Name: Tithra Chap
2020

End-to-end project delivery on cyber-security data analytics

1: Data Exploration (Understanding the data)

- Identify the attribute names (columns)

['duration', 'src_bytes', 'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in', 'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells', 'num_access_files', 'is_host_login', 'is_guest_login', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate', 'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'protocol_type_tcp', 'protocol_type_udp', 'service_X11', 'service_Z39_50', 'service_aol', 'service_auth', 'service_bgp', 'service_courier', 'service_csnet_ns', 'service_ctf', 'service_daytime', 'service_discard', 'service_domain', 'service_domain_u', 'service_echo', 'service_eco_i', 'service_ecr_i', 'service_efs', 'service_exec', 'service_finger', 'service_ftp', 'service_ftp_data', 'service_gopher', 'service_harvest', 'service_hostnames', 'service_http', 'service_http_2784', 'service_http_443', 'service_http_8001', 'service_imap4', 'service_iso_tsap', 'service_klogin', 'service_kshell', 'service_ldap', 'service_link', 'service_login', 'service_mtp', 'service_name', 'service_netbios_dgm', 'service_netbios_ns', 'service_netbios_ssn', 'service_netstat', 'service_nnsp', 'service_nntp', 'service_ntp_u', 'service_other', 'service_pm_dump', 'service_pop_2', 'service_pop_3', service_printer', 'service_private', 'service_red_i', 'service_remote_job', 'service_rje', 'service_shell', 'service_smtp', 'service_sql_net', 'service_ssh', 'service_sunrpc', 'service_supdup', 'service_systat', 'service_telnet', 'service_tftp_u', 'service_tim_i', 'service_time', 'service_urh_i', 'service_urp_i', 'service_uucp', 'service_uucp_path', 'service_vmnet', 'service_whois', 'flag_REJ', 'flag_RSTO', 'flag_RSTOS0', 'flag_RSTR', 'flag_S0', 'flag_S1', 'flag_S2', 'flag_S3', 'flag_SF', 'flag_SH']

- Check the length of training and testing datasets:

Length of training set: 125973
Length of testing set: 22544

- Datasets of the five classes of training set:

Benign : 67343
Dos : 45927
Probe : 11656
r2l : 995
u2r : 52

2: Predictive Modeling (Prediction of the classes)

Table 1: Decision Tree Classifier

Attack Class	Accuracy	Precision	Recall	F1-Score	FP Rate	FN Rate
Benign	67.56%	68.90%	97.20%	80.64%	31.10%	2.80%
DOS	76.09%	96.39%	78.33%	86.42%	3.61%	21.67%
Probe	64.55%	78.83%	78.09%	78.46%	21.17%	21.91%
R2L	8.17%	95.91%	8.20%	15.10%	4.09%	91.80%
U2R	4.78%	52.63%	5.00%	9.13%	47.37%	95.00%

Table 2: MLP Classifier

Attack Class	Accuracy	Precision	Recall	F1-Score	FP Rate	FN Rate
Benign	62.71%	65.63%	92.87%	76.91%	34.37%	6.61%
DOS	73.65%	90.74%	79.07%	84.51%	8.52%	20.93%
Probe	66.74%	86.90%	74.21%	80.05%	13.10%	25.79%
R2L	2.75%	91.03%	2.76%	5.35%	8.97%	97.24%
U2R	0.00%	0.00%	0.00%	0.00%	00%	100.00%

NOTE: MLP uses random mechanism, so the results might be slightly different each time the code is run.

Table 3: RandomForest Classifier

Attack Class	Accuracy	Precision	Recall	F1-Score	FP Rate	FN Rate
Benign	65.37%	66.61%	97.25%	79.06%	33.39%	2.75%
DOS	76.53%	95.83%	79.16%	86.70%	4.17%	20.84%
Probe	60.69%	85.38%	67.73%	75.54%	14.62%	32.27%

R2L	5.17%	99.25%	5.17%	9.82%	0.75%	94.83%
U2R	0.50%	100.00%	0.50%	1.00%	0.00%	99.50%

Table 4: Support Vector Classifier						
Attack Class	Accuracy	Precision	Recall	F1-Score	FP Rate	FN Rate
Benign	63.49%	66.63%	93.08%	77.66%	33.37%	6.92%
DOS	78.08%	94.52%	81.78%	87.69%	5.48%	18.22%
Probe	57.92%	80.18%	67.60%	73.35%	19.82%	32.40%
R2L	11.37%	96.39%	11.42%	20.42%	3.61%	88.58%
U2R	7.21%	65.22%	7.50%	13.45%	34.78%	92.50%

Table 5: Naïve Bayes Classifier						
Attack Class	Accuracy	Precision	Recall	F1-Score	FP Rate	FN Rate
Benign	68.78%	75.50%	88.55%	81.50%	24.50%	11.45%
DOS	65.29%	90.56%	70.06%	79.00%	9.44%	29.94%
Probe	65.41%	67.72%	95.05%	79.09%	32.28%	4.95%
R2L	41.78%	76.58%	47.90%	58.94%	23.42%	52.10%
U2R	9.30%	15.74%	18.50%	17.01%	84.26%	81.50%

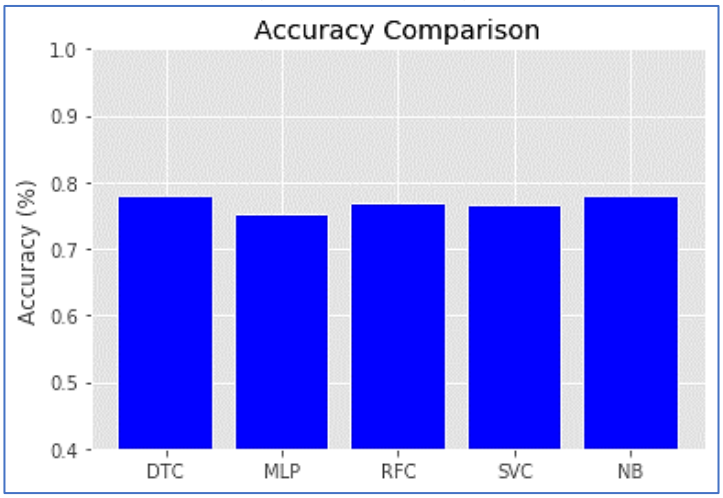
Table 6. Summary	Weighted Average				
Algorithm name	Accuracy	Precision	Recall	F1-Score	Process Time
Decision Tree Classifier	77.77%	82.22%	77.77%	74.25%	2.17s
MLP Classifier	75.08%	78.66%	74.51%	70.33%	145.57s
RandomForest Classifier	76.58%	82.55%	76.58%	72.67%	2.11s
Support Vector Classifier	76.43%	80.62%	75.84%	72.84%	16.99s
Naïve Bayes Classifier	77.72%	97.35%	77.72%	77.25%	1.19s

3: Visualizing Data

- Plot accuracy comparison between models

  Figure 1: Accuracy Comparison

  (Note: since the algorithms have very close performances, the starting point of y-axis was adjusted to 0.4 so that we can see clearly the differences.)



- Plot confusion matrix for each scenario

The following plots are normalized to avoid lengthy/scientific notations which are difficult to read.

Figure 2: Confusion matrix of DTC model

Figure 5: Confusion matrix of SVC model

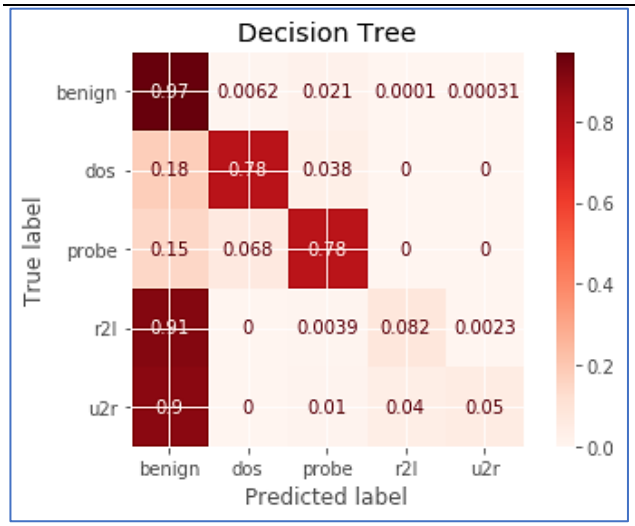


Figure 3: Confusion matrix of MLP mode

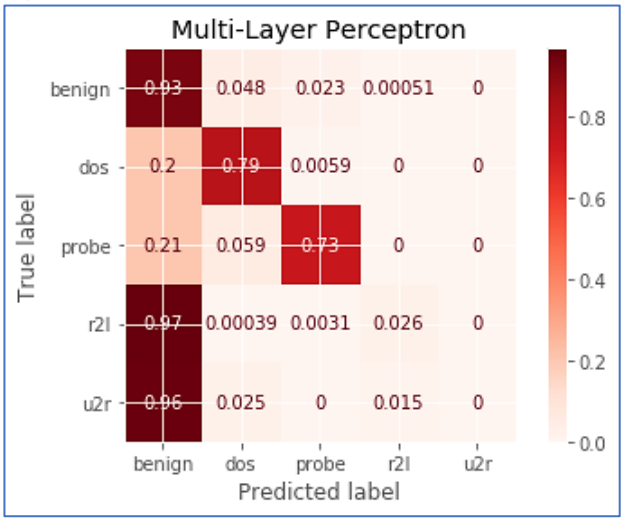
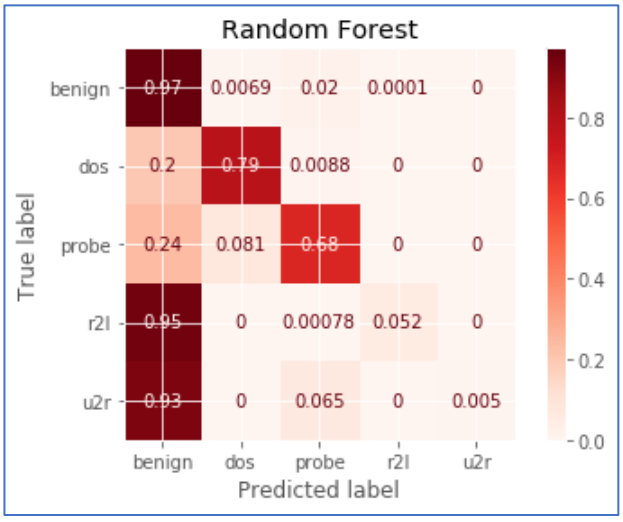


Figure 4: Confusion matrix of RFC model



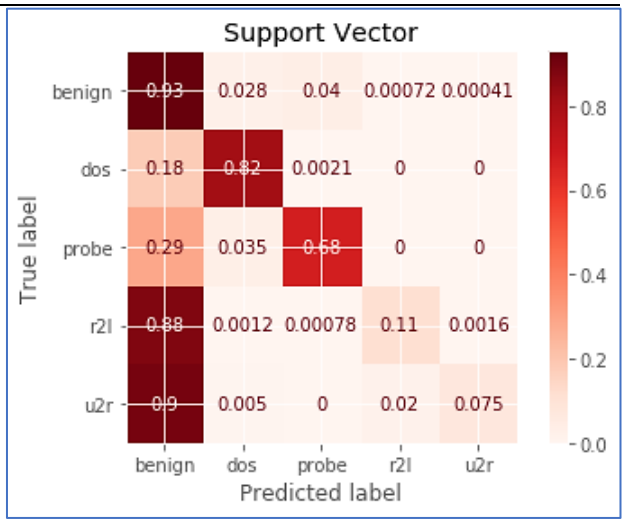
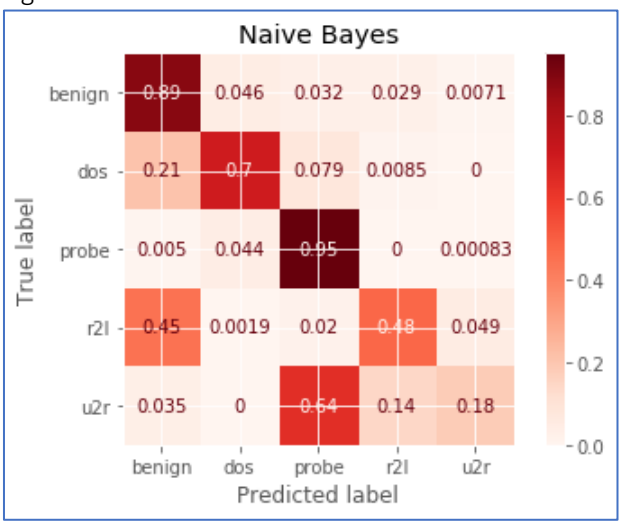


Figure 6: Confusion matrix of NB model



4: Performance evaluation report

This report discusses the key mechanism, parameters, and the performance of 5 algorithms used in the above experiments. Let's have a look at the following detail mechanism and performance of each algorithm.

The first algorithm, Decision Tree (DT), contains nodes which functions as branches of tree and leaf nodes at the last part of each branch. All nodes hold decision rules (conditional logics learned from the training data) which is used to compare to the input data attribute (testing data). After comparing and making decision, the process is forwarded to the appropriate node (downstream node) whose condition is met. This forwarding process will continue until it reaches the last node called leaf where the prediction or classification is made. The major parameter of this algorithm is criterion. Criterion determines which approach should be used to rank the order of nodes of the tree. DT may generate a very low performance if node order is not well managed. Either 'gini' or 'entropy' can be used to analyze and rank the most important node on the top (root) and least important one at the bottom (leaf). Besides, it is so critical to determine the 'random_state' of DT. 'random_state' randomizes the data sample and decide on the division of nodes at each branch. In this experiment, 'entropy' criterion is utilized together with integer 15 'random_state'. From the table 1, it can be seen that DT produces a very high score at 97.20% of Recall for normal data and a great precision at 96.39% and 95.91% of DOS and R2L attacks respectively. Though it does not show a good judgement of R2L and U2R attacks (high numbers at FN rate) which leads to poor accuracy rates of 8.17% and 4.78% respectively. DT seems to better classify normal data, DOS and Porbe but poor at predicting R2L and U2R type of attacks.

The second algorithm is Multi-Layer Perceptron (MLP). This algorithm uses the idea of perceptron to make decision or classify data. The perceptron evaluates data attribute by giving weight (a linear function) to the input data combined with other bias gaps learned from trained data. In a multi-layer perceptron, a set of related weighted functions is formulated to make decision or classification. Each function represents a layer. Those layers include input, hidden, and output layers. MLP has some important parameters to consider when classifying input data. Parameter 'solver' determines the type of weighted function, 'activation' determines how to activate the hidden layer/function, 'hidden_layer_size' refers number of neurons in the hidden layer. In this experiment; 'solver' is set to 'sgd', while 'activation' and 'hidden_layer_sizes' is configured to use 'logistic' and integer 100 respectively. Table 2 explains the performance results of this experiment. As seen in the table, MLP is highly efficient at identify DOS attacks at 84.51% of F1-score with its second best F1-score at 80.05% of Probe attacks. It is able to locate up to 92.87% of normal data in the Recall rate, although the accuracy of that is only 62.71%. The downside of MLP is the low ability to predict the R2L and U2R attacks. Up to 97.24% of FN rate in predicting R2L can be seen in the table. Lastly, the worst case for MLP is the compete undetectable rate of U2R attacks which results in zero accuracy.

The third algorithm is Random Forest (RF). The concept of this algorithm is similar to DT, except that RF generates individual tree nodes which has low correlation with one another. Each tree node acts as one part of a community. Therefore, the whole forest would contain a diversity of populated trees whose decisions are relatively distinctive. In data classification, a few nodes might predict wrongly but there are still many more will predict differently and correctly. The key parameters of the algorithm include: 'n_estimator' determines number of trees in the forest, 'random_state' manage the random sampling and splitting nodes, and 'criterion' control the ranking mechanism of the forest trees. In this experiment, integer 10 is assigned to 'n_estimator' parameter. 'Random_state' is set to integer 45. 'Criterion' takes 'entropy', as this mechanism also shows promising performance in DT's experiment. Table 3 reveals the performance results of this experiment. It is viewable that RF works greatly at locating DOS attacks with the accuracy of 76.53% and F1-score up to 86.70%

rate. The second best performance is to predict the normal data which obtains the accuracy level of 65.37% and F1-score of 79.06%. It is also noticeable that RF produces more FP mistakes than FN on normal data, and oppositely produces more FN than FP on DOS attacks. Third best performance is the ability of RF to locate DOS attacks with about 60% accuracy. Simliar to other experience, the accuracies of R2L and R2R are very low at the rate 5.17% and 0.50% respectively.

Support Vector (SV) is the fourth algorithm in these experiments. SV mathematically generates a hyperplane/line which separates or classifies the dataset. This line is also called support vector. To make the hyperplane, a linear equation is computed by SV, taking into consideration the margin between different classes of data. In a non-linear dataset, SV produces curve hyperplanes to distinguish between each classification. SV utilizes three important parameters 'C', 'kernel' and 'gamma'. 'C' determines the strictness of the curve line to pinned point the classification. Too high 'C' will result overfitting and opposite result for too low 'C'. 'Kernel' refers the mechanism for identifying pattern of data. 'Gamma' defines the weight that makes up the decision boundary of hyperplane. In SV experiment, 'linear' kernel is selected, together with 'scale' weighting mechanism, and decimal 0.5 for 'C' parameter. Table 4 above discusses the outcomes of SV experiment. Results reveal the most noticeable and high accuracy level of the algorithm, up to 78.08%, to predict the DOS attacks. The 33.37% of FP rate of predicting normal data explains that SV has limited ability to handle normal data. This error rate drags down the prediction of normal data to the second best accuracy in the table, 63.49%. The third best prediction of SV is on Probe attacks which produces about 58% of accuracy. Lastly, SV does not cover R2L and U2R very well, making a great reduction of accuracy to 11.37% and 7.21% respectively. The main reason of these poor performance that SV is unable to classify the data as the R2L or U2R attacks in the FN scenario.

The last algorithm is Naïve Bayes (NB). NB is the simplest algorithm among others, yet can promise a great performance in many cases. This algorithm employs probability technique, by using Bayes' Theorem, to come up with decision of data classification. It uses the prior knowledge of the training data to predict by studying the level of probability. The key parameter of NB is 'var_smoothing'. This parameter is to decide the stability level of calculation which depends on the value of variances. Here, NB experiment is conducted based on decimal 0.05 value of variances in 'var_smoothing' parameter. Table 5 demonstrates the experiment results of NB. As seen, BN works best in locating normal data comparing with other types of data. At 88.55% of Recall and 75.50% of Precision, NB is able to produce 81.50% of F1-score which is the highest rate comparing to other types of data. Further, NB yields an approximately the same level of accuracy of DOS and Probe at about 65%. These equivalent results are caused by the opposite balance between Precision and Recall of DOS and Probe. In other word, DOS result shows greater score in Precision than Recall (90.56% and 70.06% respectively), while Probe does the opposite with 67.72% of Precision and 95.05% of Recall. NB is able to predict R2L data at an accuracy rate of 41.78%, making it the fourth best performance of classification comparing to other types of data. The poorest outcome of predicting rate of NB is U2R whose accuracy rate is only 9.30%. It is worth to reason this problem by looking at the high rate in FP and FN which explains tons error in predictions of U2R data.

Since we have done all the details with individual algorithm, now we are working to compare the performance of those algorithms under the same measurement setting of Accuracy, Precision, Recall, F1-score, and Timing. The results of all algorithms experimented are in Table 6 and Figure 1. As a noticeable part of table, Accuracy column contains the most useful information of this comparison. At 77.77% level of accuracy, DT can be said to be the top candidate of this data analysis. The second best candidate is NB whose accuracy reaches 77.72%, very close to DT. The third position is given to RF with the accuracy of 76.58%, followed by SV at 76.43%, and lastly MLP at 75.08%. We also notice that all of the five algorithms tend to perform poorly when it comes to

analyzing the R2L and U2R data. The possible reason for this phenomenon might come from the low training population R2L and U2R which have been used to train the algorithms. From step 4.3 above, we can see that the amount of training R2L and U2R is only 995 and 52 respectively. Therefore only 0.79% of R2L and 0.04% of U2R haven been used in training which causes poor learning experience of the algorithms to predict in the experiments.

On the other hand, we also realize that the three top algorithms are the well-known algorithms in term of their simplicity of mechanism. The evidences of processing time explain the fast and simple mechanism of DT, NB and RF in processing the data classification. In the same sense, we can further imply that the nature of dataset of this analysis works better with logical decision-based algorithms like DT and RF, and simple probability technique like NB. A more linear based algorithm like MLP is unable to reach an outstanding classification efficiency with this given dataset.

An additional evaluation of the experiments can also be discussed based on the plots produced by confusion matrix. Those plots are arranged from figure 2 to figure 6 which display visual matrix of each algorithm. The darker spots on the matrix plots explain the higher values received during the experiments and, in the opposite, the lighter spots refer to the lower values. Another important point to notice is the diagonal line of each plot starting from the top-left down to bottom-right. This line tells the correct predictions of each class in the dataset. All of the plots show similar pattern, however, there are some differences that we can pinned point to reveal some interesting facts about each algorithm.

As seen from those plots, all of the algorithms have great prediction rate of normal data at the starting point of the diagonal line and their accuracies tend to get lower as the line move downward, except NB algorithm. NB has greatest classification performance of Probe and R2L comparing to all other algorithms. In a similar sense, when all other algorithms produce poor results by wrongly judging many normal/benign data as R2L or U2R, NB algorithm is able to reduce this problem remarkably. In an opposite way, NB gets more mistakes by wrongly judging Probe data as U2R (FN rate of Probe) while other algorithms do not. SV and RF show their highest rates of wrong judgement of Probe data as benign data among all other algorithms.

Based on trails and errors during the experiments, we can conclude that DT and NB are the top efficient algorithms for classifying KDD dataset among the five algorithms above. This conclusion is drawn based mainly on the assumption of KDD data nature which tends works better with logical decision-oriented DT and RF, and conditional-probability oriented NB. Though, this assumption may not apply to other situation where a distinctive dataset will be used for classification analysis instead of KDD data. In addition to efficiency of classification, we also prioritize the simple and fast speed in processing time of DT, RF and NB. In other way, MLP and SV require heavy computing power for their complex mechanism which may be inapplicable for an instant responsive scenario of real-world data analytics.

The end