Detecting SSH and FTP Brute Force Attacks in Big Data

John Hancock and Taghi M. Khoshgoftaar and Joffrey L. Leevy
Florida Atlantic University
jhancoc4@fau.edu, khoshgof@fau.edu, jleevy2017@fau.edu

***Abstract*—We present a simple approach for detecting brute force attacks in the CSE-CIC-IDS2018 Big Data dataset. We show our approach is preferable to more complex approaches since it is simpler, and yields stronger classification performance. Our contribution is to show that it is possible to train and test simple Decision Tree models with two independent variables to classify CSE-CIC-IDS2018 data with better results than reported in previous research, where more complex Deep Learning models are employed. Moreover, we show that Decision Tree models trained on data with two independent variables perform similarly to Decision Tree models trained on a larger number independent variables. Our experiments reveal that simple models, with AUC and AUPRC scores greater than 0.99, are capable of detecting brute force attacks in CSE-CIC-IDS2018. To the best of our knowledge, these are the strongest performance metrics published for the machine learning task of detecting these types of attacks. Furthermore, the simplicity of our approach, combined with its strong performance, makes it an appealing technique.**

***Keywords*—Decision Tree, Cyber-security, CSE-CIC-IDS2018, Big Data, Intrusion Detection, Brute-Force Attack**

1. Introduction

Securing information systems connected to the Internet is of paramount importance since they are constantly under attack [1]. Furthermore, since aggressors are continually improving and changing their methods for attacking, machine learning (ML) is a practical means of protection. Because ML algorithms can learn to recognize patterns in network traffic that indicate attacks, they offer an automated means of detecting malicious traffic. Our contribution in this paper shows that a simple ML algorithm, Decision Tree [2], operating on a dataset with only two features is an effective tool for detecting certain types of attacks.

We use Decision Tree to identify attacks in the Communications Security Establishment[1] (CSE) and Canadian Institute of Cybersecurity[2] (CIC) Intrusion Detection System (IDS) dataset from 2018 [3]. Hereafter, we refer to this dataset as CSE-CIC-IDS2018 [4]. Due to the nature of the data it contains, CSE-CIC-IDS2018 qualifies as Big Data [5]. The primary metrics we use to report performance are Area Under the Receiver Operating Characteristic Curve (AUC) [6], and Area Under the Precision Recall Curve (AUPRC) [7].

The types of attacks we detect in CSE-CIC-IDS2018 are Secure Shell Protocol (SSH) [8] Brute Force, and File Transfer Protocol (FTP) [9] Brute Force attacks. SSH is a communications protocol for executing commands and transferring data between hosts on a network that uses encryption to protect data as it is in transit over the network. FTP is a protocol for transferring data in the form of files between hosts on a network. Unlike SSH, FTP does not require data to be encrypted. A brute force attack [10] involves the submission of many username and password combinations by an attacker, the goal being to access applications, data, and resources that are password-protected. Here, we are concerned specifically with attacks where bad actors are attempting to gain access to systems by logging into a system's SSH or FTP services.

The experiments we perform seek to answer two research questions. The first research question, **Q1**, is, "Can Decision Tree classify CSE-CIC-IDS2018 SSH-Brute Force, FTP-Brute Force, and Combined-Brute Force attacks data with strong results, thus obviating the need for more complex algorithms?" The second research question, **Q2**, is, "What is the smallest number of features in CSE-CIC-IDS2018 we can use to achieve consistently strong classification performance?"

Our mission in this paper is to portray how we go about finding answers to research questions **Q1** and **Q2**. We begin with the Related Work section to ascertain whether existing research already answers our questions. We find it does not. Therefore, in the Methodology section, we explain how we perform our experiments. Then, in the Results section, we report experimental results. In the Statistical Analysis section, we do further computations on the results to determine if differences in experimental outcomes are statistically significant. Finally in the Conclusions section, we discuss the implications of the results and analysis, and suggest some topics for future work.

2. Related Work

Our search for CSE-CIC-IDS2018 papers that focused primarily on brute force attack detection concluded on June 15, 2021. To the best of our knowledge, there is only one published study related to our research.

The related study [11], authored by Wanjau et al., proposes a Convolutional Neural Network (CNN) [4], [12].

model to detect SSH-Brute Force network attacks. The CNN, consisting of four hidden layers of 64 units per layer, was implemented with Keras and TensorFlow. Values for hyper-parameters such as the dropout, learning rate, optimizer, activation, and loss function were determined empirically. Preprocessing of the dataset (normal and SSH-Brute Force instances) involved feature selection, data transformation to images, normalization, and dimensionality reduction. For training and testing, the dataset was split in a ratio of 70 to 30, respectively, and a hold-out validation set was used during training iterations. To gauge model performance, experimentation was performed using the full feature set and an unspecified set of minimal features. The optimal classification performance for the CNN model was obtained with the full feature set. For this set, the final SSH-Brute Force detection scores for accuracy, precision, recall, and F-measure were 0.943, 0.925, 0.978, and 0.918, respectively. The

1. https://www.cse-cst.gc.ca/en
2. https://www.unb.ca/cic/

CNN model was subsequently evaluated against five other classifiers: Decision Tree [2], [13], Naive Bayes [14], [15], Logistic Regression [16], [17], KNN [18], and SVM [19]. Results indicate that the CNN model was the top performer overall.

In our study, it is important to note that both SSH-Brute Force and FTP-Brute Force instances have been used. However, in their study, Wanjau et al. only covered SSH-Brute Force instances. One shortcoming of their paper is the lack of information on the set of minimal features, particularly the quantity and name of the features constituting the minimal set. Another shortcoming is the lack of information on the sample size of normal and SSH-Brute Force instances used for training and testing. As a consequence of these two limitations, replication of Wanjau *et al*.'s results is not possible. The limitations also mean that a useful comparative study cannot be performed between their performance results and ours.

3. Methodology

We find the performance of Decision Tree in classifying several datasets of network traffic data to be quite satisfactory. Specifically, we use the Scikit-learn [20] Decision Tree implementation. We employ three datasets in our experiments. The datasets are related, and derived from a primary dataset. The primary dataset is CIC-CSE-IDS2018, which contains data labeled with multiple attack types. We do not use all of the attack types from CSE-CIC-IDS2018. We only use data related to SSH-Brute Force attacks, and FTP-Brute Force attacks. The way we derive the three datasets from the CSE-CIC-IDS2018 dataset is to filter by attack type. This yields one dataset consisting of normal and SSH-Brute Force attack network traffic data, and one consisting of normal and FTP-Brute Force attack network traffic data. Finally, we combine SSH and FTP-Brute Force attack data with normal network traffic data to obtain a combined dataset with labels in two classes, attack or normal.

For brevity, hereafter, we refer to these three datasets as the SSH, FTP, and Combined "attack types data", or when the context is clear, simply as "attack types". Also, we perform feature selection. We report results by features selected and attack type, and we refer to the datasets collectively by the features they contain, except for the so-called 6-Agree dataset. For details on how this dataset is constructed please see [13]. In order to better explain the 6-Agree dataset, we must first explain the ensemble feature selection technique (FST) that we use to create the 6-Agree dataset.

Preliminary feature selection experiments were conducted to discover important features. We do ensemble feature selection as described in [21]. In their study, Leevy *et al* do feature selection as follows: we employ three filter-based feature ranking techniques and 4 supervised learning-based feature ranking techniques to rank features of a dataset. We then truncate the 7 feature rankings, so each ranking has at most 20 attributes. Next, we select features in common to n, out of the 7 rankings, where n ranges from 4 to 7. When we apply the approach to the SSH, FTP, and Combined Brute Force datasets for the case where we require a feature to appear in 6 out of 7 rankings, we obtain datasets with 4 features. We refer to these datasets as the 6-Agree datasets. We list the features in the 6-Agree datasets in Table 3. We find that the Decision Tree classifier yields strong performance with the 6-Agree datasets in detecting SSH, FTP, and Combined Brute Force attacks. We report these results in Table 4. Since the models trained on 4-feature datasets yield such strong performance, we have a rationale for conducting experiments where we further reduce the number of features to 1 or 2.

Furthermore, we find that for the Combined and SSH data, 7 out of 7 rankers rank the Bwd_Packets_s feature among the top 20 most important features. Our goal is to intelligently select as few features as possible to obtain results similar to the results we report in Table 3. To do the selection, we recall the 7-Agree FST finds one feature that appears in 7 out of 7 rankings for the SSH and Combined datasets. That is the Bwd_Packets_s feature. Furthermore, there are 4 features other than Bwd_Packets_s that the 6-Agree FST discovers. These are listed in Table 3. We see for each of the three attack types, the 6-Agree ensemble FST selects min_seg_size_forward, Flow_IAT_Mean, Flow_Packets_s and Flow_IAT_Max. Therefore, we conduct experiments where we select two features from each dataset. We select Bwd_Packets_s as one feature, and one of the remaining 4 features as the other feature. Hence, we have a total of 4 feature selection techniques. Since we have the SSH, Combined, and FTP datasets, we conduct 12 experiments for all combinations of FSTs and datasets. We list definitions of features in Table 1. These feature definitions are from documentation on features the CICFlowMeter-V3[3] utility extracts from raw network traffic data to form CSE-CIC-IDS2018.

TABLE 1. DEFINITIONS OF FEATURES USED

Feature : Description
min_seg_size_forward : minimum segment size observed in the forward direction
Bwd_Packets_s : number of backward packets per second
Flow_IAT_Mean : mean time between two packets sent in the forward direction
Flow_Packets_s : flow packets rate; number of packets transferred per second
Flow_IAT_Max : maximum time between two flows

We disclose the counts of instances in each dataset, as well as the numbers of instances labeled as attack or normal in Table 2.

TABLE 2. COUNTS OF INSTANCE LABELS IN DATASETS

Dataset	Total	Normal	Attack
Combined	13,771,177	13,390,234	380,943
SSH	13,577,823	13,390,234	187,589
FTP	13,583,588	13,390,234	193,354

For all experiments, we use all available SSH, FTP, and Combined Brute Force attacks data in the process of performing 5-fold cross validation. We perform 10 iterations of 5-fold cross validation to obtain a total of 50 measurements of AUC and AUPRC for every experimental outcome. Furthermore, we run experiments as Python [22] programs, and rely on functions in the Scikit-learn library to facilitate stratified 5-fold cross validation, as well as computation of metrics we report. Finally, for statistical analysis we use the R [23] programming language. Hence, we use R for conducting analysis of variance (ANOVA) tests [24], and R's Agricolae [25] package for conducting Tukey's Honestly Significant Difference (HSD) tests [26]. This concludes our discussion on methodology. In the next section, we move on to discuss our results.

3. https://www.unb.ca/cic/research/applications/html#CICFlowMeter

4. Results

We work with several combinations of attack types and FSTs; therefore, we need to define some nomenclature to enable us to report results efficiently. As mentioned previously, the 6-Agree FSTs yield datasets with features listed in Table 3. Since the other FSTs we employ yield one or two features, we refer to these FSTs by the names of the features they contain. We separate feature names with a '/' character. However, in order for the tables and figures to remain legible, we abbreviate the names of these FSTs as follows: Bwd_Packets_s / min_seg_size_forward we abbreviate as BPS/MSSF, Bwd_Packets_s / Flow_IAT_Mean we abbreviate as BPS/FMean, Bwd_Packets_s / min_seg_size_forward we abbreviate as BPS/FMin, Bwd_Packets_s / Flow_Packets_s datasets we abbreviate as BPS/FPS, and Bwd_Packets_s / Flow_IAT_Max we abbreviate as BPS/FMax.

We present results in terms of AUC or AUPRC since these metrics are not sensitive to classifier threshold choice [7].

We report the results of the 6-Agree ensemble FST side-by-side to highlight the features in common for each dataset. We then report performance for each dataset and attack type.

TABLE 3. FEATURES COMMON TO 6 OUT OF 7 RANKINGS

FTP	SSH	Combined
min_seg_size_forward	min_seg_size_forward	min_seg_size_forward
Bwd_Packets_s	Bwd_Packets_s	Bwd_Packets_s
Flow_IAT_Mean	Flow_IAT_Mean	Flow_IAT_Mean
Flow_Packets_s	Flow_IAT_Max	Flow_Packets_s

The classification performance results for models built with these features are in Table 4. The near-perfect results indicate that we may be able to further reduce the number of features used to train models.

TABLE 4. DECISION TREE CLASSIFICATION RESULTS FOR THE 6-AGREE FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.99978	0.00002	0.99918	0.00004
FTP	0.99999	1.12828×10^{-06}	0.99991	0.00002
SSH	0.99824	0.00023	0.99295	0.01696

The first two-feature FST we report results for is the Bwd_Packets_s / Flow_IAT_Mean FST, in Table 5. For each of the Combined, FTP, and SSH attack types (indicated in the Attack_Type column), we observe performance is good, but not quite as strong as in the case of the 6-Agree FST, and AUPRC values suffer more than AUC values.

TABLE 5. DECISION TREE CLASSIFICATION RESULTS FOR THE BWD_PACKETS_S / FLOW_IAT_MEAN FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.94072	0.00082	0.84944	0.00104
FTP	0.93504	0.00087	0.87599	0.00136
SSH	0.94487	0.00180	0.81928	0.03658

We use the Bwd_Packets_s / min_seg_size_forward FST to make the second group of two-feature datasets. As before, we classify these datasets with Decision Tree. Table 6 contains results that are quite strong, comparable to performance we report in Table 4 for experiments involving the 6-Agree FST. Moreover, one may notice that the results for the FTP attack type are exactly the same for in Tables 4 and 6. In order to rule out a programming error, we rewrote the software for performing the experiments that produce the results for the FTP attack type in Tables 4 and 6, and ran experiments a second time. Results did not differ. The reason the results are the same for the FTP attack type in Tables 4 and 6 is that we seed random number generators for classifiers and samplers for stratified 5-fold cross validation in both experiments with the same sequence of seed numbers. We use the same sequence of seed numbers so that we can reproduce our results precisely. Such control enables us to discover how an algorithm is learning to classify data. Here, we can be certain the decision tree algorithm discards two of the four features, since we get exactly the same results in both cases. Put another way, using the same sequence of seed numbers shows that the Scikit-learn Decision Tree implementation discovers the same rules for classification since Bwd_Packets_s and min_seg_size_forward are strong predictors for FTP-Brute Force attacks.

TABLE 6. DECISION TREE CLASSIFICATION RESULTS FOR THE BWD_PACKETS_S / MIN_SEG_SIZE_FORWARD FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.99719	0.00013	0.99904	0.00005
FTP	0.99999	1.12828×10^{-06}	0.99991	0.00002
SSH	0.99507	0.00022	0.97542	0.05652

Since the two-feature models yield strong results across all attack types, we train and test additional models with two remaining pairs of features: Bwd_Packets_s / Flow_Packets_s, and Bwd_Packets_s / Flow_IAT_Max. The performance of these models is summarized in Tables 7 and 8.

In Table 7 we see models trained with the Bwd_Packets_s / Flow_Packets_s datasets yield performance that is not as strong as the other two-feature datasets. This aligns with the fact that the Flow_Packets_s feature is not in each of the 6-Agree datasets – that our ensemble feature selection technique did not yield Flow_Packets_s in every case.

TABLE 7. DECISION TREE CLASSIFICATION RESULTS FOR THE BWD_PACKETS_S / FLOW_PACKETS_S FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.83537	0.00073	0.81366	0.00116
FTP	0.93506	0.00087	0.88066	0.00132
SSH	0.77687	0.00118	0.72582	0.02301

Finally, we report results for the Bwd_Packets_s / Flow_IAT_Max datasets. In Table 8, we notice performance similar to results we report in Table 5; performance is strong but not as good as what we find for models trained with datasets from the 6-Agree or Bwd_Packets_s / min_seg_size_forward FST.

TABLE 8. DECISION TREE CLASSIFICATION RESULTS FOR THE BWD_PACKETS_S / FLOW_IAT_MAX FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.94861	0.00047	0.87329	0.00577
FTP	0.93504	0.00087	0.87723	0.00136
SSH	0.96502	0.00704	0.87603	0.03408

We do not see consistently strong performance when we reduce the number of features to one. Table 9 shows strong performance for detecting FTP-Brute Force attacks, and the one-feature dataset consisting of the min_seg_size_forward feature. However, for the SSH and Combined attack types, we see weaker performances. This is true especially in the case of the SSH data, where performance in terms of AUC is 0.5, which is practically as bad as possible. We do not report further results regarding one-feature datasets since performance is not consistently strong, as is the case with some two-feature datasets, where we see strong performance for the SSH, FTP, and combined data.

TABLE 9. CLASSIFICATION RESULTS FOR THE MIN_SEG_SIZE_FORWARD FST; MEAN AND STANDARD DEVIATIONS OF AUC AND AUPRC, (10 ITERATIONS OF 5-FOLD CROSS-VALIDATION)

Attack_Type	Mean AUC	SD AUC	Mean AUPRC	SD AUPRC
Combined	0.75236	0.00069	0.77383	0.00110
FTP	0.99854	0.00001	0.91590	0.00069
SSH	0.50000	less than 1×10^{-6}	0.64024	0.00031

Since the Bwd_Packets_s / min_seg_size_forward datasets have two features, we speculate that the Decision Trees we fit to these datasets would be simple and we could report easy-to-read decision rules from the trained models. As an example, we report one Decision Tree in Figure 1. Due to space limitations, more information cannot be provided on the operation of the Decision Tree algorithm.

This concludes the presentation of results. In the next section, we move on to investigate whether the differences in reported performance are meaningful, or only due to random chance.

Figure 1. Decision tree rules for classifying FTP Traffic into Brute Force Attack or Normal classes with the Bwd_Packets_s / min_seg_size_forward FST

```
|--- min_seg_size_forward <= 38.00
|   |--- class: Normal
|--- min_seg_size_forward >  38.00
|   |--- Bwd_Packets_s <= 9398.50
|   |   |--- class: Normal
|   |--- Bwd_Packets_s >  9398.50
|   |   |--- min_seg_size_forward <= 42.00
|   |   |   |--- class: Attack
|   |   |--- min_seg_size_forward >  42.00
|   |   |   |--- class: Normal
```

5. Statistical Analysis

The data used for this analysis is the same results data that has been summarized in Tables 4, 5, 6, 7, and 8. The confidence level we set for ANOVA and HSD tests is 99%. The first analysis we do is a two-factor ANOVA test to determine the impact of attack type (Attack_Type), and FST on performance in terms of AUC.

TABLE 10. ANOVA FOR ATTACK_TYPE AND FST AS FACTORS OF PERFORMANCE IN TERMS OF AUC

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Attack_Type	2	0.08	0.04	51.32	0.0000
FST	4	2.24	0.56	707.85	0.0000
Residuals	743	0.59	0.00		

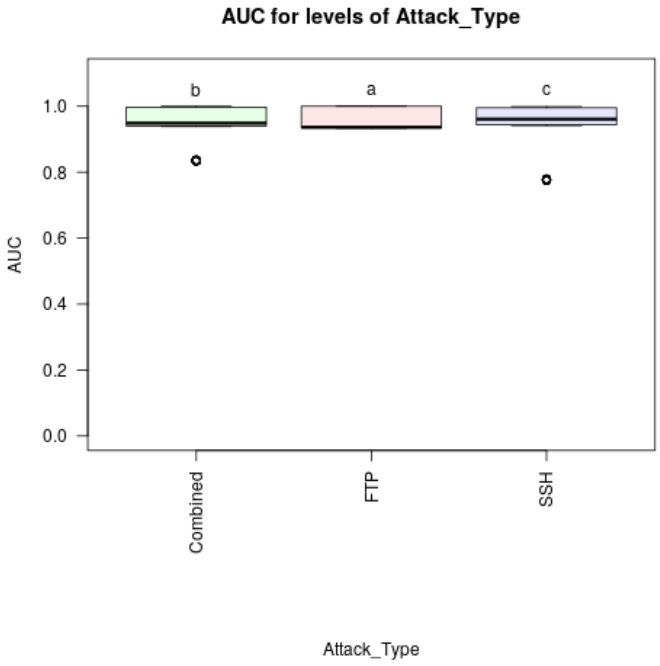
The p-values, or Pr(>F) values, we report in Table 10 are practically 0. This indicates there are statistically significant differences within each factor. Therefore, we conduct HSD tests to rank the Attack_Type and FST factors.

TABLE 11. HSD TEST GROUPINGS AFTER ANOVA OF AUC FOR THE ATTACK_TYPE FACTOR

Group a consists of: FTP
Group b consists of: Combined
Group c consists of: SSH

We see that performance of models for classifying the FTP-Brute Force attack type data is the strongest. The small variance in performance causes the differences to be statistically significant. We see in the box plots in Figure 2 that results are strong for all three attack types and close to the maximum possible value for AUC.

Figure 2. Boxplots of AUC for levels of Attack_Type; HSD group printed above box; color corresponds to HSD group



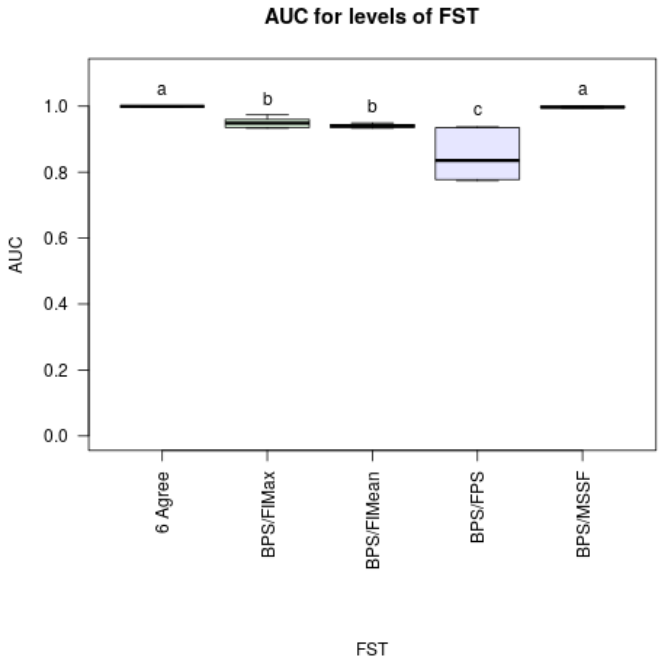
The second HSD test we undertake is to compare the performance of feature selection techniques. Since the p-value from the ANOVA test in Table 10 is practically 0 for the FST factor, we conclude FST has a significant impact on the outcome of experiments. Therefore, an HSD test is appropriate. We report the outcome of the test in Table 12.

TABLE 12. HSD TEST GROUPINGS AFTER ANOVA OF AUC FOR THE FST FACTOR

Group a consists of: 6 Agree, BPS/MSSF
Group b consists of: BPS/FIMax, BPS/FIMean
Group c consists of: BPS/FPS

Finally, to give a sense of how the FSTs relate, we provide boxplots of performance in terms of AUC by FST. Figure 3 shows again that the two-feature Bwd_Packets_s / min_seg_size_forward FST is in the same HSD category the as the 6-Agree FST.

Figure 3. Boxplots of AUC for levels of FST; HSD group printed above box; color corresponds to HSD group



Now we move on to do a similar analysis for AUPRC. Once again, the outcome of the ANOVA test reported in Table 13 has p-values that are practically 0. Therefore, we conduct HSD tests to rank factors by their performance in terms of AUPRC as well.

TABLE 13. ANOVA FOR ATTACK_TYPE AND FST AS FACTORS OF PERFORMANCE IN TERMS OF AUPRC

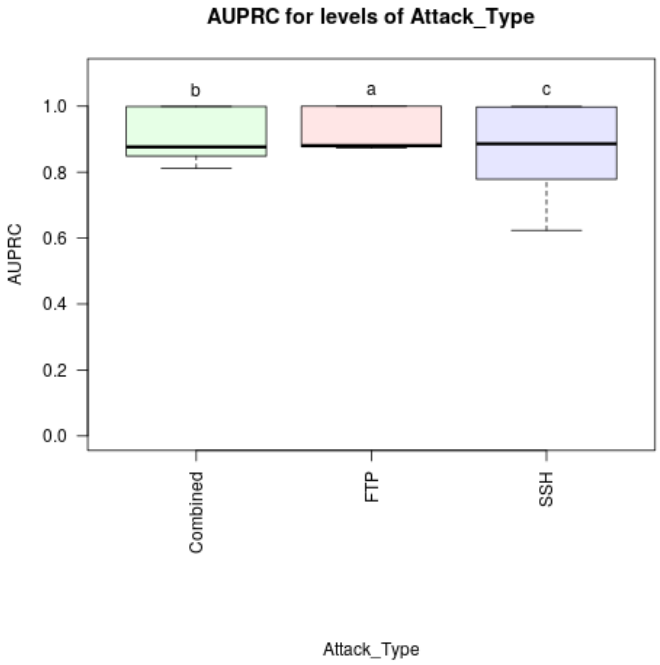
Attack_Type	2	0.30	0.15	153.38	0.0000
FST	4	4.46	1.12	1134.47	0.0000
Residuals	743	0.73	0.00		

In Table 14, we find the HSD test results are similar to what we see for classifying attack types for performance in terms of AUC. Again, Decision Tree is most adept at recognizing FTP-brute force attacks, with significant differences revealed in its ability to identify Combined and SSH attack types.

TABLE 14. HSD TEST GROUPINGS AFTER ANOVA OF AUPRC FOR THE ATTACK_TYPE FACTOR

Group a consists of: FTP
Group b consists of: Combined
Group c consists of: SSH

Figure 4. Boxplots of AUPRC by Attack_type; HSD group printed above box; color corresponds to HSD group



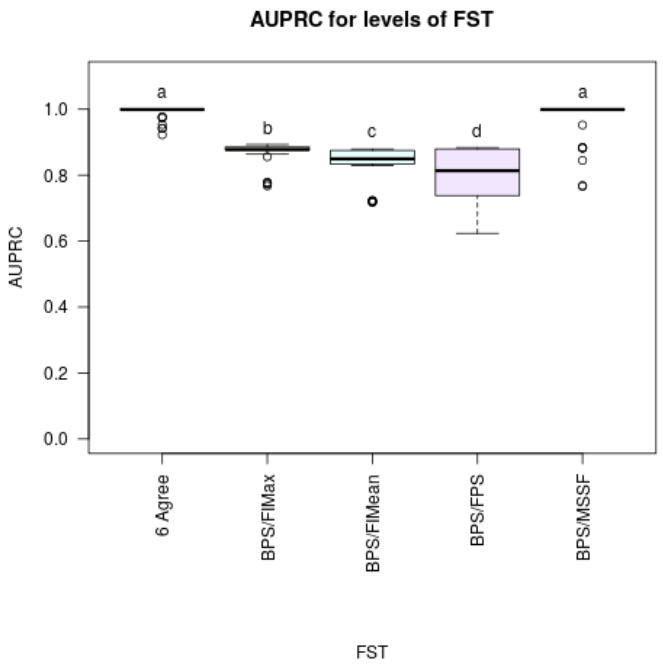
As in the case for the HSD test in terms of AUC, in Table 15, we see that the Bwd_Packets_s / min_seg_size_forward FST is again in the same HSD group as the 6-Agree FST for AUPRC performance.

TABLE 15. HSD TEST GROUPINGS AFTER ANOVA OF AUPRC FOR THE FST FACTOR

Group a consists of: 6-Agree, BPS/MSSF
Group b consists of: BPS/FIMax
Group c consists of: BPS/FIMean
Group d consists of: BPS/FPS

Lastly, we provide boxplots of performance in terms of AUPRC, grouped by FST in Figure 5. The narrow boxplots for the best performing HSD groups indicate that Decision Tree's classification results are stable with respect to sampling and shuffling that occurs during 5-fold cross validation for those FSTs.

Figure 5. Boxplots of AUPRC for levels of FST; HSD group printed above box; color corresponds to HSD group



6. Conclusion

In closing, we return to our research questions. Research question **Q1** is, "Can Decision Tree classify CSE-CIC-IDS2018 SSH-Brute Force, FTP-Brute Force, and Combined-Brute Force attacks data with strong results, thus obviating the need for more complex algorithms?" Our results and statistical analysis indicate that the answer to **Q1** is "yes." Our results show Decision Tree yields AUC and AUPRC values that are largely greater than 0.99 for a dataset with only two features. These results are close to the theoretical maximum value of 1. Therefore, it may not be necessary to utilize a more complex ML algorithm to detect SSH-Brute Force and FTP-Brute Force attacks in CSE-CIC-IDS2018 data.

Next we consider the second research question, **Q2**, "What is the smallest number of features in CSE-CIC-IDS2018 we can use to achieve consistently strong classification performance?" Table 9 contains inconsistent results for models trained on a one-feature dataset. Therefore, we find models trained on one-feature datasets are not reliable. However, the HSD groups reported in Figures 3 and 5 show that Decision Tree classifiers trained and tested on the Bwd_Packets_s / min_seg_size_forward (two-feature) datasets perform statistically similar to Decision Tree classifiers trained and tested on the four-feature 6-Agree datasets. Therefore, the answer to question **Q2** is that the minimum number of features we can use to achieve robust classification performance is 2.

The answers to these research questions lead to some ideas for further research. One area for future work is to find other network traffic datasets where Decision Tree may provide performance comparable to what it yields with the datasets we use here. To sum up, Decision Tree, trained and tested on just two features, Bwd_Packets_s and min_seg_size_forward from CIC-CSE-IDS2018 Big Data, is demonstrably and effectively capable of detecting SSH-Brute Force and FTP-Brute Force attacks.

References

[1] A. G. Roesener, C. Bottolfson, and G. Fernandez, "Policy for us cybersecurity," AIR UNIV MAXWELL AFB AL AIR FORCE RESEARCH INST, Tech. Rep., 2014.

[2] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSP*, 2018, pp. 108–116.

[4] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020.

[5] A. De Mauro, M. Greco, and M. Grimaldi, "A formal definition of big data based on its essential features," *Library Review*, 2016.

[6] M. Bekkar, H. K. Djemaa, and T. A. Alitouche, "Evaluation measures for models assessment over imbalanced data sets," *J Inf Eng Appl*, vol. 3, no. 10, 2013.

[7] K. Boyd, K. H. Eng, and C. D. Page, "Area under the precision-recall curve: point estimates and confidence intervals," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 451–466.

[8] D. J. Barrett, D. J. Barrett, R. E. Silverman, and R. Silverman, *SSH, the Secure Shell: the definitive guide*. " O'Reilly Media, Inc.", 2001.

[9] J. Postel and J. Reynolds, "File transfer protocol," 1985.

[10] M. M. Najafabadi, T. M. Khoshgoftaar, C. Calvert, and C. Kemp, "Detection of ssh brute force attacks using aggregated netflow data," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 283–288.

[11] S. K. Wanjau, G. M. Wambugu, and G. N. Kamau, "Ssh-brute force attack detection model based on deep learning," *International Journal of Computer Applications Technology and Research*, vol. 10, no. 1, pp. 42–50, 2021.

[12] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[13] R. Zuech, J. Hancock, and T. M. Khoshgoftaar, "Detecting web attacks using random undersampling and ensemble learners," *Journal of Big Data*, vol. 8, no. 1, pp. 1–20, 2021.

[14] J. L. Leevy, J. Hancock, R. Zuech, and T. M. Khoshgoftaar, "Detecting cybersecurity attacks across different network features and learners," *Journal of Big Data*, vol. 8, no. 1, pp. 1–29, 2021.

[15] I. Rish *et al.*, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

[16] D. G. Kleinbaum, K. Dietz, M. Gail, M. Klein, and M. Klein, *Logistic regression*. Springer, 2002.

[17] R. Zuech, J. Hancock, and T. M. Khoshgoftaar, "Investigating rarity in web attacks with ensemble learners," *Journal of Big Data*, vol. 8, no. 1, pp. 1–27, 2021.

[18] H. Shapoorifard and P. Shamsinejad, "Intrusion detection using a novel hybrid method incorporating an improved knn," *Int. J. Comput. Appl*, vol. 173, no. 1, pp. 5–9, 2017.

[19] J. Gu, L. Wang, H. Wang, and S. Wang, "A novel approach to intrusion detection using svm ensemble with feature augmentation," *Computers & Security*, vol. 86, pp. 53–62, 2019.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[21] J. L. Leevy, J. Hancock, R. Zuech, and T. M. Khoshgoftaar, "Detecting cybersecurity attacks using different network features with lightgbm and xgboost learners," in *2020 IEEE Second International Conference on Cognitive Machine Intelligence (CogMI)*. IEEE, 2020, pp. 190–197.

[22] G. Van Rossum and F. Drake, "Python 3 reference manual createspace," *Scotts Valley, CA*, 2009.

[23] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2018. [Online]. Available: https://www.R-project.org/

[24] G. R. Iversen and H. Norpoth, *Analysis of variance*. Sage, 1987, no. 1.

[25] Felipe de Mendiburu and Muhammad Yaseen, *agricolae: Statistical Procedures for Agricultural Research*, 2020, r package version 1.4.0.

[26] J. W. Tukey, "Comparing individual means in the analysis of variance," *Biometrics*, pp. 99–114, 1949.