A cross-comparison of feature selection algorithms on multiple cyber security data-sets

Alexander Powell, Darren Bates, Chad Van Wyk, and Adrian Darren de Abreu

Stellenbosch University, Stellenbosch, South Africa

Abstract. In network intrusion detection, it is essential to detect an attack in real-time and to initiate preventive measures accordingly. This paper aims to evaluate whether SciKit Learn feature selection algorithms improve or worsen the accuracy and processing time of machine learning algorithms when used for network intrusion detection and classification. We develop recommendations of potential machine learning and feature selection algorithms that can be used to obtain a desirable level of accuracy whilst significantly reducing the total processing time of the algorithm.

Keywords: Cyber Security · Feature Selection · Machine Learning · Network Intrusion Detection· SciKitLearn · SK Learn.

1 Introduction

During the last decade, the number of devices handling data has increased exponentially and as a result cyber security has become a major concern and priority for many companies globally. Othman [12] states that recent advancements in technology cause an increase in volume, variety and speed of network generated data and has made data analysis and processing of intrusion detection a challenge for traditional algorithms, namely K-means, Naïve Bayesian, K-Nearest Neighbors and Support Vector Machine. This increase in data volume has a linear relationship with the number of well executed cyber-attacks, which have increased in both their occurrence and their capability of incapacitating individuals, businesses and organizations whilst remaining undetected [12].

Intrusion detection comprises of identifying malicious activities that may hinder or disrupt the functionality of a system. A challenging aspect of intrusion detection is distinguishing a clear boundary between normal and malicious network activity [5].This paper provides a comprehensive investigation of various feature selection algorithms to evaluate their performance and accuracy when applied to more than one security data-set. Additionally, this research paper presents an evaluation of the accuracy and the various performance metrics of the machine learning algorithms used. Feature selection algorithms are then paired with machine learning algorithms to determine which feature selection algorithms produce the largest differences in accuracy and in computational processing times. A cross-comparison of the most compatible machine learning and feature selection algorithms will be presented. The resulting matrix provides users with a heuristic to select the most suitable combination of machine learning and feature selection algorithms based on each data-set.

The objective of this paper is to determine which feature selection algorithm is the most robust when tested on more than one security data-set for intrusion detection. Another main objective of the paper is to determine which of the selected machine learning algorithms produce the highest accuracy scores amongst the three data-sets with no feature selection. This will be achieved by baselining the following machine learning algorithms: Logistic Regression, K-Nearest Neighbors, Decision Tree, Random Forest, and K-Means. Only commonly used sklearn machine learning and feature selection algorithms were considered. The feature selection algorithms will be used in conjunction with the baselined machine learning algorithms to determine which feature selection algorithm produced the most consistent accuracy ratings whilst reducing processing times. A final objective of our paper is to identify the two best performing combinations of machine learning and feature selection algorithms per data-set. Additionally, this paper will address the complexities of working with network intrusion data and moreover, the difficulties in applying machine learning and feature selection onto it.

The rest of this paper is structured as follows: first, related work and cyber security complexities are presented. Following this, the methodology used in this paper is presented. Machine learning algorithms will then be analysed and compared with regards to accuracy and processing time in order to identify the best performing baselined machine learning algorithms. Following this, each combination of machine learning and feature selection algorithms will be examined in terms of their accuracy and processing time on each data-set. A cross-comparison is then presented, which determines the best performing combination of machine learning and feature selection algorithms for each data-set. After this, findings, conclusions and future work are discussed.

2 Related Work

Çavuşoğlu [4] proposed an intrusion detection system (IDS) that pairs multiple machine learning algorithms and feature selection techniques, in order to achieve high detection and performance rates.The NSL-KDD data-set was used. Çavuşoğlu [4] used feature selection in an attempt to improve the performance and detection rates of the chosen algorithms by establishing new relationships between specific attributes and condensing the overall number of attributes present in the data-set. The machine learning algorithms exhibited high accuracy and low false-positive rates when identifying any attack type. The paper concludes that the proposed system successfully identified attacks at higher rates than that of existing detection systems.

Gunduz et al [9] attempted to determine the most important feature sets within the KDD Cup'99 data-set by using feature selection techniques to increase the accuracy and reduce computational time by decreasing the number of features analysed.The results of the accuracy scores of each classification algorithm were compared to each other to determine which feature selection set produced the most accurate results. Several authors have discussed feature selection algorithms and their application in cyber security. Relevant works include those of Al-Jarrah et al [2], Anusha and Sathiyamoorthy [3] and Xue et al [17]. Therefore, previous research has applied machine learning intrusion detection onto one data-set, whereas our paper will conduct a cross-comparison among

three cyber security data-sets. Furthermore, the application of SciKit Learn's machine learning and feature selection algorithms has been undervalued in existing literature and to our knowledge, a comprehensive evaluation of such nature does not exist.

3 Complexities of working with cyber security data-sets

The cyber security data-sets utilized in this paper are representative of network flow data taken directly from a network at the Canadian Institute for Cybersecurity. There is thus a large amount of traffic in this data. For example, the CICIDS-2017 data-set is 3 GB, resulting in difficulty when analysing and experimenting on the data-set. The relatively large size of the data-set resulted in long processing times and difficulty in finding hardware that had sufficient computing power to support the algorithms being run. Short processing times are important for an intrusion detection system since any delay can compromise the effectiveness and speed at which a system can detect intrusions [1].

The issue of over-fitting is said to have occurred when a statistical model has successfully and accurately fit the data, however has failed to describe the pattern or trends that underlie it [16]. Often due to a large amount of data-set features relative to "events" (in this case benign instances), over-fitting leads to an increased likelihood of inaccurate predictions [6]. Feature counts of the network data-sets utilized in this paper were 43, 80, and 79 for NSL-KDD, ISCX-URL-2016 and CICIDS-2017, respectively. Additionally, excluding the ISCX-URL-2016 data-set, only a small portion of each data-set has been classified as malicious, thus prompting direct measures to be employed in order to prevent over-fitting. The preventative measures used within this paper, both being widely accepted methods of combating over-fitting, are feature selection and the partitioning of each data-set through the use of a train_test_split [16].

Network intrusion data-sets inherently have many features that are neither intuitive nor have an obvious meaning. A lack of information on specific data-set features means that one cannot be certain how necessary these features are for the machine learning algorithms. High feature counts impact accuracy, increases processing and loading times and increases the difficulty of analysing the data. The speed of analysis is very important as delays could result in malicious attacks infiltrating a system [1]. The NSL-KDD data-set comprised of 43 features, the ISCX-URL-2016 data-set comprised of 80, and the CICIDS-2017 data-set comprised of 79 features. Once the feature selection algorithms were run, the number of features were dramatically reduced. The more features a data-set possess, the higher the computational power needed to process it. The problem with many features is that some may be irrelevant or redundant, and not contribute to the final result or may negatively disturb end results [1].

Previous studies of the CICIDS-2017 data-set show a total of 3,119,345 observations, 83 features and 288,602 missing values [13]. The data-set used in this research paper contains 2,271,320 observations, 79 features and 2867 missing values, and this may be the result of data-set tampering. A lack of information with regards to what has been tampered with, removed, or added to the data-set may skew results retrieved and potentially remove valuable insights. As discussed previously, the NSL-KDD data-set is a revised version of the KDD'99 data-set. Therefore, it is apparent that the KDD'99 was tampered with by researchers in an attempt to remove several issues present in the

data-set, such as duplicate records. Although this may have led to an improvement in the NSL-KDD data-set, tampering with network intrusion data-sets may skew the data unintentionally. Lastly, this is an indication that data-sets may be curated or tampered with by researchers, and thus cannot be true representations, but rather curated iterations of proposed network activity.

Defined as the over-representation of one or more classes in a data-set, class imbalance with regards to machine learning classification is a major challenge that is actively being studied [8]. Inherently, certain services are more often consumed than others in network environments, and network data-sets are thus in most cases non-uniform and display relatively large class imbalances. Subsequently, network data-sets used for classifying malicious activity often contain only a small proportion of malicious data. In our research paper, except for the ISCX-URL-2016 data-set, this trend follows. This, coupled with the fact that standard machine learning algorithms were developed under the assumption that training data would have an equal distribution of classes, is likely to cause classifier bias towards the majority class, thus skewing results [8]. The percentage of maliciously classified network data in the three data-sets in this paper is 78.8%, 19.7%, and 19.05% for ISCX-URL-2016, CICIDS-2017, and NSL-KDD, respectively. To prevent classifier bias and the skewing of results, great care was taken to ensure that the training data-sets included instances representative of both benign and of each attack type present in the original data-set. Lastly, the network data-sets utilized in this paper are the result of a simulation and are not an accurate representation of a real-world network environment.

4 Methodology

All data-sets used are publicly accessible and have been collected from the Canadian Institute for Cybersecurity [15]. This paper utilises three network intrusion data-sets: ISCX-URL-2016, NSL-KDD, and CICIDS2017.

The CICIDS-2017 data-set contains 2,830,743 observations with 79 features. Of this, 2 271,320 instances were flagged as benign and 556,556 as malicious activity. A total of 14 different attack types are represented within the data-set, and 24.5% of the traffic has been classified as malicious and the remainder benign. The ISCX-URL-2016 data-set, consisting of five types of URLs, namely: benign, spam, phishing, malware, and defacement, was created by Mamun et al [10] for the purpose of detection and categorization of malicious URLs. The data-set contains 36,707 network observations and 80 features, with 19.70% of observations being malicious. The NSL-KDD data-set contains malicious network observations belonging to one of four distinct attack groups: denial of service, user-to-root, remote-to-local or probing attack [14]. The dimensions of the data-set are 4,898,431 by 43, with 19.05% of the data being malicious.

The SciKit Learn package was selected as it is a reliable package for data preprocessing, machine learning and feature selection, providing various functions and efficient processing times for such tasks [11]. In order to ensure standardization, one computer was used to run both the machine learning and feature selection algorithms and compute performance statistics for every data-set. The device specifications are as follows: Dell Inspiron 7577, 16 GB RAM, Intel Core i7-7700HQ and 64-bit Windows

10. During pre-processing, a binary column was created to identify network traffic as either benign (0) or malicious (1), and thereafter the pre-processed data-sets were split using the SK Learn's train_test_split function. The NSL-KDD data-set is pre-packaged into train and test data-sets. The resulting ratio of a 85% train and 15% test split was applied to the other two data-sets to ensure standardisation. The selected machine learning algorithms were trained and tested on all three data-sets, with no feature selection or hyper-parameter tuning, and ranked according to their overall accuracy and computational time (training and testing duration). The top four performing baselined algorithms were then used together with the following SciKit Learn feature selection algorithms: Extra-Tree Classifier/SelectFromModel, SelectKbest, Variance Threshold, and SelectPercentile. The SelectFromModel feature selection algorithm was used to remove unimportant features that were weighted by the Extra-Tree Classifier. Finally, a cross-comparison was conducted to compare how each machine learning and feature selection pair performed against its original baselined statistics in order to determine which combinations of algorithms improved or retained the highest accuracy rates whilst decreasing overall computational time. Lastly, a total of six recommendations (two per data-set) are presented.

5 Analysis of baselined machine learning algorithms

Analysis was initiated by bench-marking the machine learning algorithms. This was achieved by training and testing each machine learning algorithm on each of the three data-sets with no hyper-parameter tuning or feature selection, in order to determine which algorithms natively produce the highest accuracy rates in the lowest computational times. Only the top four machine learning algorithms were chosen based on their accuracy and processing times, and used in the following feature selection analysis. The chosen performance metrics include: accuracy, precision, recall, F1-score and total processing time (training and testing duration).

Upon examination of the ISCX-URL-2016 data-set, it is apparent that the four best performing algorithms, with regards to accuracy, are: Logistic Regression, K-Nearest Neighbors, Decision Tree, and Random Forest. Baselined, these algorithms produced accuracy ratings of 92%, 99%, 99%, and 99%, respectively. Upon further examination, Logistic Regression displayed the lowest computational time of 0.68 seconds, differing only by 0.10 seconds from the best performing baselined algorithm, Decision Tree. Furthermore, Random Forest was the third best performing algorithm with an accuracy of 99% and processing time of 4.67 seconds. The fourth top performing algorithm was K-Nearest Neighbors which produced an accuracy of 99% in a total processing time of 4.96 seconds.

In analysing how each machine learning algorithm performed on the NSL-KDD, it is apparent that the Logistic Regression and Decision Tree algorithms produced the highest accuracy rating (76%) with execution times of 1.02 and 1.40 seconds. Random Forest is the third best performing algorithm with an accuracy of 72% and a processing time of 1.23 seconds. Lastly, K-Nearest Neighbors is the fourth best performing baselined machine learning algorithm with an accuracy of 69% but the second longest processing time of 121.33 seconds. The remaining algorithms (supervised and unsu-

pervised K-Means) exhibit a significant decrease in accuracy, an increase in processing time and do not rank in the top four best performing algorithms and are thus, not used in further analysis. Therefore, the four best performing baselined algorithms on the NSL-KDD data-set include: Logistic Regression, Decision Tree, Random Forest, and K-Nearest Neighbors with accuracy ratings of above 69% and run-times as low as 1.02 seconds.

In terms of the CICIDS-2017 data-set, Logistic Regression achieved an overall accuracy of 93% with a total run-time of 308.71 seconds. While this is not the top performing algorithm, it managed to process the results in one of the shorter time frames. Although producing an accuracy of 99%, K-Nearest Neighbours displayed the longest processing time of 13970.01 seconds. Decision Tree and Random Forest both achieved an accuracy of 100% with short processing times of 230.21 and 137.30 seconds, respectively. Comparing this to K-Nearest Neighbors, one can see that it achieved a relatively similar accuracy but in a shorter processing time. From these results it can be determined that Random Forest was the best performing algorithm, since it was the fastest and most accurate algorithm followed closely by Decision Tree. The third best performing algorithm is Logistic Regression, as it is 6% less accurate than K-Nearest Neighbors but was less computationally intensive as it produced results 13 661 seconds faster. The last algorithm is K-Nearest Neighbors which generated a accuracy of 99% but with the longest processing time of 13970.01 seconds.

It is evident that Logistic Regression, K-Nearest Neighbors, Decision Tree and Random Forest are the machine learning algorithms that have yielded the highest accuracy ratings within the lowest computational times across all three data-sets and will therefore, be further analysed in terms of their changes in performances when used together with feature selection.

6 Analysis of feature selection algorithms

Each of the four best performing machine learning algorithms were paired with the following feature selection algorithms: Extra-Tree Classifier and SelectFromModel, SelectKBest, SelectPercentile, and VarianceThreshold.In this section, each machine learning algorithm will be compared to each other, in order to determine which machine learning algorithm is the most suitable for each of the four feature selection algorithms, on each data-set. Thereafter, a total of 12 algorithm combinations are selected for further analysis.

The feature selection algorithms: Extra-Tree Classifier, SelectKBest, SelectPercentile and VarianceThreshold each reduced the number of features in the ISCX-URL-2016 data-set from 80 features to 28, 10, 8, and 27, respectively. When paired with the Extra-Tree Classifier, Decision tree was the best performing machine learning algorithm, producing an accuracy of 99% and a computational time of 0.29 seconds. Similarly, the pairing of Decision Tree with the SelectKBest algorithm yielded the best results in terms of accuracy and computational time, producing an accuracy of 97% in 0.11 seconds, the lowest computational time relative to the four highest performing pairs. SelectPercentile performed excellently when paired with the Random Forest algorithm, reducing the data-set from 80 features to 8, and producing an accuracy of 96% in a

computational time of 0.18 seconds. Finally, although displaying one of the highest computational times (0.41 seconds) and the second highest number of features (27), Random Forest produced a high accuracy of 99% when paired with VarianceThreshold.

Each of the four chosen feature selection algorithms were run on the NSL-KDD data-set. When the Extra-Tree Classifier and SelectFromModel algorithms were applied, the number of features reduced from 43 to 32 and Logistic Regression produced the highest accuracy rate (74%) and a relatively low computational time (1.32 seconds). The SelectKBest algorithm reduced the number of features from 43 to 10. After each machine learning algorithm was paired with the SelectKBest algorithm, it is evident that the Decision Tree algorithm produced the most desirable performance rates, despite these statistics being relatively low. The Decision Tree algorithm yielded an accuracy rating of 47% with a total computational run-time of 0.39 seconds. The SelectPercentile algorithm subsequently reduced the number of features from 43 to 13. The algorithm that performed best when combined with the SelectPercentile feature selection algorithm was Decision Tree, with an accuracy of 74% and a run-time of 0.46 seconds. It is important to note that when SelectPercentile was combined with Logistic Regression, K-Nearest Neighbors and Random Forest all produced significantly lower accuracy rates that range between 43% to 47%, respectively. Lastly, VarianceThreshold reduced the number of features from 43 to 40. To this end, the Decision Tree algorithm performed best when paired with the VarianceThreshold algorithm, with an accuracy of 74% and a computational run-time of 0.71 seconds. Therefore, it is evident that the best combinations of machine learning and feature selection algorithms when applied to the NSL-KDD data-set include, Logistic Regression and Extra-Tree Classifier/SelectFromModel, Decision Tree and SelectKBest, Decision Tree and SelectPercentile, and lastly, Decision Tree and VarianceThreshold.

When the Extra-Tree Classifier was run on the CICIDS-2017 data-set, the original 80 features were reduced to 26. When the Logistic Regression algorithm was trained on this reduced data-set, it produced an accuracy of 91% in 82.87 seconds. The Decision Tree and Random Forest algorithms produced 100% accuracy ratings with run-times of 83.03 and 112.37 seconds. Secondly, SelectKBest reduced the total 80 features to 10 features. This generated an accuracy for Logistic Regression of 88% with a processing time of 17.83 seconds. This is a decrease of 3% in accuracy but a massive decrease of 65.04 seconds in processing time. The same pattern follows for Decision Tree and Random Forest in that the accuracy decreased by 3% to 97% but the processing times decreased by 56.54 and 41.98 seconds. The SelectPercentile feature selection algorithm managed to reduce the number of features to 8. This resulted in Logistic Regression producing an accuracy of 88% in 12.03 seconds. When compared to the SelectKBest algorithm the number of features was reduced by 2, the processing time decreased by 5.8 seconds, whilst maintaining the same accuracy. Decision Tree and Random Forest obtained an accuracy of 97% and a processing time of 23.2 and 53.21 seconds which show improvements from the SelectKBest algorithm. The VarianceThreshold feature selection algorithm decreased the total features to 23. Logistic Regression produced an accuracy of 87% and a processing time of 87.74 seconds. Decision tree and Random Forest retained an accuracy of 100% in processing times of 78.87 and 123.98 seconds.

From analysing the four feature selection algorithms, the best performing combination of algorithm appears to be the Decision Tree and VarianceThreshold combination.

7 Cross-comparison of baselined and feature selection algorithms

The next phase of our analysis entails the cross-comparison of the four combinations of machine learning and feature selection algorithms against the original baselined machine learning algorithm performances, to determine whether for each algorithm the baselined machine learning accuracy rate either improved or retained its original rating, and whether the overall processing duration was decreased substantially. Additionally, the number of true negatives (correctly classified attacks) and false negatives (incorrectly classified attacks) are compared and the two best performing combinations of machine learning and feature selection algorithms are recommended per data-set.

Paired with the Extra-Tree Classifier on the ISCX-URL-2016 data-set, the accuracy achieved by the Decision Tree algorithm remained 99%, however its computational time decreased by 0.49 seconds, dropping from a baselined computational time of 0.78 seconds to 0.29 seconds. Additionally, the number of true negatives identified increased from 4323 to 4334, along with a decrease in false negatives from 45 to 34. Similarly, when paired with SelectKBest, the computational time of the Decision Tree algorithm dropped from 0.78 seconds to 0.11 seconds, the shortest computational time of each of the four combinations. This drop in time was accompanied by a 2% decrease in accuracy, lowering the accuracy attained at baseline, 99%, to an accuracy of 97%. Examination of both the true negatives and false negatives reveal a similar trend, with true negatives dropping from 4323 to 4281, and false negatives increasing from 45 to 87. The baselined accuracy of the Random Forest algorithm, 99%, was also lowered when paired with the SelectPercentile feature selection algorithm, and dropped to 96%. This 3% drop in accuracy, was reinforced by the number of true negatives dropping from 4346 to 4246 and the false negatives increasing from 22 to 122, but however was accompanied by a substantially lower computational time. A total of 4.49 seconds was dropped when combined with the SelectPercentile algorithm, lowering the baselined computational time from 4.67 seconds to 0.18 seconds. Lastly, when paired with the VarianceThreshold algorithm, the baselined accuracy of the Random Forest algorithm was matched at 99%, with the number of true negatives decreasing from 4346 to 4337 and the false negatives increasing from 22 to 31, and accompanied by a drop in computational time of 4.26 seconds, dropping from 4.67 to 0.41 seconds.

With regards to the NSL-KDD data-set, comparing the baselined Logistic Regression performance metrics against that of the Extra-Tree Classifier and Logistic Regression feature selection combination, it is evident that accuracy decreased from 76% to 74%, whereas total processing time increased from 1.02 seconds to 1.32 seconds. Originally, the Logistic Regression algorithm correctly identified 8738 attacks and misclassified 4096 attacks as normal network traffic, but after Logistic Regression was run with the Extra-Tree Classifier and SelectFromModel, the number of correctly classified network attacks was 8780 and incorrectly classified attacks was 4053. The baselined Decision Tree performance metrics produced an accuracy of 76% and a run-time of 1.40 seconds. When the Decision Tree algorithm was paired with SelectKBest algo-

rithm, one can see that accuracy decreased drastically from 76% to 47% accuracy and processing time decreased from 1.40 seconds to 0.39 seconds.Additionally, the baselined Decision Tree algorithm successfully classified 8307 attacks and incorrectly classified 4526 attacks as benign network traffic and after being paired with the SelectKBest feature selection algorithm, Decision Tree successfully classified 3125 attacks and misclassified 9708 attacks as benign. The Decision Tree algorithm was also the best performing algorithm when partnered with the SelectPercentile algorithm. To this end, accuracy decreased from 76% with no feature selection to 74% with SelectPercentile feature selection and run-time subsequently decreased from 1.40 seconds to 0.46 seconds, whilst correctly classifying 7085 attacks and mis-classifying 5748 attacks. Lastly, Decision Tree was the best performing algorithm when used with VarianceThreshold feature selection. The Decision Tree baselined accuracy of 76% decreased to 74% whilst computational processing time reduced from 1.40 to 0.71 seconds. The combination of Decision Tree and VarianceThreshold was able to detect and correctly classify 8044 network attacks and incorrectly mis-classified 4789 attacks as benign network traffic. Therefore, the Decision Tree algorithm is the most robust to feature selection as it was the top performing algorithm when paired with three separate feature selection algorithms (SelectKBest, SelectPercentile, VarianceThreshold).

The best performing algorithms on the CICIDS-2017 data-set were Decision Tree and Random Forest for each feature selection algorithm. Decision Tree produced a baselined accuracy of 100% and a run-time of 230.21 seconds. When the Decision Tree and SelectKBest algorithms were paired, accuracy decreased to 97% and processing time to 26.49 seconds. The false negatives increased from 276 to 10508 and the true negatives decreased from 82947 to 72762. When Decision Tree was run with ExtraTreeClassifier/SelectFromModel, it produced 341 false negatives and 82929 true negatives with a processing time of 83.03 seconds and an accuracy rating of 100%. Decision Tree and VarianceThreshold was able to produce a high accuracy rating of 100%, of which the number of incorrectly and correctly classified attacks were 117 and 83153, respectfully. The processing time was relatively low at 78.87 seconds for Decision Tree, which is a large decrease from the original computational time (230.21), but it remains a reasonable time when compared with the other algorithms. Decision Tree and SelectPercentile generated an accuracy of 97%, a processing time of 23.22 seconds, and correctly classified 72716 attacks whilst incorrectly classifying 10554 attacks as benign. Drawing from the above analysis, it is apparent that the most suitable combination of machine learning and feature selection algorithms is Decision Tree and VarianceThreshold. Paired together, these algorithms have a low false negative rate, greatly reduced the number of features to 23 whilst retaining an accuracy of 100%, and decreasing the processing time to 78.87 seconds. Secondly, the combination of Decision Tree and ExtraTree Classifier produced an accuracy of 100% in 83.03 seconds but incorrectly classified 341 attacks, which is 224 more incorrectly classified attacks than that of the Decision Tree and VarianceThreshold algorithms.

8 Findings

It is evident that when machine learning algorithms were used in combination with feature selection algorithms, there was either a significant decrease in both accuracy and processing time or the algorithm's baselined accuracy rate was maintained whilst drastically reducing computational time. Given that network intrusion detection requires the identification of malicious traffic in real time with as little downtime as possible, we are interested in the cases where machine learning algorithms maintained their accuracy and decreased in total processing time when feature selection was used. This section will present combinations of machine learning and feature selection algorithms that are best suited for network intrusion detection and classification.

In our analysis on the ISCX-URL-2016 data-set, it was identified that the algorithm combination that achieved the most desirable results was Decision Tree and ExtraTree Classifier, reducing the number of features from 80 to 28 and producing an accuracy of 99% in 0.29 seconds. Moreover, this combination successfully classified 4334 attacks and mis-classified only 34 attacks. The second top performing algorithm was Decision Tree and SelectKBest which reduced the data-set to 10 features and displayed a total processing time of 0.11 seconds whilst retaining a high accuracy of 97%. This combination successfully classified 4281 attacks and incorrectly classified 87 attacks as benign. The best performing algorithm combination on the NSL-KDD data-set was Decision Tree and VarianceThreshold. During the combination, the number of features was reduced to 40 and a relatively high accuracy rate of 74% was maintained with a total processing time of 0.71 seconds. The algorithms correctly classified 8044 attacks whilst incorrectly labelling 4789 malicious attacks as benign network traffic. The second combination of algorithms is Decision Tree and SelectPercentile, as these algorithms reduced the number of features from 43 to 13 and successfully classified 7085 attacks and incorrectly classified 5748 attacks, whilst producing an accuracy of 74% in an execution time of 0.46 seconds. Lastly, on the CICIDS-2017 data-set, when Decision Tree was executed with VarianceThreshold, the number of features was reduced from 79 to 23 and the combination successfully classified 83153 attacks whilst incorrectly labelling 117 attacks as normal network traffic, and produced an accuracy 100% in 78.87 seconds. Secondly, when Decision Tree was run with ExtraTreeClassifier/SelectFromModel, the feature count was reduced to 26 and the combination managed to correctly label 82929 malicious network traffic and incorrectly classified 341 attacks in an execution time of 83.03 seconds whilst retaining an accuracy rating of 100%.

Therefore, is evident that these six combinations provide a useful foundation for combinations of machine learning and feature selection algorithms that meet the main application and task of feature selection on network intrusion data, being the reduction of data-set features and processing time to ensure that malicious network activity is promptly identified. Despite the findings being data-set-specific, general trends can be seen across the three data-sets. It is evident that the most robust machine learning algorithm is Decision Tree, as it performed best with all feature selection algorithms and on each data-set and ultimately, managed to retain high accuracy rates in short execution times. The most robust feature selection algorithm changes depending on the data-set it was tested on. However, a pattern emerged that showed that ExtraTree Classifier performed better on the larger and smaller data-sets, whereas SelectKBest and SelectPer-

centile performed better on the smaller data-sets. Therefore, an important outcome of these findings is a set of algorithm combinations for anyone who is seeking to use the SK Learn library for machine learning and feature selection on network intrusion data.

9 Conclusion and future research

In this paper, feature selection and machine learning algorithms were applied to the ISCX-URL-2016, NSL-KDD, and CICIDS-2017 data-sets to perform network intrusion detection. Each machine learning algorithm was baselined. Next, machine learning algorithms were paired with feature selections algorithms. Thereafter, a cross-comparison of the four best performing algorithms was conducted in terms of accuracy and processing time. The recommendations made provide network intrusion detection practitioners with varying combinations of feature selection and machine learning algorithms. Our research forms the basis for future work on the development of a recommendation system for SK Learn machine learning and feature selection algorithms, in the application of intrusion detection. It was identified that the top performing machine learning and feature selection algorithm combinations, across the data-sets, are as follows: Decision Tree and ExtraTree Classifier, Decision Tree and SelectKBest, Decision Tree and VarianceThreshold, Decision Tree and SelectPercentile. Each of these combinations retain their baselined accuracy whilst significantly reducing the total processing time. The Decision Tree machine learning algorithm is the most robust to feature selection as it retains its original accuracy with a significant reduction in processing time after feature selection. Additionally, there are several complexities that one faces when working with cyber security data-sets, including: large data-sets and long processing times, significant number of features, data tampering and class imbalances. A major constraint when working with cyber security data is the complexity and non-intuitive nature of the data-set features. This complexity resulted in our research focusing on automated feature selection as opposed to manually removing highly correlated features as the underlying meaning and structure of the data is non-obvious and complex. This paper's research could be extended through the implementation of both cross-validation and hyper-parameter tuning when obtaining the initial baseline machine learning statistics. A further extension would be the inclusion of more advanced machine learning (neural networks) algorithms within SK Learn. Lastly, future work would include a shift away from total processing time, to unit processing time (time per sample).

10 Acknowledgements

Our paper acknowledges the Canadian Institute for Cybersecurity for their publicly available network intrusion data-sets. This paper would also like to acknowledge Professor Louise Leenen from the University of the Western Cape for consultation on our methodology and research objectives. This paper would like to acknowledge Professor Bruce Watson and Professor Arina Britz as our paper supervisors and for consultations throughout the development of this paper. Lastly, our paper acknowledges the valuable feedback from the two anonymous FAIR2019 peer-reviewers.

References

1. Ait Tchackoucht, T. and Ezziyyani, M.: Building A Fast Intrusion Detection System For High-Speed-Networks: Probe and DoS Attacks Detection. Procedia Computer Science. **127**, 521–530 (2018)
2. Al-Jarrah, O. and Arafat, A.: Network Intrusion Detection System using attack behavior classification. In 2014 5th International Conference on Information and Communication Systems. pp. 1–6. IEEE, (2014)
3. Anusha, K. and Sathiyamoorthy, E.: Comparative study for feature selection algorithms in intrusion detection system. Automatic Control and Computer Sciences. **50**(1), 1–9 (2016)
4. Çavuşoğlu, Ü.: A new hybrid approach for intrusion detection using machine learning methods. Applied Intelligence. **49**(7), 2735–2761 (2019)
5. Chattopadhyay, M., Sen, R. & Gupta, S.: A Comprehensive Review and Meta-Analysis on Applications of Machine Learning Techniques in Intrusion Detection. Australasian Journal of Information Systems. **22**, (2018)
6. Foster, K.R., Koprowski, R. and Skufca, J.D.: Machine learning, medical diagnosis, and biomedical engineering research-commentary. Biomedical engineering online. **13**(1), 94. (2014)
7. Gharib, A., Sharafaldin, I., Lashkari, A.H. and Ghorbani, A.A.: An evaluation framework for intrusion detection dataset. In: 2016 International Conference on Information Science and Security. pp. 1–6. IEEE (2016)
8. Gómez, S.E., Hernández-Callejo, L., Martínez, B.C. and Sánchez-Esguevillas, A.J.: Exploratory study on class imbalance and solutions for network traffic classification. Neurocomputing. **343**, 100–119 (2019)
9. Gündüz, S.Y. & Çeter, M.N.: Feature Selection and Comparison of Classification Algorithms for Intrusion Detection. Anadolu University Journal of Science and Technology: Applied Sciences and Engineering. **19**(1), 206–218 (2018)
10. Mamun, M.S.I., Rathore, M.A., Lashkari, A.H., Stakhanova, N. and Ghorbani, A.A.: Detecting malicious urls using lexical analysis. In: International Conference on Network and System Security,pp. 467–482. (2016)
11. Mishra, P. Varadharajan, V., Tupakula, U., Pilli, E.S.: A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection. IEEE Communications Surveys & Tutorials. **21**(1), 686-728 (2019)
12. Othman, S. M., Ba-Alwi, F. M., Alsohybe, N. T. & Al-Hashida, A. Y.: Intrusion detection model using machine learning algorithm on Big Data environment. Journal of Big Data. **5**(34) (2018)
13. Panigrahi, Ranjit & Borah, Samarjeet.: A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. **7**, 479–482, (2018)
14. Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A.: A detailed analysis of the KDD CUP 99 dataset. In: 2nd IEEE Symposium on Computational Intelligence for Security and Defence Applications. (2009)
15. Canadian Institute for Cybersecurity, https://www.unb.ca/cic/datasets/nsl.html. Last accessed August 2019
16. Wagner, N. and Rondinelli, J.M.: Theory-guided machine learning in materials science. Frontiers in Materials. **3**, 28 (2016)
17. Xue, Y., Jia, W., Zhao, X. and Pang, W.: An evolutionary computation based feature selection method for intrusion detection. Security and Communication Networks. (2018)