Machine Learning Study to Optimize Anomaly Detection Speed and Accuracy for CyberSecurity Network Monitoring

Final Term Project for

CSC8515 – Machine Learning

Fall 2018

Hongwei Will Li

CyberSecurity Graduate Program,

Department of Electrical and Computer Engineering,

Villanova University

Table of Contents

[1. Introduction 3](#_Toc531480330)

[2. Data Source 3](#_Toc531480331)

[3. Methodology 5](#_Toc531480332)

[3.1. Initial Data Study and Feature Engineering 5](#_Toc531480333)

[3.2. Feature Reduction Study 6](#_Toc531480334)

[3.3. Machine Learning Algorithm Comparison 7](#_Toc531480335)

[4. Results 8](#_Toc531480336)

[4.1 Early Data Set Evaluation Results 8](#_Toc531480337)

[4.2 Feature Reduction Study Results 9](#_Toc531480338)

[4.3 Machine Learning Algorithm Comparison Results 11](#_Toc531480339)

[4.4 Final Machine Learning Report Using Test Data 11](#_Toc531480340)

[5. Conclusion and Future Directions 12](#_Toc531480341)

# 1. Introduction

Ever since the Morris worm of November 2, 1988, (Stoll, Clifford (1989). "Epilogue". The Cuckoo's Egg. Doubleday. ISBN 978-0-307-81942-0) cyber security professionals have focused on developing methods to sift through the mountains of system logs, and trying to find that needle in the haystack in order to alert on malicious traffic and start taking appropriate incident response actions.

In the pursuit of improving cyber security monitoring, machine learning methodologies had long been applied to the large quantities of data available. The most well known of the cyber security public data set is the KDD Cup 1999 data. (<http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>)

However, as the adoption of internet has increased exponentially since that time with the rise and burst of the dotcom bubbles, the cyber attack tactics, techniques, and procedures (TTP) have also increased in magnitude and sophistication to the point where nation state actors and advanced persistent threats are part of the everyday news-lingo.

One such example in cyber attack sophistication and the resulting impacting in the physical realm can be seen from the CrashOverride Malware and its severe impact on the Ukrainian power grid between 2016 and 2017. Details see analysis from Dragos: <https://dragos.com/blog/crashoverride/CrashOverride-01.pdf>. This report not only dissected the details of the specific impact on a Ukraine transmission substation which resulted in electric grid operations impact. It also concluded that the documented cyber attack appears to be just a proof-of-concept activity for a group of sophisticated cyber criminals. The future applications of this proof-of-concept malware could be much farther reaching and far more damaging to physical critical infrastructure.

# 2. Data Source

As a result, the author attempts to apply machine learning techniques to study more recent public data sets on cyber security domain, which provides more modern normal data points, as well as more current attack traffic. One such public data set is published by Moustafa and Slay. (Moustafa, Nour, and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)."Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015; Moustafa, Nour, and Jill Slay. "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 (2016): 1-14. ) and available at <https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/>

These data sets were generated using normal activities and synthetic attack traffic from an IXIA PerfertStorm tool network simulation tool. The resulting network traffic was then analyzed by Argus tool and Bro-IDS tool to provide the 49 features that can be resonably made available to a modern day cyber analyst.

|  |  |  |
| --- | --- | --- |
| Attack Category Counts |  |  |
|  | Normal | 37000 |
|  | Generic | 18871 |
|  | Exploits | 11132 |
|  | Fuzzers | 6062 |
|  | DoS | 4089 |
|  | Reconnaissance | 3496 |
|  | Analysis | 677 |
|  | Backdoor | 583 |
|  | Shellcode | 378 |
|  | Worms | 44 |
| Total # of Data Points: |  | 82332 |

Table 1 describes the sample size and label distribution of the training data set.

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| dur | Float | Record total duration |
| proto | nominal | Transaction protocol |
| service | nominal | http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service |
| state | nominal | Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, and (-) (if not used state) |
| Spkts | integer | Source to destination packet count |
| Dpkts | integer | Destination to source packet count |
| sbytes | Integer | Source to destination transaction bytes |
| dbytes | Integer | Destination to source transaction bytes |
| rate | Float |  |
| sttl | Integer | Source to destination time to live value |
| dttl | Integer | Destination to source time to live value |
| Sload | Float | Source bits per second |
| Dload | Float | Destination bits per second |
| sloss | Integer | Source packets retransmitted or dropped |
| dloss | Integer | Destination packets retransmitted or dropped |
| Sintpkt | Float | Source interpacket arrival time (mSec) |
| Dintpkt | Float | Destination interpacket arrival time (mSec) |
| Sjit | Float | Source jitter (mSec) |
| Djit | Float | Destination jitter (mSec) |
| swin | integer | Source TCP window advertisement value |
| stcpb | integer | Source TCP base sequence number |
| dtcpb | integer | Destination TCP base sequence number |
| dwin | integer | Destination TCP window advertisement value |
| tcprtt | Float | TCP connection setup round-trip time, the sum of synack and ackdat. |
| synack | Float | TCP connection setup time, the time between the SYN and the SYN\_ACK packets. |
| ackdat | Float | TCP connection setup time, the time between the SYN\_ACK and the ACK packets. |
| smeansz | integer | Mean of the ?ow packet size transmitted by the src |
| dmeansz | integer | Mean of the ?ow packet size transmitted by the dst |
| trans\_depth | integer | Represents the pipelined depth into the connection of http request/response transaction |
| res\_bdy\_len | integer | Actual uncompressed content size of the data transferred from the server's http service. |
| ct\_srv\_src | integer | No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26). |
| ct\_state\_ttl | Integer | No. for each state (6) according to specific range of values for source/destination time to live (10) (11). |
| ct\_dst\_ltm | integer | No. of connections of the same destination address (3) in 100 connections according to the last time (26). |
| ct\_src\_dport\_ltm | integer | No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26). |
| ct\_dst\_sport\_ltm | integer | No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26). |
| ct\_dst\_src\_ltm | integer | No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26). |
| is\_ftp\_login | Binary | If the ftp session is accessed by user and password then 1 else 0. |
| ct\_ftp\_cmd | integer | No of flows that has a command in ftp session. |
| ct\_flw\_http\_mthd | Integer | No. of flows that has methods such as Get and Post in http service. |
| ct\_src\_ ltm | integer | No. of connections of the same source address (1) in 100 connections according to the last time (26). |
| ct\_srv\_dst | integer | No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26). |
| is\_sm\_ips\_ports | Binary | If source (1) and destination (3)IP addresses equal and port numbers (2)(4) equal then, this variable takes value 1 else 0 |
| attack\_cat | nominal | The name of each attack category. In this data set , nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellcode and Worms |
| Label | binary | 0 for normal and 1 for attack records |

Table 2 describe the features of the data set.

# 3. Methodology

## 3.1. Initial Data Study and Feature Engineering

After ingesting the data set, the author first attempted to study the data, looking for missing data points, and performing some early visualization and tabulation from the data set.

The next part of this step involves of feature engineering work, where discrete text data was transformed into numeric data, and early experimentations on normalization of data were performed. As later stages of the study would demonstrate, certain ML algorithms would produce better results with normalized data, but other algorithms would produce better results with the original data format.

## 3.2. Feature Reduction Study

The second step involves feature reduction study. Univariate Feature Selection, Recursive Feature Elimination (RFE), and Principal Component Analysis (PCA) algorithms were used to study the UNSW-SB15 data set with the goal of trying to identify the most dominant features.

The feature reduction process with Univariate and RFE techniques are well documented in Scikit Learn documentation on feature selection methods.

* The SelectKBest method was used in this study with mutual\_info\_classif as the test function. This feature test function work well for discrete target variables, and it essentially tries to identify the most independent variables out of a given list.
* The RFE methods essentially assigns tries to assess the importance of each feature, and the least important feature is then pruned form the current feature set. Recursive steps are used to then work on the pruned set, until the desired number of features is reached. The feature importance test function used in this study is LogisticRegression, another way to measure the relationships between variables.

Due to the lack of readily available library function on feature selection using PCA based methods, the author developed a library from the stock Scikit Learn PCA functions.

* The ranking of the original features was achieved by building composite weights from each of the principle component vector and their respective weight.
* The original variables can be noted as vi, i from 1 to N;
* For each Principle Component j, j from 1 to K;
  + The eigenvector for PCj = (w1, w2, … wN)
  + The weight of PCj or eigenvalue = WPCj
* The composite weight for each of the original variable Wvi can be calculated as:
  + Wvi = sum(wi of PCj \* WPCj) for j from 1 to K

Based on insights gleaned from the previous section of the study, the author realized each attack category has some unqiue feature distributions. So, using the previous mentioend feature reduction techniques, the author first extracts the data points with only two labels, a single attack type, plus the Normal traffic. Then the dominant features will be identified from that “binary label data set”. The final collective feature set is computed and reported for each feature reduction technique when all the attack category labels have been studied in this fashion. (One example screenshot is shown in Figure 1)

Random Forest algorithm will then be used to compare the results of these three feature reduction techniques to the baseline full-feature data set. This will experimentally select the best feature reduction technique and the future feature sets to be used for the next stage of machine learning study. The practical value of this work will be appreciated when the improved speed of detection is needed in the real-world anomaly detection with the resulting much smaller data size.

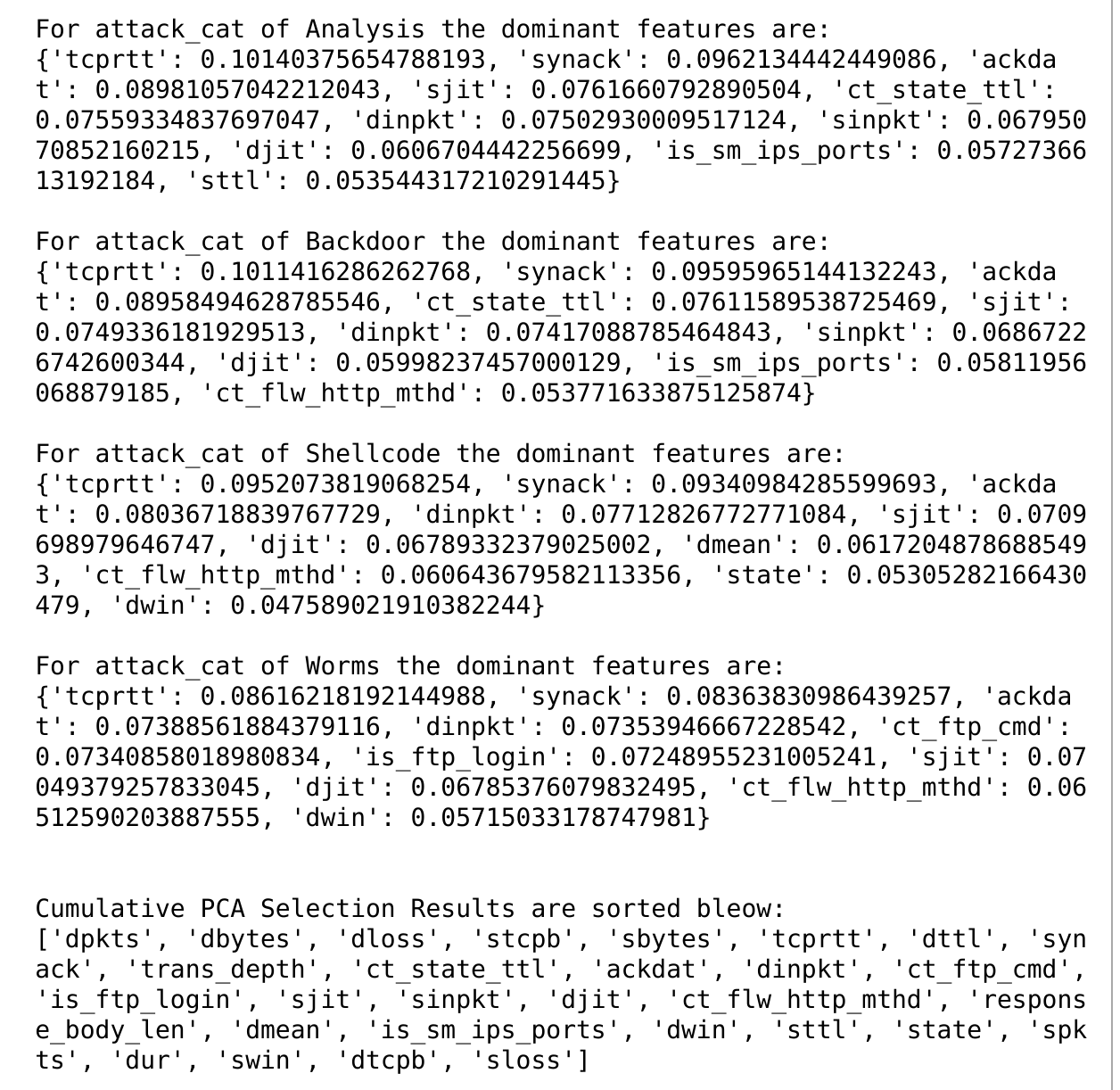


Figure 1. Sample screenshot of the PCA based feature selection study.

## 3.3. Machine Learning Algorithm Comparison

The final stage of this work involves comparing the efficacy of various supervised and unsupervised machine learning algorithms. Using the reduced feature set, the author will compare the efficacy of Random Forest algorithm, K-Nearest Neighbor, Gaussian Mixture Models, K-Means algorithm, Isolation Forest algorithm, and a artificial neural net algorithm such as Multi-Layer Perceptron (MLP). To speed up the analysis of multiple algorithms, the efficacy comparison was conducted with just the two-class label: Normal vs Attack or Anomaly traffic.

Precision, Recall, and F1 Scores will be used to compare the performance of the various algorithms.

* Accuracy = (TP + TN) / (TP + TN + FP + FN)
* Precision = TP / (TP + FP)
* Recall = TP / (TP + FN)
* F1 Score = 2 \* (Precision \* Recall) / (Precision + Recall)

# 4. Results

## 4.1 Early Data Set Evaluation Results

Table 3 and Figure 2 demonstrated some patterns and intricacies of the data set. Using the “service” feature as an example, different attack type demonstrated different distribution patterns of network services employed. While there were a lot of attach traffic using http packets, Generic-attack employed a lot of DNS packets, and Exploit-attack had a fair amount of smtp (or Internet Mail) traffic.

This analysis resulted the author to adopt attack type specific feature reduction technique as mentioned in the Methodology section to ensure the final study can account for unique behavior of each attack type.

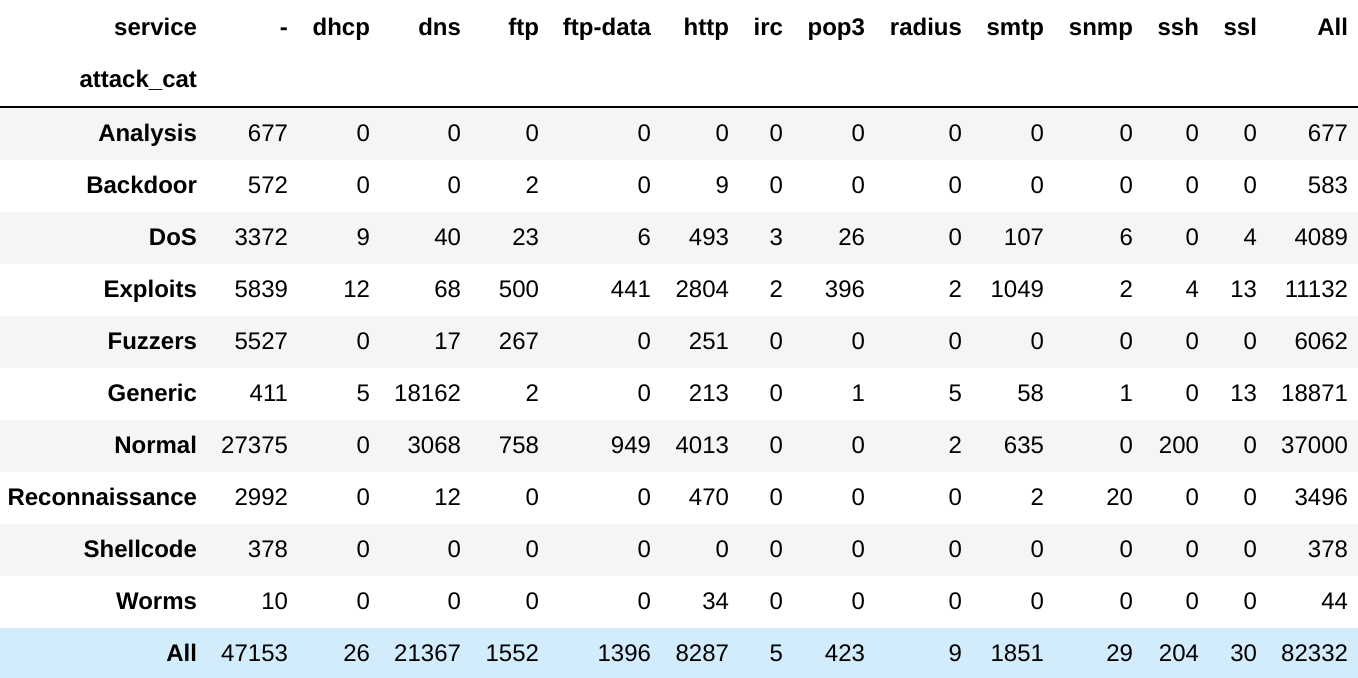


Table 3 describes the further service distribution of the training data set.

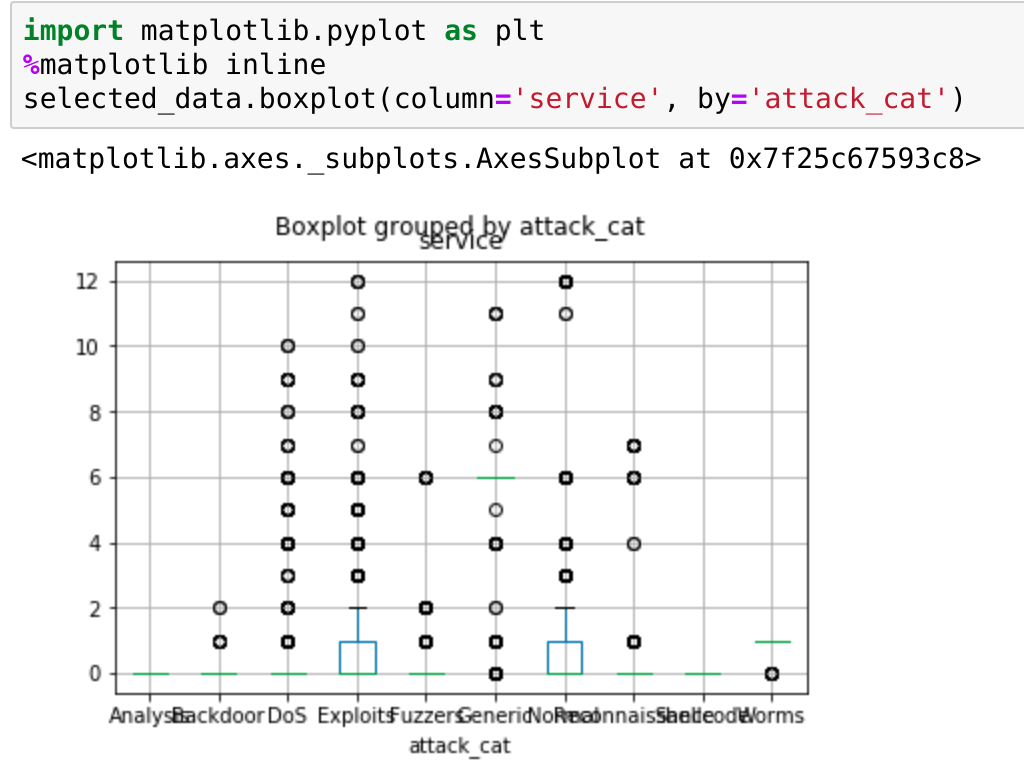


Figure 2 shows a sample plot of the training data set.

## 4.2 Feature Reduction Study Results

The results of the feature reduction technique study were shown in Figure 3, and Table 4. Recursive Feature Elimination technique resulted in a 43% feature reduction and improved the resulting accuracy. This is the benefit of reducing noise and overfitting by eliminating unimportant features.

A surprising result is that when the author tried to formulate an ultra-short list of only 9 features that are common amongst all 3 feature reduction techniques, the resulting accuracy was not that worse compared to the baseline. This means the 9 features actually encoded 96% of the information from the original 42 features!

Figure 3 shows the results of feature reduction study.

|  |  |  |
| --- | --- | --- |
|  | Total Number of Features | Sorted Dominant Feature Set  (Most to Least, except the ultra short list) |
| Baseline | 42 |  |
| Univariate Feature Selection | 22 | ['service', 'ct\_src\_dport\_ltm', 'synack', 'tcprtt', 'sinpkt', 'dinpkt', 'ct\_srv\_dst', 'dpkts', 'proto', 'ct\_dst\_sport\_ltm', 'dload', 'ct\_state\_ttl', 'state', 'sbytes', 'dttl', 'sload', 'smean', 'dmean', 'dur', 'dbytes', 'rate', 'sttl'] |
| Recursive Feature Elimination | 24 | ['state', 'dttl', 'dload', 'swin', 'dwin', 'synack', 'ct\_state\_ttl', 'ct\_dst\_sport\_ltm', 'ct\_dst\_src\_ltm', 'ct\_srv\_dst', 'proto', 'dbytes', 'dloss', 'ct\_src\_dport\_ltm', 'spkts', 'sbytes', 'sttl', 'ct\_srv\_src', 'sjit', 'dmean', 'sloss', 'smean', 'trans\_depth', 'dur'] |
| PCA Based Feature Selection | 29 | ['dpkts', 'dbytes', 'dloss', 'stcpb', 'sbytes', 'tcprtt', 'dttl', 'synack', 'trans\_depth', 'ct\_state\_ttl', 'ackdat', 'dinpkt', 'ct\_ftp\_cmd', 'is\_ftp\_login', 'sjit', 'sinpkt', 'djit', 'ct\_flw\_http\_mthd', 'response\_body\_len', 'dmean', 'is\_sm\_ips\_ports', 'dwin', 'sttl', 'state', 'spkts', 'dur', 'swin', 'dtcpb', 'sloss'] |
| Ultra Short List of Features Common Amongst All 3 Methods | 9 | ['ct\_state\_ttl', 'dbytes', 'dmean', 'dttl', 'dur', 'sbytes', 'state', 'sttl', 'synack'] |

Table 4 shows the resulting features from the feature selection study

## 4.3 Machine Learning Algorithm Comparison Results

The results of the machine learning algorithm efficacy comparison were shown in Figure 4. Random Forest algorithm produced the best result and had not shown any degradation in performance with the reduced feature set when compared to the baseline full feature set data. Additional benefit of the random forest algorithm is that even though the training process was very fast, the predicting process can be very light weight and even faster with modern multi-core multi-threat computing platforms, so it is very suitable for high-speed anomaly detection required in real-life situations.

Worth mentioning is that normalization of the data matters to certain algorithm, such as GMM and MLP. This study compiled results from both original data and normalized data, and compared results that with optimal performance.

Figure 4. Results of the ML algorithm efficacy study using two class labels.

## 4.4 Final Machine Learning Report Using Test Data

Similar to the popular machine learning competitions, the result (Figure 5) from test data set from the UNSW\_NB15 is reserved at the end to show case the result of this study. This result can be used to compare with studies from other researchers.

From the test data report, reducing the number of features by 43%, from 42 features to 24 features, resulted in no decrease in predicting performance.

Further, by selecting a set of 9 features, the prediction maintained comparable overall performance to the baseline 42 features.

Figure 5. Final machine learning report of the UNSW\_NB15 test data

# 5. Conclusion and Future Directions

In this study, systematic machine learning methodology was applied to study network monitoring data. Random Forest algorithm was proven to have the best efficacy amongst the selected ML algorithms.

Further, this study showed that out of the 40+ features in the original data set, most of the information is encoded in a handful of features. Again, using systematic techniques, the author showed the nine (9) features listed in Table 5 can produce as good a prediction result as all of the 42 features combined.

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| state | nominal | Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN, and (-) (if not used state) |
| sbytes | Integer | Source to destination transaction bytes |
| dbytes | Integer | Destination to source transaction bytes |
| sttl | Integer | Source to destination time to live value |
| dttl | Integer | Destination to source time to live value |
| synack | Float | TCP connection setup time, the time between the SYN and the SYN\_ACK packets. |
| ct\_state\_ttl | Integer | No. for each state (6) according to specific range of values for source/destination time to live (10) (11). |
| dmean | integer | Mean of the flow packet size transmitted by the destination |
| dur | Float | Record total duration |

Table 5. The ultra short 9 features learned from this study

This drastic feature reduction, coupled with the high speed Random Forest algorithm, can be much appreciated in practical applications when time and computing resources are two important constraints in the fight against cyber adversaries.

This study also showcased the importance of data and feature engineering, especially when it comes to normalization for certain algorithms.

* One episode occurred early in the study, where the TCP sequence number field showed up as the only important features in PCA analysis. This is because the sequence numbers are naturally 32-bit numbers and most frequently randomized to avoid being easily predicted. As a result, it ended up dominating the rest of the features. Only after normalization, did the author start to obtain meaningful insight.

And finally, for future work, the author is also interested in further transformation of the features based on the specific domain knowledge the author has accumulated over the years. Some examples of such transformation that can aid in cyber attack detection could be time of day, repetitive patterns of certain transmissions, of other non-human generated elements of certain data fields.