International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 8, No.5 (Sep-2019)
http://dx.doi.org/10.12785/ijcds/080505

Network Based Intrusion Detection Using the UNSW-NB15 Dataset

Souhail Meftah[1], Tajjeeddine Rachidi[1] and Nasser Assem[1]

[1]*School of Science and Engineering, Al Akhawayn University in Ifrane, Ifrane 53000, Morocco*

Received 4 Nov. 2018, Revised 5 Jul. 2019, Accepted 15 Jul. 2019, Published 1 Sep. 2019

Abstract: In this work, we apply a two stage anomaly-based network intrusion detection process using the UNSW-NB15 dataset. We use Recursive Feature Elimination and Random Forests among other techniques to select the best dataset features for the purpose of machine learning; then we perform a binary classification in order to identify intrusive traffic from normal one, using a number of data mining techniques, including Logistic Regression, Gradient Boost Machine, and Support Vector Machine. Results of this first stage classification show that the use of Support Vector Machine reports the highest accuracy (82.11%). We then feed the output of Support Vector Machine to a range of multinomial classifiers in order to improve the accuracy of predicting the type of attacks. Specifically, we evaluate the performance of Decision Trees (C5.0), Naïve Bayes and multinomial Support Vector Machine. Applying C5.0 yielded the highest accuracy (74%) and F1 score (86%), and the two-stage hybrid classification improved the accuracy of results by up to 12% (achieving a multi-classification accuracy of 86.04%). Finally, with the support of our results, we present constructive criticism of the UNSW-NB15 dataset.

Keywords: Intrusion Detection, NIDS, UNSW-NB15, Data Mining, Decision Trees, SVM, Naïve Bayes, GBM, Logistic Regression, Attack Detection, Cybersecurity

1. INTRODUCTION

With the proliferation of network services exposed to the outside world (web servers, application servers, remote procedure call services, etc.), over the last couple decades the attack surface area has significantly increased. Therefore, enterprises and government institutions alike have paid increasing attention to the security of their information systems. This is understandable given the ever growing volume, value, and sensitivity of information collected and stored in digital form by network services. Despite increased attention, recent surveys noted that 90% of enterprises have reported a security attacks in their systems [1].

Attackers target vulnerable application software reachable through the network. Typically, an attacker will make use of carefully crafted pieces of code called remote exploits which are capable of granting or extending privileges on a computer system to unauthorized users. These exploits postulate the existence of software vulnerabilities in network services programs or security deficiencies in protocols. Such vulnerabilities can be the result of misconfigurations, faulty design or programming errors all of which can be taken advantage of remotely through the network. Examples include buffer overflow, integer overflow or format string vulnerabilities stemming from poor security coding practices, backdoors embedded in compromised software versions, misconfigured internet servers with cross-site scripting, SQL injection vulnerabilities or excessive file and directory controls [2].

An unauthorized mechanism designed to access system resources and/or data is called an intrusion [3]. Intruders follow different approaches to penetrating a system depending on whether the attack emanates from within or outside of the target network. The former are often carried out by a virus infection (reaching the target network via a USB key, or by email), while the latter, need to perform complex sequences of attacks in order to access the target network/server from outside. To counter both internal and external intrusions, Intrusion Detection System (IDS) are deployed by network administrators to protect key network and enterprise services from both internal and external intrusion attempts.

E-mail: S.Meftah@aui.ma, T.Rachidi@aui.ma, N.Assem@aui.ma

Two classes of IDSs have emerged, namely (i) those that operate on network traffic called Network Intrusion Detection Systems (NIDS) [4]; they are often collocated with Firewalls and use network traffic traces for detecting intrusions, and (ii) Host Intrusion Detection Systems (HIDS) [5,6]; this breed is deployed on each network host, and uses information other than network traffic to detect intrusions. Such information includes application activity, traces, system calls and their parameters. In this work we are exclusively concerned with NIDS.

A NIDS is a computing system that monitors network traffic, analyses it, and raises alarms, whenever a security violation is detected [3]. Traditionally NIDs have used signatures of attacks (in the form of presence or absence of flags in TCP/IP headers, and to a certain extent bits in the payload) to identify and declare intrusions. Signature based NIDSs are very effective against known attacks. Though they may be ineffective against zero-day attacks (attacks for which signatures have not been identified yet), they continue to be a major defense against intrusion and as such are widely deployed.

Over the years, the role of NIDSs has evolved from inspecting IP packet header only to include inspection of payloads of the protocol stack entities up to the application layer. These techniques are often referred to as *Deep Packet Inspection*, since not only they parse the IP datagram (as router do to perform their work) and TCP/UDP segment headers (firewalls filter packets not only by IP but also by source/destination port), the application layer headers and data. Therefore, more sophisticated techniques and computing power are required to be able to understand the application layer protocols such as HTTP, SMTP and the data they carry and perform inspection at wire speed. Both the research community, software editors and hardware manufacturers have dedicated a great deal of their efforts to improve the efficiency of such systems.

Solutions for high speed deep packet inspection can be roughly divided into software solutions [7] and hardware solutions [8-12]. The former focus on developing optimized algorithms and data structures to represent the signatures while the latter aim at accelerating the matching process using specialized structures [10-11]. Nevertheless, the separation is not strict as hardware techniques often rely on suitable data structures and algorithms to optimize their performance while software-only solutions deployed on general-purpose hardware are generally not sufficient to reach the required throughput.

However, with the ever increasing attack surface area, new attacks are crafted by intruders that bypass signatures all the time. In their Internet Security Report for 2018 [13], Symantec reports that attack variants globally increased by 54% clearly posing a challenge for signature based NIDs. For these reasons it is believed that in order to cope with the ever emerging new zero-day attacks, there is need for NIDs that can detect unconventional traffic and label it as a potential intrusion without having their signatures. This perspective gave birth to another class of NIDs called anomaly-based Intrusion Detection Systems (ABNID). ABIDSs are receiving increased attention as the proliferation attacks is starting to shift from a mere nuisance to a major threat [2].

ABNIDSs are systems that use Artificial Intelligence (AI) techniques such as (Bayesian Approach, Neural Networks, Fuzzy Logic, Genetic Algorithms, data mining, Machine Learning etc.) [14-17] to learn the characteristics of both intrusive and normal traffic from the network traffic, then use the learned models to detect attacks/intrusions without signatures [18].

The taxonomy of ABIDS split into four major categories, namely:

1. *Statistical anomaly detection:* This comprises Operational models or Threshold metric, Markov model or Marker models, Moments or Mean and Standard Deviation models, Multivariate models, and Time Series models.
2. *Data mining based detection:* such as Clustering, Association rule discovery, and Classification.
3. *Machine learning based detection:* This is composed of Bayesian models, Neural Networks, Fuzzy logic models, Genetic Algorithms, and Support Vector Machine.
4. *Knowledge based detection:* State transition analysis, Expert Systems, Signature Analysis, and Perti Nets.

One has to note that anomaly-based techniques have abundantly been applied to HIDS too learning from activities taking place inside a host computer [5]. The learned characteristics can either be in the form of parameter to general models (such as Markov models), rules, graph weights etc.

A major advantage of ABNIDSs is their ability to improve their own detection performance by learning the patterns from network traffic on the fly, thus being able to detect unknown attacks; however, this is also a cause for concern as false alarms can be triggered. Such false alarms can happen both ways i.e., labeling a normal traffic and intrusive (False Positive) or labelling intrusive alarm as normal (False Negative). Often anomaly-based detection can be tuned to minimize either false alarms depending on the context of application (public network services vs. private network services) at the expense of the detection rate.

A key challenge for ABNIDS systems is therefore to increase the detection rate while decreasing/minimizing false alarms. To standardize the evaluation of the performance of ABNIDS standard metrics have been devised by the research community. These metrics factor in not only the detection rate but also the false alarms generated by the system under evaluation.

Contrary to signature based NIDs Anomaly-based NIDSs are systems require a training dataset in order to learn models for both intrusive and normal traffic. The quality of such datasets is paramount in order for such systems to learn and perform well. Good data sets should among other things be representatives of attacks, balanced [19], and contain traces of both normal and intrusive traffic etc. However, for the longest time ABIDSs have been trained using datasets such as KDDCUP 99, NSL-KDD, DARPA, and ADFA, which date from over two decades and are no longer representative of modern networks and attack environments. Hence, a new dataset is needed to be able to progress in the field of anomaly based NIDS.

Motivated by the introduction of the state of the art new network intrusion detection dataset UNSW-NB15, this work aims at developing a hybrid (two-stage) classification-based network intrusion detection system that tests the efficiency, effectiveness and robustness of that dataset.

In order to solve the complex classification problem at hand, we will, after appropriate data preparation and feature extraction using Recursive Feature Elimination (RFE) and Random Forests (RF) [20,21], proceed to a two-step process (see Figure 1), which consists of a binary classification benchmarking Logistic Regression, Support Vector Machine and Gradient Boost Machine to learn the various types of attacks with the output of the best classifier fed to a second-stage multi-classifier for improving performance metrics (e.g., accuracy) for each type of attack. The second-stage classifiers considered in this study are Naïve Bayes, Support Vector Machine and the C5.0 algorithm for Decision Trees. The usage of a wide range of statistical analysis will help us extract insight on a multitude of factors pertaining to the UNSW-NB15 dataset.

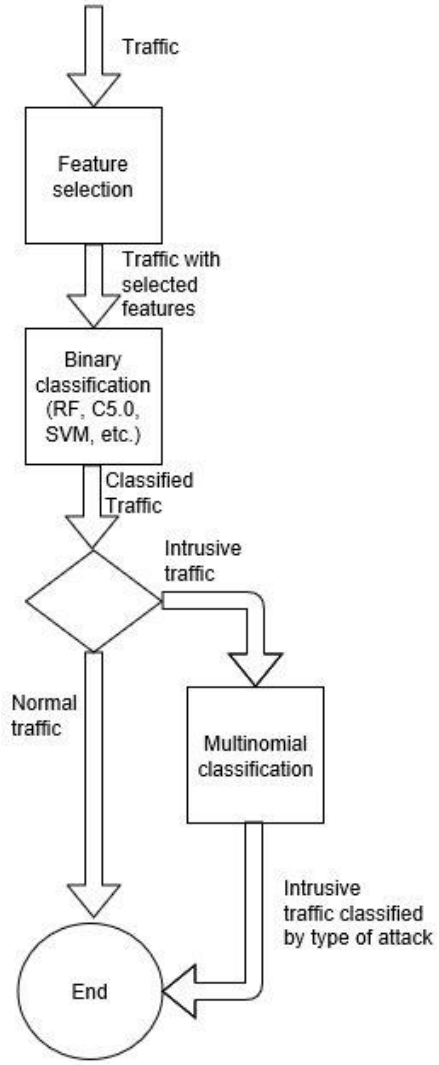

Figure 1. Proposed two-stage detection process with feature selection.

The remaining of this paper is organized as follows: Section 2 showcases some of the related works that have been carried out in the field. Section 3 provides an overview on Network Intrusion Detection. Section 4 is a high level description of the UNSW-NB15 dataset. Section 5 exposes our methodology for intrusion detection before discussing the results in Section 6. Finally, Section 7 concludes the work and presents our future intentions.

2. Related Works

Numerous works have been carried out using historical datasets, such as the KDDCUP 99, NSL-KDD, DARPA and the ADFA dataset trying to come up with the best anomaly-based NIDS. This includes implementation of various techniques and data mining algorithms for Intrusion Detection Systems. In [18], the authors contrasted the performance of UNSW-NB15 with the KDD 99 dataset. The data mining techniques employed in

their approach were Decision Trees, Logistic Regression, Naïve Bayes, ANN and EM clustering. Results showed poorer performance of the UNSW-NB15 compared to the KDD 99. They explained the results by the nature of their dataset which, in order to effectively model modern networks, added to the complexity of the problem. In [22], a study of the significant features of the UNSW-NB15 dataset, the authors of the dataset performed feature selection using an Association Rule Mining (ARM) technique in order to evaluate the precision of results with regards to each attack rather than limiting performance metrics to only the overall accuracy. The algorithms adopted for that study were Naïve Bayes and Expectation Maximization, both of which did not yield satisfactory results. Table I depicts very low accuracy percentages coupled with high false alarm rates resulting of a study conducted by the dataset's authors.

TABLE I. THE EVALUATION OF THE UNSW-NB15 DATASET AS REPORTED IN [22]

Category	The UNSW-NB15 Features			
	NB		EM	
	Acc.	FAR	Acc.	FAR
Normal	0	100	45.27	54.73
Analysis	0	100	0	100
Backdoor	20	80	0	100
DoS	71.1	28	0	100
Exploits	54.6	46.41	74.5	24.67
Fuzzers	33.2	66.8	23.5	76.8
Generic	94.3	5.68	95.1	4.81
Reconnaissance	69.9	30	0	100
Shellcode	0	100	0	100
Worms	0	100	0	100
Average	37.5	62.58	23.83	75.80

In a different attempt [23], the authors of the dataset used the Central points of attributes values and Association rule mining for feature selection on a high level of abstraction. Without going into the details of detecting each type of attack independently, they extracted a set of feature that would optimize the overall accuracy of the model used, despite the low precision that can be obtained on different types of attacks. The work contrasted UNSW-NB15's performance with NSLKDD using EM, LR, NB and other data mining techniques. The results were more satisfying than their previous attempts; especially that it took processing time optimization into account.

Recently, multiple research groups demonstrated interest in the potential of the UNSW-NB15 dataset in Intrusion Detection. Different techniques and methods were used in multiple papers to determine which machine learning algorithms are best fit to the dataset's properties in order to optimize its results. In [24], the authors demonstrated that the Averaged One Dependence Estimator (AODE) ML algorithm has the potential to reach up to 98.5% true positive rate (TPR) on certain types of attacks (i.e. generic). However, due to the complex nature of the dataset, TPR can drop to as low as 16.7% on other types of attacks (i.e. Backdoor). Similarly, [25] experimented with Random forests after careful data-preprocessing using principal component analysis. Their results showcases a Precision and Recall of respectively 84.9% and 85.1%. A recent study conducted in [30] explored initial results for various algorithms on binomial detection only (intrusion, normal) and found that RF performed the best. This study omitted to use feature selection, which potentially can change the performance of other algorithms. Moreover, detecting the type of attacks (i.e., multinomial detection) was left for a further study.

In our study, one of the objectives is to evaluate the dataset over other ML/data mining algorithms than previously mentioned. [26, 16] emphasize that ensemble methods other than Random Forest have not been explored and have the potential to vastly improve the F1 performance on minority classes. Although papers like [27] tackled a large range of algorithms, no attempt on the improvement of performance metrics has been made through the introduction of a hybrid classification solution. Additionally, in the absence of reporting on the data pre-processing, feature selection, training and testing sets in [27], we were unable to reproduce and validate the extremely high accuracies reported in their paper.

3. DESCRIPTION OF THE UNSW-NB15 DATASET

TABLE II. DESCRIPTION OF ATTACK CATEGORIES OF THE UNSW-NB15 DATASET

Traffic Type	Description
Normal	Traffic that contains no threat
Fuzzing	Automated process of finding 'hackable' software bugs by randomly feeding different permutations of data into a target program until one of those permutations reveals a vulnerability
Analysis	Generic type for describing port scanning, spam and html file penetration
Backdoor	Malware type that negates normal authentication to grant remote access to resources such as databases and file servers
DOS	Deprives legitimate users from using web services through flooding the network/server with invalid authentication attempts forcing it to crash or stall
Exploits	Code that take advantage of a software vulnerability or security flaw. Often incorporated into malware, allowing a fairly easy and fast propagation.

Generic	Collision attack on the secret keys of ciphers. Works against all block ciphers.
Reconnaissance	Collection of simple techniques that gather information about the target network/server, such as *nmap*.
Shellcode	Set of instructions/statements which are injected and executed by a flawed program. Directly manipulates registers and the functions of a program
Worms	Malicious self-replicating code. Consumes too much system memory and network bandwidth. Decreases availability of systems

The effectiveness of a Network Intrusion Detection System is evaluated based on their performance in accurately identifying attacks which requires a comprehensive data set that contains normal and abnormal traffic [28]. Older datasets that have been used for NIDS research are NSLKDD, KDD98, KDDCUP 99, CIDDS-001, DARPA and ADFA [29]. This is illustrated through the wealth of studies in the area. Nonetheless, these datasets (most of which conceived over two decades ago) have a number of limitations that make of them not very reliable and outdated. Using such datasets can no longer be considered as accurate or comprehensive representation of modern attack environments, and algorithms trained on these will not reflect realistic output performance. These datasets do not include recent attack types and misrepresents normal traffic in such a way that stealthy/spy attacks can easily slip in as normal behavior. Other existing limitations are dataset specific, such as: - unbalance of the number of records from different types of traffic – non-comprehensive training sets that do not describe all attacks present in the testing set – absence of validation work – data generation methods – Low data rates, etc.

For all of these reasons, the Australian Center for Cyber Security (ACCS) in collaboration with a number of researchers worldwide took the initiative to counter this challenge and create the UNSW-NB15 dataset. The IXIA PerfectStorm tool was utilized to generate a rich hybrid set of normal and abnormal modern network traffic [28]. The IXIA tool proactively harvests and aggregates publicly known vulnerabilities and exposures relative to information systems security. It represents a valuable source of information on modern public attacks.

TABLE III. TRAINING AND TESTING SET RECORD DESTRIBUTION

Category	Training Set	Testing set
Normal	56,000	37,000
Analysis	2,000	677
Backdoor	1,746	583
DOS	12,264	4,089
Exploits	33,393	11,132
Fuzzing	18,184	6,062
Generic	40,000	18,871
Reconnaissance	10,491	3,496
Shellcode	1,133	378
Worms	130	44
Total Records	175,341	82,332

The UNSW-NB15 dataset tries to simulate modern network environments through embedding most modern low-key attacks. Table II details the ten types of traffic featured in the dataset: Normal, Fuzzing, Analysis, Backdoor, DOS, Exploits, Generic, Reconnaissance and Worms. The detailed distribution of these categories in terms of number of records per attack as well as training/testing set distribution are illustrated in Table III.

4. OUR PROPOSED METHODOLOGY

A. *Evaluation Metrics*

In order to evaluate the effectiveness of algorithms trained on the UNSW-NB 15 dataset, a number of metrics are taken into consideration; namely, TP, TN, FP, FN, where TP (True positive) is the number of the correctly classified attacks, TN (True Negative) denotes the number of the correctly non-attack rows, FP (False Positive) is the number of the misclassified attacks, and FN (False Negative) refers to the number of the misclassified non-attack records. Based on these we define *recall, accuracy, precision* and F1 score as follows:

$$Recall = \frac{TP}{TP+FN} \quad (1)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

$$F1 = 2 . \frac{Precision . Detection}{Precision + Detection} \quad (4)$$

Considering that our study consists of both binary and multi classification and given the unbalanced nature of the dataset at hand in terms of number of entries, accuracy alone may not be sufficient as a performance evaluation metric. Precision and Recall have to be taken into account in comparing the different ML algorithms. An effective way of doing so is to consider the F1-score which is the harmonic mean of precision and recall as explained in equation (4).

B. Data Pre-Processing

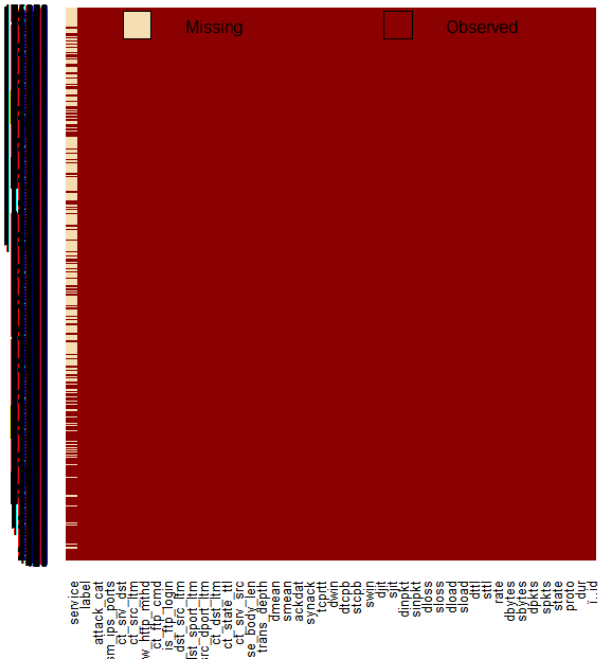

Figure 2. Missing vs. observed values for the training set. Horizontal axis indicates features. Vertical axis indicates dataset records. White spots indicate missing feature withing a record. Feature "service" (first in the list) is largely missing.

The sheer volume and dimensionality of the data makes it compulsory to perform proper data pre-processing to clean and format data prior to any training. The first thing to account for is that some of the data may be missing or corrupt. As shown in Figure 2 the data in UNSW-NB 15 are mostly complete except for the feature "service", which we chose based on this result to discard from our sample in order to avoid noise.

After this initial cleansing and formatting of UNSW-NB 15 dataset, the horizontal complexity imposes some preliminary feature selection to guide our tests. For that purpose, we use the Random Forest (RF) algorithm with 10-Fold Cross Validation. The tree-based strategies utilized by random forests inherently provides the ability to rank features based on how well they improve the purity of the node.

The mechanics in using RF are based on the following: When building the multiple Decision Trees, a score is assigned for each feature. This score represents how much this feature, when used in a tree node or more, reduces the impurity over the whole forest. These scores are then scaled and used as an index of feature importance.

The produced results summarized in Figure 3 show us a ranking of all the features of our dataset based on their impact. This further highlights the inter-dependencies between different features in the dataset, which if used independently, may severely damage the quality of the results. From this analysis we deduced that the 5 top features providing 98% accuracy to work with are: **ct_dst_src_ltm** (percentage of connections of the same source to the same destination address during the last recordingtime), **ct_srv_dst** (percentage of connections containing the same service port and destination address during the last recording time), **ct_dst_sport_ltm** (percentage of connections containing the same destination address and source port during the last recording time), **ct_src_dport_ltm** (percentage of connections containing the same source address and destination port during the last recording time), and **ct_srv_src** (percentage of connections containing the same service and source address during the last recording time) .

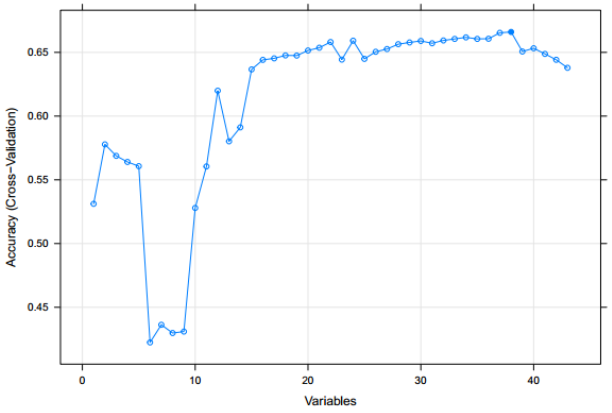

Figure 3. RFE/RF feature selection graph

C. Logistic Regression

Subsequent to the feature selection step, we proceed with the first stage of intrusion detection learning process. This stage consists in a binary classification benchmark using a number of algorithms to learn, compare, and improve on the accuracy of the results. The first of these algorithms is Logistic Regression.

To prepare the model, we relied on the R function *glm()*. It is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor

and a description of the error distribution. The model provides insight as to what features are statistically significant. However, to get a better grasp of our model's characteristics, we further analyze the table of deviance using the *anova()* function. More precisely, we used the Chi-Square test (Chisq). The Chi-Square test of independence is used to determine whether there is a significant connection between two nominal (categorical) variables. The frequency of each category for one nominal variable is compared across the categories of the second nominal variable. The difference between the null deviance and the residual deviance shows how our model is doing against the null model (a model with only the intercept). While no exact equivalent to the R2 of linear regression exists, the McFadden R2 index can be used to assess the model fit. Logistic regression is estimated by maximizing the likelihood function. Let L_0 be the value of the likelihood function for a model with no predictors, and let LM be the likelihood for the model being estimated. McFadden's R2 is defined as:

$$R2MFc = 1 - \frac{\ln(L_M)}{\ln(L_0)} \quad (5)$$

where ln() is the natural logarithm. The rationale for this formula is that ln(L0) plays a role analogous to the residual sum of squares in linear regression. Consequently, this formula corresponds to a proportional reduction in "error variance". It's sometimes referred to as a "pseudo" R2. Its reported results are illustrated in Figure 4.

```
          llh       llhNull            G2
-2.677541e+04 -1.098335e+05  1.661162e+05
     McFadden          r2ML          r2CU
7.562182e-01  6.122480e-01  8.571349e-01
```

Figure 4. R2 Index results

By setting the parameter type='response', R will output probabilities in the form of $P(y = 1|X)$. Our decision boundary is set to 0.5; that is; If $P(y = 1|X) > 0.5$ then $y = 1$ otherwise $y = 0$.

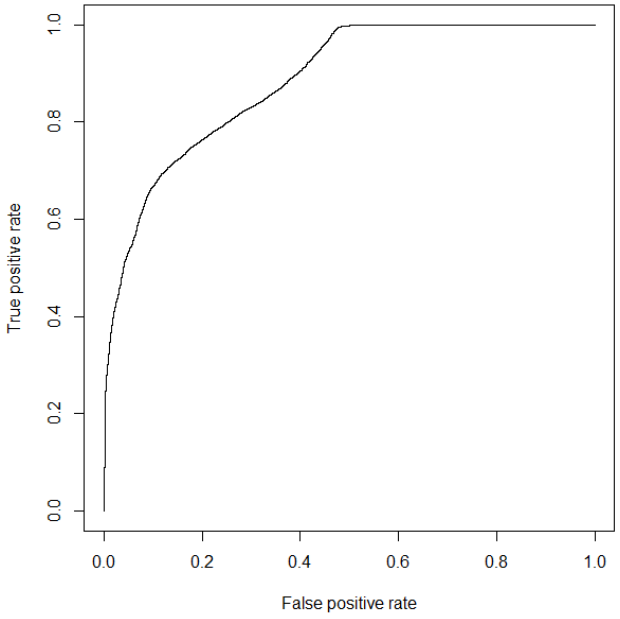

Figure 5. ROC curve for linear regression model. AUC = 0.884385

In order to double check the model's reliability, we use the function *cv.glm()*. This function calculates the estimated K-fold cross-validation prediction error for generalized linear models. For K = 10 folds, the delta value we considered was <6%.

As a last step, we plot the ROC (Receiver Operating Characteristic) curve, as depicted in Figure 5, and calculate the AUC (area under the curve), which are typical performance measurements for a binary classifier is 0.884385 indicating a very good accuracy of the model.

D. Gradient Boosting Machine

The motivation behind using GBM is to compute Logarithmic-Loss in order to evaluate the complexity of our models and get a feel of how that affects our results. For that purpose, we compute the Logarithmic Loss.

Log-Loss quantifies the accuracy of a classifier by penalizing false classifications. Minimizing the Log Loss is basically equivalent to maximizing the accuracy of the classifier. For this, we define Log Loss as:

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\, log(p_{ij}) \quad (6)$$

where N is the number of samples or instances, M is the number of possible labels, y_{ij} is a binary indicator of whether or not label j is the correct classification for instance i, and p_{ij} is the model probability of assigning label j to instance i. A perfect classifier would have a Log Loss of precisely zero. Less ideal classifiers have

progressively larger values of Log Loss. If there are only two classes then the expression above simplifies to:

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(p_i) + (1 - y_i) * \log(p_i)] \quad (7)$$

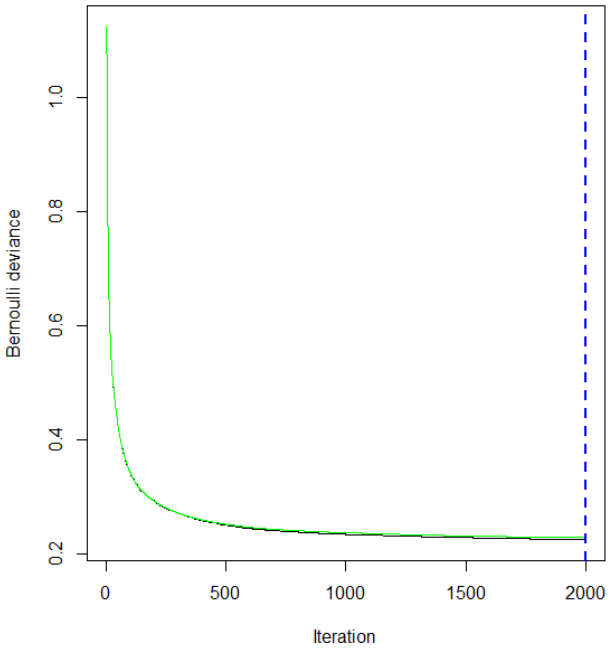

Figure 6. Bernouli Deviance vs. Iterations

As depicted in Figure 6, we determined a number of 2000 trees with a shrinkage value of 0.01 in order to achieve a good log loss of 0.1567176.

E. Support Vector Machine (Binary)

SVM is known to be the best binary classifier because of its kernels flexibility. We decided to use a radial kernel considering the complex nature of the dataset.

The full set of features used in our first run does not allow testing as the features *state*, *proto* and *service* introduce new levels in the testing set. This is considered a notable limitation of the training / testing set distribution that comes with the UNSW-NB15 dataset.

F. Second Stage Multi-Classification

Based on the results from our binary classification, the output from the best binary classification algorithm is fed back as input to a best performing multi-classification algorithm that would cater for attack category classification. For that purpose, we benchmark decision trees (C5.0), SVM, and Naïve Bayes. Accordingly, we perform a 10-Fold cross validation of the produced models so as to minimize the margin for error. Results are reported in the next section.

5. RESULTS AND DISCUSSION

Results from the first stage of the study are depicted in Figure 7. SVM as a binary classification algorithm outperformed its rivals GBM and LR with an accuracy of 82.11%. We believe that it had the potential to achieve a much higher accuracy. Nonetheless, there have been certain categories of attacks advertised by the UNSW-NB15 that could not be trained for using the information available in the dataset; so they cannot be tested for. Namely, *Backdoor*, *Analysis*, *Shellcode* and *Worms*. These attacks either do not have enough records to represent them in the dataset or are in need of extra features. Either way, many of the algorithms we tested in the scope of this study failed to train a model that could detect them with a reliable accuracy. This is further exemplified in the results of multi-classifiers in Table IV. These results were generated independently from the first Stage binary classification in order to determine the most efficient multi-classifier to use in conjunction with SVM as a binary classifier.

Category	Accuracy
Logistic Regression	77.21%
Gradient Boost Machine	61.83%
Support Vector Machine	82.11%

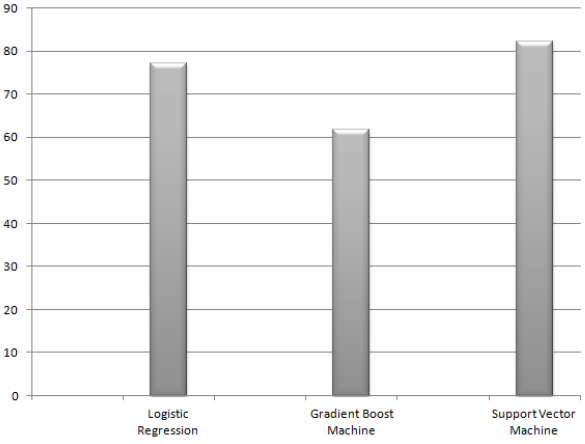

Figure 7. Summary for Accuracy obtained for binary classifiers

TABLE IV. SUMMARY TABLE OF ACCURRACY COMPARISON FOR STAGE TWO CLASSIFICATION INDEPENDATELY OF STAGE ONE

Category	Decision Trees (C5.0)		Naïve Bayes		Support Vector Machine	
	Train	Test	Train	Test	Train	Test
Normal	93.73	74.93	83.21	64.54	97.53	62.41
DOS	10.38	8,83	0	0	34.01	1.07
Fuzzers	79.10	55.24	63.11	36.28	48.8	76.26
Backdoor	17.75	4.97	53.26	22.47	0	0
Exploits	96.68	90.08	14.89	24.97	69.47	85.22
Analysis	21.35	0	0.55	0	0	0
Generic	98.72	96.96	97.89	96.29	96.27	96.24
Reconnaissance	76.29	80.77	39.05	49.57	54.09	68.44
Shellcode	84.02	60.84	0.97	1.32	0	0
Worms	**63.63**	**72.72**	**64.61**	**38.64**	**0**	**0**
Total	**85.41**	**75.53**	**61.22**	**60.70**	**82.87**	**70.21**

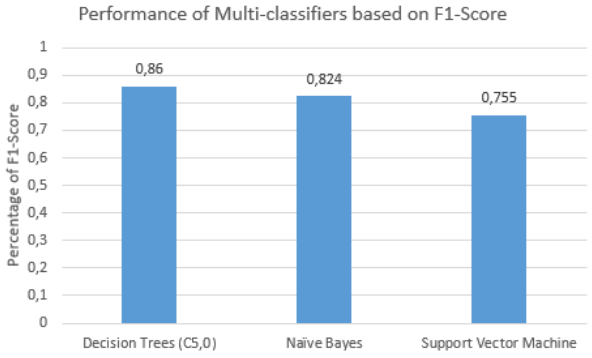

Figure 8. Performance of Multi-classifiers based on F1-Score

Based on Figure 7, we have picked SVM as the binary classifier with the highest accuracy to feed its results to a multi-classifier in Stage II. Before doing so, we needed to test for the models behavior with regards to our fine-tuned dataset in order to: - Have a scheme of reference and; - Pick only the best fit algorithm. Table IV summarizes a comprehensive overview of the training and testing accuracies of the three algorithms. As mentioned earlier, it is important to note that there are categories of attacks that fail not only at the testing phase, but rather at the training phase. The still quite high overall accuracies occur because of imbalance of number of records pertaining to each type of attack. That said, we needed to validate the accuracy results produced by computing the F1-score of each of the three ML algorithms in order to take into consideration the imbalance of the dataset in terms of number of entries. Figure 8 illustrates the results and confirms that the C5.0 algorithm outperforms the two other ML algorithms both in Accuracy and F1-score as a harmonic mean of precision and recall. As such, we mount our two-stage classification using SVM for stage I (binary classification) and C5.0 for stage II (multi-classification). Table V used for both the purpose of validating and illustrating our results shows a considerable improvement in the accuracy achieved by C5.0 on the UNSW-NB15 dataset after its integrating with binary SVM. The highest accuracy improvement achieved is observed in fold 3, enhancing a 74% accuracy to an 86% with a maximum increase of almost 12% and an average increase of 9%.

TABLE V. 10-FOLD CROSS VALIDATION OF C5.0 WITH AND WITHOUT STAGE ONE SVM

$Fold01 [1] 0.753423	$Fold01 [1] 0.847249
$Fold02 [1] 0.7585829	$Fold02 [1] 0.8486007
$Fold03 [1] 0.7499697	$Fold03 [1] 0.8604375
$Fold04 [1] 0.755821	$Fold04 [1] 0.847302
$Fold05 [1] 0.7564773	$Fold05 [1] 0.8476675
$Fold06 [1] 0.7588135	$Fold06 [1] 0.8424438
$Fold07 [1] 0.7473851	$Fold07 [1] 0.8382424
$Fold08 [1] 0.7449583	$Fold08 [1] 0.8394251
$Fold09 [1] 0.7598829	$Fold09 [1] 0.8438263
$Fold10 [1] 0.7589837	$Fold10 [1] 0.8478605

6. CONCLUSION

While our two-stage classification for NIDS using the UNSW-NB15 dataset has proven itself effective in improving the performance of attack-category classifiers, the study revealed a number of limitations to the UNSW-NB15 dataset. In this study, we performed a number of tests for data pre-processing before initiating our experiments. This showed that certain features in the dataset introduce a big amount of noise while having negligible contribution to the intrusion detection. The standard training and testing sets advertised with the dataset could be further optimized. Certain features of the testing set introduces new levels unavailable in the training set making it impractical to train on these features using certain algorithms such as decision trees. Results from the two-stage classification using LR, GBM and SVM for binary classification, then C5.0, NB and SVM for Stage II classification shows that the dataset is unable

to train with reliable accuracies for certain types of attacks claimed to be supported by the dataset. This limitation is further persisted with the imbalance of data entries for each type of attack in both the training and testing sets.

On the other hand, thanks to the connection features and other specificities of the UNSW-NB15, the dataset holds important potential for attack pattern recognition and analysis. As future work, we recommend the investigation of Hidden Markov Models and Deep Learning techniques in this dataset. The rationale behind is to be able to learn from sequentiality in complex attacks which would add a new valuable dimension to our predictive models.

REFERENCES

[1] Siponen, M.; Willison, R.; and Baskerville, R., "Power and Practice in Information Systems Security Research" (2008). ICIS 2008 Proceedings. Paper 26. http://aisel.aisnet.org/icis2008/26

[2] Scambray J., McClure S., and Kurtz G.. 2012. Hacking Exposed (7nd ed.). McGraw-Hill Professional.

[3] Hubballi, N., & Suryanarayanan, V. (2014). False alarm minimization techniques in signature-based intrusion detection systems: A survey. Computer Communications, 49, 1-17. doi:10.1016/j.comcom.2014.04.012

[4] Garuba M., Liu C., Fraites D., "Intrusion Techniques: Comparative Study of Network Intrusion Detection Systems", *Fifth International Conference on Information Technology: New Generations*, pp. 592-598, 2008.

[5] Koucham, O., Rachidi, T., & Assem, N. (2015). Host intrusion detection using system call argument-based clustering combined with Bayesian classification. 2015 SAI Intelligent Systems Conference (IntelliSys). doi:10.1109/intellisys.2015.7361267

[6] Mouttaqi, T., Rachidi, T., & Assem, N. (2017). Re-evaluation of Combined Markov-Bayes Models for Host Intrusion Detection on the ADFA Dataset. Intelligent Systems Conference.

[7] Jamshed M. A., Lee J., Moon S., Yun I., Kim D., Lee S., Yi Y., and Park K., "Kargus: a highly-scalable software-based intrusion detection system," in Proceedings of the 2012 ACM conference on Computer and communications security, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 317–328. [Online]. Available: http://doi.acm.org/10.1145/2382196.2382232

[8] Razan Abdulhammed, Miad Faezipour, Khaled M. Elleithy, "Network intrusion detection using hardware techniques: A review", Systems Applications and Technology Conference (LISAT) 2016 IEEE Long Island, pp. 1-7, 2016.

[9] M. Rathod, Prashantkumar & Marathe, Nilesh & V. Vidhate, Amarsinh. (2014). A survey on Finite Automata based pattern matching techniques for network Intrusion Detection System (NIDS). 1-5. 10.1109/ICAECC.2014.7002456.

[10] G.cS. Shenoy, J. Tubella, A., Gonzalez, "A Performance and Area Efficient Architecture for Intrusion Detection Systems," Parallel & Distributed Processing Symposium (IPDPS), IEEE International, pp.301-310, 2011.

[11] Thinh Y. T. N., Hieu T. T., Dung V. Q., and Kittitornkun S., "A fpga-based deep packet inspection engine for network intrusion detection system," in Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2012 9th International Conference on, may 2012, pp. 1 –4.

[12] Chengcheng Xu, Shuhui Chen, Jinshu Su, Yiu S. M., Hui Lucas C. K., "A Survey on Regular Expression Matching for Deep Packet Inspection: Applications Algorithms and Hardware Platforms", Communications Surveys & Tutorials IEEE, vol. 18, no. 4, pp. 2991-3029, 2016.

[13] Symantec (2018) *2018 Internet Security Threat Report.* Available at: *http://www.symantec.com/security_response/publications/threatreport.jsp* (Accessed 21st Jan 2019).

[14] Depren O., Topallar M., Anarim E., Ciliz M. K., "An Intelligent Intrusion Detection System (IDS) for Anomaly and Misuse Detection in Computer Networks", *Expert Systems with Applications*, pp. 713-722, 2005.

[15] Gang W., Jinxing H., Jian M., "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering", *Expert systems with applications*, vol. 376, pp. 6225-6232, 2011.

[16] Shaohua T., Hongle D., Naiqi W., Wei Z., Jiangyi S., "A Cooperative Network Intrusion Detection Based on Fuzzy SVMs", *Journal of Networks*, vol. 5, pp. 475-483, 2010.

[17] Zuhairi Megat F. ; Shadil Akimi Z.A ; Dao Hassan. Anomaly-based NIDS: A review of machine learning methods on malware detection Raffie Z.A Mohd ; 2016 International Conference on Information and Communication Technology (ICICTM) pp. **DOI:** 10.1109/ICICTM.2016.7890812

[18] Moustafa, N., & Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1-3), 18-31. doi:10.1080/19393555.2015.1125974.

[19] Galar M., Fernandez A., Barrenechea E., Bustince H., and Herrera F., "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," pp. 463–484, 2012.

[20] Sasan, H. P., & Sharma, M. (2016). Intrusion Detection Using Feature Selection and Machine Learning Algorithm with Misuse Detection. International Journal of Computer Science and Information Technology, 8(1), 17-25. doi:10.5121/ijcsit.2016.8102

[21] Amiri F., Mohammad R. Y., Caro L., Azadeh S., Nasser Y., "Mutual Information - Based Feature Selection for Intrusion Detection System", *Journal of Network and Computer Applications*, vol. 34, pp. 1184-1199, 2011.

[22] Moustafa, N., & Slay, J. (2015). The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems. 2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS). doi:10.1109/badgers.2015.014.

[23] Moustafa, N., & Slay, J. (2015). A hybrid feature selection for network intrusion detection systems: Central points. 16th Australian Information Warfare Conference, 5-13. doi:10.4225/75/57a84d4fefb.

[24] Nawir, M., Amir, A., Yaakob, N., and Lynn, O. (2018). Multi-classification Of UNSW-NB15 Dataset For Network Anomaly Detection System. *Journal of Theoretical and Applied Information Technology*, *96*(15), 5094-5104.

[25] Meghdouri, F., Zseby, T., & Iglesias, F. (2018). Analysis of Lightweight Feature Vectors for Attack Detection in Network Traffic. *Applied Sciences*, *8*(11), 2196th ser. doi:10.3390/app8112196

[26] Divekar, A., Parekh, M., Savla, V., Mishra, R., & Shirole, M. (2018). Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives. *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*. doi:10.1109/cccs.2018.8586840

[27] Belouch, M., Hadaj, S. E., & Idhammad, M. (2018). Performance evaluation of intrusion detection based on machine learning using Apache Spark. *Procedia Computer Science, 127*, 1-6. doi:10.1016/j.procs.2018.01.091

[28] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). 2015 Military Communications and Information Systems Conference (MilCIS). doi:10.1109/milcis.2015.7348942.

[29] Verma, A., & Ranga, V. (2018). Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine Learning. Procedia Computer Science,125, 709-716. doi:10.1016/j.procs.2017.12.09

[30] Belouch M., El Hadaj S., and Idhammad M., "Performance evaluation of intrusion detection based on machine learning using Apache Spark," Procedia Computer Science, Vol. 127, 2018, pp 1-6, https://doi.org/10.1016/j.procs.2018.01.091

Souhail Meftah received his B.S. Hons. (2016) in Computer Science from Al Akhawayn University in Ifrane, Morocco and his M.S. (2018) in Information Systems Security from Al Akhawayn University in Ifrane. He is currently working on his Ph.D. in Electrical and Computer Engineering at the National University of Singapore. His research interests include Intrusion Detection, Cloud Security, Deep Learning and Homomorphic Encryption.

Tajjeeddine Rachidi is a graduate of the ENSIMAG, Grenoble, France in Computer Systems Engineering. He received his Ph.D. in Machine Vision from the University of Essex, UK. His research interests include Cryptography, the Applications of Machine Learning and AI to intrusion detection, and Nano-satellite systems. He is currently the director of "Systems and Communications Laboratory", at Al Akhawayn University, Ifrane Morocco. Dr Rachidi is co-editor of many Springer LCNS proceedings.

Nasser Assem obtained an Electrical Engineering degree from the Mohammadia School of Engineering in Rabat, Morocco (1988), a Master of Science and a Ph.D. in Computer Science (2002) from Michigan State University, USA. His teaching and research interests include business intelligence and data science, including machine learning, data mining, and big data. Prior to his career in academia, he worked as a software engineer in the Ministry of Public Works.