Feature Selection Improves Tree-based Classification for Wireless Intrusion Detection

Shilpa Bhandari*
Avinash K Kukreja*
sbhandari05@student.ysu.edu
akkukreja@student.ysu.edu
Youngstown State University
Youngstown, OH

Alina Lazar
alazar@ysu.edu
Youngstown State University
Youngstown, OH

Alex Sim
Kesheng Wu
asim@lbl.gov,kwu@lbl.gov
Lawrence Berkeley
National Laboratory
Berkeley, CA

ABSTRACT

With the growth of 5G wireless technologies and IoT, it become urgent to develop robust network security systems, such as intrusions detection systems (IDS) to keep the networks secure. These IDS systems need to detect unauthorized access and attacks in real-time. However, most of the modern IDS are built based on complex machine learning models that are time-consuming to train. In this work, we propose a methodology using the SHapley Additive exPlanations (SHAP) in combination with tree-based classifiers. SHAP can be used to select consistent and small feature subsets to reduce the execution time and improve classification accuracy. We demonstrate the proposed approach with the Aegean Wi-Fi Intrusion Dataset (AWID) dataset in a series of multi-class classification experiments. Among the four classes ("normal", "injection", "flooding" and "impersonation"), it is well-known that the class impersonation is hard to be classified accurately. Tests show that we can use about 10% of the initial feature set without reducing the overall prediction accuracy. With this reduced set of features, the training time could be reduced as much as a factor of four, while slightly improving the discriminating ability to identify impersonation instances. This study suggests that by reducing the number of features, the classification algorithms are able to focus on key trends that differentiates the "attacks" classes from the "normal" class. Using a reduces subset of features improves IDS's accuracy and performance. Also, SHAP dependence plots capture the relationship between individual features and the classification decision.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; • **Networks →** *Network security*; • **Computing methodologies → Ensemble methods**; **Feature selection**.

KEYWORDS

Wi-Fi network, intrusion detection, classification, feature importance

*Both authors contributed equally to this research.

SNTA '20, June 23, 2020, Stockholm, Sweden

ACM ISBN 978-1-4503-7980-9/20/06…$15.00
https://doi.org/10.1145/3391812.3396274

ACM Reference Format:
Shilpa Bhandari, Avinash K Kukreja, Alina Lazar, Alex Sim, and Kesheng Wu. 2020. Feature Selection Improves Tree-based Classification for Wireless Intrusion Detection. In *3rd International Workshop on Systems and Network Telemetry and Analytics (SNTA '20), June 23, 2020, Stockholm, Sweden.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3391812.3396274

1 INTRODUCTION

Today, smaller and smaller sensors and wireless devices interconnected through the Internet are becoming extremely important in the day-to-day life of almost everyone. Meanwhile, cyberattacks threatening banking, online shopping, e-health and other digital services are launched everyday around the world [26, 35].

Due to the global increase of cyberattacks, building reliable IDSs is vital. IDSs must become an integrated part of any network anomaly detection system. Network providers should tune and continuously monitor their hardware and software systems in order to guarantee reliable, high quality, uninterrupted services. Extensive research has been conducted on building efficient IDS systems. As cyberattacks are becoming more sophisticated, security experts see machine learning and artificial intelligence (AI) as promising technologies to help identify and combat cyberattacks. In this context the use of machine learning algorithms can provide an accurate, efficiant and automated approach to detect intrusions.

Although has been proven by experts that machine learning models can detect [3, 5, 9, 10, 13, 20, 29, 33] most types of cyberattacks with high performances, the mechanics that affect accuracy are not usually analyzed and discussed. Also, building performant machine learning models require large datasets of labeled instances [16, 30]. Furthermore, tuning these machine learning models usually rely on a time-consuming parameter search process that depends on the underlying distribution in the training and testing data. In general, the complex mathematical nature of machine learning models is not easily interpretable. The reason for this is because machine learning models are typically considered black boxes, and while the performance might be impressive, researchers may not know the connection between the dataset, the model and its performance.

Datasets used to build IDS systems consists of passive measurements in the network where existing network traffic is observed without adding overhead. This can be done by capturing packets, monitoring interface counters or other ways collecting information using popular software systems such as Wireshark [27] or *tstat*[21]. The data collected this way, is usually noisy, contains a lot of missing values, some features only have constant values while others are highly correlated. Therefore, utilizing a preprocessing

step could potentially improve the efficiency and accuracy of IDS systems.

As a preprocessing step, feature selection is instrumental to identify the most important features and eliminate the non-essential features from datasets. By reducing the size of the problem, the computational time is also reduced and sometimes the performance increases. Relevant features contain useful information that the models can capture and use for better prediction. Irrelevant features, on the other side, result in performance degradation because these features carry no additional information that is not already provided by the selected features. This problem is exacerbated in the case of multi-class classification. In the end, feature selection and implicit dimensionality reduction promote better data understanding and interpretability, and may also reduce the required storage space and computational cost.

The main contribution of this paper is applying the SHapley Additive exPlanations (SHAP) to select the most important features from the benchmark AWID wireless network intrusion datasets, and demonstrate that we can not only reduce the time to classify the attacks but also improve the accuracy of classification of the most difficult cases. Our tests show that we can reduce the number of features used for classification by a factor of ten (from about 150 features to 15) while reducing the training time by as much as a factor of four. Using only a subset of 15 most important features, we show that the relative contribution of some features noticeably increases. We show evidence that this change is likely the reason to be able to successfully classifying the instance from the "impersonation" class, which are generally difficult to differentiate from the other classes.

This paper is organized as follows. Section 2 presents recent research advances on IDS using machine learning and compares our approach with previous work. Section 4 is an overview of the tree-based classification methodology and the feature importance approach we used. Section 3 provides an overview of the AWID datasets, while section 5 thoroughly discusses and compares the experiments performed on full and reduced features datasets. Section 5 also shows the dependence plots for the best ranked features. Last section 6 draws conclusions and presents future developments for this work on IDS.

2 LITERATURE REVIEW

In 2015, Kolias [14] collected a set of wireless intrusion detection datasets and made them publicly available for research. In a small office simulated environment, they perform sixteen different types of attacks, categorized in four main classes. In the same paper they report the results of performing intrusion detection using various machine learning algorithms on the AWID dataset. Manually reducing the set of features from 155 to 20 resulted in an improved performance of most classifiers, however these experiments show that instances for classes ”impersonation” and ”flooding” are harder to predict compare to ”injection” instances.

A curated subset of 36 features was used by Rezvy et al. [24, 25] to improve the overall prediction process, however, no details of the feature selection process were given in the paper. They used a combined dataset, that includes instances from both the train and test AWID datasets to improve the performance. The method is based on a two-step process. First, a stacked autoencoder (SAE) was utilised for unsupervised pre-training to remove potential noise from the dataset. Next, the output from the SAE was fed to a three layer convolutional neural network (CNN) for classification with a softmax activation layer for processing the output. To prevent overfitting the authors used dropout and batch normalization in their set up for the training process. For result, this algorithm gives an overall classification accuracy of 99.9%, with very good precision and recall.

Selecting subsets of important feature manually is a complex process, time consuming, therefore automated methods of selection or reduction are preferred in practice. In [1] Aminanto proposed a feature ranking and selection procedure, also based on SAE and tested it in conjunction with three classification methods to improve the prediction of the ”impersonation” attack class. Their method showed an overall accuracy rate of 99.91% and a false alarm rate of 0.012%. The disadvantage of their binary classification approach is that it has to be applied separately for each class in order to detect other attacks except impersonation attacks.

Most papers on AWID intrusion detection, approach the problem as a multi-class classification problem and do not address the problem that some instances are more difficult to classify especially in this highly imbalanced dataset. From the experimental results reported in previous papers we can draw the conclusion that the instances part of the ”injection” class are easier to classify in comparison with ”impersonation” and ”flooding” instances. One cause of this problem is the different distribution of training and testing datasets. Also, it looks like, especially in the testing set the ”injection” instance are linearly separable for the other classes, while ”impersonation” and ”flooding” instances overlap with ”normal” instances. One approach used by Aminanto [1] is to replace the multi-class problem with three binary classification problems, one for each attack class. This approach has the advantage of allowing a different feature selection for each class.

A series of recent research papers have developed various deep learning methods to address the intrusion detection challenge proposed by the AWID dataset. Comprehensive surveys about the applications of deep learning to cyber security [15] and intrusion detection have been recently published. Shone et al. [28] were first to apply a combination of deep and shallow autoencoders to a variate of well-known intrusion detection datasets. They proved that SAEs are capable of accurately predicting intrusions in a variety of network traffic datasets.

Thing [31] was one of the first to proposed using SAEs for intrusion detection and ran the experiments on the AWID dataset. Experiments with several different activation functions for intrusion detection problem show that to achieve optimal results the Parametric Rectified Linear Unit (PRelu) activation function works best. The proposed 2 and 3-hidden layer architectures showed good results for the ”impersonation” class, but not for the ”flooding” class (50% and 8% accuracy).

Wang et al. [34] analysed and compared multiple architectures SAEs and CNNs, using the PRelu activation function, for wireless intrusion detection. As parameters for the network architecture they used three and seven hidden layers, the hidden layers had the same large number of neurons (128, 96 and 64) and the output layer was a *softmax* regression layer. Although overall results show great

detection rate especially for the ”impersonation” and significant improvement over the results reported by Thing, the accuracy for identifying the ”flodding” attacks is still low (73%).

The ladder model proposed by Ran [22] can be categorized as a semi-supervised deep learning learning approach that combines supervised and unsupervised training. This method is based on a combination of back-propagation and SAEs. In order to improve the results and especially to correctly classify some of the difficult instances, this approach uses the AWID testing set to update the model weights in an unsupervised way. They slightly improve the results obtained by Wang et al. [34].

A overview survey of Random Forest (RF) applications for IDSs was presented by Resende in [23]. RF models are suitable for IDS because of their ability to deal with imbalanced datasets, large number of features and categorical features as well as numerical features. Another advantage of this approach is the high performance models that can be trained in shorter amount of time compared to other machine learning algorithms such as deep learning.

Another recent approach proposed by Kasongo [11] consists of an IDS implementation that combines feature selection approach with CNNs for classification. The feature selection method used here is based on the Extra Trees (ET) algorithm [7], an extension of the Random Forest (RF) algorithm [17]. Both RF and ET classification methods are iterative tree-based methods that at each iterative step have to identify one or multiple features providing the best split for the tree. In this way the features can be ranked and a feature subset selected. The weights of the trained CNN model were also adjusted by using the test instances. They compared the results obtained by running the CNN models on the full set of features with results obtained by training the CNN with a reduced set of 26 features. This approach obtained overall classification accuracy of 99.77% for the multi-class classification, however they don't specify the breakdown by attack class and this can be misleading and can't be compared with previous results. They also report better accuracy on the reduced feature set (99.77%) compared with the full feature set(98.57%).

Our approach is similar to the approach developed by by Kasongo [11], with several differences. Here we are classifying the network traffic using several tree-based classification models. The feature selection is performed using the new SHAP method. We use a smaller subset of 15 features compared to 26 features. Finally, most significant, in our approach the interaction plots show the dependencies between individual features and the Shapley values.

3 DATASETS

The Aegean WiFi Intrusion Dataset (AWID), a popular publicly available collection of datasets, contain both normal and attack real network flows [14]. The tabular dataset includes 154 features, plus the class. There are 4 classes represented, that include ”normal” and 3 types of attacks ”injection”, ”impersonation” and ”flooding”. The features mainly store MAC layer information collected from network traces using Wireshark. The data is already divided into two datasets on called training (AWID-CLS-R-Trn) and one called testing (AWID-CLS-R-Tst). In [14] it is specified that the two datasets were collected at different times. The distribution of the 4 classes in the training and testing dataset is shown in Table 1. The ratio between the number of ”normal” and ”attack” instances is 10:1 in the training dataset and 12:1 for the testing dataset.

Table 1: Data Distribution in Terms of Type of Attacks

	Normal	Injection	Impers.	Flooding
AWID-R-Trn	1,633,190	65,379	48,522	48,484
AWID-R-Tst	530,785	16,682	20,079	8,097
Total	2,371,281	82,061	68,601	56,581

3.1 Data Preprocessing

As previously mentioned, the two AWID datasets contain multiple features with different data types and value ranges. There is only one string feature, namely SSID, and all the other ones have numeric or nominal values. Features that represent MAC addresses are stored as hexadecimal values and need to be converted before the analysis. Particularly, in the training dataset there are many missing values and after converting these to zeros, many features have more than 99% zero values. After removing the mostly "zero" features, 100 heavily imbalanced features were left. While, a typical MAC address takes values in the $[-2^{31}, 2^{31} - 1]$, the typical value of subtypes (feature wlan.fc.subtype) is an integer between 0 and 12. Even if tree-based methods deal well with features representing different scales, a Min-Max scaling step can improve the running time of any kind of machine learning algorithms.

When collecting data in the real world, getting data that is unrepresentative or data that shift because of location or time changes is not uncommon. These changes and biases can have a dramatic effect on the machine learning models and result in models that lack generalization and predictability. The covariate shift [8, 32] problem refers to the change in data distribution present in the training and the test data. The other problem is class imbalance, that occurs when classes are not equally represented in the dataset. Many real-world dataset suffers from these problems and AWID dataset is just one of them. To find out which features differ the most in terms of their distribution between train and test, we can run the Kolmogorov-Smirnov test (KS test). Figure 1 show the histograms of the first five features with the highest score for the KS test. All of them have different distributions but the first two frame.time_epoch and radiotap.mactime don't even have common values between the training and the testing datasets.

A simple solution to the covariate shift is to mix the train and test file, and based on the full dataset generate new train or test sets that could be classify with reasonable accuracy. In our case combining training and the test dataset result in a total number of 2,371,281 instances. We used 20% of the entire dataset as test set, the remaining 80% of the data for training. To solve the large imbalanced problem of 10:1 ratio, we only select a number of "normal" instance equal with the number of all "attacks".

4 METHODS

4.1 Tree-based Machine Learning Methods

4.1.1 Random Forest (RF). For RF models simple decision trees are the building blocks. This algorithm [2] combines bagging ensemble methods and small decision trees for better performance. Two specific things make RF special and give it the name ”random”.

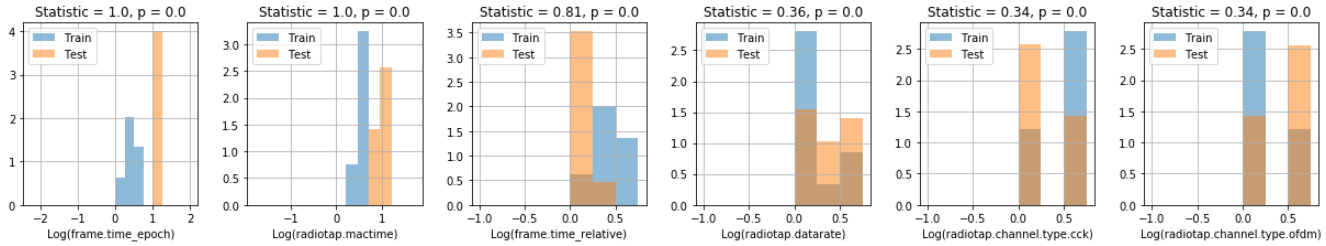

Figure 1: Feature Differences between Training and Testing Datasets

First, each subtree is build based on a random sampling of the training dataset. Second, only random subsets of features are used for calculating the splitting nodes. Therefore, is becoming very easy to evaluate the importance of the features for the overall model. The final predictions of the RF models are made by averaging the predictions over hundreds of individual shallow trees. RFs have the advantages of short computation time and interpretable models.

4.1.2 XGboost, LightGBM and Catboost. All these tree-based methods, developed in the recent years are improvements over the classical decision trees (DT). In addition to binary classification, tree-based algorithms can be used for multi-class classification, ranking and regression. They are all based on the idea of node splitting based on some impurity measure, usually calculated as entropy. The three algorithms differ by the objective function, that is minimized during gradient descend part of the algorithm. A choice of multiple pre-build metrics is available not only to measure the accuracy of the model, but to use it in the optimization process.

Standard gradient boosting methods build upon classical boosting methods, managed to solve over-fitting problems, but not very efficiently. In an effort to overcome gradient tree boosting shortcomings, Chen [4] created the XGBoost algorithm, an optimized gradient boost. XGBoost employs both LASSO and Ridge regularization techniques to minimize the over-fitting. Also, it has a built-in cross-validation, automatically deals with missing values based on the training loss and uses a Sketch algorithm to compute the split nodes. These improvements allow XGBoost to be faster and more robust during training.

The light gradient boosting machine (LGBM) [12] also improves over the standard gradient boosting methods. Developed after XGBoost, LGBM supports GPU computations, with faster training time, better accuracy, and for larger datasets with many featues. The main idea differentiating this algorithm from other tree-based algorithm is that the algorithm spends more time training the data instances with larger gradients, considered more important for the final model.

A slightly different gradient boosting method, CatBoost [6], includes a step of processing categorical features using permutation techniques and target-based statistics. This helps especially for problems with such features, where the accuracy is better compared with other tree-based methods.

4.2 Feature Ranking based on Shapley Values

The tree-based methods discussed in this paper, such as RF, XGboost, LGBM and CatBoost, are suitable to provide quick and performant

Table 2: Accuracy Metrics for the Initial Dataset

	Acc.	Prec.	Recall	F1	Time(s)
GaussNB	84.37	61.92	82.96	65.17	0
RF	99.99	99.91	100	99.95	50.5
XGBoost	99.85	98.63	99.94	99.28	241
LGBM	99.99	99.87	100	99.93	15.7
CatBoost	99.98	99.74	99.98	99.86	47.6

models for tabular datasets. These models usually comprise of large sets of shallow trees can be used for binary and multi-class classification,regression and ranking tasks. All the algorithms mentioned here need to rank the features in order to decide which one provides the best split. Therefore, computing feature importance is already engrained in these algorithms and the new method TreeExplainer proposed by Lundberg et al. [18, 19] takes advantage of that. The TreeExplainer or SHAP method, based on game theory, computes Shapley values for all the features. Shapley values work for both classification and regression and provides a consistent method to rank the features used to build the models. The ranking can be used for feature selection, while the Shapley values can help practitioners decide the cut off point (Figure 3).

In general machine learning models are considered black-boxes, that means it is hard to explain how the input is transformed into the output, by the model, during the prediction. Explaining classifications and predictions made by machine learning models is important, but not trivial. Tree-based algorithms are well-known for their interpretabilty power, which is one of their advantages beside fast training time. The SHAP dependence plots, shown in Figure 4 and Figure 5 capture the impact of one feature, wlan.da in this case, on the final classification task. Two features can be also represented in the same time, one feature (wlan.da) on the x-axsis and the other feature by coloring the individual data points based on the values of this second feature (wlan.fc.type_subtype).

5 EXPERIMENTS AND RESULTS

Before passing AWID train and test dataset to models, we ensure those datasets have balanced number of normal and intrusive data and all those data were in integer format. Then, we processed data a variety of clustering methods including Random Forest, XGBoost, CatBoost and LightGBM machine learning models. The accuracy, precision, recall and fitting time were calculated. As shown in Table 2 and Figure 2 (a), the most accurate model was found to be Random Forest, which had an accuracy of 99.99%.

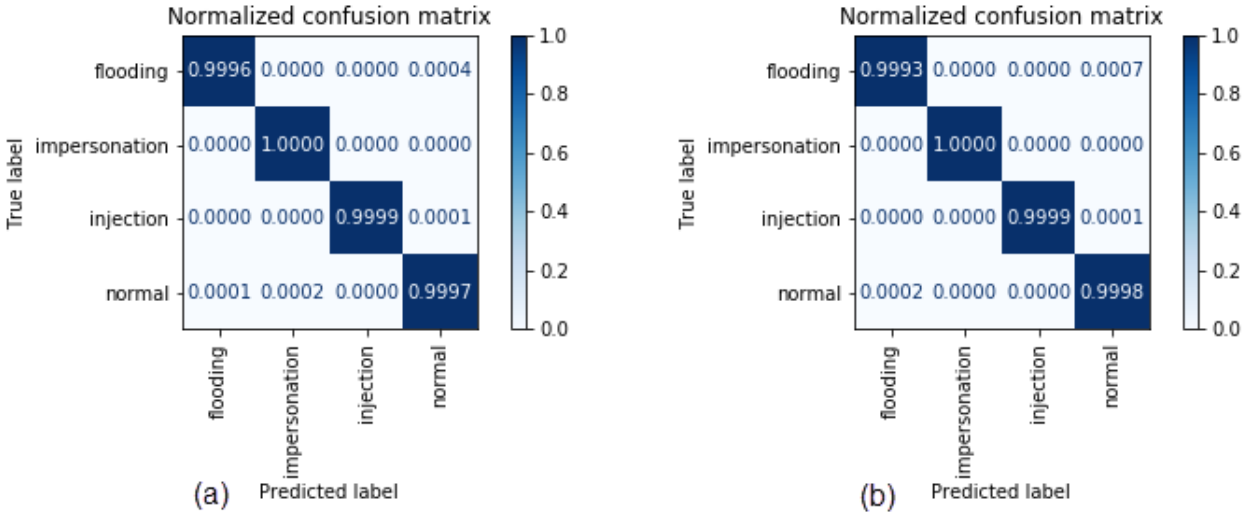

Figure 2: Normalized Confusion Matrix: (a) Initial Feature Set; (b) Reduced Feature Set.

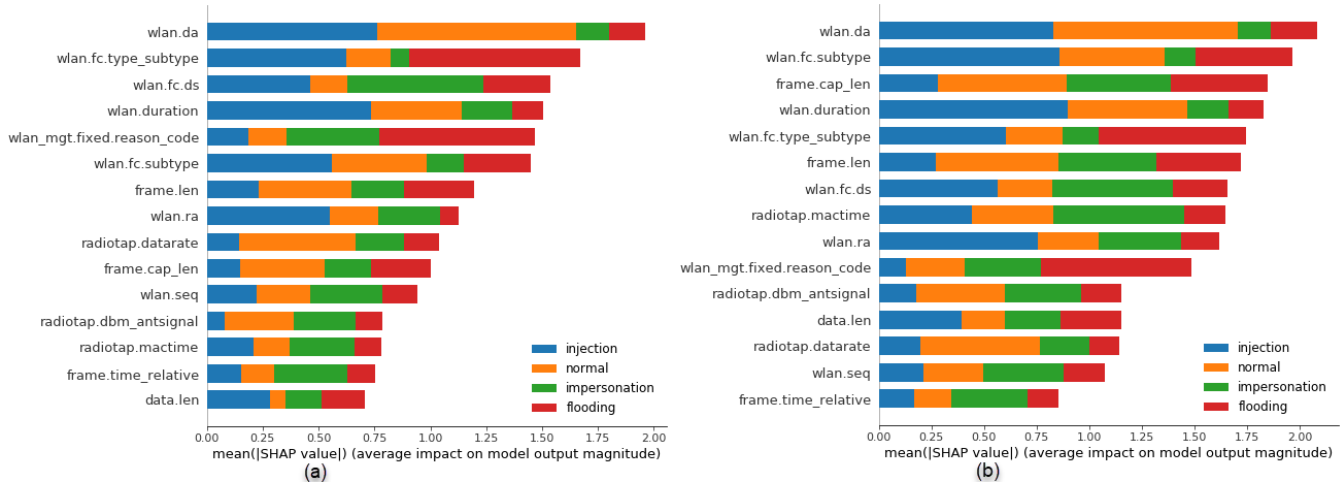

Figure 3: SHAP feature importance plots show higher importance for the injection class after feature selection relative change helps improve classification accuracy: (a) Initial Feature Set; (b) Reduced Feature Set.

Table 3: Accuracy Metrics for the Reduced Dataset

	Acc.	Prec.	Recall	F1	Time(s)
GaussNB	76.51	63.42	91.19	65	0
RF	100	99.94	100	99.97	27.1
XGBoost	99.8	98.25	99.88	99.05	61
LGBM	99.99	99.87	100	99.93	7.36
CatBoost	99.98	99.72	99.98	99.85	31.2

5.1 Feature Importance

To determine which features are the most important for a tree-based model, we calculate and plot the first 15 largest SHAP values over the training dataset. The summary plot in Figure 3(a) shows the most important features, sorted by the mean absolute value of the SHAP values for each feature over all samples. The three most important features are wlan.da, wlan.fc.type_subtype and wlan.fc.ds. The plot also shows how each feature contributes differently at classifying instances with the four classes. For example, the contribution of wlan.fc.type_subtype to classify instances for the "impersonation" class is very small in comparison with the contribution of wlan.fc.ds.

Second, we take the subset of the first 15 most important features designated by the SHAP values and repeat the classification experiments. In terms of accuracy, precision and recall the results obtain on the reduced feature subset are almost the same as the results ordained for the full feature set. As shown in Table 3, Figure 2(b) for all the tree-based methods the results are higher or similar.

For these balanced training and testing datasets, we can predict the 3 different attack classes with the same accuracy. The summary plot in Figure 3(b) shows the ranking of the features in the reduced subset. We can see that some feature become more important and the Shapley values become larger. By looking at the green bars, that represent the "impersonation" class, we can conclude that the model built on the reduced subset will make better prediction for this class.

5.2 Feature Dependence Plots

Next, we choose the most important feature wlan.da and the most important time related feature radio.tap.mactime and plotted the partial dependence plots for each class. These figures show the marginal effect that each individual variable has on the prediction outcome. For all these plots, the x-axis represents the value of one feature versus the SHAP values on the y-axis. Each dot on the plot denotes on instance. They are colored by the values on another feature, wlan.fc.type_subtype in this case. Blue represents the low values of the feature while red the high ones.

The plots tell whether the relationship between the target and the variable is linear (the data points fit on a line), monotonic (the data points fit on a monotonic function) or more complex (the points are scattered on large chunks of the plot like in Figure 5 (a)). Figure 4 (c) plot shows that there is an approximately linear and positive trend between the wlan.da and the target variable for class "injection", and wlan.da interacts with wlan.fc.type_subtype frequently. It looks like high values of wlan.da are easier to classify when the instance are from class "injection" but harder when they are from class "normal". Figure 4 (a) and (b) plots show that in general instances from the "flooding" and "impersonation" classes are harder to classify using the wlan.da feature.

Figure 5 show the dependence plots for feature radio.tap.mactime. These plots clearly show the time gap between the instances in the initial training dataset (x < .6) and the initial testing dataset (x > .9). This confirms the covariate shift existing in the initial datasets, that can be seen in Figure 1, especially for the time related features.

6 CONCLUSIONS

Based on our finding, Random Forest, XGBoost, LightGBM and CatBoost provide similar or improved results in terms of accuracy, precision and recall when applied on a reduced feature subset selected by the SHAP method. While there have not been many work on feature selection and analysis for building robust IDSs, we found wlan.da, wlan.fc_subtype and wlan.lc.ds are the features which contribute the most while the machine learning models are detecting attacks. Experiments on subset of 15 features provide similar accuracy results, but ran in less time. Knowledge of these important features can be used to remove insignificant features and also to better understand how the models work and what data should be collected in future.

In future, we plan to find better ways to attenuate the distribution differences between the training and the testing datasets. AWID has a second pair of larger training and testing datasets, 20 and 9 times as large as the AWID dataset used in this paper respectively. To validate the results presented in this paper we plan to rerun the experiments on the larger versions of the AWID dataset. We will also compare the tree-based classification results of the reduced dataset with results based on other popular classification approaches, such as stack autoencoders (SAE) and convolutional neural networks (CNN).

ACKNOWLEDGMENTS

This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center.

REFERENCES

[1] Muhamad Erza Aminanto, Rakyong Choi, Harry Chandra Tanuwidjaja, Paul D Yoo, and Kwangjo Kim. 2018. Deep Abstraction and Weighted Feature Selection for Wi-Fi Impersonation Detection. *IEEE Trans. Inf. Forensics Secur.* 13, 3 (March 2018), 621–636.

[2] Leo Breiman. 2001. Random Forests. *Mach. Learn.* 45, 1 (Oct. 2001), 5–32.

[3] Liwei Cao, Xiaoning Jiang, Yumei Zhao, Shouguang Wang, Dan You, and Xianli Xu. 2020. A Survey of Network Attacks on Cyber-Physical Systems. *IEEE Access* 8 (2020), 44219–44227.

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 785–794.

[5] Anil V Deorankar and Shiwani S Thakare. 2020. Survey on Anomaly Detection of (IoT)- Internet of Things Cyberattacks Using Machine Learning. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*. ieeexplore.ieee.org, 115–117.

[6] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. 2018. CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363* (2018).

[7] Pierre Geurts, Damien Ernst, and Louis Wehenkel. 2006. Extremely randomized trees. *Machine learning* 63, 1 (2006), 3–42.

[8] Patrick Glauner, Petko Valtchev, and Radu State. 2018. Impact of Biases in Big Data. (March 2018). arXiv:cs.LG/1803.00897

[9] Yogita Hande and Akkalashmi Muddana. 2020. A Survey on Intrusion Detection System for Software Defined Networks (SDN). *International Journal of Business Data* (2020).

[10] Elike Hodo, Xavier Bellekens, Andrew Hamilton, Christos Tachtatzis, and Robert Atkinson. 2017. Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey. (Jan. 2017). arXiv:cs.CR/1701.02145

[11] Sydney Mambwe Kasongo and Yanxia Sun. 2020. A deep learning method with wrapper based feature extraction for wireless intrusion detection system. *Comput. Secur.* 92 (May 2020), 101752.

[12] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30*, I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett (Eds.). Curran Associates, Inc., 3146–3154.

[13] Khalid Khan, Amjad Mehmood, Shafiullah Khan, Muhammad Altaf Khan, Zeeshan Iqbal, and Wali Khan Mashwani. 2020. A survey on intrusion detection and prevention in wireless ad-hoc networks. *Int. J. High Perform. Syst. Archit.* 105 (May 2020), 101701.

[14] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis. 2016. Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset. *IEEE Communications Surveys Tutorials* 18, 1 (2016), 184–208.

[15] Soman KP, Mamoun Alazab, et al. 2020. A Comprehensive Tutorial and Survey of Applications of Deep Learning for Cyber Security. (2020).

[16] Alina Lazar, Kesheng Wu, and Alexander Sim. 2018. Predicting Network Traffic Using TCP Anomalies. *Conference on Big Data (Big Data)* (2018).

[17] Andy Liaw, Matthew Wiener, et al. 2002. Classification and regression by randomForest. *R news* 2, 3 (2002), 18–22.

[18] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2019. Explainable AI for Trees: From Local Explanations to Global Understanding. (May 2019). arXiv:cs.LG/1905.04610

[19] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence* 2, 1 (Jan. 2020), 56–67.

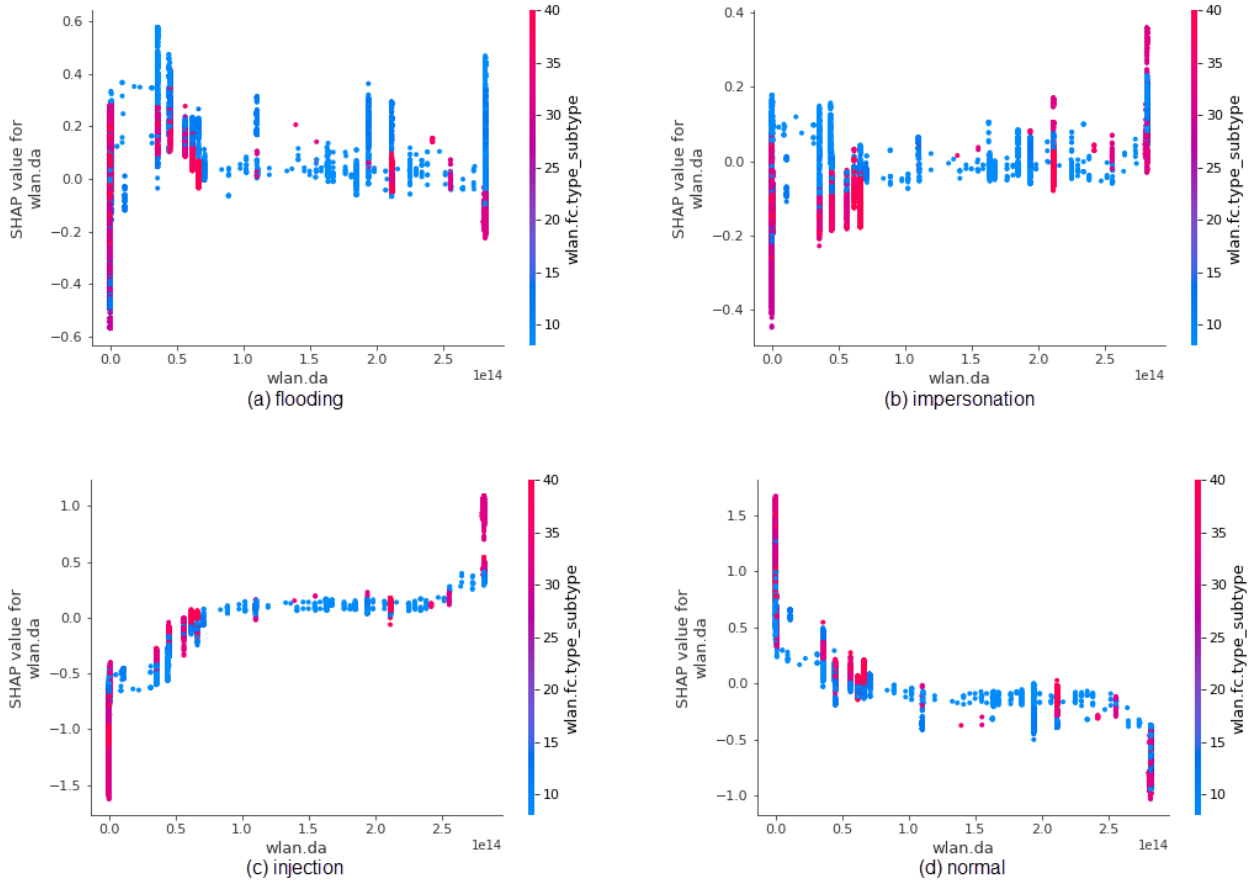

Figure 4: SHAP Individual Feature Dependence Plots for Feature wlan.da. The trends of the scatter plots (a) and (b) are quite similar to each other suggesting that it would be hard to use this feature to differentiate flooding from impersonation.

[20] Mohammad Masdari and Hemn Khezri. 2020. A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. *Appl. Soft Comput.* 92 (July 2020), 106301.

[21] Marco Mellia, Michela Meo, Luca Muscariello, and Dario Rossi. 2008. Passive analysis of TCP anomalies. *Computer Networks* 52, 14 (2008), 2663–2676.

[22] Jing Ran, Yidong Ji, and Bihua Tang. 2019. A Semi-Supervised Learning Approach to IEEE 802.11 Network Anomaly Detection. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. 1–5.

[23] Paulo Angelo Alves Resende and André Costa Drummond. 2018. A Survey of Random Forest Based Methods for Intrusion Detection Systems. *ACM Comput. Surv.* 51, 3, Article Article 48 (May 2018), 36 pages. https://doi.org/10.1145/3178582

[24] Shahadate Rezvy, Yuan Luo, Miltos Petridis, Aboubaker Lasebae, and Tahmina Zebin. 2019. An efficient deep learning model for intrusion classification and prediction in 5G and IoT networks. In *2019 53rd Annual Conference on Information Sciences and Systems (CISS)*. 1–6.

[25] Shahadate Rezvy, Miltos Petridis, Aboubaker Lasebae, and Tahmina Zebin. 2019. Intrusion Detection and Classification with Autoencoded Deep Neural Network. In *Innovative Security Solutions for Information Technology and Communications*. Springer International Publishing, 142–156.

[26] Gaganjot Kaur Saini, Malka N Halgamuge, Pallavi Sharma, and James Stephen Purkis. 2020. A Review on Cyberattacks: Security Threats and Solution Techniques for Different Applications. In *Cyber Warfare and Terrorism: Concepts, Methodologies, Tools, and Applications*. IGI Global, 98–126.

[27] Chris Sanders. 2017. *Practical packet analysis: Using Wireshark to solve real-world network problems*. No Starch Press.

[28] Nathan Shone, T N Ngoc, V D Phai, and Q Shi. 2018. A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions on Emerging Topics in Computational Intelligence* 2, 1 (Feb. 2018), 41–50.

[29] Nasrin Sultana, Naveen Chilamkurti, Wei Peng, and Rabei Alhadad. 2019. Survey on SDN based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications* 12, 2 (March 2019), 493–501.

[30] Astha Syal, Alina Lazar, Jinoh Kim, Alexander Sim, and Kesheng Wu. 2019. Automatic detection of network traffic anomalies and changes. *of the ACM Workshop on Systems …* (2019).

[31] Vrizlynn L. L. Thing. 2017. IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. ieeexplore.ieee.org, 1–6.

[32] Ryan J Tibshirani, Rina Foygel Barber, Emmanuel Candes, and Aaditya Ramdas. 2019. Conformal Prediction Under Covariate Shift. In *Advances in Neural Information Processing Systems 32*, H Wallach, H Larochelle, A Beygelzimer, F dAlché-Buc, E Fox, and R Garnett (Eds.). Curran Associates, Inc., 2530–2540.

[33] Putra Wanda and Huang Jin Jie. 2020. A Survey of Intrusion Detection System. *International Journal of Informatics and Computation* 1, 1 (Jan. 2020), 1–10.

[34] Shaoqian Wang, Bo Li, Mao Yang, and Zhongjiang Yan. 2019. Intrusion Detection for WiFi Network: A Deep Learning Approach. In *Wireless Internet*. Springer International Publishing, 95–104.

[35] Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. 2017. A survey of intrusion detection in Internet of Things. *Journal of Network and Computer Applications* 84 (April 2017), 25–37.

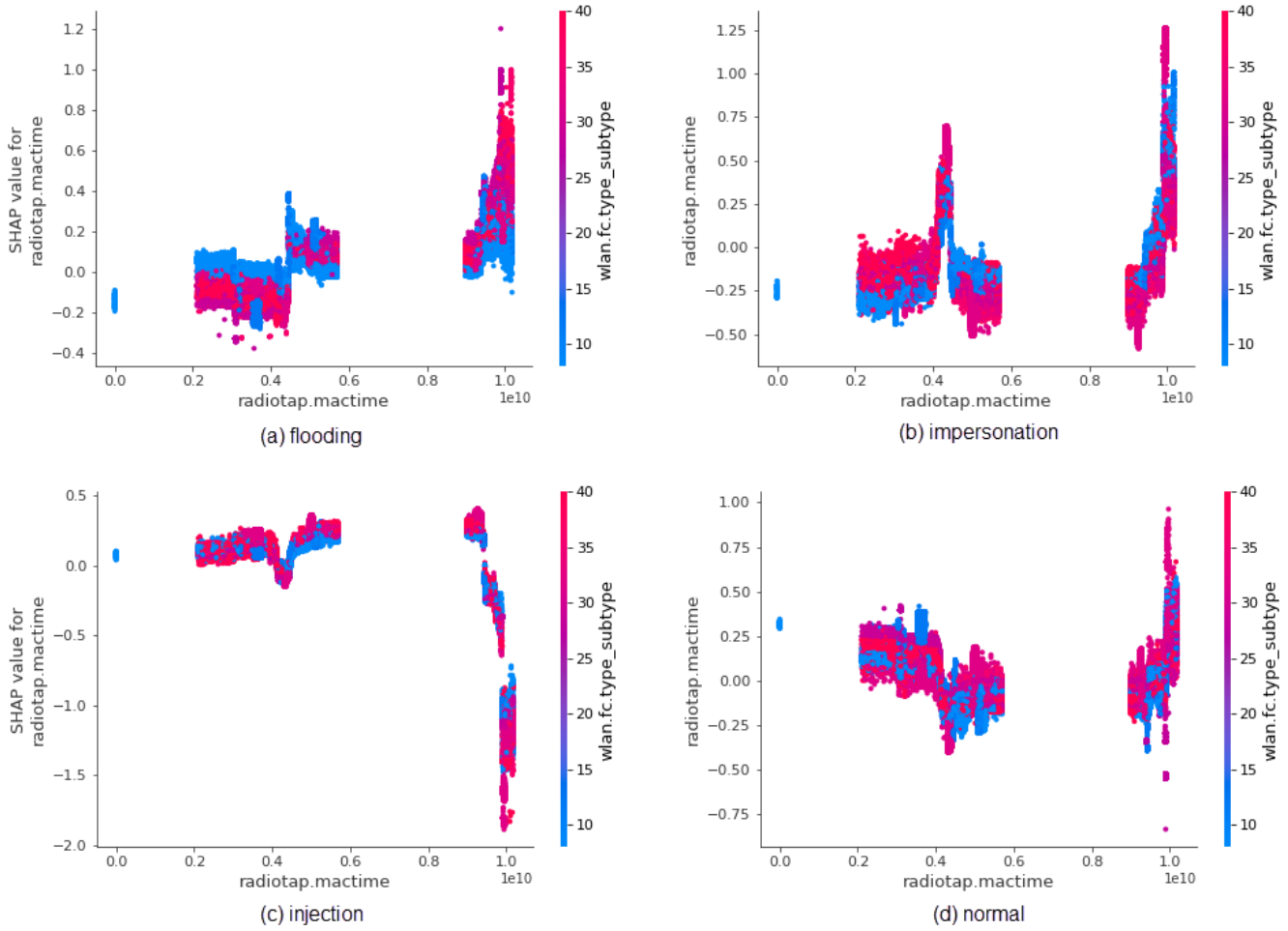

Figure 5: SHAP Individual Feature Dependence Plots for Feature radiotap.mactime. Since the scatter plots for (a) and (b) are distinctive (especially around 0.4E10) this feature increases the likelihood that clustering methods could differentiate flooding from impersonation.