Optimal Machine Learning Algorithms for Cyber Threat Detection

Hafiz M. Farooq
Information Security Division
Expec Computer Center, Saudi Aramco
Dhahran, Saudi Arabia
hafiz.farooq@aramco.com

Naif M. Otaibi
Information Security Division
Expec Computer Center, Saudi Aramco
Dhahran, Saudi Arabia
naif.otaibi.19@aramco.com

Abstract — **With the exponential hike in cyber threats, organizations are now striving for better data mining techniques in order to analyze security logs received from their IT infrastructures to ensure effective and automated cyber threat detection. Machine Learning (ML) based analytics for security machine data is the next emerging trend in cyber security, aimed at mining security data to uncover advanced targeted cyber threats actors and minimizing the operational overheads of maintaining static correlation rules. However, selection of optimal machine learning algorithm for security log analytics still remains an impeding factor against the success of data science in cyber security due to the risk of large number of false-positive detections, especially in the case of large-scale or global Security Operations Center (SOC) environments. This fact brings a dire need for an efficient machine learning based cyber threat detection model, capable of minimizing the false detection rates. In this paper, we are proposing optimal machine learning algorithms with their implementation framework based on analytical and empirical evaluations of gathered results, while using various prediction, classification and forecasting algorithms.**

Keywords: SOC, Machine Learning, Anomaly Detection, Prediction, Classification, Numerical Clustering, Dimensionality, Regression, Decision Trees, Ensemble Learning, Deep Learning.

I. INTRODUCTION

Many organizations are now hitting physical limits with traditional approaches and technologies to collect, parse, normalize, search, analyze, visualize and explore the huge volume of security events collected by Security Information and Management Systems (SIEM). Traditional SIEM devices in any Security Operations Center (SOC) are designed to statically correlate security events and generate alerts for potential cyber threats. Most SIEM systems rely on these static threats correlation rules to report security incidents and kick-start the Incident Response (IR); but these correlation rules lack context and memory of normal behavioral baselines, and cause significant operational overheads to maintain, and thus falling short for providing real-time data mining capability that is of crucial importance to detect advanced and complex cyber-attacks.

A typical enterprise network with thousands of IT systems, generates billions of security events per day. Generally, only a subset of these security events are ingested by SIEMs to run through threat correlation rules, however organizations are now aiming to ingest as much security events as possible to meet regulatory requirements, and to leverage this security big data to enhance investigative visibility, conduct Threats Hunting (TH) and expedite the Incident Response (IR) process through live and historical analysis of security events. This trend has made the situation more worst and unmanageable by SOC Analysts in large-scale Security Operations Centers.

Therefore, organizations have now realized that the traditional monitoring complemented by an effective and versatile Machine Learning based Threat Hunting will be a necessary part of any Security Monitoring portfolio. Detection of complex cyber-attacks due to its noisy security log data will require utilization of classification, regression and forecasting algorithms to spot anomalies and detection new patterns to cater cyber-based as well as insider threats.

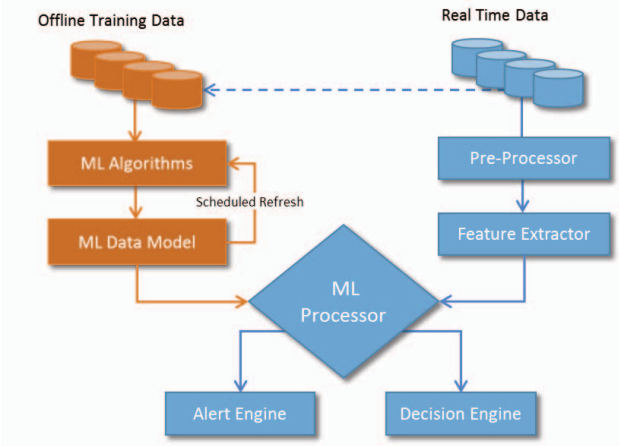

Figure 1. Machine Learning Analytical Work-flow (MLAW).

Nevertheless, Machine Learning based security analytics should be performed through an Optimal Workflow (as shown in figure-1) in order to ensure efficient preprocessing of data before applying a nicely trained machine learning predictor or classifier for subsequent analytics. Such a workflow can help in addressing all challenges by reducing the huge volume of security events to a few outlier events and providing security analysts with potential indicators of malicious activities to feed into cyber threats detection and hunting processes. Different algorithms work best for different security events types, and analysts should choose

0-7695-6405-4/18/$31.00 ©2018 IEEE
DOI 10.1109/UKSim.2018.00018

ones that better model the data and lead to least number of false-positives.

II. C YBER THREAT CASES TAXONOMY

Before performing machine learning based analytics for Cyber Threat Detection, we deliberated multiple MITRE's taxonomies (ATT&CK, CAPEC, MAEC) in order to select a standard reference for cyber threat cases. ATT&CK and CAPEC are two suitable references that also recommended by [1], where the first reference provides explanation of wide range of post-compromise techniques and latter just enumerates general types of attack patterns for overall cyber-attack life cycle. ATT&CK threats cases, being more practical and detection oriented (figure-2), is more suitable for demonstrating the machine learning analytics. MAEC is limited in scope [2] as it only envisages explanation of malware specific standardization. Moreover, Lockheed Martin's 7-Phase Cyber Kill Chain model is a wholesome guideline for both pre-compromise and post-compromise stages, without any classification of techniques and procedures used at different attack phases.

Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery
Lateral Movement	Executions	Collection	Exfiltration	Command & Control

Figure 2. MITRE ATT&CK Matrix (Techniques/Tactics).

Out of ten types explained in above ATT&CK matrix, below example cases were carefully selected, analyzed and modeled using machine learning algorithms in this paper.

TABLE I. ATT&CK USE CASES

Techniques	Example Use Cases
Executions	Anomalous Process Executions
Discovery	Predicting User's Processes
Ex-Filtration	Data Rate Analytic
Exploitation	Parent Child Analytic

III. OPTIMAL MACHINE LEARNING ALGORITHMS FOR CYBER THREAT DETECTION

A. Data Rate Analytic using Clustering Algorithms

Analyzing internet upload and download traffic is acutely important for initial detection of cyber-attacks; as well as for subsequent triage and network forensics. In larger enterprises, due to the expected high volume of network traffic, conventional statistical tools are insufficient to detect abnormal network sessions/flows. So, Numerical Clustering as highlighted in [3] & [4] can be used as an effective method to filter normal and abnormal network traffic. In this use case, we analyzed and compared Partitional Clustering algorithms (including KMeans, DBSCAN, BIRCH) for upload/download traffic analysis because of the numerical nature of the dataset, and due to KMeans' better performance in comparison to many other Hierarchical algorithms (Agglomerative, Divisive, etc.).

KMeans: Out of many available non-hierarchical and partitional algorithms, we selected K-Means because of its ability to cluster numerical data quickly and its ease of use. KMeans partitions n observations (data points) into *k* spatial clusters, where each of the observation is assigned to a cluster based on Euclidean distance from iteratively selected centroid. Estimating the optimal value of the variable *k* (number of cluster) is a challenge here, however best option available is elbow method [5][6][7] that finds the optimal value of *k* by gradually increasing its value to reach a relatively stable dispersion or distance.

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \|x_i^j - \mu_i\|^2$$

Conventionally, K-Means Clustering creates clusters of homogeneous shapes and spatial extents [6] thus converting large space into Voronoi cells, whereas (other contemporary algorithms) try to find centroids by using expectation-maximization principle thus leading dissimilar shapes clusters. From Cyber Security perspective, K-Means can be comprehensively used for clustering enterprise users based on their download and upload rates. Below K-Means analysis (figure-3) explicates the results sought by clustering the users' traffic data from enterprise internet gateway (firewall) into three clusters (k=3) thus earmarking low, medium and highly active internet users. Additional pre-processing could be done through PCA (Principle Component Analysis) for dimensionality reduction and StandarScalar for scaling data without impact data variance.

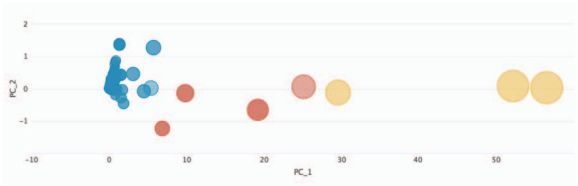

Figure 3. KMeans Clustering - Upload/Download Rates.

Algorithm 1 Data Rate Analytic using KMeans

1: **DataSource:** WindowsLog
2: **if** BytesSent $>$ 0 OR BytesReceived $>$ 0 **then**
3: *do* **StandardScaler** $BytesSent,\ BytesReceived$
4: *do* **PCA** $SS_BytesSent,\ SS_BytesReceived$
5: *do* **KMeans** k=3 ,
6: **end if**

These K-Means generated clusters can be diagrammatically analyzed with the help of below ex-filtration quadrant (figure-4), where clusters positioned at quadrants-II & IV pose relatively major risk of data ex-filtration subjected to any suspected Command-and-Control (C2) beacons, data leakage, remote access trojans (RAT) or an unauthorized

software download activity. Therefore, user data-points from such abnormal clusters are then further fed to the advanced threat hunting and machine learning analytic process for detailed forensics by the incident handler teams.

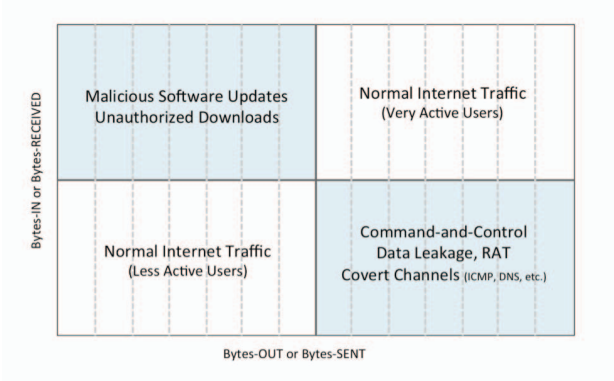

Figure 4. Ex-Filtration Quadrant.

Density-Based Spatial Clustering (DBSCAN) creates cluster regions with relatively higher spatial-density of data points, whereas low density regions are earmarked as clusters of noise and outliers. This algorithm is based on two important input parameters [8], radius of neighborhood and minimum no of points *minPts* that must exist in cluster neighborhood.

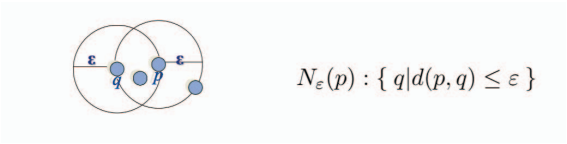

Figure 5. DBSCAN Algorithm.

DBSCAN creates more meaningful clusters in case operational and behavioural studies being resistant to noise and quite adaptive, can handle clusters of different shapes and sizes [9]. However, being dependent on its initialization parameters ε and *minPts,* DBSCAN is generally less accurate for cyber security use cases, like upload/download internet traffic due to the dynamically varying traffic densities and highly scattered data values.

Balanced Iterative Reducing & Clustering using Hierarchical (BIRCH) is an efficient multi-level hierarchical clustering algorithm that builds an in-memory dendrogram [10], known as clustering feature (CF) tree. Each feature or cluster in CF tree is represented through three parameters (N: no of data points, LS: linear sum, SS: square sum) and represented mathematically:

$$N, LS = \sum_{i=1}^{N} X_i, \; SS = \sum_{j=1}^{N} X_j^2$$

BIRCH clustering can be very slow for larger datasets as in this use case, since it involves multiple clustering iterations depending upon the data resolution and partitioning level required. Hence, its roles in cyber security is only limited to micro-level clustering on subset of data, in conjunction with a macro-level algorithm (KMeans, DBSCAN, etc.) clustering data at the higher level.

B. Anomaly Detection in Process Executions using One-Class SVM (OCSVM) Classifier

Finding anomalous Windows Processes in millions of processes executed in a large-scale Data Center needs an army of SOC Analysts to investigate; erstwhile deterministic or static-correlation based detection rules cannot encompass diversity of anomalous process execution patterns. In this study, we used Microsoft SYSMON Tool for analyzing process creation logs (Event ID: 1) installed locally on multiple enterprise hosts [11]. 'Curse of Dimensionality' [12] in in this hyper-dimensional process execution datasets can be effectively handled through Support Vector Machines (SVM) because of their kernel based classification approach.

However, One-Class SVM (OCSVM) algorithm, proposed by *SchÖlkopf et al*.[13] is relatively well suited for novelty detection scenario where only "normal" dataset is available and no outliers are practically known. So, anomaly detection in Windows Process Execution is a similar use case where execution behavior differing the normal is suspicious and generally unknown. So we extracted Windows' process command line parameters in the feature-extraction phase using Microsoft's SYSMON events and then fed to the OCSVM classifier.

Algorithm 2 Process Anomaly Detection using OCSVM

1: **DataSource:** SYSMON-Logs
2: **if** EventID $== 1 AND$ isNormal! $= 1$ **then**
3: *do* **OneClassSVM**
4: *set kernel* $= poly$ *nu* $= 0.01$ *coef* $= 0.5$
5: *set gamma* $= 0.01$ *tol* $= 1$ *deg* $= 3$ *shrinking* $= f$
6: *save* **Model** $ProcExecOCSVM$
7: *do* **Dedup** $ProcessName$
8: **end if**

OCSVM uses kernel approach to map the input dataset in a hyper-dimensional feature space H and iteratively finds the maximal margin hyper-plane that best separates the data subsets from the origin.

$$f(x) = \begin{cases} +1 & \text{if x} \in S \\ -1 & \text{if x} \in \overline{S} \end{cases}$$

In our context, let $\{x_1, x_2, \ldots, x_l\}$ be training examples belonging to one class X, where X is a compact subset of R^N. Let $\Phi : X \rightarrow H$ be a kernel map which transforms the training examples to another space. Then, to separate the data set from the origin, one needs to solve the following quadratic programming problem:

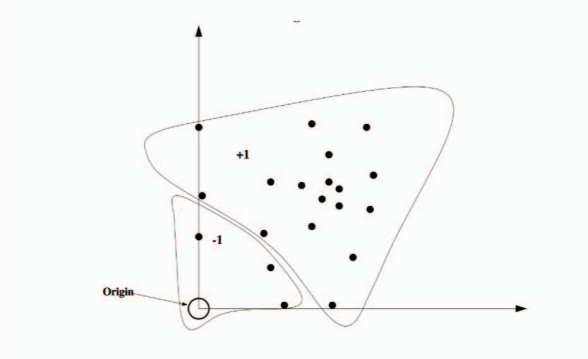

Figure 6. One Class SVM (OCSVM).

$$min\frac{1}{2}\|w\|^2 + \frac{1}{vl}\sum_i \xi_i - \rho$$

$$\text{subject to } (w.\Phi(x_i)) \geq \phi - \xi_i, \xi_i \geq 0.$$

If ω and ρ solve this problem, then the decision function

$$f(x) = sign((w.\Phi(x)) - \rho$$

will be positive for most examples x_i contained in the training set.

We used different *Kernels* (*Gaussian, Linear, Polynomial, Radial)* during this analysis of command line parameters. Selecting the relatively 'suitable' kernel with appropriate kernel parameters is a crucial task for OCSVM, and heavily depends on the specific task at hand. During our analysis, linear kernel offered relatively 'stable' decision boundaries and generated more true anomalies in comparison to others (radial, 2-degree polynomial and other available kernels). Similar results were inferred by *Katherine A. Heller* [10] while analyzing anomalous windows registry entries.

C. *Predicting User Behaviour using Linear Regression*

One potential indicator of malicious activity is an abnormal count of process executions, which would indicate abnormal activity of systems/users when compared to their behavioral baseline. In our experiments, linear regression algorithm worked best to model this threat case given the simple, numerical and linear dataset with low number of input features and with the fact that the features are not co-linear. The below model consists of *DayOfTheWeek* and *UserAccount* as input features, and the dependent variable is the number of process executions for that user in that day of the week (*execount*). The model was trained was four months of known good data to envisage all possible know scenarios. A security analyst can use such a model to spot behavioral anomalies by zooming in on data points where actual process executions is much larger than predicted values generated by the model, indicating potential suspicious/malicious activity.

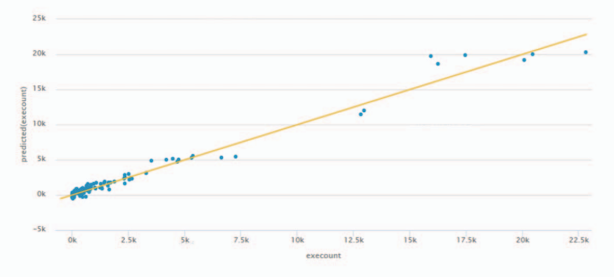

Figure 7. Abnormal Process Executions Detection Using Linear Regression.

Same analysis was done through Random Forest (RF) regressor that also gave much accurate prediction due to its inherent ensemble boosted behaviour [14]. However, Decision Tree (DT) regressor generated more false-positives as it needs more accurate training data.

D. *Anomaly Detection in Process Executions using One-Class SVM (OCSVM) Classifier*

In this case study, we used Logistic Regression to do a transactions analysis of Microsoft Windows Processes and thus finding anomalous Child and Parent process pairs. This machine learning analysis was helpful in disinterring malwares exploiting vulnerabilities by attacking unpatched or unhardened OS components, e.g. *PowerShell* invocation by a JavaScript executed through macro-enabled Microsoft Word document or through Internet Explorer (IE).

Algorithm 3 Parent Child Analytic using LR

1: **DataSource:** WindowsLogs
2: **if** EventID = $4688 AND$ CommandLine! = $EMPTY$ **then**
3: $join$ **WindowsLogs** $\{NewProcID = CreatorProcID\}$
4: do **REGEX** $FindProcNames$
5: do **LogisticRegression** $ChildProc\ ParentProce$
6: $where\ ChildProc <> ParentProce$
7: **end if**

==WMI Command Executed==
Namespace: "root\CIMV2"
Method Executed: Create
Command Executed: powershell.exe -Command {IEX (New-Object Net.Webclient).DownloadString ('http://192.168.110.141/execPayload.ps1')}

Figure 8. Results: Abnormal Child Process.

Logistic regression is a binomial predictive algorithm based on binary logistic function that computes a dichotomous dependent variable value (Yes/No, 0/1, True/False) in relation to a set of independent variables, which can be either discrete and/or continuous. Logistic regression deals with this problem of binomial outcome *Y=p(x)* with continuous input variable *x*, by using a logit transformation which allows modeling of a nonlinear association in a probabilistic linear way. If $ß_0$ and $ß_1$ be the regression coefficients, logistic regression model is presented as:

$$log\frac{p(x)}{1-p(x)} = \beta_0 + x.\beta_1$$

solving for $p(x)$

$$Y = p(x) = \frac{1}{1+e^{-(\beta_0+x.\beta_1)}}$$

Logistic Regression (LR) worked well on child-parent process execution due to the absence of multicollinearity between the independent predictor variables (i.e. time, host, parent). Intrinsically we have limited number of parent process outcomes for every child process, so it makes LR more suitable for process execution analytics. Also, availability of limited training examples (parent-process pairs) and huge feature dataset makes Logistic Regression more suitable for this analytics [15] in comparison to different Decision Trees, *NaiveBays (NB)* or *DecisionTree* classifiers.

E. Message Classification using Random Forest

In this case study, we used Random Forest Classifier (RF) machine learning algorithm for the classification of enterprise mobile messaging dataset. Prepossessing of textual data was done through TF-IDF (Term Frequency-Inverse Document Frequency), a text mining algorithm used for document categorization.

TF-IDF is a numeric measure that is use to score the importance of a word in a document based on how often did it appear in that document and a given collection of documents. The intuition for this measure is: If a word appears frequently in a document, then it should be important and we should give that word a high score. But if a word appears in too many other documents, it's probably not a unique identifier, therefore we should assign a lower score to that word. The math formula for this measure:

$$idf(t, D) = log\frac{|D|}{1 + |\{d \in D : t \in d|}$$

Where *t* denotes the terms; *d* denotes each message or document; *D* denotes the collection of documents. In the following documentation, we'll break down this formula using four small documents to illustrate the idea. Being categorical data, *RandomForestClassifier* decision tree based estimator was used to predict the value of categorical fields (texts). It is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. We did use *RandomForestClassifier* instead of *SVM* as our sample dataset was no reasonably clean and outlier free, this it was the best algorithm to predict spam textual data.

A random forest (RF) classifier algorithm is a predictor consisting of a collection of randomized base regression trees. These random trees are combined to form the aggregated regression estimate. Due to multi-tree estimation random forest classification is relatively more suitable to decrease the bias or overfitting without impacting the characteristic variance.

IV. CONCLUSION

We have presented multiple examples for using Machine Learning analytics to enhance cyber security monitoring along with analysis on optimal algorithms for common cyber threats cases. ML based analytics is an excellent tool to provide context derived from learning security events normal behavioral baselines and leading to minimal number of false-positive security alerts. In addition, ML analytics are best suited to analyze huge volume of security events and feed deviations from normal baselines into proactive threat hunting processes as indicators or leads of potential malicious activity. Moreover, supervised (binary or multi-class classification) algorithms will have limited role to play in the future because of operational cost of preparing good and bad training data to train algorithms models. Semi-supervised (one-class classification) algorithms like One-Class SVM (OCSVM) are relatively easier to train, more cost effective and better suited to enable SOC Analysts to perform novelty detection and uncover new indicators of compromise (IOCs).

REFERENCES

[1] Bromander, Siri, Audun Jøsang, and Martin Eian. "Semantic Cyberthreat Modelling." STIDS. 2016.

[2] Mavroeidis, Vasileios, and Siri Bromander. "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence." Proceedings of the IEEE (2017).

[3] Marchette, David J. "A Statistical Method for Profiling Network Traffic." Workshop on Intrusion Detection and Network Monitoring. 1999.

[4] Gu, Guofei, et al. "BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection." USENIX security symposium. Vol. 5. No. 2. 2008.

[5] Kodinariya, Trupti M., and Prashant R. Makwana. "Review on determining number of Cluster in K-Means Clustering." International Journal 1.6 (2013): 90-95.

[6] Paparrizos, John, and Luis Gravano. "k-shape: Efficient and accurate clustering of time series." Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015.

[7] Celebi, M. Emre, Hassan A. Kingravi, and Patricio A. Vela. "A comparative study of efficient initialization methods for the k-means clustering algorithm." Expert systems with applications 40.1 (2013): 200-210.

[8] Yang, Caihong, Fei Wang, and Benxiong Huang. "Internet traffic classification using dbscan." Information Engineering, 2009. ICIE'09. WASE International Conference on. Vol. 2. IEEE, 2009.

[9] Chakraborty, Sanjay, N. K. Nagwani, and Lopamudra Dey. "Performance comparison of incremental k-means and incremental dbscan algorithms." arXiv preprint arXiv:1406.4751 (2014).

[10] Virpioja, Sami. "BIRCH: Balanced Iterative Reducing and Clustering using Hierarchies." (2008).

[11] Halsey, Mike. "Microsoft Sysinternals." Windows 10 Troubleshooting. Apress, 2016. 393-408.

[12] Chen, Lei. "Curse of Dimensionality." Encyclopedia of Database Systems. Springer US, 2009. 545-546.

[13] Schölkopf, Bernhard, and Alexander J. Smola. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002.

[14] Wang, Aiping, et al. "An incremental extremely random forest classifier for online learning and tracking." Image Processing (ICIP), 2009 16th IEEE International Conference on. IEEE, 2009.

[15] Ng, Andrew Y., and Michael I. Jordan. "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes." Advances in neural information processing systems. 2002.