malware artifacts related to registry modifications as well as calls to library files and operating system routines using descriptions of memory structures are recovered. Following an evaluation of the retrieved features, samples are categorized using the chosen attributes. The best outcomes show a 98% detection rate and a 16% false positive rate, demonstrating the efficiency of the suggested behavior extraction method. In this work, the data imbalance problem affects the true positive rate and error rate.

Selvi, (2019) offers a machine learning method that uses Random Forest to identify algorithmically produced domains only based on the lexical properties of the domain names. They particularly recommend combining other statistics gleaned from the domain name with masked N-grams. Additionally, they offer a dataset created for experimentation that contains domain names that were randomly and artificially produced from various malware families. They additionally group these families based on the domain creation algorithm used. As a result, masked N-grams offer detection accuracy that is comparable to that of other methods now in use while also offering substantially better performance. This approach can eventually be expanded to big, imbalanced datasets.

Takase *et al*.(2019) proposed employing values taken out of the CPU as part of a malware detection procedure. By utilizing processor information, they offload the malware detection technique to hardware while reducing the demand on the hardware's resources. Virtual machine QEMU was used to create a prototype of the method described. A demonstration on how the suggested technique can distinguish between malicious and good applications using processor information and can also identify malware variants that belong to the same family was done. However, evaluation results show that it is possible Trace Data could be incorrectly categorised based on the combination of training programs.

Ijaz*et al*. (2019) used dynamic malware analysis and different feature combinations. The various combinations are produced via APIs, Summary Data, DLLs, and RegKey Changed. Dynamic malware analysis is done with the help of the adaptable and accurate Cuckoo Sandbox. Using PEFILE, more than 2300 features are statically collected from binary malware and 92 features are dynamically extracted from malware analysis. From 10000 benign files and 39,000 dangerous binaries, static features are retrieved. In the Cuckoo Sandbox, 800 benign files and 2200 malware files are dynamically examined, and 2300 characteristics are extracted. Static analysis is 99.36% accurate, compared to 94.64% for dynamic malware analysis. Analysis of dynamic malware is ineffective due to malware's cunning and intelligent behavior. Due to regulated network behavior and limited network access, dynamic analysis has some restrictions and cannot be fully examined.

Foley *et al.* (2019) identified coupled attacks against two well-known objective function (OF). OF is one of the important features of routing protocol for low-power and lossy networks (RPL). They investigate their vulnerability assessments utilizing various simulated situations and machine learning algorithms. To do this, they developed a unique IoT dataset based on network and power parameters, which is implemented as a component of an RPL IDS/IPS system to improve information security. Regarding the results that were recorded, their machine learning approach is effective at spotting combination attacks against two well-known RPL OFs based on the power and network metrics, where MLP and RF algorithms are the most successful classifier deployment for both single and ensemble models.

Ieracitano *et al.* (2020), presented a novel statistical analysis-driven intelligent intrusion detection system (IDS). To extract more optimal, strongly correlated characteristics, the proposed IDS specifically blend data analytics and statistical methods with current developments in machine learning theory. By comparing it to the benchmark National Security Database (NSL-KDD), the suggested IDS are evaluated. Comparative experimental results demonstrate that, when compared to conventional deep and shallow machine learning and other recently proposed state-of-the-art methodologies, the planned statistical analysis and stacked Autoencoder (AE) based IDS obtain greater classification performance.

He and Kim,(2019) researched how well Convolutional Neural Networks (CNN) defends against the injection of redundant API. By converting malware files into picture representations and classifying the image representation with CNN, they created a malware detection system SPP or spatial pyramid pooling layers used to create CNN to handle input of different sizes. By assessing the performance of this system on both unaltered data and hostile data with redundant API injection, they assess the usefulness of SPP and picture color space (grayscale/RGB). Results indicate that greyscale imaging is effective against redundant API injection whereas naïve SPP implementation is problematic owing to memory limitations.

Ren*et al*.(2020) presented hierarchical clustering with a decision tree model, a hybrid malware detection model on a limited malware dataset. It has difficulty in identifying multi-class attacks on real-time malware dataset

Alhanahnah*et al*.(2019) transformed the program OpCodes into a vector space and applied fuzzy and fast fuzzy pattern tree algorithms for malware identification and categorization. Particularly for the fast fuzzy pattern tree, a high level of accuracy was achieved with a manageable run-time. Edge computing malware detection and categorization methods become more effective as a result of the robust use of both feature extraction and fuzzy classification. However, it applies to samples built on ARM and there are not enough samples in the dataset.

Given the various literature reviewed, the authors' works addressed the following issues:

i. Investigations on integrated risk and damage assessment in the digital economy using fuzzy approaches.
ii. Anomaly Detection on IOT system based on Random Forest.
iii. A hybrid malware detection model using hierarchical clustering with a decision tree to identify multi-class attacks on real-time malware datasets.
iv. Creating strategies for distinguishing between signatures for IOT malware.

Due to the unbalanced nature of datasets used in the reviewed works, problems of true positive rates and error rates were encountered. Also, the authors did not take into account the different categories of digital economy threats. It is as a result of the aforementioned drawbacks that this research adopts Random Forest and Support Vector Machine algorithms to classify digital economy threats into eight categories using a balanced dataset to compensate for the large true positive recorded in the reviews above

3. METHODOLOGY

The methodology adopted in this research work is Rational Unified Process (RUP). RUP is an iterative software development process framework that makes heavy use of Unified Modeling Language (UML) in its phases.

Support Vector Machine and Random Forest algorithms are used to develop a model for classifying malware attacks in the activities of the digital economy in Nigeria. A data set in the form of .csv file was used to train and test the model. The results helped evaluate how malware attacks can be prevented.

3.1 Analysis of Existing System

Malware classification and detection is an emerging prospect that seeks to resolve cybersecucurity concerns arising from the activities of fraudulent persons. Some research works such as Ren *et al (2020)* have been carried out and a hybrid malware detection model was built using hierarchical clustering with a decision tree model however; it had difficulty in identifying multi-class attacks on real-time malware dataset. In Nigeria, malware incidents are not detected and recorded real-time thereby creating room for malware incidences to go undetected and unabated. The common practice is the use of antivirus software which is inadequate against the evolving nature of malware because malware takes advantage of security vulnerabilities in hardware and network traffic.

3.2 Analysis of The Proposed System

Random Forest and Support Vector Machine was the machine learning algorithms used in the training and testing of our model

Random Forest (RF): is an algorithm for ensemble machine learning. Several decision trees (DT) are combined into a forest to operate this algorithm. DT is a frequently used data mining ML method that is able to solve classification and regression problems, it is highly suited for resolving classification tasks. This algorithm categorizes a population into branch-like segments that construct a reversed tree with a root node, internal nodes, and leaf nodes. The ML method is non parametric and can competently deal with large, complex, and complicated data without imposing a complex parametric structure. DT learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a DT. Learned trees can also be shown as sets of if-then rules to improve human legibility. The DT model makes analysis based on three basic nodes, namely:

• Root node: principal node based on this node all-other nodes functions.

• Interior node: handles various features or attributes of the dataset.

• Leaf node: represent the outcome of each test, i.e., the class of the dependent variable in classification problems.

The DT algorithm divides the data into two or more analogous sets based on the most significant indicators. The entropy of each feature is computed, and then the dataset is divided, with predictors having the minimum entropy. The formula for calculating the entropy of an attribute is shown in Eq. (1).

Entropy s ( ) c ∑ = − i = 1 p log 2 p i i (1)

Where in Eq. (1), c is the total number of classes and the probability of samples belonging to a class at a given node can be denoted as p i

The class with the highest votes becomes the model's forecast in the RF algorithm, which generates a class expectation from each distinct tree. The likelihood of improved accuracy increases with the number of trees in an RF classifier. While it is more effective at classification tasks and can overcome the drawbacks of DT and other ML approaches, like overfitting and missing values, it can be utilized for both regression and classification tasks (Ekle *et al,* 2023)

A supervised machine learning approach that may be applied to both regression and classification problems is the support vector machine (SVM). With support vector machines, data points are plotted as points in an n-dimensional space, where n is the number of features. The value of each feature is represented by a specific coordinate. By identifying the ideal hyperplane that best distinguishes the two classes, the classes are classified into their appropriate categories.

The SVM Classifier:

The hypothesis function is defined as:

$$h\,(x_i) = \begin{cases} +1 & if\ w.x + b \geq 0 \\ -1 & if\ w.x + b < 0 \end{cases}$$

Class +1 will be assigned to the point that is above or on the hyperplane, while class -1 will be assigned to the point that is below the hyperplane.

To compute the (soft-margin) SVM classifier, one must minimize the following expression:

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(w \cdot x_i - b)\right)\right] + \lambda\|$$

Since selecting a small enough value for lambda produces the hard-margin classifier for linearly classifiable input data, we concentrate on the soft-margin classifier.(Kecman, Vojislav, 2005)

3.3. System Design

The proposed system uses the support vector machine algorithm and random forest algorithm to classify and detect threats from nine families of attacks identified from security vulnerability in system and network traffic.

3.3.1 Architectural Model of the System

The architecture of the model consists of: –

i. data collection-data is from an online data source https://www.kaggle.com in the form of .csv file.

ii. Data Extraction- The dataset is UNSW-NB15 (computer network security dataset released in 2015).Nine families of attack namely Fuzzers, Analysis, Backdoors, DoS, Exploits, Reconnaissance, Shellcode, and Worms are extracted. The total number of records is two hundred and fifty seven thousand, six hundred and seventy three (257,673) The number of records in the training set is 175,341 records and the testing set is 82,332 records from different types of attack and normal.

iii. The data is normalized and then fed into the Random Forest and Support vector classifier to train and test the model. The results of the model will be displayed and evaluated based on the following parameters: F1, recall, accuracy, and precision. The architectural model is given in Figure 1. The architectural model is broken down into five phases as seen in figure 1.

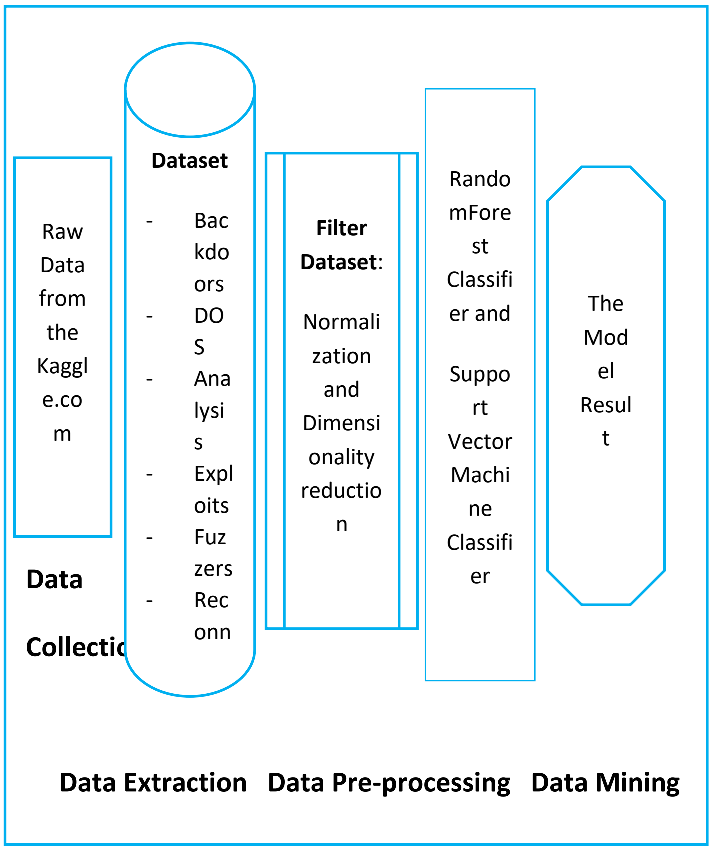


Figure 1: Architectural Model

3.3.2 Activity Diagram

is a system design tool that describes all the activities of a system from start to finish. The small black circle depicts the beginning of the activities of the system as seen in figure 2; the arrow is a pointer to the next activity. The machine learning algorithms in this study are random forest and support vector machine. There are two algorithms in this activity, so the arrow has to be split in two different activities support vector machine and random forest, so it requires an object called the fork node. The fork node in an activity diagram is used specify where a single activity is divided into two or more activity. The small, dark inner shaded circle with an outer unshaded circle marks the end of all activity. The activity diagram is shown in figure 2:

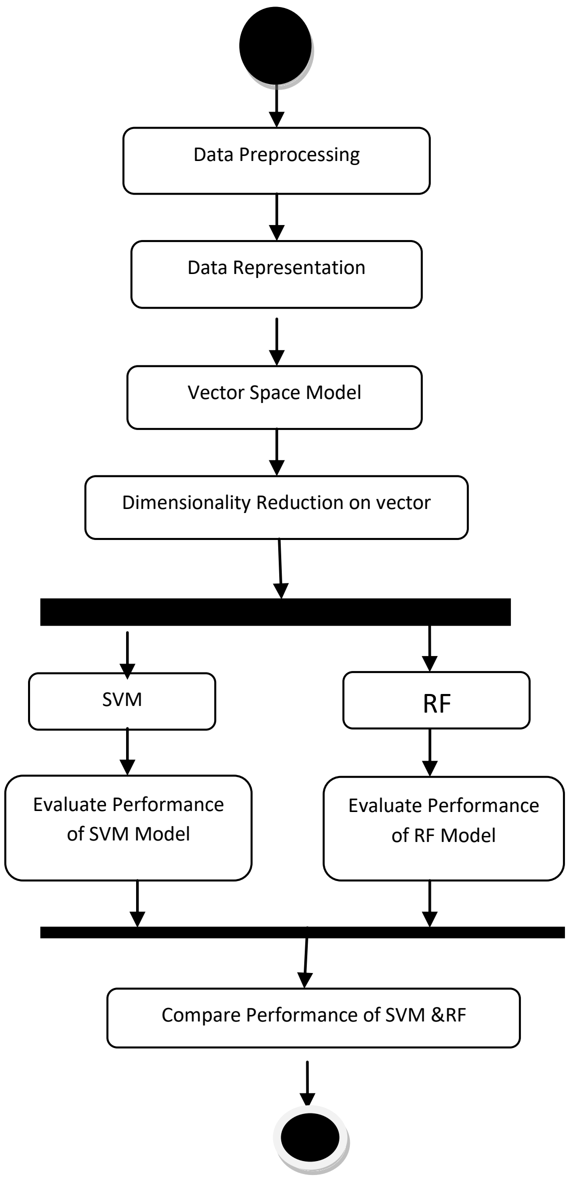

Data Preprocessing
Data Representation
Vector Space Model
Dimensionality Reduction on vector
SVM
RF
Evaluate Performance of SVM Model
Evaluate Performance of RF Model
Compare Performance of SVM &RF

Figure 2: Activity Diagram

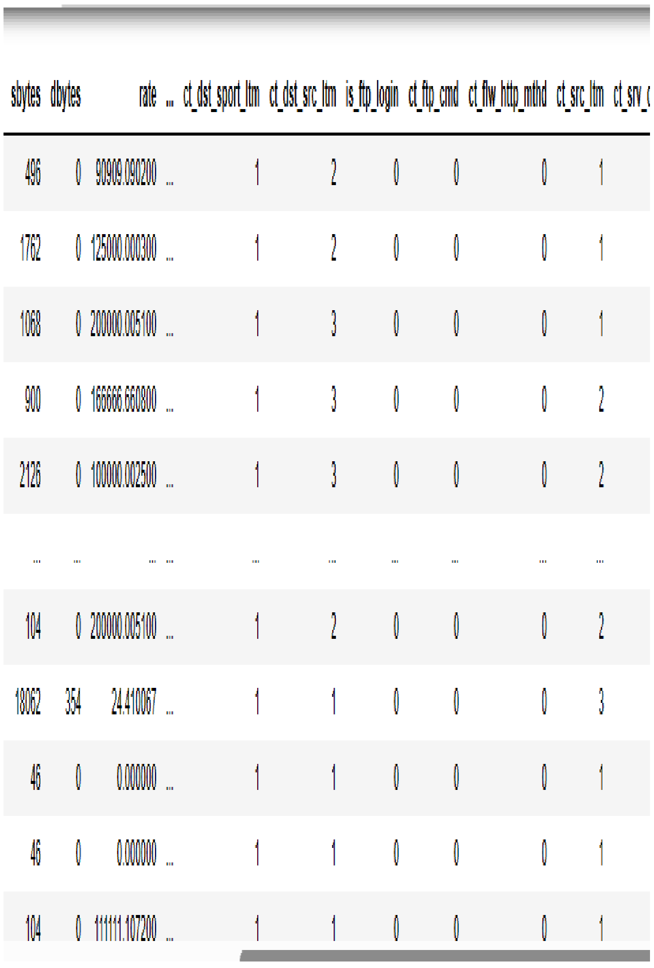
4. IMPLEMENTATION AND RESULTS

The threat classification technique is implemented using Python programming environment version 3.7. This chapter describes the nature of the datasets, data preprocessing, experimental set-up, experiment evaluation, and finally experimental results.

4.1 Description of Datasets

The experiment in this research was conducted on raw network packets of the data set created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviours. This data set has nine families of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The data set utilised 49 features with the class label. A partition from this data set is configured as a training set and testing set respectively. The number of records in the training set is 175,341 records and the testing set is 82,332 records. Table 1 represent an extract of the data set of digital economy threats used for the classification problem.

Table 1: An Extract of the Data set



sbytes	dbytes	rate	...	ct_dst_sport_ltm	ct_dst_src_ltm	is_ftp_login	ct_ftp_cmd	ct_flw_http_mthd	ct_src_ltm	ct_srv_
496	0	90909.090200	...	1	2	0	0	0	1	
1762	0	125000.000300	...	1	2	0	0	0	1	
1068	0	200000.005100	...	1	3	0	0	0	1	
900	0	166666.660800	...	1	3	0	0	0	2	
2126	0	100000.002500	...	1	3	0	0	0	2	
...	...	...	...	...	...	...	...	...	...	
104	0	200000.005100	...	1	2	0	0	0	2	
18062	354	24.410067	...	1	1	0	0	0	3	
46	0	0.000000	...	1	1	0	0	0	1	
46	0	0.000000	...	1	1	0	0	0	1	
104	0	111111.107200	...	1	1	0	0	0	1	

Table 2: Transposed Data set

Index	Rate	Attack_ Cat	Label
2709	16.4677	7	1
2710	29.8079	3	1
2711	125000	3	1
2712	58.466	2	1
2713	12.5000	4	1
2714	41.887	4	1
2715	3333333	7	1
2716	111111	4	1
2717	111111	7	1
2718	125000	7	1
2719	125000	2	1
2720	111111	3	1
2721	105.548	2	1
2722	333333	2	1

4.2 Data Preprocessing

The preprocessing on data set is done by removing the unwanted columns that are present in the data set. The columns named ‘rate’, ‘attack_cat’ and ‘label’are the features of interest; every other aredropped as it’s not needed. The ‘attack_cat’ column contains nominal values that cannot be used to train the models of interest, so we have converted the nominal values in the ‘attack_cat’ column into numeric values using the “One Hot Encoder” algorithm. Now, the data is ready to work with as seen in Table 2. Figure 6 indicates heat map for visualizing missing values in the dataset. Figure 7 indicates the dataset's correlation matrix. This matrix clarifies that the ‘label’ attribute is dependent on the ‘attack_cat’and‘rate’ attributes

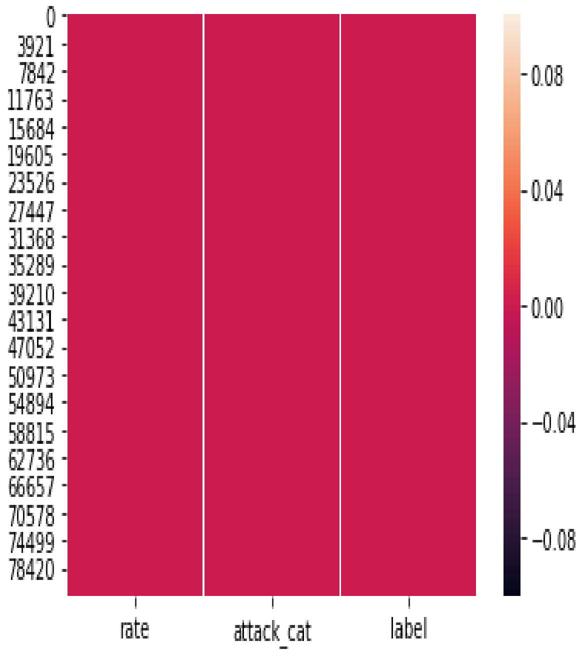


Figure 6: Heat map of the dataset

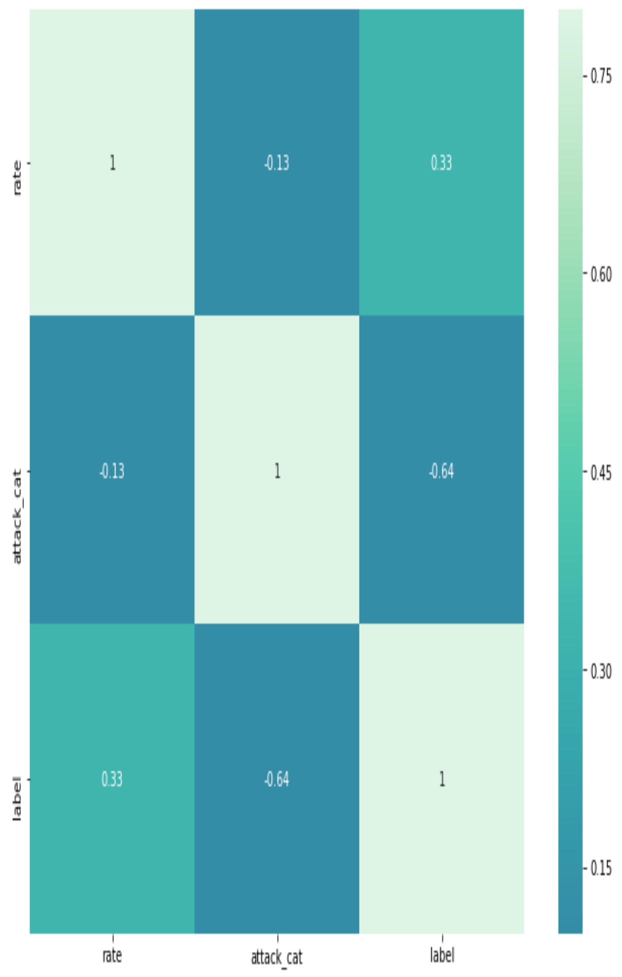


Figure 7: Dataset Correlation Matrix

4.2 Evaluation of Experiments

In this research, experiments were conducted to check the performance of the models based on the accuracy, precision, recall rate and F1 - Score of the models. The results of the experiments are recorded in Table 3 and Table 4 below. In Table 3, the classification metrics shows that the Random Forest Classifier scored 98.9% in accuracy, precision, recall rate, and F1–score. The classification metrics in Table 4 shows that Support Vector Machine scored 98.9% in accuracy, precision, recall rate, and F1 – Score. The experimental result implies that Random Forest Classifier and Support Vector Machine Classifier scored the same in performance when compared.

Table 3: Random Forest Classification Reports

Random Forest Errors:	0			
The Matthews correlation coefficient is:	1.0			
	Precision	Recall	f1-score	support
0	0.98	0.98	0.98	11147
1	0.98	0.98	0.98	13553
Accuracy			0.98	24700
macro avg	0.98	0.98	0.98	24700
weighted avg	0.98	0.98	0.98	24700

Table 4: Support Vector Machine Classification Reports

Support Vector Machine Errors:	0			
The Matthews correlation coefficient is:	1.0			
	Precision	Recall	f1-score	support
1	0.98	0.98	0.98	1903
Accuracy			0.98	1903
macro avg	0.98	0.98		
weighted avg	0.98	0.98		

5. DISCUSSIONS

5.1 Experimental Results and Discussions

The results of the experiment conducted in table 2 are shown and discussed in figures 8,9,10,11 and 12 respectively

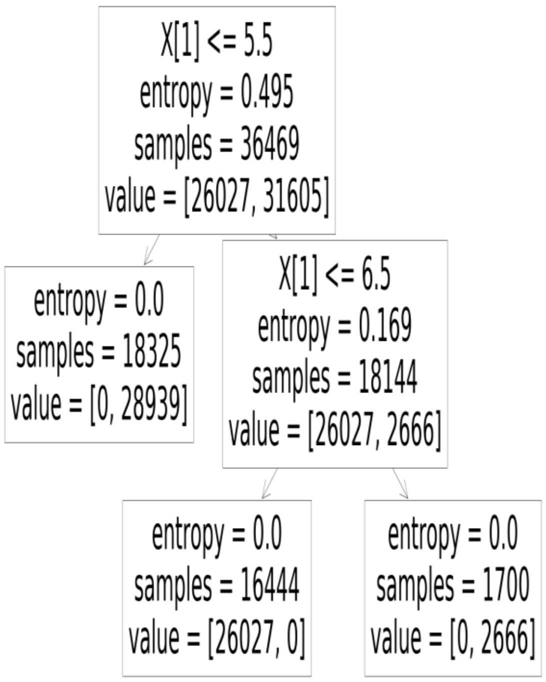


Figure 8: First Decision Tree

Figure 8 shows a decision tree with root node that consists of X [1] <= 5.5, entropy = 0.495, samples = 36469, and value = [26027, 31605]. Since the entropy value (0.495) is relatively high indicating uncertainty in our dataset, the root node is splitted into left (if X[1] <= 5.5 is True) and right (if X[1] <= 5.5 is False) nodes. The left intermediate node is now a leaf node (entropy = 0.0, samples = 18325 and value = [0, 28939]) and cannot be splitted further since it has zero entropy value which indicates certainty in our dataset. The right intermediate node (X[1] <= 6.5, entropy = 0.169, samples = 18144 and value = [26027, 2666]) will be splitted further into left (if X[1] <= 6.5 is True) and right (if X[1] <= 6.5 is False) nodessince it has entropy value (0.169) which indicates uncertainty in our dataset. The left (entropy = 0.0, samples = 16444 and value = [26027, 0]) and right (entropy = 0.0, samples = 1708 and value = [0, 2666]) nodes are now leaf nodes and cannot be splitted further since it has zero entropy values which indicates certainty in our dataset. Since the entropy values of all the leaf nodes are zero, it implies that information is 100% present and the path of the deepest leaf node rightmost is the best decision path since the node has the least number of samples (1700).

=

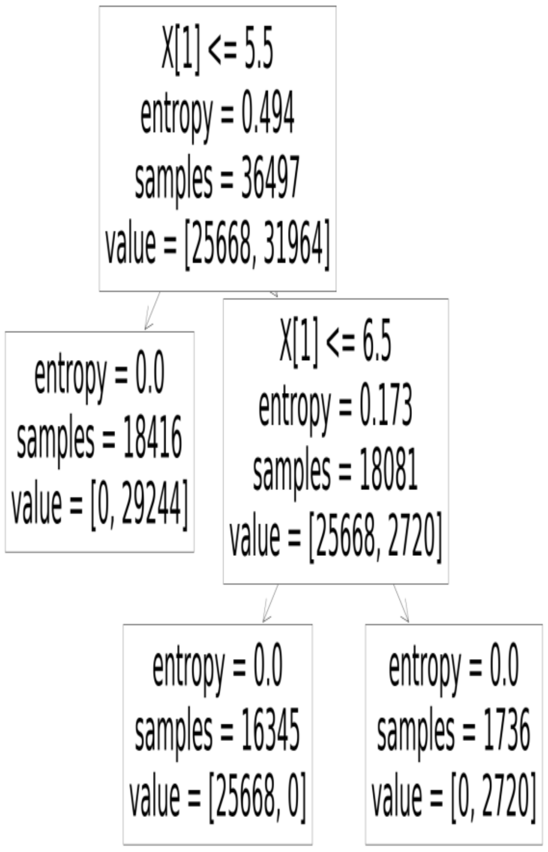


Figure 9: Second Decision Tree

Figure 9 shows a decision tree with root node which consists of X[1] <= 5.5, entropy = 0.494, samples = 36497 and value = [25668, 31964]. Since the entropy value (0.494) is relatively high indicating uncertainty in our dataset, the root node is splitted into left (if X[1] <= 5.5 is True) and right (if X[1] <= 5.5 is False) nodes. The left intermediate node is now a leaf node (entropy = 0.0, samples = 18416 and value = [0, 29244]) and cannot be splitted further since it has zero entropy value which indicates certainty in our dataset. The right intermediate node (X[1] <= 6.5, entropy = 0.173, samples = 18081 and value = [25668, 2720]) will be splitted further into left (if X[1] <= 6.5 is True) and right (if X[1] <= 6.5 is False) nodes since it has entropy value (0.173) which indicates uncertainty in our dataset. The left (entropy = 0.0, samples = 16345 and value = [25668, 0]) and right (entropy = 0.0, samples = 1736 and value = [0, 2720]) nodes are now leaf nodes and cannot be splitted further since it has zero entropy values which indicates certainty in our dataset. Since the entropy values of all the leaf nodes is zero, it implies that information is 100% present and the path of the deepest leaf node rightmost is the best decision path since the node has the least number of samples (1736).

Figure 10: Third Decision Tree

Figure 10 shows a decision tree with a root node that consists of X[1]<= 5.5, entropy = 0.494, samples = 36378, and value = [25707, 31925]. Since the entropy value (0.494) is relatively high indicating uncertainty in our dataset, the root node is splitted into left (if X[1] <= 5.5 is True) and right (if X[1] <= 5.5 is False) nodes. The left intermediate node is now a leaf node (entropy = 0.0, samples = 18462, and value = [0, 29204]) and cannot be splitted further since it has zero entropy value which indicates certainty in our dataset.The right intermediate node (X[1] <= 6.5, entropy = 0.173, samples = 17916 and value = [25707, 2721]) wassplitted further into left (if X[1] <= 6.5 is True) and right (if X[1] <= 6.5 is False) nodessince it has entropy value (0.173) which indicates uncertainty in our dataset. The left (entropy = 0.0, samples = 16208 and value = [25707, 0]) and right (entropy = 0.0, samples = 1708 and value = [0, 2721]) nodes are now leaf nodes and cannot be splitted further since it has zero entropy values which indicates certainty in our dataset. Since the entropy values of all the leaf nodes are zero, it implies that information is 100% present and the path of the deepest leaf node rightmost is the best decision path since the node has the least number of samples (1708).

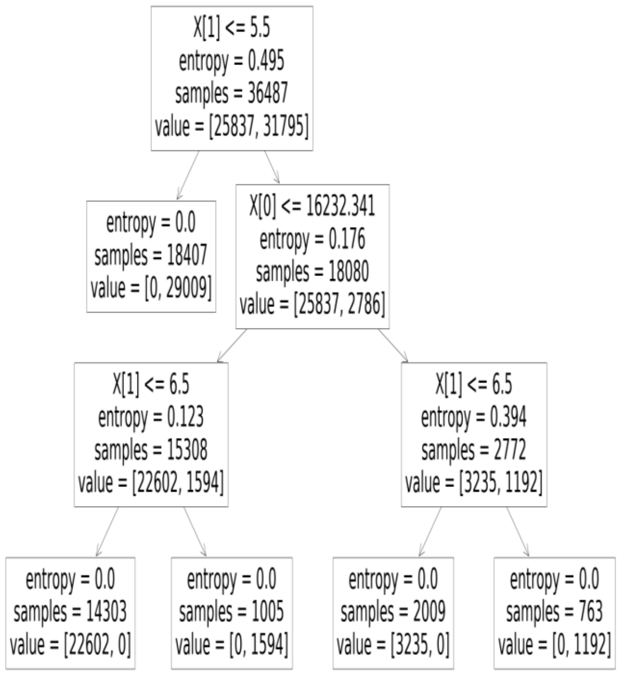


Figure 11: Fourth Decision Tree

Figure 11 shows a decision tree with root node that consists of X[1] <= 5.5, entropy = 0.495, samples = 36487, and value = [25837, 31795]. Since the entropy value (0.495) is relatively high indicating uncertainty in our dataset, the root node is splitted into left (if X[1] <= 5.5 is True) and right (if X[1] <= 5.5 is False) nodes. The left intermediate node is now a leaf node (entropy = 0.0, samples = 18407 and value = [0, 29009]) and cannot be splitted further since it has zero entropy value which indicates certainty in our dataset. The right intermediate node (X[0] <= 16232.341, entropy = 0.176, samples = 18080 and value = [25837, 2786]) wassplitted further into the left (if X[1] <= 6.5 is True) and right (if X[1] <= 6.5 is False) nodessince it has entropy value (0.176) which indicates uncertainty in our dataset. Repeat the split process until all nodes become leaf nodes with zero entropy values indicating that information is 100% present and the path of the deepest leaf node rightmost is the best decision path since the node has the least number of samples (763).

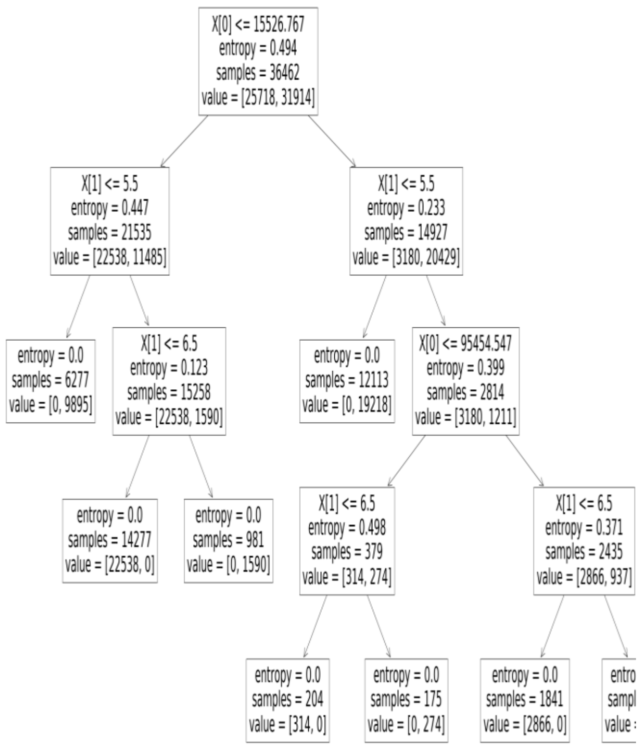


Figure 12: Fifth Decision Tree

Figure 12 shows a decision tree with a root node that consists of X[0] <= 15526.767, entropy = 0.494, samples = 36462, and value = [25718, 31914]. Since the entropy value (0.494) is relatively high indicating uncertainty in our dataset, the root node is splitted into left (if X[1] <= 5.5 is True) and right (if X[1] <= 5.5 is False) nodes. Repeat the split process until all nodes become leaf nodes with zero entropy values indicating that information is 100% present and the path of the deepest leaf node with samples = 175 is the best decision path since the node has the least number of samples.

6. CONCLUSION AND RECOMMENDATION(S)

6.1 Conclusion

This research motivation is to enhance threat classification in the digital economy and make a proper decision (s) as to the rate of occurrence of specific types of threats using Random Forest and Support Vector Machine. Numerous experiments were conducted on digital threat datasets using Random Forest and Support Vector Machine algorithms with different decision trees showing the best decision path. The experiment was conducted in Python version 3.7 programming environment. The experiment indicated 98.9% performance of the proposed classification techniques on the training dataset. Based on the analysis conducted the following conclusions were reached:

i. The both algorithm scored 98.9% in performance (accuracy, precision, recall and f1-score) of the classification technique onthe digital threats dataset.
ii. The Support Vector Machine is not suitable for training large-size dataset since it takes a lot of time to train.

6.2 Recommendation(s)

In this research, we described Random Forest and Support Vector Machine classification methods for detecting threats in a digital economy. In addition to the results shown in this research, the following recommendations were made:

i. The Support Vector Machine is too slow to train large chunks of dataset as in the case of our research dataset which can be a disadvantage, hence an alternative approach be employed to compensate for the drawback of the Support Vector Machine. The results of the Random Forest classification technique are tree-like structures that is easy to visualize compared with clusters of points.
ii. Since classification only indicates whether digital services are threat or not, therefore it will be interesting work to visualize the physical locations of digital threat services on a map.
iii. Implementation of the technique on different supervised machine learning algorithms to measure the performance.

REFERENCES

Aghaeikheirabady, M., Farshchi, S. M. R., & Shirazi, H. (2014, November). A new approach to malware detection by comparative analysis of data structures in a memory image. In *2014 International Congress on Technology, Communication and Knowledge (ICTCK) (pp. 1-4). IEEE.*

Alhanahnah, M., Lin, Q., Yan, Q., Zhang, N., & Chen, Z. (2018). Efficientsignature generation for classifying cross-architecture IoT malware. In *2018 IEEE conference on communications and network security (CNS) (pp. 1-9). IEEE.*

Bukht, Rumana & Heeks, Richards (2017). Defining, Conceptualising and Measuring the Digital Economy Centre for Development Informatics, University of Manchester, UK

Erdal Ozkaya, Milad Aslaner (2019). Hands-On Cybersecurity for Finance.Available *@subscription.packtpub.com*

Ekle*†,F.A, Agu†,N.M, Bakpo†,F.S, Udanor†,C.N and Eneh†,A.H.(2023)A machine learning model and application for heart disease prediction using prevalent risk factors in Nigeria.

Foley, J., Moradpoor, N., &Ochen, H. (2020). Employing a machine learning approach to detect combined internet of things attacks against two objective functions using a novel dataset. Security and Communication Networks, 2020, 1-17.

Hamad, A., Noor, F. and Arjun, B. C. (2019). Survey learning for Cybersecurity. *International Journal of Research in Electronics and Computer Engineering, 7(2). 2393-9028.*

Hansen, S. S., Larsen, T. M. T., Stevanovic, M., & Pedersen, J. M. (2016). An approach for detection and family classification of malware based on behavioral analysis. In *2016 International conference on computing, networking and communications (ICNC) (pp. 1-5). IEEE.*

He, K. and Kim, D. S. (2019). Malware detection with malware images using deep learning techniques. In 2019 *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE) (pp. 95-102). IEEE.*

Ieracitano, C., Adeel, A., Morabito, F. C. and Hussain, A. (2020). A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. Neurocomputing, 387, 51-62.

Ijaz, M., Durad, M. H., & Ismail, M. (2019). Static and dynamic malware analysis using machine learning. In *2019 16th International bhurban conference on applied sciences and technology (IBCAST) (pp. 687-691). IEEE.*

Kecman, Vojislav (2005) "Support vector machines–an introduction." *Support vector machines: theory and applications. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.1-47.*

Ren, Z., Wu, H., Ning, Q., Hussain, I.and Chen, B. (2020). End-to-end malware detection for android IoT devices using deep learning. Ad Hoc Networks, 101, 102098

Selvi, J., Rodríguez, R. J. and Soria-Olivas, E. (2019). Detection of algorithmically generated malicious domain names using masked N-grams. Expert Systems with Applications, 124, 156-163.

Takase, H., Kobayashi, R., Kato, M., & Ohmura, R. (2020). A prototypeimplementation and evaluationof the malware detection mechanism for IoT devices using the processor information. *International Journal of Information Security, 19, 71-81.*