# ML-based Novelty Detection and Classification of Security Threats in IoT Networks

Marcelo V. C. Aragão, Gabriel P. Ambrósio, Felipe A. P. de Figueiredo

***Abstract*— This article presents a practical evaluation of machine-learning models to detect novelties and classify threats in IoT networks using an ML-based approach. Given the escalating significance of analyzing network traffic amidst the proliferation of devices and sensitive data exchange, this research holds significant relevance. The IoT Network Intrusion dataset was chosen for experimentation, followed by data processing and imbalance handling techniques. Four distinct models encompassing novelty detection and classification were trained, allowing for an in-depth comparison of their performance in terms of accuracy and time. Notably, after attaining the results, it was evident that these models achieved remarkably high accuracy in novelty detection and classification tasks, emphasizing that techniques based on machine learning can be successfully applied to this context.**

***Keywords*— Machine Learning, IoT, Network Traffic Analysis, Novelty Detection, Classification.**

## I. Introduction

Detecting network traffic novelties has become crucial with the rapid increase in connected devices and the exchange of sensitive data over networks [1]. This article introduces a novel approach that combines supervised and unsupervised machine learning (ML) techniques to address this challenge. By training ML models on a real-world IoT-network-based dataset encompassing novelty detection and classification, this work aims to develop a robust method for accurately identifying and categorizing novel patterns in network traffic data.

The objectives are twofold: conduct a comprehensive analysis of network traffic data and construct an ML model capable of effectively detecting and classifying novelties within this data.

This article contributes to the field by providing a theoretical background, reviewing related literature, presenting a novel approach that combines supervised and unsupervised techniques, and reporting the results of rigorous experiments.

## II. Theoretical Review

The proposed solution presented in this paper involves various fields of study, including ML, Novelty Detection, IoT Security, and Network Traffic Analysis. In the following sections, a brief theoretical overview of each of these topics is provided.

Marcelo V. C. Aragão, Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí-MG, e-mail: marcelovca90@inatel.br; Gabriel P. Ambrósio, Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí-MG, e-mail: gabriel.pivoto@inatel.br; Felipe A. P. de Figueiredo, Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí-MG, e-mail: felipe.figueiredo@inatel.br;

### A. Machine Learning

Artificial Intelligence (AI) is a broad field encompassing techniques enabling computers to imitate human behavior and solve complex problems. However, it is limited by the fact that humans often cannot articulate all the implicit knowledge required for performing intricate tasks. ML overcomes this limitation by utilizing algorithms that iteratively learn from specific training data, allowing computers to uncover complex patterns and hidden insights without the need for explicit programming [2].

### B. Novelty Detection

Novelty detection and outlier detection can be understood as specialized subcategories of anomaly detection. Anomaly detection identifies deviations from expected patterns within a dataset, encompassing novelties (unseen patterns) and outliers (extreme values). Novelty detection focuses on identifying samples that significantly differ from the training data, while outlier detection targets data points that deviate notably from the dataset's majority. This work emphasizes novelty detection, representing a noteworthy learning paradigm recently attracting considerable attention from the research community [3]. The models used in this study are:

- Elliptic Envelope: This method detects novelties by fitting a robust covariance estimator to the data without being affected by outliers. It uses Mahalanobis distances to measure outlyingness, focusing on central data points, disregarding outliers outside the central mode, and reliably identifying novelties in a dataset [4].
- Isolation Forest: This algorithm detects anomalies by calculating an anomaly score for each sample using a tree-based approach. It isolates observations by randomly selecting features and split values, measuring the path length from the root node to the terminating node in the tree structure. Anomalies have shorter path lengths due to random partitioning, indicating a high likelihood of anomalies when multiple trees produce shorter path lengths for specific samples [5] [6].
- Local Outlier Factor (LOF): This method assigns a local outlier factor (LOF) to each object in a dataset based on its isolation compared to its local neighborhood. The LOF measures the difference in density between the object and its neighborhood, with high LOF values indicating likely outliers and low LOF values indicating likely regular objects within their local neighborhood. High LOF values suggest low-density neighborhoods and a higher potential for being an outlier [7].

- SGD One-Class Support Vector Machine (SVM): It is an unsupervised outlier detection model that estimates the support of a high-dimensional distribution through a Stochastic Gradient Descent (SGD) optimization. It identifies abnormalities by constructing a frontier in an embedding space and considering observations outside it as abnormal. The choices of the kernel and scalar parameter are necessary to define the frontier, with the radial basis function (RBF) kernel commonly used despite the absence of an exact formula or algorithm to set its bandwidth parameter [8].

### C. IoT Security

The Internet of Things (IoT) comprises a growing number of interconnected devices that exchange data over the Internet. Ensuring device security is challenging due to weak passwords and inadequate authentication measures. Default or easy-to-guess passwords on IoT devices make them vulnerable to hackers, risking privacy and enabling large-scale attacks. Securing interconnected IoT devices is vital for protecting consumer privacy, critical infrastructure, and websites [9].

### D. Network Traffic Analysis

Network Traffic Analysis (NTA) is essential in networking, particularly with new networks like IoT. NTA techniques, including anomaly/novelty detection, traffic classification, fault management, and prediction, assess network security, QoS, and resource use. Techniques can be active/passive. However, rapid growth poses daily challenges in data handling, integration, security, and more, with most research addressing specific NTA aspects [10].

### E. Classification

Classification is a supervised learning technique in data mining, assigning items to categories. Its primary goal is to predict the target class for each example in the dataset, with no inherent order among the classes [11]. The study uses the following models:

- Decision Tree (DT): They are non-parametric models for classification and regression tasks. They extract simple decision rules from data features to predict the target variable. Decision Trees segment the feature space into distinct regions, approximating the underlying relationship. Such models can also handle multi-class scenarios. Training the classifier requires input arrays: **X**, representing training samples in sparse or dense format, and **Y**, an array of integer values representing the labels of the training samples [12].
- LightGBM (LGBM): It is a high-speed model suitable for large datasets with more than 10,000 values. It stands out from other classifiers due to its lower memory usage. In addition, LGBM prioritizes accuracy over other factors and grows trees vertically, which reduces loss compared to horizontal growth used by other boosting algorithms [13].
- Random Forest (RF): This ensemble algorithm uses multiple decision tree classifiers, each trained on different subsets of the dataset, to enhance predictive accuracy and reduce overfitting through averaging predictions. The size of each subset is determined by the *max_samples* parameter, adjustable for flexibility. When *bootstrap=True* (default), subsets are created through bootstrapping; when *bootstrap=False*, the entire dataset is used for building each tree. This approach creates a robust and versatile model for diverse classification tasks [14].
- eXtreme Gradient Boosting (XGBoost): It is a highly scalable ML system for tree boosting that excels in speed and scalability. It outperforms existing solutions more than tenfold on a single machine and can handle billions of examples in distributed or memory-limited settings. This scalability is achieved through the system and algorithmic optimizations, including a tree learning algorithm designed for sparse data and a weighted quantile sketch procedure. Parallel and distributed computing techniques accelerate learning, enabling rapid model exploration [15].

## III. Literature Review

This section presents other articles and studies concerning anomaly detection in network traffic using ML.

Vikram and Mohana [16] implemented an unsupervised ML project for network traffic anomaly detection, using the Isolation Forest algorithm for anomaly detection and the One-Class SVM as the classifier. Extensive preprocessing handled dataset size and imbalance, leading to a 98.3% AUC score. Key considerations include data quality, the importance of the *contamination* parameter (set at 4%), and scalability. They propose improvements: feature normalization, combining ML algorithms, and incorporating deep learning. Combining supervised and unsupervised ML could enhance results, while parallelization improves performance of real-time data handling and response suggestions.

Hwang et al. [17] introduces D-PACK, an efficient unsupervised deep learning approach for network traffic anomaly detection, specifically focusing on the initial two packets of each flow to enhance efficiency by reducing packet capture and analysis overhead. The method achieves nearly 100% accuracy in detecting malicious traffic, with false negative and false positive rates below 1%, highlighting the benefits of the USTC-TFC2016 dataset. Minimal packet and byte examination contribute to faster detection. The article calls for additional research to optimize deep learning for real-time anomaly detection with minimal delay.

Zhang and Zulkernine [18] addresses training data challenges in Network Intrusion Detection Systems (NIDSs) and proposes using random forests for anomaly-based NIDSs to detect intrusions by identifying outliers. The study modified the outlier detection algorithm for computational efficiency by assuming unique patterns for each network service's everyday activities. Experimental results on the KDD'99 dataset show decreased performance of unsupervised systems with increased attack connections. The authors recommend integrating anomaly-based and misuse-based approaches to enhance NIDS performance.

Based on a literature review, the articles concur on detecting novel instances as outliers or anomalies. This research paper incorporates novelty detection and network traffic classification. A notable contribution is the comparative analysis of novelty detection and classification models, distinguishing it from other single-model-focused articles.

## IV. Proposed Methodology

This work aims to detect novel threats and classify known attacks through an ML-based network traffic analysis. For this, novelty detection models and classification models were trained and compared. The proposed methodology for this project entails several vital stages, which are depicted in Figure 1 and described in sequence.

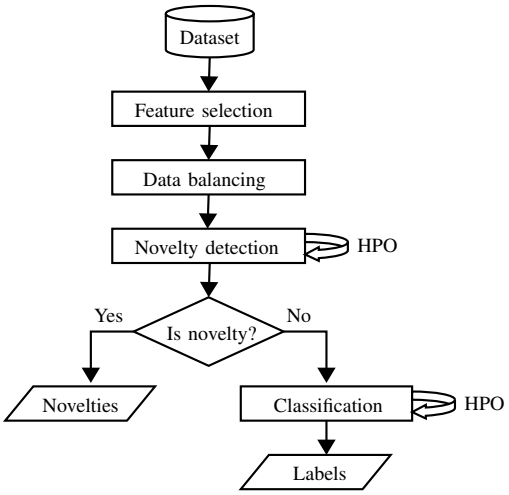


Fig. 1: Proposed methodology.

- **Dataset**: The IoT Network Intrusion Dataset [19] is an aggregate of real-world IoT network traffic captures. It contains nearly 3 million samples among five scenarios (including benign and malicious traffic), subsequently divided into subcategories such as ARP spoofing and SYN flooding. This work used a fraction (10%) of a pre-processed version of the dataset, where only the categories (and not subcategories) of the samples were considered. The distribution of the classes in this reduced version is Normal (53.8%), Mirai Botnet (39.0%), Man-in-the-Middle (MITM) (3.8%), Denial of Service (DoS) (2.4%), and Scanning (0.9%). The Pandas [20] library was used to load the dataset into memory.
- **Feature Selection**: The FeatureWiz library [21] attempts to improve model performance by analyzing the dataset and selecting essential features. This reduces dimensionality, prevents overfitting, and enhances classification accuracy. The algorithm has two stages: SULoV (Searching for Uncorrelated List of Variables), which identifies highly correlated variables and calculates the Mutual Information Score (MIS) to choose the most relevant pairs, and RFE (Recursive Feature Elimination), which recursively feeds the selected variables through and XGBoost model to identify the best features while skipping similar data.
- **Balancing**: Over-sampling and under-sampling methods were employed to balance the dataset. The over-sampling aspect is handled by SMOTE (Synthetic Minority Oversampling Technique) [22], which generates new samples for under-represented categories. SMOTE offers three additional options to generate samples, specifically targeting instances near the decision function boundary. These methods generate samples in the direction opposite to the nearest neighbors' class. Tomek's Links [23] was incorporated into the pipeline after applying SMOTE over-sampling to remove less relevant samples. This step is essential for classification because an imbalanced dataset can lead to biased models and, consequently, misleading results. It was performed using the imbalanced-learn [24] library.
- **Hyperparameter Optimization (HPO)**: This step aims to optimize model performance by exploring various combinations of hyperparameters for novelty detection and threat classification. To achieve this, the Optuna [25] library is utilized. The process involves configuring the hyperparameter space for each model and creating and running a study that evaluates different combinations. Promising trials are expanded, while less relevant or invalid trials are pruned using a TPE (Tree-structured Parzen Estimator) [26] sampler.
- **Novelty Detection**: This step aims to verify whether the anomaly detection models, presented in section II-D, can identify samples that characterize an unknown type of traffic (i.e., a "novelty"). For this, different scenarios were considered. In each one of them, the models were trained with only one traffic class (considered as "normal"). They were used to classify the samples of the other classes (considered as "novelties"). The models were provided by the scikit-learn [27] library.
- **Classification**: This step aims to verify whether the classification models presented in section II-E can classify samples among known types of traffic. For this, different scenarios were considered, and in each one of them, the models were trained with "all but one" traffic classes since the class that was left out is precisely the one used in the training of the novelty detection models. The models were also provided by the scikit-learn [27] library.

## V. Experiments and Discussion

The experiments were conducted on a virtual machine with Intel® Xeon™ E5-2650L v4 processor, 896 GB of RAM, and Windows 10 Education operating system version 22H2. Each scenario underwent 200 trials of hyperparameter optimization to refine the model's detection/classification accuracy and evaluate its robustness/sensitivity with different hyperparameter configurations.

The experimental results were systematically organized into tables for analysis. Tables I and II present the outcomes of the studied novelty detection models. The former includes statistical measures like mean, standard deviation, and maximum accuracy values for the trained models, while the latter indicates training and test durations.

TABLE I: Novelty detection accuracy.

| Scenario | Elliptic Envelope | | Isolation Forest | | Local Outlier Factor | | SGD One-Class SVM | |
|---|---|---|---|---|---|---|---|---|
| | Mean ± SD | Max | Mean ± SD | Max | Mean ± SD | Max | Mean ± SD | Max |
| DoS | 0.557 ± 0.494 | **1.000** | 0.594 ± 0.465 | **1.000** | 0.109 ± 0.105 | 0.263 | 0.620 ± 0.485 | **1.000** |
| Mirai | 0.747 ± 0.293 | 0.991 | 0.913 ± 0.087 | 0.993 | 0.414 ± 0.128 | 0.574 | 0.650 ± 0.477 | **1.000** |
| MITM | 0.391 ± 0.395 | **1.000** | 0.074 ± 0.093 | 0.474 | 0.208 ± 0.119 | 0.330 | 0.740 ± 0.439 | **1.000** |
| Scan | 0.537 ± 0.466 | **1.000** | 0.828 ± 0.238 | 0.949 | 0.601 ± 0.311 | 0.894 | 0.595 ± 0.491 | **1.000** |

TABLE II: Novelty detection total (training and test) time.

| Scenario | Elliptic Envelope | | Isolation Forest | | Local Outlier Factor | | SGD One-Class SVM | |
|---|---|---|---|---|---|---|---|---|
| | Mean ± SD | Min | Mean ± SD | Min | Mean ± SD | Min | Mean ± SD | Min |
| DoS | 01:36 ± 00:29 | 00:49 | 00:14 ± 00:27 | **00:00** | 32:17 ± 57:34 | 00:07 | 00:06 ± 00:01 | 00:03 |
| Mirai | 00:57 ± 00:29 | 00:27 | 00:34 ± 00:50 | **00:00** | 26:06 ± 34:12 | 00:05 | 00:04 ± 00:01 | 00:02 |
| MITM | 01:44 ± 00:31 | 00:37 | 00:06 ± 00:19 | **00:00** | 49:14 ± 62:25 | 00:06 | 00:05 ± 00:01 | 00:02 |
| Scan | 01:17 ± 00:26 | 00:38 | 00:24 ± 00:31 | **00:00** | 37:08 ± 61:26 | 00:07 | 00:05 ± 00:01 | 00:02 |

TABLE III: Classification accuracy.

| Scenario | Decision Tree | | LightGBM | | Random Forest | | XGBoost | |
|---|---|---|---|---|---|---|---|---|
| | Mean ± SD | Max | Mean ± SD | Max | Mean ± SD | Max | Mean ± SD | Max |
| DoS | 0.875 ± 0.141 | **0.969** | 0.765 ± 0.241 | 0.954 | 0.917 ± 0.054 | 0.957 | 0.947 ± 0.008 | 0.957 |
| Mirai | 0.851 ± 0.162 | **0.969** | 0.781 ± 0.243 | 0.953 | 0.917 ± 0.057 | 0.958 | 0.948 ± 0.008 | 0.957 |
| MITM | 0.864 ± 0.152 | **0.967** | 0.798 ± 0.213 | 0.953 | 0.916 ± 0.060 | 0.959 | 0.947 ± 0.008 | 0.956 |
| Scan | 0.865 ± 0.154 | **0.966** | 0.776 ± 0.242 | 0.953 | 0.916 ± 0.057 | 0.957 | 0.947 ± 0.008 | 0.955 |

TABLE IV: Classification total (training and test) time.

| Scenario | Decision Tree | | LightGBM | | Random Forest | | XGBoost | |
|---|---|---|---|---|---|---|---|---|
| | Mean ± SD | Min | Mean ± SD | Min | Mean ± SD | Min | Mean ± SD | Min |
| DoS | 00:00 ± 00:00 | **00:00** | 00:27 ± 00:12 | 00:01 | 05:03 ± 02:44 | 00:28 | 02:26 ± 00:34 | 00:59 |
| Mirai | 00:00 ± 00:00 | **00:00** | 00:24 ± 00:12 | 00:02 | 02:49 ± 01:31 | 00:20 | 02:26 ± 00:41 | 00:55 |
| MITM | 00:00 ± 00:00 | **00:00** | 00:25 ± 00:12 | 00:02 | 03:13 ± 02:01 | 00:21 | 02:22 ± 00:38 | 00:45 |
| Scan | 00:00 ± 00:00 | **00:00** | 00:24 ± 00:11 | 00:01 | 04:06 ± 02:35 | 00:29 | 02:35 ± 00:40 | 00:54 |

Similarly, Tables III and IV pertain to the classification models, serving a similar purpose to novelty detection ones but focusing on this specific category. Both sets of tables facilitate a comparative analysis of the models across different scenarios. Each scenario involves variations in the target value of the dataset, including DoS, Mirai, MITM, and Scan.

A lower standard deviation implies reduced sensitivity to hyperparameters' values, indicating that a model with a low standard deviation produces satisfactory results regardless of the hyperparameters used. Tables I and III show that specific models yield nearly identical outcomes. SGD One-Class SVM and Elliptic Envelope exhibit remarkably similar results, while all models perform well in the latter.

Tables II and IV show the training and test times for novelty detection and classification models, respectively, for easy comparison.

Although the training and test times may seem negligible, it is worth noting that SGD One-Class SVM emerges as the superior novelty detection model. It demonstrates exceptional detection accuracy while requiring minimal time for both training and testing, in contrast to Elliptic Envelope, which also exhibits high detection accuracy but demands more time.

All classification models demonstrate commendable accuracy. The DT model stands out for its high accuracy, fast training and testing times, and moderate tolerance to hyperparameter variations. In contrast, LightGBM is hyperparameter-sensitive and less precise, while XGBoost is slower but more robust. Interestingly, the obtained classification results align with Aragão, Mafra, and Figueiredo [1], where a DT proved to be the fastest and most accurate model for threat identification, despite using a different dataset.

In summary, the SGD One-Class SVM would be recommended for novelty detection. At the same time, the DT model would be a suitable choice for classification due to its high accuracy and low computation times, enabling fast hyperparameter optimization to address the sensitivity issue. It is important to note that while the proposed technique shows strong performance on this dataset, it's crucial to assess its effectiveness and robustness across diverse datasets.

The code and instructions for reproducing the experiments and all the results obtained are available in the following GitHub repository: `https://github.com/GabrielPivoto/iot-detection-and-classification`.

## VI. Conclusion

The work aimed to compare four novelty detection and four classification models for network traffic analysis using the IoT Network Intrusion dataset. The following libraries were used in this work: Pandas (dataset reading), FeatureWiz (feature selection), imbalanced-learn (dataset balancing), Optuna (hyperparameter optimization), and scikit-learn (detection and classification models).

Results show that SGD One-Class SVM performed best for novelty detection, while Decision Tree was the top classification model.

This work stands out by comparing multiple models and differentiating between novelty detection and outlier/anomaly detection, providing clarity on their differences.

The proposed methodology provides a valuable foundation for network traffic analysis and security management tools. Future improvements can enhance novelty detection by exploring new scenarios and models. This study contributes to ML-based network security and sets the stage for further research in novelty detection.

## Acknowledgements

This work was partially funded by FAPEMIG via grant no. 2070.01.0004709/2021-28; by Huawei, under the project Advanced Academic Education in Telecommunications Networks and Systems, Grant No. PPA6001BRA23032110257684; by CNPq under Grant Nos. 313036/2020-9 and 403827/2021-3; by FAPESP under Grant No. 2021/06946-0; by CAPES and RNP, with resources from MCTIC, under Grant Nos. 01250.075413/2018-04, 01245.010604/2020-14, and 01245.020548/2021-07 under the Brazil 6G project of the Radiocommunication Reference Center (CRR) of INATEL, Brazil. The authors also thank Prof. Dr. Guilherme P. Aquino (INATEL) for providing the computational resources to run the experiments.

## References

[1] M. V. C. Aragão, S. Mafra, and F. A. P. de Figueiredo. "Análise de Tráfego de Rede com Machine Learning para Identificação de Ameaças a Dispositivos IoT". In: *Anais do XL Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*. Sociedade Brasileira de Telecomunicações, 2022.

[2] C. Janiesch, P. Zschech, and K. Heinrich. "Machine learning and deep learning". In: *Electronic Markets* 31.3 (Sept. 2021), pp. 685–695. ISSN: 1019-6781, 1422-8890.

[3] M. A. F. Pimentel et al. "A review of novelty detection". In: *Signal Processing* 99 (2014), pp. 215–249. ISSN: 0165-1684.

[4] P. J. Rousseeuw and K. V. Driessen. "A Fast Algorithm for the Minimum Covariance Determinant Estimator". In: *Technometrics* 41.3 (Aug. 1999), pp. 212–223.

[5] F. T. Liu, K. M. Ting, and Z.-H. Zhou. "Isolation Forest". In: *2008 Eighth IEEE International Conference on Data Mining*. 2008, pp. 413–422.

[6] F. T. Liu, K. M. Ting, and Z.-H. Zhou. "Isolation-Based Anomaly Detection". In: *ACM Transactions on Knowledge Discovery from Data* 6.1 (Mar. 2012), pp. 1–39.

[7] M. M. Breunig et al. "LOF: identifying density-based local outliers". In: *ACM SIGMOD Record* 29.2 (June 2000), pp. 93–104. ISSN: 0163-5808.

[8] B. Schölkopf et al. "Estimating the Support of a High-Dimensional Distribution". In: *Neural Computation* 13.7 (July 2001), pp. 1443–1471.

[9] R. Ahmad and I. Alsmadi. "Machine learning approaches to IoT security: A systematic literature review". In: *Internet of Things* 14 (2021), p. 100365. ISSN: 2542-6605.

[10] M. Abbasi, A. Shahraki, and A. Taherkordi. "Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey". In: *Computer Communications* 170 (2021), pp. 19–41. ISSN: 0140-3664.

[11] N. S. Ahmed and M. Hikmat Sadiq. "Clarify of the Random Forest Algorithm in an Educational Field". In: *2018 International Conference on Advanced Science and Engineering (ICOASE)*. 2018, pp. 179–184.

[12] L. Breiman et al. *Classification and regression trees*. 1st ed. June. Boca Raton, FL, USA: Chapman & Hall/CRC, 1984, p. 358. ISBN: 978-0412048418.

[13] K. S. Naik. "Predicting Credit Risk for Unsecured Lending: A Machine Learning Approach". In: (2021).

[14] D. H. Wolpert. "Stacked generalization". In: *Neural Networks* 5.2 (1992), pp. 241–259. ISSN: 0893-6080.

[15] T. Chen and C. Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, 785–794. ISBN: 9781450342322.

[16] A. Vikram and Mohana. "Anomaly detection in Network Traffic Using Unsupervised Machine learning Approach". In: *2020 5th International Conference on Communication and Electronics Systems (ICCES)*. 2020, pp. 476–479.

[17] R.-H. Hwang et al. "An Unsupervised Deep Learning Model for Early Network Traffic Anomaly Detection". In: *IEEE Access* 8 (2020), pp. 30387–30399.

[18] J. Zhang and M. Zulkernine. "Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection". In: *2006 IEEE International Conference on Communications*. Vol. 5. 2006, pp. 2388–2393.

[19] H. Kang et al. *IoT network intrusion dataset*. 2019. URL: `https://ieee-dataport.org/open-access/iot-network-intrusion-dataset`.

[20] T. pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020.

[21] R. Seshadri. *AutoViML/featurewiz: Use advanced feature engineering strategies and select best features from your data set with a single line of code.* URL: `https://github.com/AutoViML/featurewiz`.

[22] N. V. Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[23] I. Tomek. "Two Modifications of CNN". In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-6.11 (1976), pp. 769–772.

[24] G. Lemaître, F. Nogueira, and C. K. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning". In: *Journal of Machine Learning Research* 18.17 (2017), pp. 1–5.

[25] T. Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[26] J. Bergstra et al. "Algorithms for hyper-parameter optimization". In: *Advances in neural information processing systems* 24 (2011).

[27] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.