XGBoost(ing) Network Intrusion Detection

Arnaldo Gouveia[1,2] and Miguel Correia[2]

[1] IBM Belgium – Brussels, Belgium
[2] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa – Lisboa, Portugal

Abstract. XGBoost is a recent machine learning method that has been getting increasing attention. It won Kaggle's Higgs Machine Learning Challenge, among several other Kaggle competitions, due to its performance. In this paper we explore the use of XGBoost in the context of anomaly-based network intrusion detection, an area in which there is a considerable gap.

1 Introduction

The detection of security-related events using *machine learning* has been extensively investigated [5,6,9,13,14,16,20,24]. *Intrusion Detection Systems* (IDSs) are security tools used to detect malicious activity. *Network Intrusion Detection Systems* (NIDS) are one of the best known contexts of machine learning application in the security field [9]. IDSs can be classified using several criteria [5]. One of these criteria is the detection approach, in terms of which IDSs (and NIDSs) can be signature-based or anomaly-based. The former class detects attacks by comparing the data flow under analysis to patterns stored in a signature database of known attacks. The later detects anomalies using a model of normal behaviour of the monitored system and flagging behavior lying outside of the model as anomalous or suspicious. Signature-based IDSs can detect well-known attacks with high accuracy but fail to detect or find unknown attacks [13,24], whereas anomaly-based IDSs have that capacity. In this paper we focus on *anomaly-based NIDSs.*

XGBoost (eXtreme Gradient Boosting) [4] is a recent decision-tree-based ensemble machine learning algorithm. XGBoost won Kaggle's Higgs Machine Learning Challenge in 2014, that consisted in searching for Higgs bosons [11] in a large dataset provided by CERN[3] [1, 2]. The dataset and the subject of the challenge correspond to particular physical phenomena, the Higgs boson to *Tau* particles decay ($H \rightarrow \tau^+\tau^-$) [22]. The Higgs boson has many different processes (called channels by physicists) through which it can decay, i.e., produce other particles. The Standard Model of particle physics has been being developed for decades and the existence of almost all subatomic particles has been confirmed experimentally in the past century. The Higgs boson was the exception, as it was detected experimentally only on 2012, after experiments in the Large Hadron Collider (LHC) [21, 22]. XGBoost found again Higgs bosons on CERN data two years later.

XGBoost is gaining popularity due to its performance and scalability [2, 4, 17]. XGBoost has been proving faster than other popular algorithms in a single

[3] Conseil Européen pour la Recherche Nucléaire.

machine and to scale to billions of examples in distributed or memory-limited settings as shown empirically in Kaggle competitions [17]. Kaggle is an online community focused on machine learning, popular for the competitions it organizes.

The goal of a NIDS is to generate alerts when adversaries try to penetrate or attack the network. Consider a *flow* to be a sequence of IP packets with similar features (e.g., same destination IP address and port). Typically an intrusion involves a few flows, hidden on a multitude of legitimate flows. In statistical terms, the problem of detecting a few flows in a large set of flows is similar to the problem of detecting Higgs bosons. In this scope we aim at answering the following questions:

1. How effectively can XGBoost be used in the context of NIDSs?
2. How to optimize XGBoost for this application?
3. Among the parameters configured for XGBoost model training, can we find correlations with performance predictors, e.g., area under curve or logloss?

We use XGBoost with two datasets carefully designed for evaluating NIDSs: UNB NSL KDD [20] and UNSW NB15 [14, 16]. To the best of our knowledge there is only one previous work that applies XGBoost to intrusion detection [6], but it is just an exercise of running XGBoost on a single dataset that does not cover the second and third questions above.

The contributions of the this work are: (1) a study of the use of XGBoost in the context of network intrusion detection; (2) an optimization method for XGBoost for achieving performance in that context; (3) a set of XGBoost parameters with strong impact/correlation on performance.

2 XGBoost and Boosting Trees

XGBoost is based on a set of machine learning concepts:

- *Weak learners:* XGBoost uses *decision trees* as *weak* learners;
- *Boosting:* combines the contributions of many weak learners to produce a strong learner;
- *Gradient boosting:* does boosting by minimizing errors by means of introducing a gradient-like term.

To address overfitting, i.e., to avoid modelling the training data too tightly, XGBoost uses L1 and L2 *regularization.* The difference between these two penalty terms is mainly:

- *Lasso Regression (L1):* adds the modulus of the coefficient vector as penalty term to the loss function.
- *Ridge regression (L2):* adds a squared modulus term of the coefficient vector as penalty term to the loss function.

The key difference between these techniques is that Lasso shrinks the less important features' coefficients to zero, thus effectively removing some features, something that Ridge does not manage to do. Feature reduction is, therefore, a more defined consequence of using Lasso. XGBoost's regularization strategy involves also configuring other parameters to be effective, e.g., *Maximum Depth*, *Minimum Child Weight*, and *Gamma*. Without proper regularization, the tree will be split until it can predict the training set perfectly, i.e., it will overfit. Therefore, it will end up losing generality and will not perform well on new data (i.e., on test data or on production data).

The trees used play the role of weak learners, providing decisions only slightly better than a random guess. XGBoost used an ensemble approach, i.e., it combines a large number of decision trees to produce a final model consisting of a forest of decision trees. An individual decision tree is grown by ordering all features and checking each possible split for every feature. The split that results in the best score for the chosen objective function becomes the rule for that node.

For a given data set with n examples and m features $\mathcal{D} = \{(x_i, y_i)\}$ ($|\mathcal{D}| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}$), a tree ensemble model uses K additive functions to predict the output [4]:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F}, \tag{1}$$

where $\mathcal{F} = \{f(x) = w_{q(x)}\}(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T)$ is the space of regression trees. In the equation, q represents the structure of each tree that maps an example to the corresponding leaf index, T is the number of leaves in the tree, each f_k corresponds to an independent tree structure q, and w are leaf weights.

In the general case an objective functions has two terms: training loss and regularization [4]: $\mathcal{L}(\theta) = L(\theta) + \Omega(\theta)$. A common choice of $L(\theta)$ is the mean squared error. With proper choices for y_i we may express regression, classification, and ranking. Training the model amounts to finding the parameters θ that best fit the training data x_i labels y_i. In order to train the model, we need to define the objective function to measure how well the model fits the training data. To learn the set of functions used in the model, we minimize the following regularized objective:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{2}$$

A common choice for the loss function $\hat{y}_i, y_i)$ is the mean squared error (MSE):

$$l(\hat{y}_i, y_i) = \sum_i (y_i - \hat{y}_i)^2 \tag{3}$$

where l is a differentiable convex loss function that measures the difference between the prediction $\hat{y}_i$ and the target y_i.

The second term Ω penalizes the complexity of the model (i.e., the regression tree functions). Depending on the type of regularization chosen, $\Omega(f)$ is usually defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\| \text{ (L1 or Lasso)} \tag{4}$$

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2 \text{ (L2 or Ridge)} \tag{5}$$

Year	Dataset	Size	Types of attack

2016	NSL-KDD	Train= 18.7MB
		Test= 3.4MB
			Probing, DoS, R2L, U2R

2017	UNSW NB15	Train= 14.7MB
		Test= 30.8MB
			Fuzzers, Shellcode, Analysis, Backdoors, DoS,
			Exploits, Generic, Reconnaissance, Shellcode, Worms

Table 1: Dataset comparison. The UNB NSL KDD files used were *KDDTest+.arff* and *KDDTrain+.arff*.

where w is the vector of scores on leaves, T is the number of leaves, γ and λ are adjustable gain factors. This regularization term helps to avoid overfitting. The splitting process continues until some condition is met.

We used the H2O XGBoost implementation, written in R, in the experiments. H2O is an open source software package machine learning. Specifically, we used the implementation in the H2O R Package 3.26.0.1. In the paper we sometimes refer explicitly to this implementation.

3 The Datasets

An important component of an effective NIDS evaluation is a good benchmarking dataset. We use two recent datasets that have been designed specifically and carefully to evaluate NIDSs. A comparative summary of the two can be found in Table 1.

3.1 The UNB NSL KDD dataset

Vasilomanolakis et al. [23] identified a number of desirable functional requirements for a dataset to be useful for anomaly detection: attack diversity, the existence of labeled data and precision in labelling (not having non malicious data incorrectly classified). Stolfo et al. [19] came up with the idea of using concepts until then used mainly in economics, fraud detection and optimization to the problem of defining the features of a dataset for intrusion detection. Cost-based assertions developed for fraud detection have shown to be generalizable to the domain of network intrusion detection as a criteria for feature finding. With this approach in mind, Tavallaee et al. defined a set of features intrinsically related to specific classes of traffic anomalies [18]. By maximizing cost in the specific problem, a number of features was identified by the authors to be used in the University of New Brunswick (UNB) NSL KDD Intrusion Detection Evaluation Dataset. In this context the features chosen were the best candidates for maximizing the types of cost characteristic to intrusions: (1) damage cost, the amount of damage caused by an attack if intrusion detection is not attained; (2) challenge cost, the cost to act upon a potential intrusion when it is detected; and (3) operational cost, the resources needed to identify the attacks.

The dataset is composed of sequences of entries in the form of records labeled as *normal* or *attack*. Each entry contains a set of characteristics of a *flow*, i.e., of a sequence of IP packets starting at a time instant and ending at another, between which data flows between two IP addresses using a transport-layer protocol (TCP, UDP) and an application-layer protocol (HTTP, SMTP, SSH, IMAP, POP3, or

Class	Train dataset attacks	Test dataset only attacks
Probing	portsweep, ipsweep, satan, guesspasswd, spy, nmap	snmpguess, saint, mscan, xsnoop
DoS	back, smurf, neptune, land, pod, teardrop, buffer overflow, warezclient, warezmaster	apache2, worm, udpstorm, xterm
R2L	imap, phf, multihop	snmpget, httptunnel, xlock, sendmail, ps
U2R	loadmodule, ftp write, rootkit	sqlattack, mailbomb, processtable, perl

Table 2: Attacks in the UNB NSL KDD train and test datasets (all attacks from the first exist also in the second).

FTP). The dataset is fairly balanced with prior class probabilities of 0.465736 for the *normal* class and 0.534264 for the *anomaly* class. The attacks represented in dataset fall in four classes:

- *Denial of Service (DoS).* A DoS attack is a type of attack in which the attacker exhausts resources (e.g. CPU, RAM), hence denying users access to a machine.
- *User to Root (U2R).* These attacks are exploits are privilege escalations attempting to abuse vulnerabilities in the system in order to access more privileges.
- *Remote to Login (R2L).* A remote to user attack is an attack in which the system is probed remotely to expose vulnerabilities and with this information a login to the system is obtained.
- *Probing.* Probing is an attack in which vulnerabilities are identified that may be exploited so as to compromise the system.

The UNB NSL KDD dataset is composed of two sub-datasets: a *train dataset*, used for training a NIDS, and a *test dataset*, used for testing. Both have the same structure and contain all four types of attacks. However, the test dataset has more attacks as shown in Table 2, to allow evaluating the ability of algorithms to generalize. Each record of the dataset is characterized by features that fall into three categories: basic, content related, and traffic related. These features are represented in Table 3.

3.2 The UNSW NB15 dataset

The second dataset comes from the Australian Centre for Cyber Security (ACCS) [14, 16]. The IXIA PerfectStorm tool[4], an application traffic simulator, has been used in the ACCS Cyber Range Lab to create a dataset with normal and abnormal network traffic. The abnormal traffic is created with the IXIA tool, which contains information about new attacks updated using MITRE's Common Vulnerabilities and Exposures (CVE) site.

The number of records in the training set is 175, 341 and the testing set has 82, 332 records from the different types, attack and normal. The UNSW NB15 data set includes nine categories of attacks:

- *Fuzzers*: in this type of attack, randomly generated data is feed into a suspended program or network;

[4] https://www.ixiacom.com/products/perfectstorm

Feature	Description
duration	length of the flow in seconds
protocol-type	type of protocol, e.g., TCP, UDP, ICMP
service	network service, e.g., HTTP, Telnet
src-bytes	num. of data bytes from source to destination
dst-bytes	num. of data bytes from destination to source
flag	status of the flow, normal or error
lang	1 if flow is for the same host/port; 0 otherwise
wrong-fragment	num. of erroneous fragments
urgent	num. of urgent packets
hot	num. of hot indicators
num-failed-logins	num. of failed login attempts
logged-in	1 if successfully logged in; 0 otherwise
num-compromised	num. of compromised conditions
root-shell	1 if root shell is obtained; 0 otherwise
su-attempted	1 if su root command attempted; 0 otherwise
num-root	num. of root accesses
num-file-creations	num. of file creation operations
num-shells	num. of shell prompt
num-access-files	num. of operations on access control files
num-outbound-cmds	num. of outbound commands in a ftp session
is-host-login	1 if the login belongs to the hot list; 0 otherwise
is-guest-login	1 if the login is a guest login; 0 otherwise
count	num. of connections to the same host as current
serror-rate	% of connections that have SYN errors
rerror-rate	% of connections that have REJ errors
same-srv-rate	% of connections to the same service
diff-srv-rate	% of connections to different services
srv-count	num. of connections to the same service as current
srv-serror-rate	% of connections that have SYN errors
srv-rerror-rate	% of connections that have REJ errors
srv-diff-host-rate	% of connections to different hosts
dst-host-count	num. of connections to the same destination host
dst-host-srv-count	num. of connections to the same service as current
dst-host-same-srv-rate	% of connections to the same service
dst-host-diff-srv-rate	% of connections to different services
dst-host-same-src-port-rate	% of connections from same source and port
dst-host-srv-diff-host-rate	% of connections to different services
dst-host-serror-rate	% of connections that have SYN errors
dst-host-srv-serror-rate	% of connections that have SYN errors per service
dst-host-rerror-rate	% of connections that have REJ errors
dst-host-srv-rerror-rate	% of connections that have REJ errors per service

Table 3: Features used to characterize each flow in the UNB NSL KDD dataset: basic (top), content (middle), traffic (bottom).

– *Reconnaissance*: The attacker gathers information about a system for later attacking it;
– *Shellcode*: code is used as the payload in an attack packet;
– *Analysis*: includes port scan, spam and HTML files penetrations;
– *Backdoors*: access to a system is gained by bypassing any secured layers;
– *Denial of Service*: the perpetrator intents resource unavailability by temporarily or indefinitely disrupting services;
– *Exploits*: the attacker exploits vulnerabilities of the system through known loopholes of the system;
– *Generic*: the attack is implemented without knowing how the cryptographic primitive is implemented and works for all block ciphers;
– *Worms*: the attack mechanism replicates itself through the network.

The UNSW NB15 dataset includes 9 different moderns attack types (compared to 21 older attack types in UNB NSL KDD dataset) and wide varieties of real normal activities as well as 49 features inclusive of the class label consisting total of $2,540,044$ records. These features are categorized into six groups of features: flow related, Basic Features, content related, time related, additional generated and labelled features. These features include a variety of packet-based features and flow related features. The packet based features assist the examination of the payload beside the headers of the packets. On the contrary, for the flow based features and maintaining low computational analysis instead of observing all the packets going through a network link, only connected packets of the network traffic are considered. Moreover, the flow related features are based on traffic direction, inter-arrival time and inter-packet length. The matched features are categorised into several groups (Table 4).

4 Performance Metrics

We have chosen *logloss* as the primary criteria for assessing the model's performance, and used the *Area Under Curve* (AUC) to obtain additional insights. Moreover, we show that there are a number of other metrics that are closely correlated to those, so optimizing one will likely provide comparable results for others (Section 6.2).

The *logloss* (logarithmic loss) metric in Equation 6 can be used to evaluate the performance of both binomial and multinomial classifiers [12]. logloss evaluates how close the values predicted by a model (uncalibrated probability estimates) are to the actual target value. In other words this metric captures the extent to which predicted probabilities diverge from class labels: it incentivates well-calibrated probabilities.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} w_i(\ y_i \ln(p_i) + (1 - y_i) \ln(1 - p_i)\ ) \tag{6}$$

In the equation, N is the total number of rows (observations) of the dataset; w is the per-row user-defined weight (defaults is 1); p is the predicted value (uncalibrated probability) assigned to a given row by the algorithm; y is the actual target value.

Receiver operating characteristic (ROC) graphs (or curves) are useful for visualizing the performance of classifiers [8]. ROC graphs have, in recent years, been increasingly used in machine learning and data mining research. The AUC, i.e., the area under a ROC curve, represents the probability that a given sample classification is (correctly) rated or ranked with greater suspicion than a randomly chosen classification [3, 10].

The ROC curve is a useful tool for a few reasons: The curves of different models can be compared directly in general or for different thresholds. Also the AUC can be used as a summary of the model skill. The shape of the curve is informative in terms of asserting the balance between the classification for the classes. Smaller values on the x-axis of the plot indicate lower false positives and higher true negatives. Larger values on the y-axis of the plot indicate higher true positives and lower false negatives. The correlation among the models metrics was calculated having as input data the specific figures for these metrics for each of

#	Name	Type	Description
1	srcip	N	Source IP address
2	sport	I	Source port number
3	dstip	N	Destination IP address
4	dsport	I	Destination port number
5	proto	N	Transaction protocol
6	state	N	The state and its dependent protocol, e.g. ACC, CLO, else (-)
7	dur	F	Record total duration
8	sbytes	I	Source to destination bytes
9	dbytes	I	Destination to source bytes
10	sttl	I	Source to destination time to live
11	dttl	I	Destination to source time to live
12	sloss	I	Source packets retransmitted or dropped
13	dloss	I	Destination packets retransmitted or dropped
14	service	N	http, ftp, ssh, dns .., else (-)
15	sload	F	Source bits per second
16	dload	F	Destination bits per second
17	spkts	I	Source to destination packet count
18	dpkts	I	Destination to source packet count
19	swin	I	Source TCP window advertisement
20	dwin	I	Destination TCP window advertisement
21	stcpb	I	Source TCP sequence number
22	dtcpb	I	Destination TCP sequence number
23	smeansz	I	Mean of the flow packet size transmitted by the src
24	dmeansz	I	Mean of the flow packet size transmitted by the dst
25	trans depth	I	the depth into the connection of http request/response transaction
26	res bdy len	I	The content size of the data transferred from the server's http service
27	sjit	F	Source jitter (mSec)
28	djit	F	Destination jitter (mSec)
29	stime	T	record start time
30	ltime	T	record last time
31	sintpkt	F	Source inter-packet arrival time (mSec)
32	dintpkt	F	Destination inter-packet arrival time (mSec)
33	tcprtt	F	The sum of 'synack' and 'ackdat' of the TCP.
34	synack	F	The time between the SYN and the SYN ACK packets of the TCP.
35	ackdat	F	The time between the SYN ACK and the ACK packets of the TCP.
36	is sm ips ports	B	If source (1) equals to destination (3)IP addresses and port numbers (2)(4) are equal, this variable takes value 1 else 0
37	ct state ttl	I	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
38	ct flw http mthd	I	No. of flows that has methods such as Get and Post in http service.
39	is ftp login	B	If the ftp session is accessed by user and password then 1 else 0.
40	ct ftp cmd	I	No of flows that has a command in ftp session.
41	ct srv src	I	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
42	ct srv dst	I	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
43	ct dst ltm	I	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
44	ct src ltm	I	No. of connections of the same source address (1) in 100 connections according to the last time (26).
45	ct src dport ltm	I	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
46	ct dst sport ltm	I	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
47	ct dst src ltm	I	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).

Table 4: Features of the UNSW NB15 dataset: flow-related (top), basic (next), content-related (next), time-related (next), and connection (bottom).

the models produced. A model has been obtained for each point of the grid search space (explained next).

XGBoost model parameter	**Values**
max_depth	5/10
eta	0.1/0.3/0.5
sample_rate	0.1/0.5
gamma	0.0/0.01
reg_lambda	0.1/0.9
reg_alpha	0.1/0.9

Table 5: Feature range used in the optimization phase (discrete values).

5 Experimental Approach and Parameters

Our experimental approach is summarised as follows:

- To define the values of the model's parameters for the two datasets, we used a grid search approach, i.e., we did a search over the hyperparameter space using H2O's grid search functionality. The range of discrete values used for the parameters in the search is in Table 5.
- In random grid search a stopping criterion can be specified to define when the random grid search is completed. In this case we have chosen to stop when performance stops improving by a specified amount (in the case 0.01).
- The models are ranked via the validation logloss minimization criteria;
- The best model is set to be the model with the minimum validation logloss value;
- A correlation calculation of the models parameters versus logloss has been performed in order to seek which types of metrics can be used as proxy to estimate others;
- The two correlations were validated with a significance test.

XGBoost, configured for acting as a tree booster, has been configured with the following parameters:

- *ntrees*: number of trees to build.
- *max-depth* (default=6): maximum depth of a tree. Used to control over fitting as higher depth will allow model to learn relations very specific to a particular sample.
- *tree-method*: construction tree method to use.
- *eta* (default=0.3): learning rate by which to shrink the feature weights. Shrinking feature weights after each boosting step makes the boosting process more conservative and prevents overfitting.
- *gamma* (default=0): a node is split only when the resulting split gives a positive reduction in the loss function. Gamma specifies the minimum loss reduction required to make a split. Makes the algorithm conservative. The values can vary depending on the loss function and should be tuned.
- *subsample* (alias sample-rate) (default=1): row sampling ratio of the training instance.
- *grow-policy* (default="depthwise"): the way that new nodes are added to the tree. "depthwise" splits at nodes that are closest to the root.
- *booster* (default="tree"): booster type.
- *lambda* (default=1): L2 regularization term on weights (Ridge regression).
- *alpha* (default=0) L1 regularization term on weight (Lasso regression).

The values used in the grid search are listed in Table 5. All other parameters were assumed to the default values.

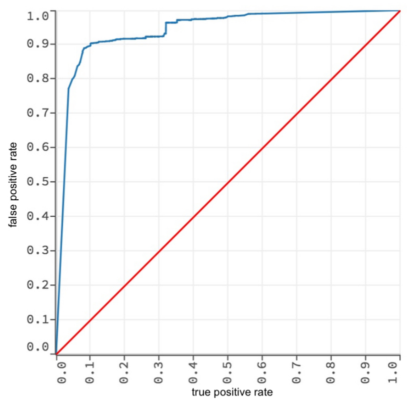


Fig. 1: Best UNB NSL KDD model ROC curve for the validation dataset, AUC = 0.941384.

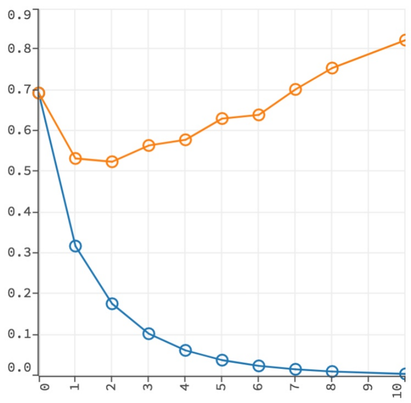


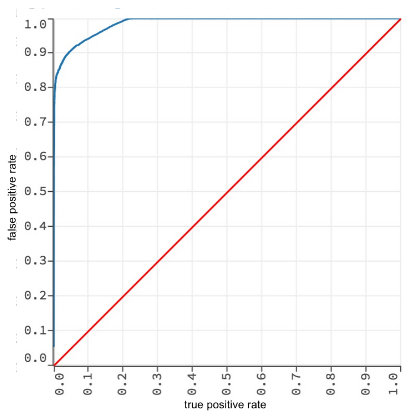
Fig. 2: UNB NSL KDD logloss values by number of trees (1 to 10). Curves obtained with the training dataset (blue) and with the validation dataset (orange).

6 Results

The objective of the experimental evaluation was to answer the three questions posed in the introduction. It started with a grid searching phase, as explained in Section 5. The parameters for the best models obtained for both datasets are depicted in Table 7. Figs. 9, 10 and 11 show performance values for the UNB NSL KDD dataset. Figs. 12, 13 and 14 do the same for the UNSW NB15 dataset.

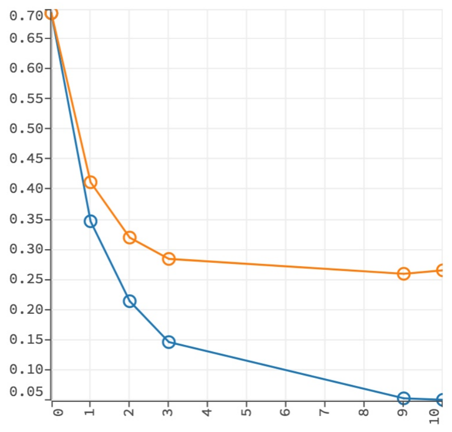
6.1 Logloss and AUC results

Regarding logloss, we can see in Figs. 2 and 4 the fast convergence obtained with the training dataset as seen by number of trees in the x-axis (blue lines). Focusing on the validation dataset, for the KDD dataset the logloss figures did not converge to a minimum value (Fig. 2, orange line). The value obtained for AUC was high,



(a)

Fig. 3: UNSW NB15 ROC curve for best model validation, AUC=0.985961.



(a)

Fig. 4: UNSW NB15 logloss values by number of trees (1 to 10). Curves obtained with the training dataset (blue) and with the validation dataset (orange).

0.959016 (see Table 6). As can be observed, the curve obtained is relatively well-balanced (Figure 1). The predictor scored better with the UNSW NB15 dataset, in line with previous results by Moustafa [15].

6.2 Interpreting the models

Figs. 5 and 6 show that there are clusters of metrics highly correlated between each other (one central cluster with strong negative correlation and two triangle vertices clusters with strong positive correlation). The figures represent values of the Pearson correlation coefficient, probably the most widely used measure of the

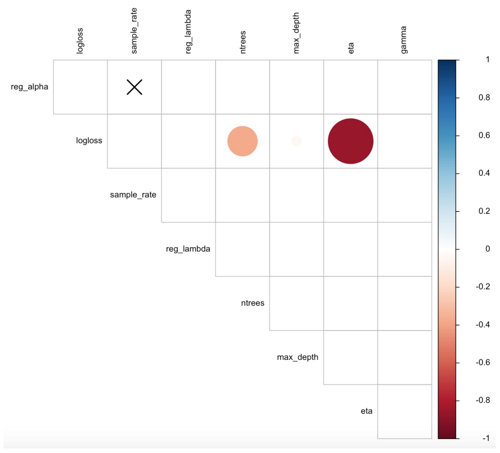


Fig. 5: UNB NSL KDD Pearson correlation for the model variables. Correlation values found to be non significant are crossed. The magnitude of the correlation links to the size of the circles. Red colors are negative and blue positive. Values processed for all the iterated models. Significance level for the $p - test = 0.001$.

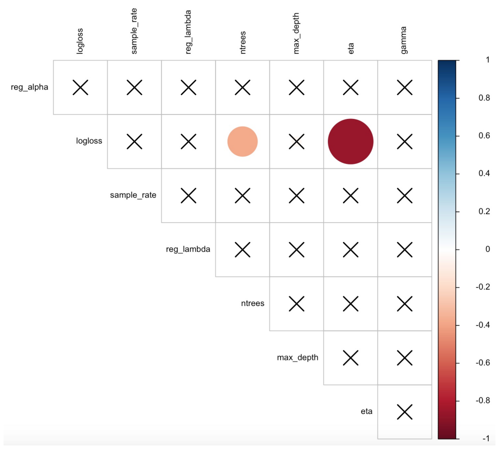


Fig. 6: Pearson correlation for the model variables for UNSW NB15. Non-significant correlations are crossed (p-test significance level of 0.001). Correlation magnitude represented by the circle diameter and color (red negative blue positive). Values processed for all the iterated models.

extent to which two variables are linearly related: X, Y: $\rho(X, Y) = \frac{\text{cov}(X,Y)}{\sigma_x \sigma_y}$. The metrics considered in this correlation test are[5]: Validation R2, Validation Classification Error, Training Logloss, Cross Validation Logloss, Training RMSE, Cross Validation RMSE, Training Classification Error, Cross Validation Classification Error, Validation AUC, Validation Lift, Validation RMSE, Validation Logloss,

[5] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/performance-and-prediction.html

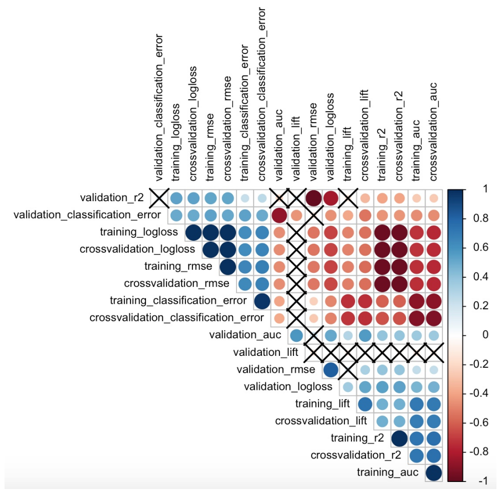


Fig. 7: UNB NSL KDD Pearson correlation tested against significance at a 0.001 level. Correlations found to be non significant are crossed. The magnitude of the correlation links to the size of the circles presented color coded.

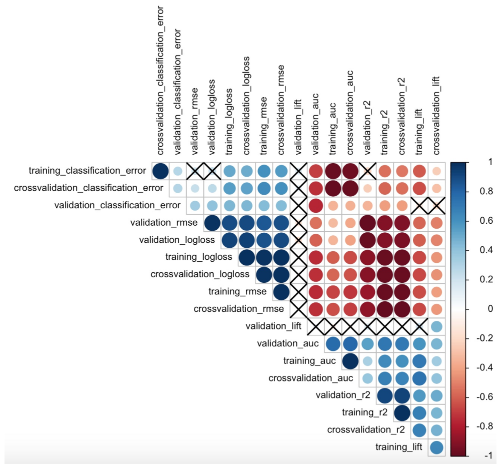


Fig. 8: UNSW NB15 Pearson correlation tested against significance at a 0.001 level. Correlations found to be non significant are crossed. The magnitude of the correlation links to the size of the circles presented color coded.

Training Lift, Cross Validation Lift, Training R2, Cross Validation R2, Training AUC. The values of the Pearson correlation are shown with a p-test at a significance level of 0.001, so many correlations were not found significant and are shown crossed.

Two main clusters are evident for both UNB NSL KDD (Fig. 5) and UNSW NB15 cases (Fig. 6), each with a high positive correlation and another with a high negative correlation among parameters. In Figs. 7 and 8, the training features

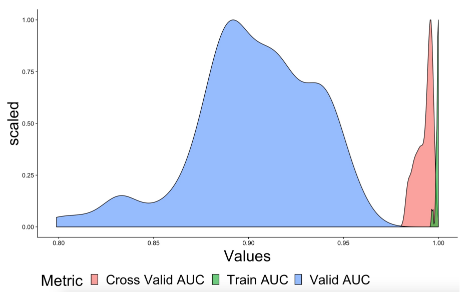


Fig. 9: UNB NSL KDD logloss values dispersion. Validation AUC has a large spread. On the opposite Cross Validation AUC and Training AUC have low spread.

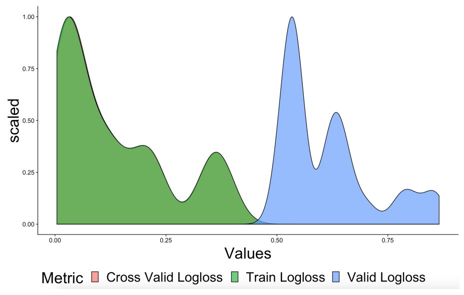


Fig. 10: UNB NSL KDD logloss values dispersion. Validation AUC has a large spread. On the opposite Cross Validation AUC and Training AUC have low spread. Cross Validation logloss superimposes with Train logloss.

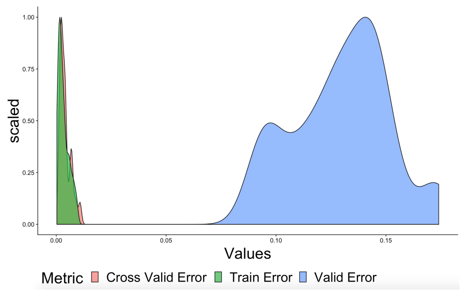


Fig. 11: UNB NSL KDD classification error values dispersion.

for both models show correlations with the training metric used, logloss. It is noticeable that for both training models the single most relevant features to affect the logloss results are the *eta* and *ntrees* features. Some of the metric correlations also did not pass the significance test. These values are shown as crossed or empty. The extreme focus on only two (the same) features for the two datasets is likely to be attributed to the effects of using Lasso as regularizer.

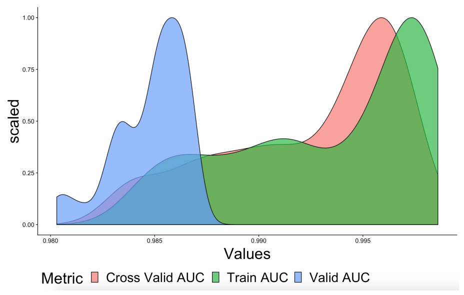


Fig. 12: UNSW NB15 AUC values dispersion.

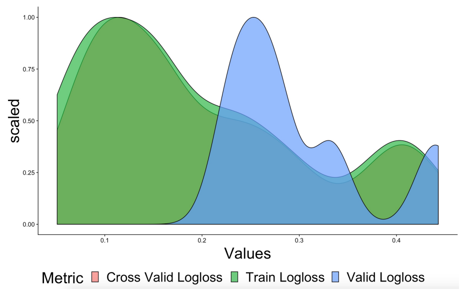


Fig. 13: NB15 logloss values dispersion.

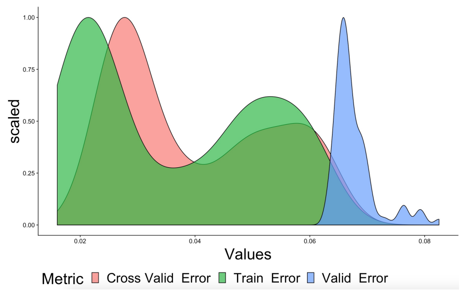


Fig. 14: UNSW NB15 classification error values dispersion.

6.3 How much information is hidden in the performance metrics?

The graphs in Figs. 9, 10 and 11 can be understood as a way of empirically assessing the amount of information XGBoost manipulate in the datasets. This information amount can be assessed as an entropy value by means of the parallel with the entropy concept (equation 7): given the graphs are density functions, the entropy magnitude can be inferred as proportional to the graphs area, as per equation 7.

$$H(X) = -\int_x p(x) \log p(x) \, dx \tag{7}$$

Dataset	KDD-NSL	UNSW NB15
Training logloss	0.0069	0.0511
Validation AUC	0.941384	0.985961
Training AUC	0.999999	0.998551
Cross-Validation AUC	0.999975	0.997226

Table 6: Best validation AUC values for the UNB NSL KDD and UNSW NB15 datasets. The minimum training logloss obtained for the stop criteria is listed.

Parameter	NSL KDD	UNSW NB15
gamma	0.01	0.01
eta	0.3	0.1
max_depth	5	5
sample_rate	0.1	0.2

Table 7: Experimental best model features for the two datasets.

It is straightforward to observe that the bigger this area is (entropy-like magnitude) for the training metrics the narrower and focused the values for the validation metrics are. The inverse is also observable.

6.4 Correlation validation

The p-test is used to check statistical significance of the correlation values obtained. Small p-values occur when the null hypothesis is false.

The validity of the test depends on the p-value used. Apparently small values of p like 0.01 still open a relatively large probability for the null hypotheses. Expanding on earlier work, especially Edwards et al. [7], it is shown that actual evidence against a null can differ by an order of magnitude from the p-value. The choice here is to choose a p-value equal or smaller than 0.001 as per Table 8.

6.5 Discussion

Here we present a short argument on how we answer the three questions posed in the introduction:

1. How effectively can XGBoost be used in the context of NIDSs? Our experiments show that XGBoost is a good approach for network intrusion detection, as the metrics we obtained can be considered very good, e.g., AUC higher than 0.990 (Table 6).
2. How to optimize XGBoost for this application? We proposed a grid search approach (Section 5). We used it to obtain the parameters for the detector for the two datasets (Table 7). The values for both datasets were similar, a fact that suggests that XGBoost is indeed a good approach as parameters can be reasonably stable for different datasets.

$\mathbf{p}$	0.1	0.5	0.01	0.005	0.001
$\alpha(\mathbf{p})$	0.385	0.289	0.111	0.067	0.0184

Table 8: Calibration table as a function of the p-values tested for significance.

3. Among the parameters configured for XGBoost model training, can we find correlations with performance predictors, e.g., area under curve or logloss? The answer is positive, as shown in Figs. 5 and 6.

7 Conclusions

We describe our approach to NIDS using XGBoost. The results show that the techniques proposed in the paper are effective. An evaluation with two well-known NIDS datasets has been performed using the XGBoost R implementation. Performances compatible with the best values obtained with other experiments has been confirmed. We proved the potential XGBoost has to be used in the context of network intrusion detection, in applying XGBoost to two representative datasets often used in machine learning research. A method for optimization and best model parameters identification is described. The method is based on a grid search approach, with the objective of logloss minimization. By minimizing logloss and imposing a stop criteria results have been obtained balancing computational time and model results.

Elements that contribute to the model interpretability were added, specifically elements linked to entropy of the resultant metrics (Section 6.3). Clear correlations were found among many important performance metrics. The correlation calculation executed with the performance metrics allowed to identify which metrics are correlated with others, and above all, identify which metrics do not correlate at all among each other.

References

1. Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B., Rousseau, D.: How machine learning won the Higgs boson challenge. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (Apr 2016)
2. Adam-Bourdarios, C., Cowan, G., Germain, C., Guyon, I., Kégl, B., Rousseau, D.: The Higgs boson machine learning challenge. In: NIPS 2014 Workshop on High-energy Physics and Machine Learning. pp. 19–55 (2015)
3. Bradley, A.P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition **30**(7), 1145–1159 (Jul 1997)
4. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794 (2016)
5. Debar, H., Dacier, M., Wespi, A.: A revised taxonomy of intrusion detection systems. Annales des Télécommunications **55**(7), 361–378 (2000)
6. Dhaliwal, S., Nahid, A.A., Abbas, R.: Effective intrusion detection system using XGBoost. Information **9**(7), 1–24 (2018)
7. Edwards, W., Lindman, H., J. Savage, L.: Bayesian statistical inference in psychological research. Psychological Review **70**, 193–242 (05 1963)
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters **27**(8), 861 – 874 (2006)
9. García-Teodoro, P., Díaz-Verdejo, J., Maciá-Fernández, G., Vázquez, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security **28**(1-2), 18–28 (Feb 2009)

10. Hanley, J., Mcneil, B.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology **143**, 29–36 (05 1982)
11. Higgs, P.W.: Broken Symmetries and the Masses of Gauge Bosons. Physical Review Letters **13**, 508–509 (1964)
12. Masnadi-Shirazi, H., Vasconcelos, N.: On the design of loss functions for classification: Theory, robustness to outliers, and savageboost. In: Proceedings of the 21st International Conference on Neural Information Processing Systems. pp. 1049–1056 (2008)
13. Mitchell, R., Chen, I.R.: A survey of intrusion detection techniques for cyber-physical systems. ACM Computing Surveys **46**(4), 55:1–55:29 (Mar 2014)
14. Moustafa, N., Slay, J.: UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military Communications and Information Systems Conference. pp. 1–6 (Nov 2015)
15. Moustafa, N.: Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic. Ph.D. thesis, University of New South Wales, Canberra, Australia (2017)
16. Moustafa, N., Slay, J.: 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 (Apr 2016)
17. Nielsen, D.: Tree Boosting With XGBoost – Why Does XGBoost Win "Every" Machine Learning Competition? Master's thesis, NTNU (2016)
18. Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A.A.: Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Computers & Security **31**(3), 357–374 (May 2012)
19. Stolfo, S.J., Fan, W., Lee, W., Prodromidis, A., Chan, P.K.: Cost-based modeling for fraud and intrusion detection: Results from the JAM project. In: Proceedings of the 2000 DARPA Information Survivability Conference and Exposition. pp. 130–144 (2000)
20. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: Proceedings of the 2nd IEEE International Conference on Computational Intelligence for Security and Defense Applications. pp. 53–58 (2009)
21. The CMS Collaboration: Evidence for the direct decay of the 125 GeV Higgs boson to fermions. Nature Physics **10** (Jun 2014)
22. The CMS Collaboration: Observation of the Higgs boson decay to a pair of τ leptons with the CMS detector. Physics Letters **B779**, 283–316 (2018)
23. Vasilomanolakis, E., Cordero, C.G., Milanov, N., Mühlhäuser, M.: Towards the creation of synthetic, yet realistic, intrusion detection datasets. In: IEEE/IFIP Workshop on Security for Emerging Distributed Network Technologies. pp. 1209 – 1214 (Apr 2016)
24. Wheeler, P., Fulp, E.: A taxonomy of parallel techniques for intrusion detection. In: Proceedings of the 45th Annual Southeast Regional Conference. pp. 278–282 (2007)