Available Online at www.ijcsmc.com/journal

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 11, Issue. 6, June 2022, pg.137 – 151

Analysis of Various Machine Learning Approach to Detect Anomaly from Network Traffic

[1]Sharad Laxman Pawar; [2]Dr. Tryambak Hiwarkar
[1]Ph.D Research Scholar; [2]Professor
[1,2]School of Engineering and Technology
[1,2]Sardar Patel University Balaghat

DOI: https://doi.org/10.47760/ijcsmc.2022.v11i06.011

Abstract:

Although conventional network security measures have been effective up until now, machine learning techniques are a strong contender in the present network environment due to their flexibility.

In this study, we evaluate how well the latter can identify security issues in a corporative setting Network. In order to do so, we configure and contrast a number of models to determine which one best our demands. In addition, we spread the computational load and storage to support large quantities of data. Our model-building methods, Random Forest and Naive Bayes.

Keywords: Decision trees, Support Vector Machines, SVM

1. Introduction

Network security threats and cyber-attacks have significantly increased in developing technologies including cloud, fog, edge computing, and the internet of things (IoT). Network-related environments, servers hosted in the cloud, and economic source data can all be compromised by these attacks [1]. Every network security system needs network anomaly detection systems (NADSs), which keep an eye on network packets for potential threats and unusual user behavior.

The frequency of malicious incidents has sharply increased over the past ten years, and such a problem has serious repercussions for individual users, organizations, and businesses. As an illustration, a denial-of-service (DoS) attack disrupts the network's normal traffic by saturating it with a huge amount of traffic. Similar to this, distributed denial-of-service (DDoS) attacks target regular network traffic by flooding it with network traffic [2]. Since the template or signature specification for these kinds of attacks is not disclosed, zero-day attack detection is likewise exceedingly difficult [3].

A machine learning-based network anomaly detection system is made of four primary modules, as shown by the typical architecture in Figure 1. The modules are listed in this order:

Packet decoder: this module receives raw network traffic packets and transfers suitable information to the pre-processing module.

Pre-processing: this module receives a portion of network traffic features and prepares corresponding normalized feature vectors, which is necessary for learning-based systems in the **detection module**.

• Classifier system: the responsibility of this module is to build a model on top of the prepared data which discriminates malicious instances against normal ones.

• Detection and recognition: two phases are defined in this module—(i) detects the malicious instances as a binary decision problem (e.g., 0 for normal and 1 for malicious) and transmits an alert to a system administrator for making a reaction; and (ii) after any malicious behavior detection, the system can recognize various types of abnormality (attack classification).

A deviation from the network's usual behavior is known as a network anomaly. That covers both regrettable accidental incidents and malicious attacks intended to jeopardize the availability of the network. It is crucial to be able to identify things immediately in both situations so that we can respond in time [1].

Machine learning has been gradually replacing policies-based classic intrusion detection systems over the last several years since it offers a number of advantages over them. Machine learning approaches enable the creation of non-parametric algorithms that are transferable across applications and adapt to our network and its modifications [1].

Despite the fact that there is still a gap between the deployment of machine learning-based intrusion detection systems and those of their predecessors as a result of various difficulties [2], it is a fact that traditional detection methods struggle to handle massive data volumes because their analysis procedures are intricate and time-consuming. However, these issues can be resolved with the aid of big data and machine learning tools and methodologies [3].

In this work, we explore several machine learning and big data alternatives while modelling and deploying the UNB ISCX IDS 2012 Intrusion Detection Evaluation Dataset to accommodate substantial data processing. The eventual solution we'll use will be built on the top model and executed on Apache Structured Streaming with PySpark and MLlib. Hadoop will disperse the storage as well. Furthermore, we'll use Zabbix and JMX for performance monitoring.

II. Literature Review

Related Work Machine Learning in Intrusion Detection and Prevention Domains

Some of the earliest works on network classification were produced by Cannady [5] and [6]. He suggests that when neural networks are built for a specific issue area with representative training data sets, they are appropriate solutions. Streaming data could not be handled by the model, and

Therefore, it is essential that the person guarding our system always takes the data offline.

He needs to execute the model using the new set of representative data after training it. Additionally, the A three-layer control feed forward method was used by the authors to provide a number of input–output mappings. As a result, the specific Intrusion Detection System (IDS) agent gains knowledge. How to recognize Internet Control Message Protocol-based flood-based Denial of Service attacks together with the User Datagram Protocol (ICMP) (UDP). The process first investigates how to regularly refresh and retrains the model in order to detect the ICMP attacks. Consequently, it is able to pick up new assaults based on the UDP protocol and learn how to identify them. Cerebellar Model Articulation Controller Neural Network was utilized to suggest an online learning method. The authors devised an elaborate system designed to produce a series of mappings of input and output. The system is originally taught how to recognize ICMP violations and to Base on the UDP, it decides how to identify new threats using prior knowledge and training. Protocol. The approach is focused solely on Denial of Service assaults that use floods, though. The analysis of the sequences of data used in one technique used to find abnormalities in IDS is based on system chimes.

The researchers in [7] used Random Forest and Regression Trees to identify systemic flaws. Principal Component Analysis (PCA) was used to identify outliers and anomalies in IDS. [8] Comparison of various approaches to attribute selection and anomaly detection. The choice of an appropriate model is crucial since it will aid in subsequent research and indicate the best course of action for balancing accuracy and complexity. Observing system call sequences is one method for identifying breaches in host-based intrusion detection systems. These calls are started by a process operating on the host, and they are organized into sets of traces. Each trace contains a file of all system calls made throughout.

Numerous IDS research papers can be distilled into machine learning classification issues that can be resolved with supervised or semi-supervised learning models. Unsupervised learning was attempted by some writers, however they had poor accuracy [9].

Although RL has been widely used in computer network disciplines, little research has been done on its application to intrusion detection or intrusion prevention. The field of routing protocols, validation procedures, entrance control, and service quality methods is thought to be very exciting by scientists. Because RL makes sense in circumstances when the environment responds, people are more alert. All of the aforementioned situations exhibit feedback that is portrayed as a reward.

2.1. Reinforcement Learning in Network Security

Reinforcement learning was used by Xu et al. in conjunction with Hidden Markov Models (HMM) to recognize breaches by learning the probability of state transitions. The state transitions on IDS might be adequately estimated, according to the authors, using HMM. Using the same training and testing sets, they employed temporal difference approaches to modify the value function and got results. Two years later, used a temporal-difference technique to model the network dynamics. With this work, they improved detection accuracy compared to earlier

HMM method implementations. They employed a kernel least-squares temporal-difference algorithm (LS-TD) to estimate the value function and carry out parameter reduction.

In order to demonstrate the viability of the proposed approach, Xu and Luo employed system call traces from the send mail application. The model Miller and Inoue utilized is Perceptual Intrusion Detection System with Reinforcement uses numerous distinct agents to detect intrusions. A self-organizing map can be used by a single agent to find malicious activity, and there is a blackboard technique for combining the information provided by all agents. A signal is disseminated to all agents for group analysis once it has been discovered within the system. Voters submit ballots to the central voting system, which computes weights and awards the agents based on their performance.

2.2 Machine Learning Techniques for Streaming Data

The idea of using partial rather than complete memory for data storage can be seen in some of the earliest work on data streaming manipulation. The sliding-window concept illustrates the idea that a window will always have the most recent data related to learning at any given time. The easiest way to solve this problem is to let the user choose the window size and keep it fixed while the algorithm is run. This is because it can be difficult to determine the window size W. Using a score that may identify the indication of the difference between the current and referred window, the change is detected. Other theories suggest that we should only remember aggregate statistics that have a "decay function," which indicates the significance of those aggregates over time.

Other methods construct the drift detection idea by tracking the rates of three operational thresholds, such as recall, precision, and accuracy. With respect to the confidence intervals for the standard sample errors of each individual indicator (using the most recent set of examples), their values are examined using a moving average. The main goal is to select a data window size that will minimize the projected error on new instances. The method makes advantage of unlabeled data. As a result, the concept is simple to apply and does not call for intricate computations.

Another method [87] for spotting changes in the distribution is to watch the algorithm's error rate as it changes in real time. This method involves a series of exams that are used to assess learning. If a fresh testing instance becomes available, the current model is used to label it. The authors decide on drift levels kd and a warning kw that are activated when the error rate reaches specific thresholds. Those levels act as a recommendation for a shift in the allocation of the instances. Two kinds of the drifting concept can be identified and. The first lesson discusses practices that the student should use on a regular basis without considering when changes might occur.

The second type of research is characterized by the fact that the learner adjusts to the change before we do. The temporal windows with a defined size and weighted instances are good examples of the methodologies described above. Because the value of an instance decreases as it ages, the weighted instances are characterized by this idea. In the event that a time frame is used, the learner is only affected by the instances that fall inside the window. Results might be skewed by using a narrow window of time. For example, it is possible to use a temporal window with an adaptable size, or to change its size in accordance with the drift notion. Widmer and Kubat's [91] FLORA2 presents a new concept for window change based on particular rules for categorization purposes. Based on the accuracy and the learner, the authors recommended a window size for recognizing distribution changes. Performance indicators were measured in terms of the accuracy and precision. A moving average was used to calculate the standard error and associated confidence ranges. This approach has a handful of drawbacks. The first is connected to the fact that most of the time we have very little information about the actual class. In the second case, there are a large number of parameters that need to be adjusted.

Support vector machines were proposed by Klinkenberg and Joachims as a reliable method for detecting changes. In order to minimize the predicted error of the new examples, it is important to set an appropriate window size.

III. Contemporary Malicious Behaviors (Network Attacks):

The quantity and variety of network attacks are dramatically growing, causing financial losses, interrupting businesses, and stealing users' confidential information. Referring to the Australian Cyber Security Center (ACSC) [24] and McAfee threat reports [25], contemporary attacks still expose network and computer systems and require further improvement using NADSs. The different types of malicious behaviors are explained as follows:

• **Trojan** is a malware often disguised as normal software but carrying out abnormal activities in the backend. Trojan malware are usually utilized by cybercriminals to penetrate victims' systems and the users are mostly cheated by hackers to execute the Trojan malware on target computers [26].

• **Scareware** is a new sort of malware created to deceive users to buy and download useless and potentially dangerous software—for instance, fake protection programs that cause many financial and security-related perils to the users [3].

• **Rootkits** are malware created to hack particular processes and enable continued privileged access to victims' machines. Rootkit malware can be executed at various levels such as application programming interface (API) calls, at the user level or interfere with OS at the device level [2]

• Analysis contains different types of port scanner attacks such as spam and .html file penetrations [14]. • Ransom ware attack is a malicious software from crypto virology that threatens to reveal the victim's sensitive information unless a ransom is paid [3].

• Zero-day attack is a computer-software vulnerability that exploits a significant security infirmity without the creator's awareness. Until the vulnerability is identified by the system, hackers can exploit it to negatively affect the computer programs and data.

IV. Supervised learning algorithms

Inductive machine learning is the process of learning a set of rules from instances (examples in a training set), or more generally speaking, creating a classifier that can be used to generalize from new instances. The process of applying supervised ML to a real-world problem is described in Figure 1.

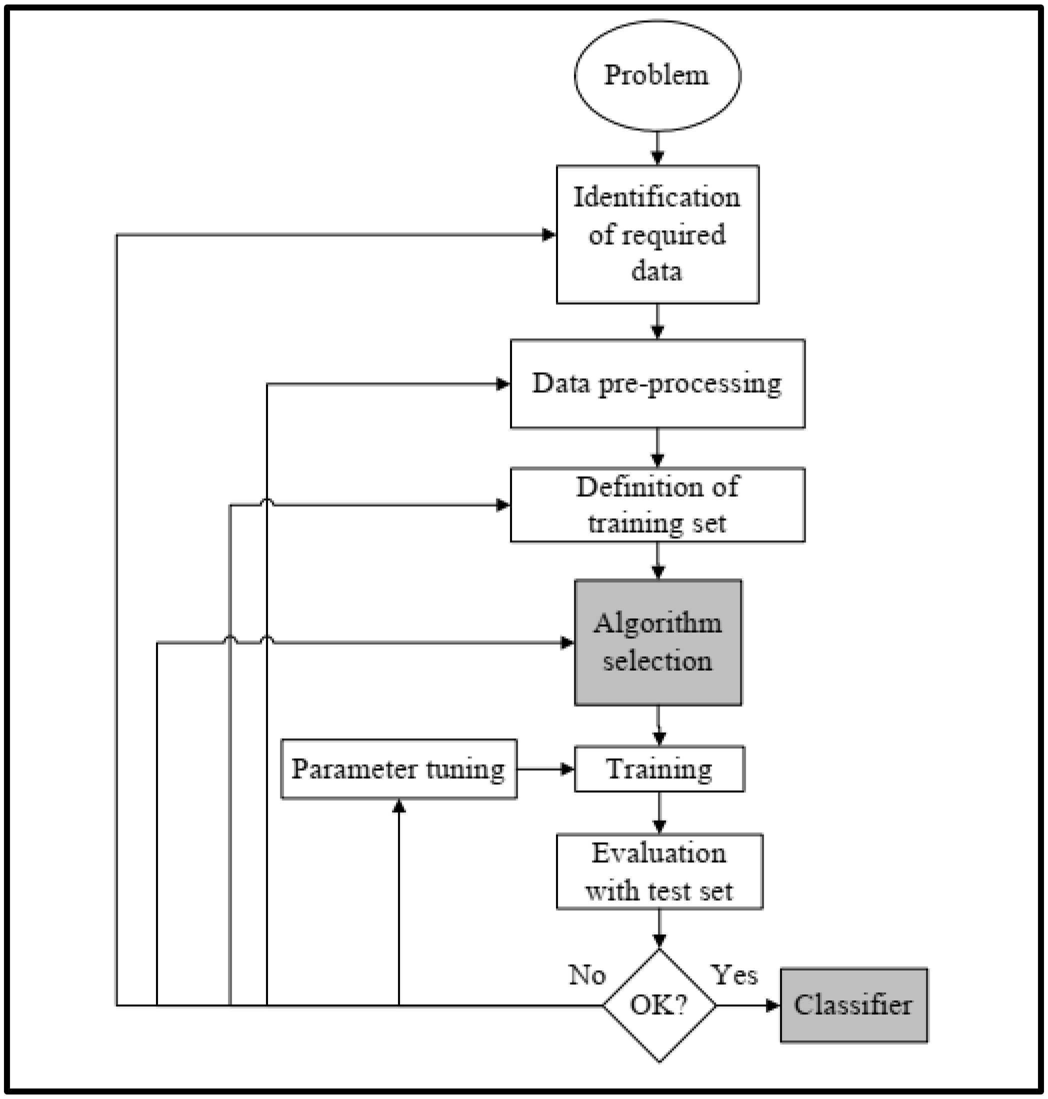


Figure 1. The process of supervised ML

The first step is collecting the dataset. If a requisite expert is available, then s/he could suggest which fields (attributes, features) are the most informative. If not, then the simplest method is that of "brute-force," which means measuring everything available in the hope that the right (informative, relevant) features can be isolated. However, a dataset collected by the "brute-force" method is not directly suitable for induction. It contains in most cases noise and missing feature values, and therefore requires significant pre-processing (Zhang et al., 2002).

The second step is the data preparation and data preprocessing. Depending on the circumstances, researchers have a number of methods to choose from to handle missing data (Batista & Monard, 2003). Hodge & Austin (2004) have recently introduced a survey of contemporary techniques for outlier (noise) detection.

These researchers have identified the techniques' advantages and disadvantages. Instance selection is not only used to handle noise but to cope with the infeasibility of learning from very large datasets. Instance selection in these datasets is an optimization problem that attempts to maintain the mining quality while minimizing the sample size (Liu and Motoda, 2001). It reduces data and enables a data mining algorithm to function and work effectively with very large datasets. There is a variety of procedures for sampling instances from a large dataset (Reinartz, 2002). Feature subset selection is the process of identifying and removing as many irrelevant and redundant features as possible (Yu & Liu, 2004). This reduces the dimensionality of the data and enables data mining algorithms to operate faster and more effectively.

The fact that many features depend on one another often unduly influences the accuracy of supervised ML classification models. This problem can be addressed by constructing new features from the basic feature set (Markovitch & Rosenstein, 2002). This technique is called feature construction/transformation. These newly generated features may lead to the creation of more concise and accurate classifiers. In addition, the discovery of meaningful features contributes to better comprehensibility of the produced classifier, and a better understanding of the learned concept.

V. Logic based algorithms

Decision trees

Murthy (1998) provided an overview of work in decision trees and a sample of their usefulness to newcomers as well as practitioners in the field of machine learning. Thus, in this work, apart from a brief description of decision trees, we will refer to some more recent works than those in Murthy's article as well as few very important articles that were published earlier. Decision trees are trees that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values. Figure 2 is an example of a decision tree for the training set of Table 2.

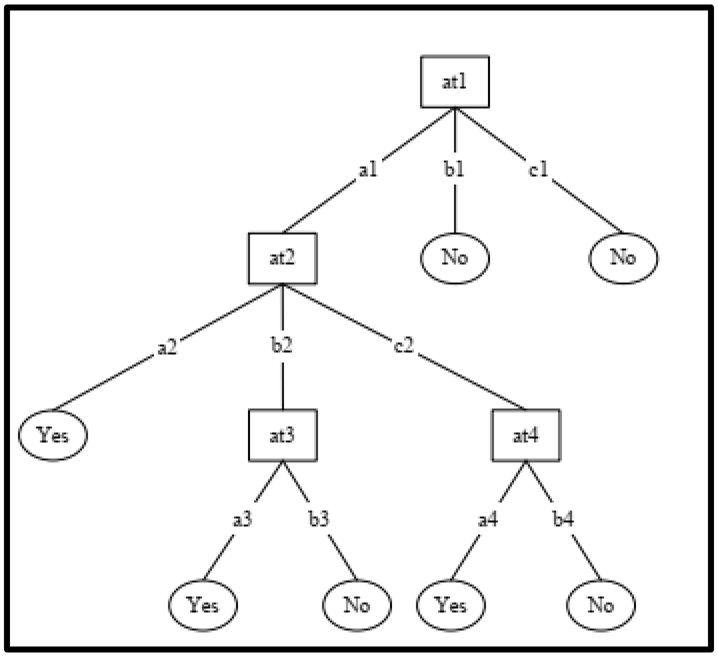


Figure 2. A decision tree

2) Naive Bayesian (NB) Networks: These are very simple Bayesian networks which are composed of directed acyclic graphs with only one parent (representing the unobserved node) and several children (corresponding to observed nodes) with a strong assumption of independence among child nodes in the context of their parent [7].Thus, the independence model (Naive Bayes) is based on estimating [14]. Bayes classifiers are usually less accurate that other more sophisticated learning algorithms (such as ANNs).However, [5] performed a large-scale comparison of the naive Bayes classifier with state-of-the-art algorithms for decision tree induction, instance-based learning, and rule induction on standard benchmark datasets, and found it to be sometimes superior to the other learning schemes, even on datasets with substantial feature dependencies. Bayes classifier has attribute independence problem which was addressed with Averaged One-Dependence Estimators [8]: These are very simple Bayesian networks which are composed of directed acyclic graphs with only one parent (representing the unobserved node) and several children (corresponding to observed nodes) with a strong assumption of independence among child nodes in the context of their parent [7].Thus, the independence model (Naive Bayes) is based on estimating. Bayes classifiers are usually less accurate that other more sophisticated learning algorithms (such as ANNs).However, [5] performed a large-scale comparison of the naive Bayes classifier with state-of-the-art algorithms for decision tree induction, instance-based learning, and rule induction on standard benchmark datasets, and found it to be sometimes superior to the other learning schemes, even on datasets with substantial feature dependencies. Bayes classifier has attribute independence problem which was addressed with Averaged One-Dependence Estimators [8]

3) Multi-layer Perceptron: This is a classifier in which the weights of the network are found by solving a quadratic programming problem with linear constraints, rather than by solving a no convex, unconstrained minimization problem as in standard neural network training .Other well-known algorithms are based on the notion of perceptron .Perceptron algorithm is used for learning from a batch of training instances by running the algorithm repeatedly through the training set until it finds a prediction vector which is correct on all of the training set. This prediction rule is then used for predicting the labels on the test set [9].

4) Support Vector Machines (SVMs): These are the most recent supervised machine learning technique [24].Support Vector Machine (SVM) models are closely related to classical multilayer perceptron neural networks. SVMs revolve around the notion of a ―margin‖— either side of a hyper plane that separates two data classes. Maximizing the margin and thereby creating the largest possible distance between the separating hyper plane and the instances on either side of it has been proven to reduce an upper bound on the expected generalization error [9].

5) Bayesian Network: A Bayesian Network (BN) is a graphical model for probability relationships among a set of variables (features). Bayesian networks are the most well-known representative of statistical learning algorithms [9].The most interesting feature of BNs, compared to decision trees or neural networks, is most certainly the possibility of taking into account prior information about a given problem, in terms of structural relationships among its features [9].A problem of BN classifiers is that they are not suitable for datasets with many features [4].This prior expertise, or domain knowledge, about the structure of a Bayesian network can take the following forms:

1. Declaring that a node is a root node, i.e., it has no parents.
2. Declaring that a node is a leaf node, i.e., it has no children.
3. Declaring that a node is a direct cause or direct effect of another node.

4. Declaring that a node is not directly connected to another node.
5. Declaring that two nodes are independent, given a condition-set.
6. Providing partial nodes ordering, that is, declare that a node appears earlier than another node in the ordering.
7. Providing a complete node ordering.

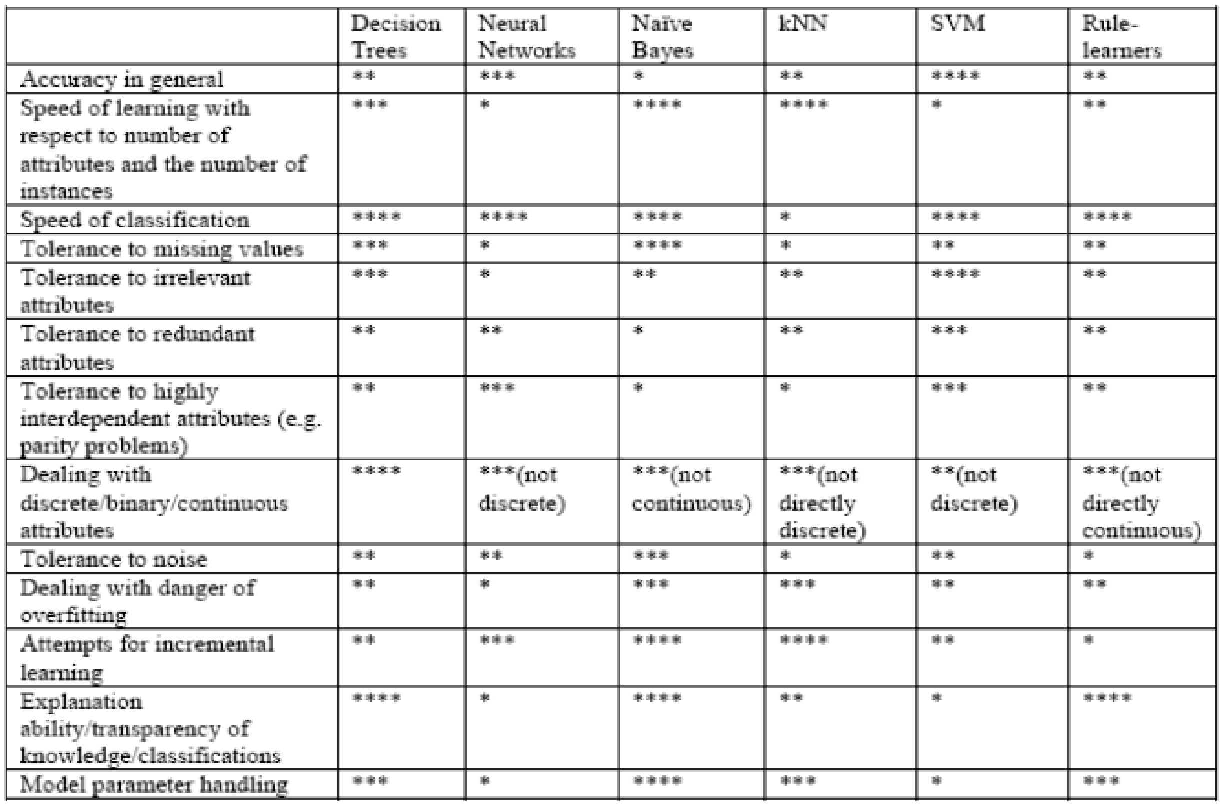
6) Neural Networks: [2]opined Neural Networks (NN) that can actually perform a number of regression and/or classification tasks at once although commonly each network performs only one. In the vast majority of cases, therefore, the network will have a single output variable, although in the case of many-state classification problems, this may correspond to a number of output units (the post-processing stage takes care of the mapping from output units to output variables).Artificial Neural Network (ANN) depends upon three fundamental aspects, input and activation functions of the unit, network architecture and the weight of each input connection. Given that the first two aspects are fixed, the behavior of the ANN is defined by the current values of the weights.

The weights of the net to be trained are initially set to random values, and then instances of the training set are repeatedly exposed to the net. The values for the input of an instance are placed on the input units and the output of the net is compared with the desired output for this instance. Then, all the weights in the net are adjusted slightly in the direction that would bring the output values of the net closer to the values for the desired output. There are several algorithms with which a network can be trained [12]

VI. Features of Machine Learning Algorithms:

Supervised machine learning techniques are applicable in numerous domains. A number of Machine Learning (ML) application oriented papers can be found. Generally, SVMs and neural networks tend to perform much better when dealing with multi dimensions and continuous features. On the other hand, logic-based systems tend to perform better when dealing with discrete/categorical features. For neural network models and SVMs, a large sample size is required in order to achieve its maximum prediction accuracy whereas NB may need a relatively small dataset. There is general agreement that k-NN is very sensitive to irrelevant features: this characteristic can be explained by the way the algorithm works. Moreover, the presence of irrelevant features can make neural network training very inefficient, even impractical. Most decision tree algorithms cannot perform well with problems that require diagonal partitioning. The division of the instance space is orthogonal to the axis of one variable and parallel to all other axes. Therefore, the resulting regions after partitioning are all hyper rectangles. The ANNs and the SVMs perform well when multicollinearity is present and a nonlinear relationship exists between the input and output features. Naive Bayes (NB) requires little storage space during both the training and classification stages: the strict minimum is the memory needed to store the prior and conditional probabilities. The basic kNN algorithm uses a great deal of storage space for the training phase, and its execution space is at least as big as its training space. On the contrary, for all non-lazy learners, execution space is usually much smaller than training space, since the resulting classifier is usually a highly condensed summary of the data. Moreover, Naive Bayes and the kNN can be easily used as incremental learners whereas rule algorithms cannot. Naive Bayes is naturally robust to missing values since these are simply ignored in computing probabilities and hence have no impact on the final decision. On the contrary, kNN and neural networks require complete records to do their work.

Finally, Decision Trees and NB generally have different operational profiles, when one is very accurate the other is not and vice versa. On the contrary, decision trees and rule classifiers have a similar operational profile. SVM and ANN have also a similar operational profile. No single learning algorithm can uniformly outperform other algorithms over all datasets.



	Decision Trees	Neural Networks	Naïve Bayes	kNN	SVM	Rule-learners
Accuracy in general	**	***	*	**	****	**
Speed of learning with respect to number of attributes and the number of instances	***	*	****	****	*	**
Speed of classification	****	****	****	*	****	****
Tolerance to missing values	***	*	****	*	**	**
Tolerance to irrelevant attributes	***	*	**	**	****	**
Tolerance to redundant attributes	**	**	*	**	***	**
Tolerance to highly interdependent attributes (e.g. parity problems)	**	***	*	*	***	**
Dealing with discrete/binary/continuous attributes	****	***(not discrete)	***(not continuous)	***(not directly discrete)	**(not discrete)	***(not directly continuous)
Tolerance to noise	**	**	***	*	**	*
Dealing with danger of overfitting	**	*	***	***	**	**
Attempts for incremental learning	**	***	****	****	**	*
Explanation ability/transparency of knowledge/classifications	****	*	****	**	*	****
Model parameter handling	***	*	****	***	*	***

Table 1: Comparing learning algorithms (* stars represent the best and * star the worst performance)***

VII. Conclusions

From a general point of view, we define the outcomes obtained as positive. They certainly align with both our hypothesis and other authors' ideas [1,3] about how machine learning in the network security domain can make a difference. About the models, and concretely respecting Naïve Bayes, its simplicity and strong assumptions about feature independency [5] probably take a part in why it falls behind in terms of capability of prediction with respect to our other choices. On the other side, Random Forest overtook every other alternative in almost every aspect. With a default configuration and using only a few of the original features, it was already the best scoring algorithm. Concerning DNN, its outcomes came close to the ones of Random Forest. In addition, this is the algorithm that benefits the most from hyper parameterization. In reference to the distribution outcomes, the results display an issue that needs to be addressed to successfully take advantage of one of the benefits that we have been claimed machine learning can offer, handling the processing of extensive volumes of data.

References

[1] Ahmed, T.; Oreshkin, B.; Coates, M. Machine Learning Approaches to Network Anomaly Detection. In Proceedings of the Second Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML07), Cambridge, MA, USA, 10 April 2007.

[2] Sommer, R.; Paxson, V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, USA, 16–19 May 2010; Volume 1.

[3] Amrollahi, M.; Hadayeghparast, S.; Karimipour, H.; Derakhshan, F.; Srivastava, G. Enhancing Network Security Via Machine Learning: Opportunities and Challenges. In *Handbook Of Big Data Privacy*; Springer International Publishing: Cham, Switzerland, 2020; p. 169

[4] Fernandez, G. Deep Learning Approaches for Network Intrusion Detection. Ph.D. Thesis, The University of Texas at San Antonio, San Antonio, TX, USA, 2019; pp. 19, 20, 50, 51

[5] Moustafa, N.; Slay, J. A network forensic scheme using correntropy-variation for attack detection. In Proceedings of the IFIP International Conference on Digital Forensics, New Delhi, India, 3–5 January2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 225–239

[6] Bridges, S.M.; Vaughn, R.B. Fuzzy data mining and genetic algorithms applied to intrusion detection. In Proceedings of 12th Annual Canadian Information Technology Security Symposium, Baltimore, MD, USA, 16–19 October 2000; pp. 109–122.

[7] Shen, X.; Agrawal, S. Kernel Density Estimation for An Anomaly Based Intrusion Detection System. MLMTA 2006, 161–167. Available online: https://www.researchgate.net/profile/Xiaoping-Shen-2/publication/221188648_Kernel_Density_Estimation_ for_An_Anomaly_Based_Intrusion_Detection_System/links/54de03c60cf22a26721dd528/Kernel-Density-Estimation-for-AnAnomaly-Based-Intrusion-Detection-System.pdf (accessed on 19 April 2021)

[8] Narouei, M.; Ahmadi, M.; Giacinto, G.; Takabi, H.; Sami, A. DLLMiner: Structural mining for malware detection. Secur. Commun. Netw. 2015, 8, 3311–3322. [CrossRef]

[9] Han, X.; Xu, L.; Ren, M.; Gu, W. A Naive Bayesian network intrusion detection algorithm based on Principal Component
Analysis. In Proceedings of the 2015 7th International Conference on Information Technology in Medicine and Education (ITME), Huangshan, China, 13–15 November 2015; pp. 325–328.

[10] Lin, D.; Stamp, M. Hunting for undetectable metamorphic viruses. J. Comput. Virol. 2011, 7, 201–214. [CrossRef]
78. Saber, M.; El Farissi, I.; Chadli, S.; Emharraf, M.; Belkasmi, M.G. Performance Analysis of an Intrusion Detection Systems Based of Artificial Neural Network. In Europe and MENA Cooperation Advances in Information and Communication Technologies; Springer: Berlin/Heidelberg, Germany, 2017; pp. 511–521.

[11] Ramadas, M.; Ostermann, S.; Tjaden, B. Detecting anomalous network traffic with self-organizing maps. In International Workshop on Recent Advances in Intrusion Detection; Springer: Berlin/Heidelberg, Germany, 2003; pp. 36–54. 80.

[12] Hawkins, S.; He, H.; Williams, G.; Baxter, R. Outlier detection using replicator neural networks. In Proceedings of the International Conference on Data Warehousing and Knowledge Discovery, Aix-en-Provence, France, 4—6 September 2002;Springer: Berlin/Heidelberg, Germany, 2002; pp. 170–180.

[13] Jirapummin, C.; Wattanapongsakorn, N.; Kanthamanon, P. Hybrid neural networks for intrusion detection system. Proc. ITC-CSCC 2002, 7, 928–931.