Universiteit Leiden
The Netherlands

Opleiding Informatica

Impact of Flow Anonymization on Cyberattack Detection in IoT

Jack Voorham

Supervisors:
prof. dr. Marco Spruit
Max van Haastrecht, MSc

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

12/07/2022

Abstract

Intrusion Detection Systems (IDS) are crucial in networking as they can detect malicious intent that other mechanisms such as firewalls fail to recognize. A technique commonly used in an IDS to detect malicious intent is the use of flow data. However, flow data often contains personal identifiers such as IP addresses that can be considered privacy-sensitive. The privacy sensitivity of these fields makes it hard to share these logs in settings such as collaborative intrusion detection systems (CIDS) or open-source threat intelligence models. This paper will assess how an IDS in an IoT setting performs when privacy is taken into consideration using various anonymization techniques, this is examined in both a binary setting and a multi-class setting with multiple attack types. We show that when we consider the sensitivity of data in a multi-class setting, the F1 score of our best baseline model drops by 12.46% when applying binning anonymization and 22.47% when applying the more conservative black-marker anonymization. Subsequently, in the binary classification setting, we show that anonymization has only a slight effect, with F1 scores dropping 2.90% when applying binning anonymization and 5.46% when applying black-marker anonymization.

Contents

1 Introduction

Cyberattacks have increased rapidly in volume, complexity, and diversity in the past few years. A recent report from Accenture [1] stated that cyberattack frequency on companies increased by 31 percent in 2021 with respect to the year prior. The same report also stated that 82 percent of company budgets for cybersecurity have increased. Additionally, attack trends and complexity are constantly changing in the cybersecurity landscape [2]. A recent trend on a threat actor level is COVID-19-related malware and phishing [3]. Also, some sectors can become more prevalent as attack targets. Figure 1 shows that attacks on critical sectors of society such as education, government, and healthcare have been increasing since 2019. This changing landscape makes it challenging for entities focused on information security to mitigate risk as mitigation strategies need constant evaluation on both a threat-actor and industry level.

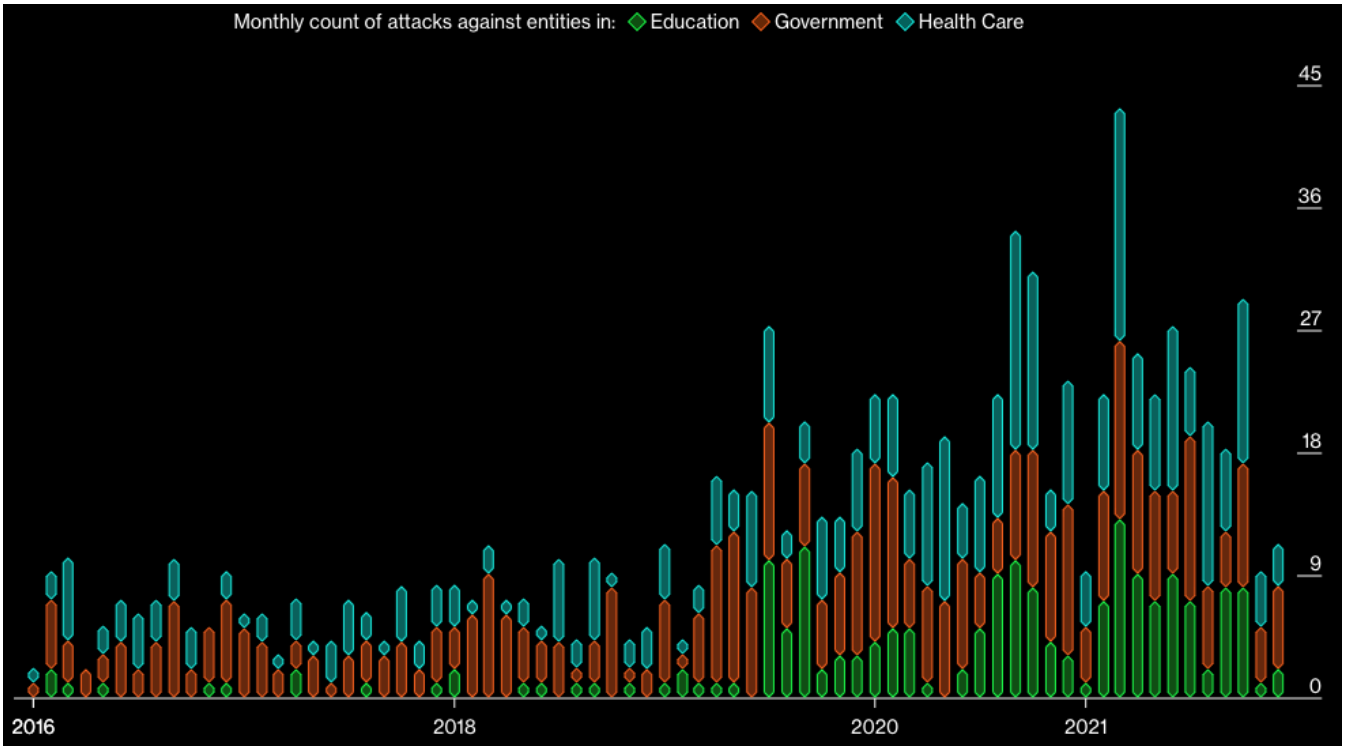

Figure 1: Monthly attacks against education, government, and health care from the period 2016-2022. [4]

Another emerging trend is cyberattacks on Internet of Things (IoT) devices [5]. We can define IoT as all devices that on them contain hardware, software, and sensors that make it possible to share information with other devices and systems. Examples of IoT can be as simple as a lightbulb or thermostat but can also be as critical as a driverless bus or entire cities [6]. The ENISA 2021 threat landscape [3] states that IoT in conjunction with 5G is accelerating attack volume, with IoT devices becoming more predisposed to DDoS attacks. Subsequently, it is shown that 98 percent of the IoT data is unencrypted and that most IoT devices run outdated software, increasing exploitation risks [7]. This combination of IoT being a vulnerable target and the potential impact of a successful cyberattack makes it easy to see that IoT can be a hotbed for malicious intent.

1.1 Problem statement

As cyberattacks are becoming more advanced and sophisticated and with new attack (zero-day) exploitation becoming more common [8], Intrusion Detection Systems (IDS) also are posed the challenge of needing to detect situations not currently known. To combat this, most IDS use forms of alert logs and network traffic to collect data and then extract needed data such as Internet Protocol (IP) addresses for usage in the IDS. This data can then be used to train models that detect anomalies in incoming network data, for example, huge connection sizes.

Another possible solution for the detection of novel attacks and intrusions are collaborative intrusion detection systems (CIDS). In a CIDS, IDSs cooperate to share their collected information and experience to achieve better intrusion detection performance [9]. CIDS can also benefit businesses and enterprises that do not have the big data of networks or the resources to set up their own IDS, such as SMEs.

However, while CIDS propose an opportunity to scale the mitigation of cyberattacks even in settings that do not have many resources, sharing data between multiple entities and organizations is prone to privacy-related issues [10]. These privacy-related issues stem from the fact that some of the data shared is considered personally identifiable information (PII). For example, sharing raw, unanonymized IP addresses could lead to the identification of end-users in a network. Privacy considerations become especially important when a system needs to comply with regulations such as the General Data Protection Regulation (GDPR). According to article 4 of the GDPR, PII concerns any data that can be directly or indirectly identified to a source [11]. This also concerns data commonly used in an IDS, such as IP addresses.

What is needed is a more general approach to solving the problem of attack prediction while still preserving the privacy of end-users. A potential solution to these problems is the use of flow data. Flow data traditionally has a significant privacy advantage over solutions such as analyzing raw network packets as it does not contain any payload information, and the end-to-end user communications are protected [12]. However, it is still recommended for the flow data to be anonymized to protect privacy [10]. This anonymization of flow data can be approached using various anonymization techniques.

As stated in the introduction, a recent trend in the cybersecurity landscape has been cyberattacks on IoT devices. The Microsoft IoT Signals Report [13] states that in 2021 90 percent of all businesses and organizations were IoT adopters. This same report also stated that one of the key developments in the IoT landscape has been the focus on security, with nearly a third of the IoT adopters keeping a close eye on the security risk of IoT with specific concerns about data privacy and network security. Thus, examining how an IoT-based IDS performs in a more private manner is vital so parties can ultimately share data and secure their devices while not risking legal and privacy-related issues.

1.2 Research Scope and Questions

This thesis will evaluate the impact of anonymizing flow-based network data on the detection performance of an IoT-based IDS. This is done by comparing different techniques for anonymization

of flow data and evaluating their impact on the performance. We approach this by examining both a setting where the model has to determine whether a flow is malicious or benign and a setting where the model has to detect specific attacks. We will look at the following (sub)questions in this process:

1. How does anonymized flow data perform in the detection of malicious traffic in comparison to raw flow data in an IoT environment?

2. How does anonymized flow data perform in the detection of traffic from specific attacks (e.g., DDoS) in comparison to raw flow data in an IoT environment?

3. What classical machine learning models are best suited for the detection of malicious traffic in an IoT environment using anonymized flow data?

4. Which features are most important for the detection of malicious traffic in an IoT environment, and how do anonymization techniques impact the importance of these features?

These questions lead us to the overarching research question of the paper, which we will define as the following:

> *To what extent can less sensitive flow data detect cyber-attack traffic in an IoT environment in comparison with privacy-sensitive, unanonymized data?*

1.3 Thesis Overview

This thesis is divided into several chapters. First, we will examine related research on flow-based cyber-attack detection and compare our work with other works. Then in Section 3, Methodology, we will lay out the foundations and techniques that will be utilized to conduct the research. Here we for example examine what data privacy entails in the context of an IDS and investigate how this relates to specific features. Section 4, Experimental Setup, details the experiment setup and data processing. We will also present techniques for anonymizing flow data based on our privacy classifications. In Section 5, Experiments, the information from the previous sections will be applied, and the results of the experiments will be presented. Section 6, Discussion, will answer our subquestions and main research questions based on the experiments. We will also state limitations that were found when conducting our work. Finally, in Section 7, Conclusion and Further Work, we will conclude our work and explore what areas could be considered in future work.

2 Background

2.1 Flow-based Intrusion Detection

The field this paper can be classified into is flow-based intrusion detection. Traditionally, most intrusion detection systems used deep packet inspection or protocol analysis. However, due to the growth in speed of modern networks and the aforementioned methods being computationally expensive, scaling these methods is not feasible in modern environments [14]. A more modern and scalable approach is flow-based intrusion detection which uses flow data from exporters such as Cisco's NetFlow. We can define a flow as the following:

> "*A flow record contains information about a specific flow that was metered at an observation point. A flow record contains measured properties of the flow (e.g., the total number of bytes of all packets of the flow) and usually characteristic properties of the flow (e.g., source IP address).*" [15, p. 5]

Flow exporters are designed to collect IP traffic metadata by traversing network devices such as routers and switches. This metadata can consist of entries such as IP addresses, port numbers, and packet volumes.

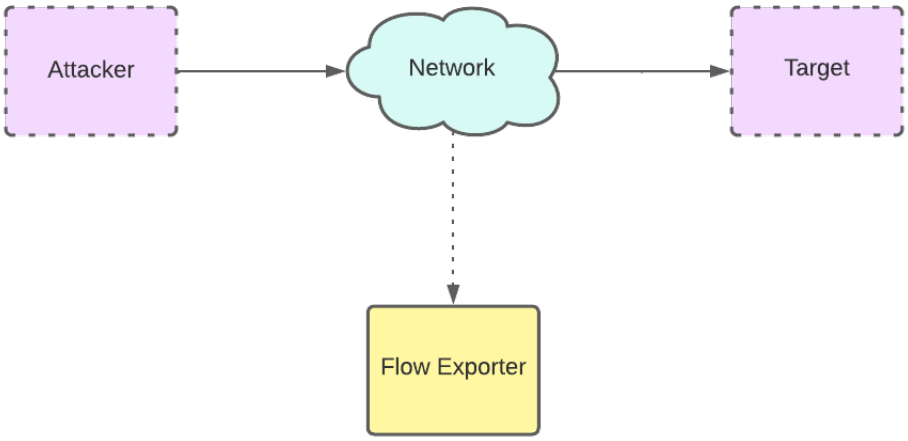

Figure 2: Flow exporter setup: based on Figure 3 by Hofstede et al. [16]

Figure 2 shows how a flow exporter will be positioned for use in a flow-based intrusion detection system. The flow exporter extracts the flow information between an attacker and a target. This flow data can then be exported for usage in the detection of harmful network activity and anomalies. This can be as simple as detecting an IP that is on a whitelist [17] but also DDoS detection with algorithms utilizing flow-based features as their parameters [18].

2.2 Related Work

In the last decade, there has been extensive research in the field of intrusion detection with flow data. In this section, we will briefly describe papers that are closely related to this work. We will also state in what aspects this paper differs from the current state of the art. Because this paper will focus on the usage of machine learning methods, we will only look at work incorporating machine learning methods.

Many approaches for flow-based intrusion detection have been proposed utilizing a wide range of methods, from classical machine learning approaches to deep learning. Tran et al. [19] proposed an Artificial Neural Network (ANN) approach using a hardware-based detection system, making it possible to process large volumes of data in real-time. Diro et al. [20] proposes a Convolutional Neural Network (CNN) approach. Yin et al. [21] proposes a deep learning approach based on Recurrent Neural Networks (RNN) with better results than with baselines of ANN, Random Forest, and Support Vector Machine (SVM) approaches. However, these works examine their approaches with datasets not from an IoT-related environment. Also, the proposed methods do not consider privacy-related issues and use the raw network data in their models.

Some work also has been done using datasets captured in IoT settings. For example, Koroniotis et al. [22] looked at the detection of botnets with the recent Bot-IoT dataset using SVM, RNN, and LSTM approaches. Koroniotis et al. also considers the correlation between features and evaluates performance using only the top 10 features. However, a shortcoming is that this is only examined in a binary setting and multiple attack types are not considered. Ullah et al. [23] also uses a deep learning approach for anomalies in an IoT network using 1D, 2D, and 3D CNNs. This is examined in both a binary and a multi-class setting. The dataset combines multiple public datasets, such as the aforementioned Bot-IoT set. While Ullah et al. examines the models' performance in predicting specific attacks, it does not look at which features influence the classification of these different attacks. Sarhan et al. [24] uses the ToN-IoT dataset, the same dataset utilized in this paper, and looks at both binary and multi-class classification performance on this dataset. However, the paper does not scrutinize privacy issues, and the data that is trained on is kept in its raw state. Also, it does not look at the classification impact of certain features. This shortcoming is also mentioned in possible future work. Meidan et al. [17] used Random Forests for the detection of unauthorized IoT devices with the use of classifying whitelists. While the paper does not specify any concerns of the sensitivity of the data or anonymization, it does look at how specific features impact detection performance of different devices and found that the min. time-to-live (TTL) was most impactul.

As we can see, there has been quite some research on attack detection using flow-based data in various settings. We have summarized the work as mentioned earlier in Table 1. This table contains columns concerning the techniques used in the paper and the name of the dataset that is used in the work. We also created columns for the properties of each work containing checkmarks for a quick overview. The Anonymization column contains a checkmark if the work concerns itself with potential privacy issues and anonymizes the features, the feature impact column contains a checkmark if it looks at the impact of certain features on the overall prediction performance or the prediction of specific classes, and the IoT column contains a checkmark if the data that is used is generated in an IoT related environment.

Table 1: Overview of related work and their properties

Research	Technique	Data	Anonymization	Feature Impact	IoT
This work	SVM, KNN, DT	ToN-IoT	✓	✓	✓
Koroniotis et al.	SVM, RNN, LSTM	Bot-IoT	✗	✓	✓
Meidan et al.	RF	Custom	✗	✓	✓
Ullah et al.	CNN	Multiple	✗	✗	✓
Sarhan et al.	Tree-based	ToN-IoT	✗	✗	✓
Wang et al.	SVM, KNN, DNN, RF	NSL-KDD	✗	✓	✗
Yin et al.	RNN	NSL-KDD	✗	✗	✗
Tran et al.	ANN	Darpa	✗	✗	✗
Diro et al.	CNN	NSL-KDD	✗	✗	✗

We have seen that all the papers mentioned use sensitive and unanonymized data for their detection algorithms. Also, we have seen that there is little research on the direct relationships between specific features and attack detection in both the multi-class and binary settings. Subsequently, no work was found on the intersection between feature relation and anonymization.

In comparison to other work, this work will distinguish itself in the following aspects:

1. Acknowledgement of privacy sensitivity of features and looking at the performance impact of using anonymized data for attack detection compared to using raw, unanonymized flows in an IoT environment (Subquestion 1,2,3).

2. Establishing the relationship between anonymized features and attack detection performance in an IoT environment (Subquestion 4).

In general, this paper can be considered to lay a foundation for developing a privacy-protecting IoT IDS, and conclusions can be made on the impact of taking a private approach to the intrusion detection problem and the impact this has on performance.

3 Methodology

3.1 Feature Privacy

To conduct the experiments, we first must classify which features tend to be privacy-sensitive. Privacy in the context of an IDS is two-way intractability, meaning the ability "to assert that endpoint X contacted endpoint Y at timestep Z" [10, p. 8] is removed.

In Tan et al., [25] it is stated that sanitizing flows is a difficult task because the line between what is sensitive and not is often not fully known and is subject to change over time. However, we do know of certain features to be privacy-sensitive. Tan et al. states that we should be careful with IP addresses, port numbers, and trace counters. The most direct identifiers are the IP addresses and ports. IP addresses can be used to identify a specific network or host, for example, by using IP address ranges. Subsequently, ports can be sensitive because they can be associated with specific services or processes that are currently active. Trace counters such as packet and byte volumes per flow, while not identifiers in and of themselves, can be used to extract sensitive data with injection attacks.

Aleroud et al. [26] also states that the last three features, IP addresses, ports, and trace counters, should be sanitized for privacy in flows. It also states that MAC addresses and timestamps should be considered privacy-sensitive. MAC addresses are 48-bit hexadecimal addresses for specific devices and can thus be used to identify a specific end device if this MAC address is known. Timestamps, while not being a direct identifier, can be seen as privacy-sensitive because they can be used with injection attacks similar to trace counters.

The aforementioned five feature groups are also stated in RFC6235 (IP Flow Anonymization Support) [10], which provides a proposed internet standard for anonymizing flow-based features. Besides the aforementioned privacy-sensitive groups, Section 4.5 of the aforementioned RFC also mentions that protocol numbers can also be seen as privacy-sensitive besides port numbers.

This leads us to six feature (groups) that we consider to be privacy sensitive; we have highlighted the six groups in Table 2, with their respective reason for anonymization.

Table 2: Overview of sensitive features and their anonymization reasons

Feature (group)	**Anonymization Reason**
MAC Addresses	Can be used to identify a specific end device
IP Addresses	Can be used to identify a specific host or network
Port Numbers	Can be used to identify the use of specific services
Protocol Numbers	Can be used to identify the use of specific services
Timestamps	Can be used for fingerprinting and injection attacks
Trace Counters	Can be used for fingerprinting and injection attacks

In Section 4.2, we will look at which of the features in our data are contained in one of these six feature groups and thus have to be anonymized for optimal privacy.

3.2 Data

The data used in this work comes from the NF-ToN-IoT dataset [24]. This dataset is a collection of multiple IoT and industrial IoT datasets that are aimed at use for testing the efficiency of cybersecurity applications and IDS.

The datasets used are all collected from an extensive testing network created at the University of New South Wales (UNSW) that connects hacking platforms, VMs, and IoT sensors [27]. A threat model is used to simulate an environment that is as realistic as possible [28]. This threat model distinguishes nine categories of attack types labeled in the dataset for each flow. The nine categories included within this threat model and their definitions are listed in Table 3.

Table 3: Attack Types present in ToN-IoT dataset

Label	Definition
Backdoor	Malware type to get remote access to resources within applications
DoS[1]	Attack that attempts to overload a system's resources from one attack source
DDoS[2]	Attack that attempts to overload a system's resources from distributed attack source
Injection	Attack supplying untrusted input that aims to alter the usual control flow
MITM[3]	Attack, where the attacker is placed between host and victim and intercepts network traffic
Password	This label covers all password-related attacks; this can be, e.g., sniffing or brute-forcing
Ransomware	Attacks that encrypt files on a host and require payment for decryption
Scanning	Group of techniques that try to discover information or vulnerabilities in networks
XSS[4]	Attack that attempts to inject malicious browser-side scripts into benign sites

[1] Denial-of-Service [2] Distributed Denial-of-Service [3] Man-in-the-middle [4] Cross-site scripting

Each flow is supplemented with 12 flow-based features extracted from network packet captures using the nProbe [29] extraction tool. The data also contains two labels, one for the attack type of one of the previously mentioned attack types and a binary label that is true based on whether the flow is part of an attack.

3.3 Data Composition

This section will examine the data used to conduct the experiments. Additionally, we will investigate the composition of the data to avoid potential bias.

First, we will look at some primary data about the dataset and the labeling composition. The raw dataset has 1,379,274 entries with no NaN or null values in both the features and labels, and we can thus use the raw dataset without many cleaning tasks beforehand. As mentioned earlier, the dataset contains both a binary label, which is true based on whether the flow is an attack, and a multi-class label for the attack type.

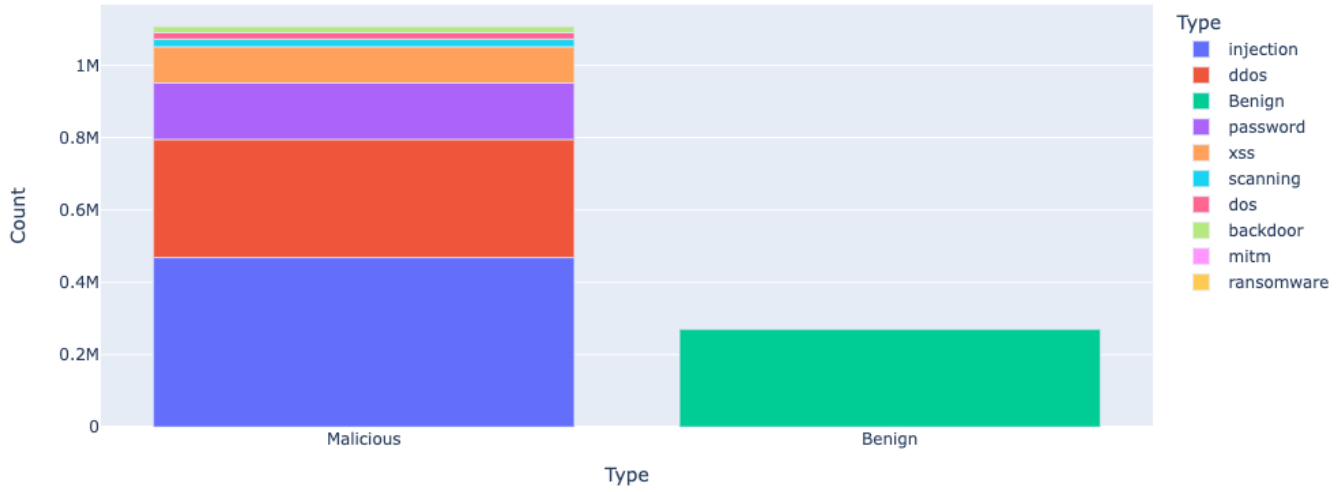

Figure 3: Data Composition

We can see in Figure 3 that the labels of benign and attack samples are slightly imbalanced with 1,108,995 attack labeled flows and 270,279 benign flows. If we look at the distribution of the labels in the dataset, we see that injection and DDoS are the most common labels for attacks, with occurrences of 33.97 and 23.66 percent of all flows. Password and XSS follow with occurrences of 11.33 and 7.25 percent of all flows. Scanning, DoS, and backdoor attacks occur to a lesser extent, with occurrences of around 1.5 percent. MITM and ransomware attacks occur much less than the other classes, with all occurring less than 0.1 percent of the time.

We can thus conclude that there is a slight imbalance in the binary labels and a pretty significant imbalance in the attacks with some outliers that have almost no occurrence and thus have an extreme imbalance. Our experiments can mitigate biased results with performance metrics suited for imbalanced data. This will be handled in the Section 3.4. We can also look at oversampling of the data to create a more uniform representation which will be looked at in Section 4.1.

3.4 Evaluation Metrics

To evaluate our models, we need to specify the metrics that will measure the performance. Additionally, we will examine what metrics should be examined when an IDS needs to meet specific needs. As we saw in Section 3.3 we have an imbalance in the binary and multi-class labels; we should thus look at performance metrics suitable for imbalanced data.

To evaluate how the models perform, we will use precision, recall, F1, and Matthews Correlation Coefficient (MCC) as our metrics for both the multi-class and the binary classification. Additionally, we will use SHapley Additive exPlanation (SHAP) values to determine feature contribution, which is essential to answer our fourth subquestion.

We define the precision metric by the following formula:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

here TP and FP are the true-positive and false-positive, respectively. Here the true-positive means the number of predictions that predicted the positive class when the positive class was also the target. The false-positive means the number of predictions where the positive class was predicted, but the target was not positive. In the context of this work, this can be seen as the number of flows that are labeled to be malicious that were malicious. This metric is important in an IDS where benign flows must not get marked as malicious.

The second metric we will use is the recall score which is defined as:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

The recall metric measures how many actual positive class labels are labeled positive in our predictions. This metric is essential in an IDS where malicious flows must get marked as malicious.

The third metric we will look at is the F1 score. The F1 score is a function of both the precision and recall metrics and is defined as follows:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{3}$$

The F1 score measures a balance between the precision metric and recall metric. This measure is thus essential when both the false positives must be equally represented. In the context of an IDS, a balance between benign flows not being marked as malicious and malicious flows getting marked as malicious is optimal.

As an additional metric in our experiments, we use the MCC metric. This metric provides a more balanced view of the model's performance against F1 because it also considers how well it can predict the negative value [30]. Thus, the model must score well in all four sections of the confusion matrix for a higher score. It is defined as:

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)\,(TP + FN)\,(TN + FP)\,(TN + FN)}} \tag{4}$$

Where TN and FN are the true-negative and false-negative, respectively. Here, the true negative means the number of predictions on negative when negative was also the target. False-negative means the number of predictions where the negative class was predicted but the positive class was the target.

To determine feature importance, we will use SHAP values. Shapley values are a metric from game theory to determine the contribution of each player to a game; however, they can also be used to determine feature importance in machine learning models. For this we use SHAP [31], an interpretability method based on the aforementioned Shapley values to assess the effect of specific features in a machine learning model.

4 Experimental Setup

This section will look at the experimental setup, which explains how the experiments will be done. Subsequently, this section will look at how the data needs to be preprocessed such that the data will be optimal and fair for the experiments. All written code for this thesis can be found in the following GitHub repository: github.com/jackvoorham/thesis_code.

4.1 Data Preprocessing

Because the dataset was created within a small subset of networks worldwide, the first preprocessing step we take is the removal of the source and destination IP addresses. We do this because the IP address is not a generalizable feature across different intrusion detection systems as the IP addresses and their ranges will change based on the network an IDS is situated in. Another feature we will remove is the `L7_PROTO` feature. Because we will anonymize features based on RFC6235 (IPFIX Anonymization Support), we will remove this feature as it is not specified in the IPFIX standard [32].

Subsequently, since machine learning algorithms can only interpret integers, it is necessary to conduct some preprocessing before the data is used in the models. While most of the features present in the data are integers, some categorical data exists. Features like the ports need to be mapped to integers to be correctly interpreted in our model. We do this mapping from categorical values to integers using one-hot encoding. With one-hot encoding, we convert each categorical value into a new column that is assigned either a one or a zero based on whether the categorical value is present or not. The features that we will encode include the ports, the IP protocol, and the TCP flags.

The second goal we want to achieve with preprocessing is the minimization of bias. The first step to achieving this goal is normalizing the numerical features so that data does not have a bias towards large or small values. We use min-max normalization, which normalizes the value between zero and one. Sarhan et al. [24], which utilizes the same dataset, also proposed this step. Additionally, we will balance all attack classes using oversampling for both binary and multi-class classification tasks. We will use the Synthetic Minority Oversampling Technique (SMOTE), which adds synthetic samples for the minority classes. Additionally, we will apply k-fold cross-validation with ten folds to prevent overfitting.

To conduct our experiments, Pycaret [33] is used. To do the preprocessing we pass the `normalize=True` and `fix_imbalance=True` parameters to Pycaret's setup function which will apply normalization and SMOTE respectively. It is important to note that SMOTE gets applied individually for each cross-validation step when running the models, and SMOTE does not get applied on the test set.

4.2 Privacy Labeling

As the dataset does not have any labels for privacy sensitivity, we need to make conclusions about this aspect manually. We can use the groups described in Section 3.1 to identify which features should be listed as sensitive. Because some features are similar (e.g., source and destination port), we will group them in a single category and label them based on that.

Table 4 contains the groups of the extracted features from the NF-ToN-IoT dataset. Each group is grouped with the NF-ToN-IoT feature names, textual descriptions, and the anonymization techniques that will be evaluated in the scope of this work. These anonymization techniques will be explained in Section 4.3. This results in us having six groups that are grouped on similar features. Four of the six groups contain PII per the privacy-sensitive groups stated in Table 2.

The ports are considered sensitive under the port numbers feature group; The IP Protocol is considered sensitive as per the sensitive protocol numbers group. The volume in octets and packets ought to be sensitive under the trace counters group. The MAC Address and timestamps groups are unused because none of the features present in the dataset are contained in one of these two groups.

Table 4: Feature groups of ToN-IoT features with respective anonymization techniques

Group	**Feature name(s)**	**Description**	**Anonymization Techniques**
Ports	`L4_SRC_PORT`	IPV4 Source Port	Binning & black-marker
	`L4_DST_PORT`	IPV4 Destination Port	
Volume in Octets	`IN_BYTES`	Incoming num. bytes	Binning & black-marker
	`OUT_BYTES`	Outgoing num bytes	
Volume in Packets	`IN_PKTS`	Incoming num. packets	Binning & black-marker
	`OUT_PKTS`	Outgoing num. packets	
IP Protocol	`PROTOCOL`	IP Protocol Number	Binning & black-marker
TCP Flags	`TCP_FLAGS`	Cumulative of TCP flags	None
Durations	`FLOW_DURATION_MILLISECONDS`	Flow duration (msec)	None

4.3 Anonymization Techniques

Now that we have distinguished feature groups based on their privacy sensitivity, we can look at anonymization techniques. This paper will look at two techniques; black-marker anonymization and binning. We will follow the guidelines of RFC6235, IPFIX anonymization support, by Boshi et al [10].

Black-marker anonymization is the simplest and most extreme form of anonymization and can be applied to any field. In black-marker anonymization, we delete the field or do not export the field in the network setup. While black-marker is the most secure technique, a trade-off is made in the usability of the dataset. Another technique that can be used for anonymization is binning. Binning can be seen as a type of data generalization where the data gets mapped into a specific bin based on its value. Multiple different values can thus be placed in the same bin and ultimately get the same value when anonymized. RFC6235 states that there is no particular one-fit solution for binning

and the binning scheme depends on the field type. However, the main goal of an excellent binning scheme is to keep precisely the information required for the analysis task. Notice that binning is closely related to black-marker anonymization, with black-marker anonymization being equivalent to binning into a single bin.

The RFC, however, proposes some ideas about how we can approach this for different IPFIX features. Binning ports can, for example, be done using a bilateral binning approach by creating a split based on low (0-1023) and high (1024- 65535) port numbers. This binning makes it possible to still tell the difference between service and ephemeral ports without the possibility to make conclusions about specific applications that are used. The pseudocode for the binning anonymization of ports is defined in Algorithm 1.

Algorithm 1 Bilateral binning of Port Numbers

Input: An arbitrary Port Number $\in \{0 \dots 65535\}$
Output: X, the binned Port Number
  $X \leftarrow$ Port Number
  if $X \leq 1023$ **then**
    $X \leftarrow 0$
  else
    $X \leftarrow 1$
  end if
  return X

Subsequently, for packet volumes, another algorithm was proposed. Packet volumes can also be binned by a bilateral approach which distinguishes between packet volumes smaller or equal to 2 or packet volumes larger than 2. The intuition behind this distinction is that connections that are opened usually contain less than or equal to 2 packets, and packets that were not opened contain three or more packets; we thus get one bin for connections that are likely to have been opened and a bin for connections that have likely not been opened. The pseudocode for the binning of the packet volumes is defined in Algorithm 2.

Algorithm 2 Bilateral binning of Packet Volumes

Input: An arbitrary Packet Volume $\in \mathbb{N}$
Output: X, the binned Packet Volume
  $X \leftarrow$ Packet Volume
  if $X \leq 2$ **then**
    $X \leftarrow 0$
  else
    $X \leftarrow 1$
  end if
  return X

A binning scheme was also provided in the RFC for the IP protocol. This scheme also uses a bilateral binning approach and looks if the protocol is part of a specific subset of IP protocols,

namely protocol numbers 1, 6, and 17 for ICMP, UDP, and TCP traffic, respectively. If the IP protocol number is not present in this specific subset, it gets placed in the other bin. Pseudocode for the binning of the IP protocol is defined in Algorithm 3.

Algorithm 3 Bilateral binning of IP Protocol Number

Input: An arbitrary IP Protocol Number $\in \{0 \dots 255\}$
Output: X, the binned IP Protocol Number
  $X \leftarrow$ IP Protocol Number
  if $X \in \{1, 6, 17\}$ **then**
    $X \leftarrow 0$
  else
    $X \leftarrow 1$
  end if
  return X

For the octet volumes, no particular binning schemes were proposed in RFC6235. We propose a simple bilateral binning scheme where the volume gets binned on whether it is higher or lower than the median of all before-seen octet volumes. This scheme is chosen, so that small and large flows get separated, and anomalies in either relatively large or small flows could still be spotted while not being influenced by outliers in either direction. The pseudocode of this binning technique is defined in Algorithm 4.

Algorithm 4 Bilateral binning of Octet Volumes

Input: An arbitrary Octet Volume $\in \mathbb{N}$ and S, the set of all Octet Volumes
Output: X, the binned Octet Volume
  $X \leftarrow$ Octet Volume
  $Y \leftarrow med(S)$
  if $X \leq Y$ **then**
    $X \leftarrow 0$
  else
    $X \leftarrow 1$
  end if
  return X

4.4 Experimental Setup

Before we conduct the experiments, we will first look at how we set up the experiments and what type of experiments will be done to answer the research questions optimally. To train in a way that minimizes bias, the models will be validated using 10-fold cross-validation on both the binary and the multi-class experiments. For the models, we will use K-nearest-neighbours (KNN), linear SVM, and Decision Trees (DT) as we think they have the optimal trade-off between performance (based on F1 scores and MCC scores) and training time, for both the multi-class as binary environments. For reference we ran all models using the `compare_models` function in Pycaret, the results can be seen in the tables in Appendix A. While the training time is on the higher side, the KNN model has

significantly better MCC scores than other models, especially in the multi-class setting. SVM and DT have low training time but perform similarly to other models with higher training time. It is also stated that these models are quick in their classification speed which is crucial if we want to use the model in a fast-paced IDS system [34]. Additionally, we saw that some of the mentioned related work also conducted their work using SVM, KNN, and tree-based models. The aforementioned models will be run using the Pycaret library [33], which is a wrapper over model implementations from, for example, the sci-kit-learn machine learning library.

To answer subquestions 1 and 2, we will train the models in binary and multi-class settings on the raw, unanonymized data. These results can act as a baseline with which we can compare the performance of the models that use anonymized data and make conclusions based on how they perform based on the metrics described in Section 3.4. To answer subquestion 4, *Which features are most important for differentiation of cyber-attack traffic from benign traffic in an IoT environment, and how do anonymization techniques impact the importance of these features?*, we will also examine the anonymization techniques on individual feature groups and see how this impacts the overall performance of using the feature set. With the results from these experiments, we can see how each feature gets impacted by its anonymization technique by simply comparing it to the unanonymized baseline. To get a better insight into the feature impact, we will also use the explainability of the DT model to look at the SHAP values for the classification performance. With this, we can also better understand the impact of specific features on the various attack types. We will use Pycaret's `interpet_model` function to calculate the SHAP values.

5 Experiments

5.1 Binary Classification

Table 5 presents the results of our baselines for the binary experiments. We see that based on both the F1 and MCC scores, KNN is the best classifier using unanonymized data from our considered models, followed by DT and SVM, respectively.

Table 5: Binary Experiments: Baselines

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
None (baseline)	KNN	—	0.9977	0.9987	0.9982	0.9908
	DT	—	0.9966	0.9748	0.9856	0.9244
	SVM	—	0.9454	0.9972	0.9706	0.8713

When looking at the Shapley values of the DT in Figure 4, we conclude that the `L4_DST_PORT`, `TCP_FLAGS` and `L4_SRC_PORT` are the three most essential features in the unanonymized baseline using the DT model. We see that the packet volumes and protocol are the least important in the binary classification task out of all the features. As for the protocols, we see that protocol numbers 17 and 6, or TCP and UDP protocols, are the most important.

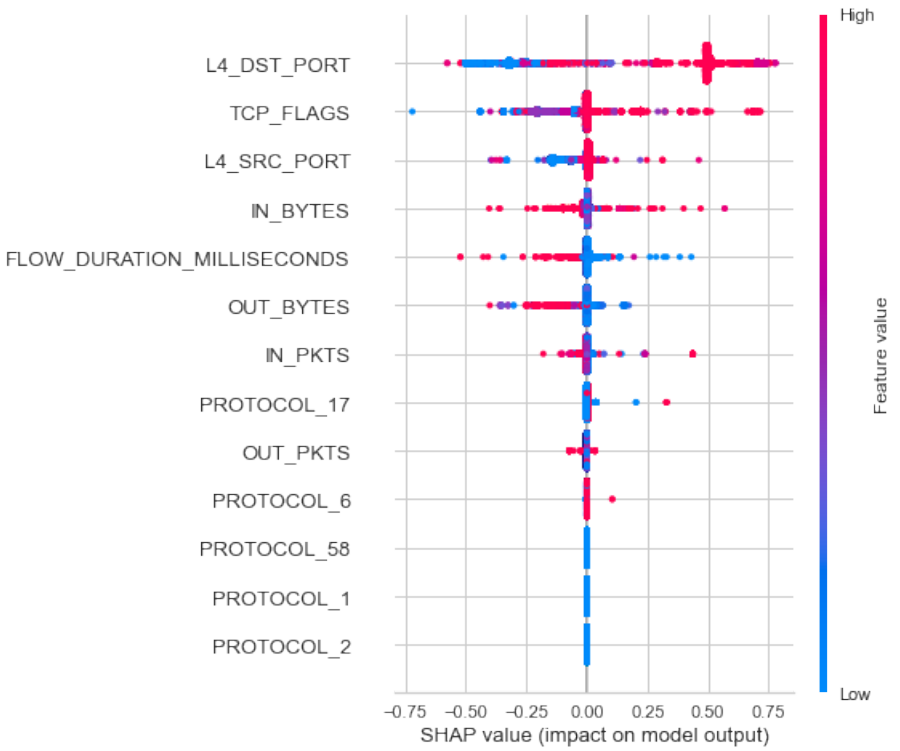

Figure 4: Binary Experiments: SHAP values of DT with no anonymization

5.1.1 Black-marker Anonymization

Table 6 shows the results of the experiments using black-marker anonymization in the binary classification setting. When using black-marker anonymization on all sensitive features, which we can see as the safest anonymization scheme, a significant decrease in performance happens on all three models. This decrease in performance is especially noticeable in the DT model, which goes from being the second-best in our baselines to the worst when using the black-marker technique. We see this in the metrics with an F1 score of 0.9856 in the baseline compared to 0.9107 when using the black-marker anonymization on all sensitive features. Subsequently, the MCC drops from the baseline of 0.9244 to 0.6064. The KNN approach is still the best classifier in the binary setting. However, anonymization significantly impacts our F1 and MCC metrics, decreasing from 0.9982 to 0.9436 and 0.9908 to 0.7198, respectively. For the SVM approach, it is important to note that while being the worst performer in the baseline, full black-marker anonymization has metric scores close to that of the KNN model with an F1 score of 0.9317 against 0.9436 and an MCC of 0.6767 against 0.7198. We can thus conclude that the SVM model is most resilient against applying full black-marker anonymization when comparing the metrics to the baseline.

When looking at the black-marker anonymization impact of the individual features, we see that anonymizing the ports using the black-marker approach significantly impacts the overall prediction performance. The black-marker anonymization of the protocol, the volumes of packets, and the volume of octets show similar results as the baseline, meaning their individual anonymization using the black-marker technique does not lead to a significant decrease in performance. Thus when considering the performance-privacy trade-off, they will be the best anonymized using the black-marker technique.

Table 6: Binary Experiments: Black-marker Anonymization

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
Black-marker	KNN	All Sensitive	0.9376	0.9496	0.9436	0.7198
		Ports	0.9777	0.9917	0.9847	0.9246
		Protocol	0.9977	0.9987	0.9982	0.9909
		Octet Volume	0.9978	0.9870	0.9923	0.9604
		Packet Volume	0.9973	0.9984	0.9978	0.9890
	DT	All Sensitive	0.8817	0.9419	0.9107	0.6064
		Ports	0.9301	0.9389	0.9345	0.6729
		Protocol	0.9859	0.9793	0.9824	0.9133
		Octet Volume	0.9908	0.9759	0.9831	0.9145
		Packet Volume	0.9801	0.9808	0.9801	0.9043
	SVM	All Sensitive	0.9160	0.9479	0.9317	0.6767
		Ports	0.9168	0.9481	0.9322	0.6785
		Protocol	0.9443	0.9986	0.9707	0.8729
		Octet Volume	0.9452	0.9980	0.9709	0.8732
		Packet Volume	0.9449	0.9975	0.9705	0.8713

Furthermore, we find that the individual anonymization of the ports in the SVM model bears similar metrics to complete black-marker anonymization. From this, we can conclude that the SVM does not find an interaction between the other features, while KNN and the DT model find interaction between the other features. This effect can be explained since the SVM model makes classifications based on linear separability.

In Figure 5 we see the SHAP values of the features in the feature set where all sensitive features are anonymized with the black-marker approach. Here we can see that the `TCP_FLAGS` is the most critical feature of the non-sensitive features when doing binary classification, with flow duration coming after that. Another thing to notice from this Figure is that `TCP_FLAGS` makes more predictions on malicious when the value is higher and more predictions for non-malicious when the value is lower. We can conclude from this that in cyberattacks, the higher bits are more often set, or more bits are set in general.

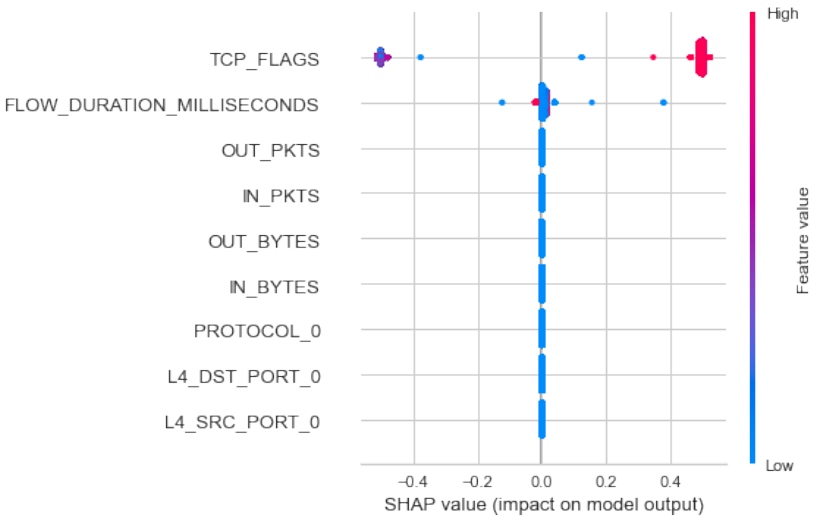

Figure 5: Binary Experiments: SHAP values of DT with black-marker on all sensitive features

5.1.2 Binning Anonymization

Table 7 shows the results for the experiments using the binning anonymization algorithms in the binary classification setting. To compare the impact of the binning approach, we again look at binning all sensitive features against the baseline described in Table 5. As we can see, the binning of all sensitive features results in a slight decline in performance on all models, but not as significant as when applying the black-marker approach. From this, we can conclude that the degradation in precision still keeps relevant information that can be used to separate benign and malicious flows. Again, we see that the DT model suffers from the most significant decrease in performance, with the F1 score decreasing to 0.9535 and the MCC score decreasing to 0.7828. We can again conclude that the SVM approach is most resilient against anonymization using binning based on the distances between the baseline and the metrics when anonymized. Comparing the metrics against the baseline, we see that the F1 and MCC metrics decline from 0.9706 to 0.9598 and 0.8713 to 0.8301, respectively. While for the binning approach, the KNN is still the best performing model, the impact of the anonymization is noticeable. This is most noticeable when comparing the MCC

score against the baseline, with a score of 0.9908 against 0.8648.

Table 7: Binary Experiments: Binning Anonymization

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
Binning	KNN	All sensitive	0.9439	0.9958	0.9692	0.8648
		Ports	0.9826	0.9988	0.9906	0.9546
		Protocol	0.9978	0.9988	0.9983	0.9913
		Octet Volume	0.9982	0.9908	0.9945	0.9716
		Packet Volume	0.9975	0.9985	0.9980	0.9898
	DT	All sensitive	0.9362	0.9715	0.9535	0.7828
		Ports	0.9375	0.9708	0.9539	0.7831
		Protocol	0.9853	0.9788	0.9818	0.9104
		Octet Volume	0.9909	0.9769	0.9837	0.9174
		Packet Volume	0.9964	0.9743	0.9852	0.9225
	SVM	All sensitive	0.9281	0.9939	0.9598	0.8301
		Ports	0.9356	0.9936	0.9637	0.8432
		Protocol	0.9447	0.9985	0.9708	0.8735
		Octet Volume	0.9812	0.9892	0.9851	0.9257
		Packet Volume	0.9452	0.9977	0.9708	0.8725

Looking at Figure 6, we can see the SHAP values of the features in the DT model using the binning anonymization technique. Here we can see that using the binning approach, the `L4_DST_PORT` and `TCP_FLAGS` still are essential features, as we saw in the SHAP values of our baseline. However, some discrepancies can be noticed when comparing the values to the SHAP values in our baseline; for example, we can see that the `OUT_PKTS` becomes significantly more important using the binning approach, while, for example, the `IN_BYTES` feature declines from being the fourth most important feature in the baseline to the second least significant value.

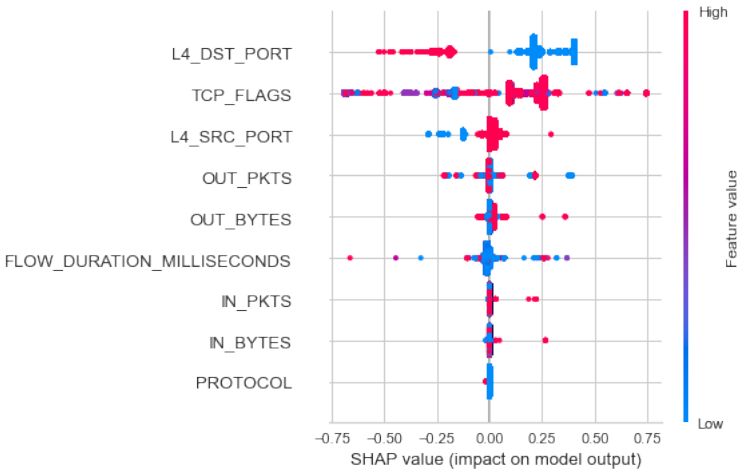

Figure 6: Binary Experiments: SHAP values of DT when binning all sensitive features

5.2 Multi-class Classification

Table 8 shows the baseline of the multi-class experiments. We can see that even on unanonymized flows, our models do not perform well on multi-class prediction in the Recall, F1, and MCC metrics. However, our models do perform quite well on precision. We see that for the multi-class prediction, the KNN model is the best choice out of our models based on the F1 score. Comparing the F1 score of the KNN model to the F1 score of 0.6 achieved by Sarhan et al. [24], we see that we get comparable scores to previous work.

Table 11 shows the feature impacts on the various classes.

Table 8: Multiclass Experiments: Baselines

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
None (baseline)	KNN	—	0.6137	0.6599	0.6315	0.5179
	DT	—	0.3546	0.6734	0.3993	0.3920
	SVM	—	0.4452	0.7257	0.4753	0.4011

Figure 7 shows the SHAP values using unanonymized flows in the multi-class classification setting. Comparable to the binary setting, the `L4_DST_PORT` is the most crucial feature for predicting all classes, with it being the most impactful to the prediction of the injection, benign and DDoS classes, respectively. An essential difference between the SHAP values we saw in our binary classification baseline is that the separation in the multi-class environment is more impacted by the volumes of both the packets and octets, with a focus on the outgoing packets and octets.

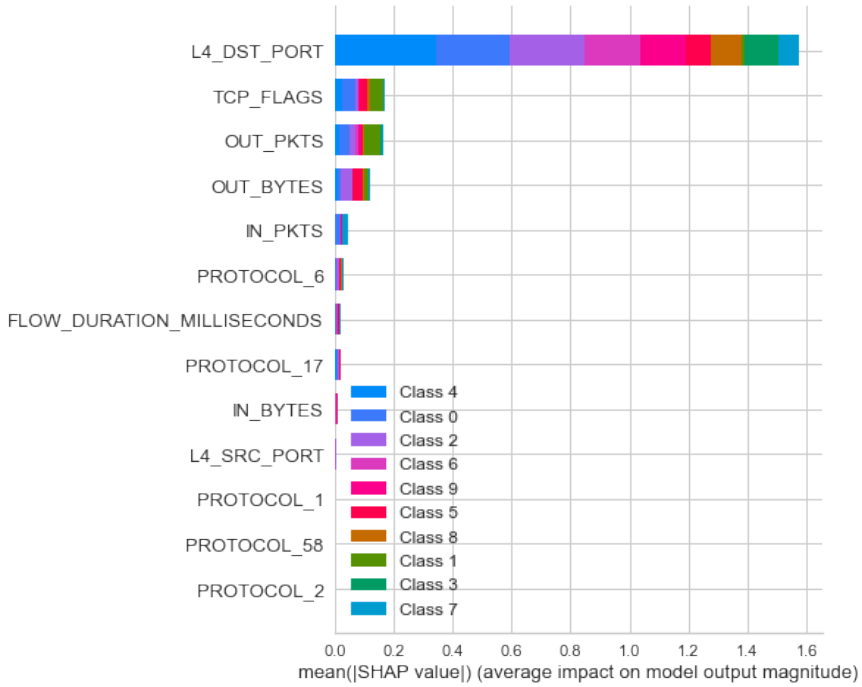

Figure 7: Multi-class Experiments: SHAP values of DT with classes benign: 0, backdoor: 1, DDoS: 2, DoS: 3, injection: 4, MITM: 5, password: 6, ransomware: 7, scanning: 8, XSS: 9

Again, like the binary classification baselines, we see that the protocols do not have much explanatory power. Here, a potential reason the protocols do not have as much explanatory power is that the most commonly used protocols (TCP, UDP, and ICMP) have their own way of accomplishing a specific attack. For example, for DDoS and DoS attacks, we can flood ICMP using ICMP echo requests, UDP by sending excessive amounts of IP packets containing UDP datagrams, and TCP by sending excessive amounts of SYN network packets [35]. While done on different protocols, the aforementioned attack approaches are all classified as DoS or DDoS attacks. Thus, based on the protocol alone, attack specification is not possible on some attacks, such as DoS and DDoS attacks.

5.2.1 Black-marker Anonymization

In Table 9 we see the impact of black-marker anonymization on the multi-class classification performance. We notice that the black-marker anonymization leads to significantly worse performance in all three tested models, with more significant relative drops than we saw in the black-marker anonymization of the flows for the binary classification. For instance, the SVM model decreases from an F1 score of 0.4753 in the baseline to an F1 score of 0.1692 and an MCC score of 0.4011 in the baseline to an MCC score of 0.1743 when all of the sensitive features were anonymized using the black-marker approach. Looking at the performance impact of using black-marker anonymization on specific features, we again notice that the ports have the most impact when anonymized using the black-marker approach.

Table 9: Multi-class Experiments: Black-marker

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
Black-marker	KNN	All Sensitive	0.5079	0.5290	0.4896	0.3697
		Ports	0.5214	0.5778	0.5462	0.4016
		Protocol	0.6146	0.6604	0.6323	0.5188
		Octet Volume	0.6108	0.6411	0.6232	0.5070
		Packet Volume	0.6137	0.6592	0.6313	0.5174
	DT	All Sensitive	0.1681	0.5555	0.2294	0.1313
		Ports	0.1960	0.5225	0.2640	0.1637
		Protocol	0.3387	0.5642	0.3676	0.3803
		Octet Volume	0.3345	0.5725	0.3725	0.3733
		Packet Volume	0.3451	0.6308	0.3908	0.3787
	SVM	All Sensitive	0.1950	0.3298	0.1692	0.1743
		Ports	0.2772	0.4342	0.2457	0.1874
		Protocol	0.4184	0.6562	0.4487	0.3639
		Octet Volume	0.4058	0.5574	0.3946	0.3581
		Packet Volume	0.3580	0.6707	0.3967	0.3492

Looking at the SHAP values of the non-sensitive features in Table 11 we see that feature impact order is the same as in the binary setting, with `TCP_FLAGS` being the most important feature followed by the `FLOW_DURATION_MILLISECONDS`. When looking at the importance of the `TCP_FLAGS` to the different classes, we again see that the injection, DoS, and benign classes have the most benefit of `TCP_FLAGS` for differentiation between other classes.

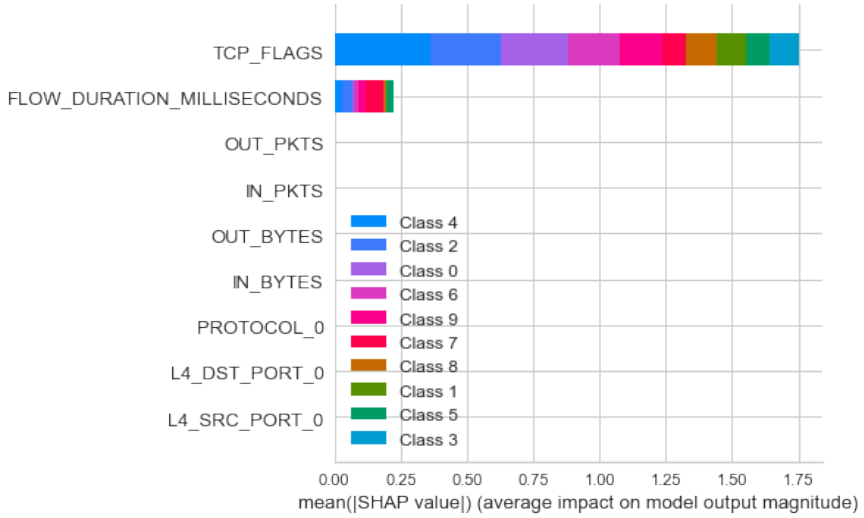

Figure 8: Multi-class Experiments: SHAP values of DT with black-marker on all sensitive features

5.2.2 Binning Anonymization

Table 10 shows the results and metrics of using the binning anonymization algorithms for the classification of the various attack types. We can see that the best performing model in the multi-class case using binning anonymization is the KNN model, which is the same model we noticed to perform best in the multi-class baseline and with complete binning anonymization in the binary setting. However, we notice that the performance of binning all sensitive features significantly decreases the performance of the models in the precision scores, which was seen in the baseline as the highest-scoring metric. Unlike the binning of the binary flows, all models significantly decrease their precision when looking at the multi-class case.

We again see that, like in previous experiments, the ports have the most impact when anonymized. This can be explained because different attacks happen on specific ports which fall both in the range of the ephemeral ports as the service ports; we can see this, for example, in the Mitre ATT&CK knowledge base [36], which presents a list of used ports that are used for different attacks.

Looking at the SHAP values in Figure 9, we can more closely argue why performance decreases in precision in the multi-class case. Comparing the SHAP values to baseline values, we can see that the `L4_DST_PORT` moves down to the second spot of the most impactful feature from the first spot. We can see that the `L4_DST_PORT` now has comparable importance to the `FLOW_DURATION_MILLISECONDS`. Thus, we can conclude from both the individual removal of ports, as the change in the order of the SHAP values that a more considerable precision in the `L4_DST_PORT` binning scheme is a significant factor in the separability between different classes.

Table 10: Multi-class Experiments: Binning

Anonymization	Model	Anonymized Feature(s)	*Recall*	*Precision*	*F1*	*MCC*
Binning	KNN	All Sensitive	0.5800	0.5726	0.5528	0.4665
		Ports	0.5331	0.5877	0.5561	0.4170
		Protocol	0.6136	0.6591	0.6311	0.5176
		Octet Volume	0.6180	0.6436	0.6285	0.5149
		Packet Volume	0.6157	0.6573	0.6318	0.5188
	DT	All Sensitive	0.2603	0.4818	0.3145	0.2151
		Ports	0.2417	0.5094	0.3007	0.2071
		Protocol	0.3193	0.4811	0.3377	0.3546
		Octet Volume	0.3298	0.6589	0.3754	0.3676
		Packet Volume	0.3536	0.6400	0.3985	0.3837
	SVM	All Sensitive	0.4274	0.5210	0.4366	0.3723
		Ports	0.2850	0.5643	0.2820	0.2598
		Protocol	0.3952	0.5726	0.4159	0.3344
		Octet Volume	0.4308	0.6312	0.4364	0.3960
		Packet Volume	0.4980	0.6941	0.5298	0.4259

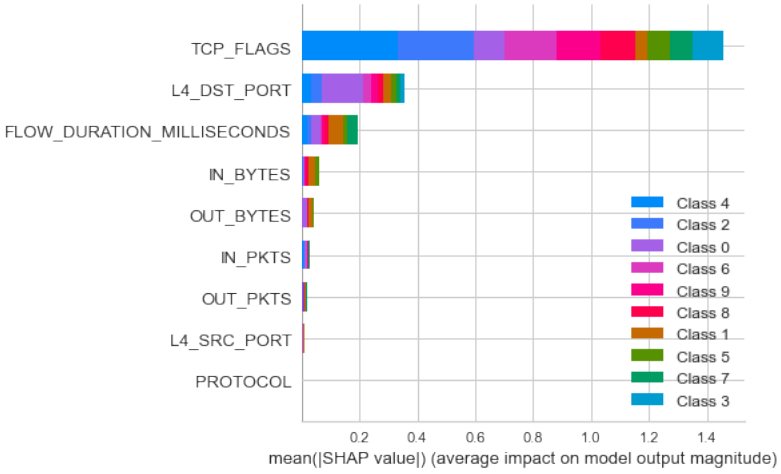

Figure 9: Multi-class Experiments: SHAP values of DT when binning all sensitive features

6 Discussion

6.1 Interpretation

In this section, we will briefly summarize the main points from our experiments by answering the subquestions and main research question. To answer the first subquestion, *how does anonymized flow data perform in detecting malicious traffic compared to raw flow data in an IoT environment?* we can interpet the experiments done in Section 5.1. We can conclude based on the experiments that binary classification can positively be done using anonymization techniques. When binning all sensitive features, we achieved an F1 score of 0.9692 and an MCC score of 0.8648, respectively, using the KNN model. Black-marker anonymization, while still achieving relatively good scores for the amount of privacy it provides, the loss of information is noticeable in the performance, with all models declining to F1 scores less than 0.95 and MCC scores less than 0.75.

Secondly, for the subquestion, *how does anonymized flow data perform in detecting traffic from specific attacks (e.g., DDoS) compared to raw flow data in an IoT environment?* we can look at Section 5.2. The first thing we notice in the experiments of this section is that the classification of different attack types performs poorly even on the raw baseline. When examining the performances of complete black-marker anonymization, we saw that the DT and SVM model could not keep up with the amount of anonymization. However, the impact on performance on the KNN model was relatively minimal compared to the other models scoring an F1 score of 0.4896 and an MCC score of 0.3697. Using the binning approach we saw that our binning schemes kept performance relatively close to the unanonymized baseline for the SVM and KNN model. Using the binning approach, KNN was the best performing model when applied on all sensitve features with an F1 of 0.5528 and an MCC of 0.4665 for the KNN model.

To answer the second subquestion, *what machine learning models are best suited for the detection of malicious traffic in an IoT environment using anonymized flow data?* we can conclude that the KNN model is the best performing model for binary as multi-class classification using anonymized features. However, declines in the KNN model were noticeable compared to the baseline. For the SVM model, we saw that this model stayed closest to its unanonymized baseline in the binary and multi-class classification task when binning all sensitive features, signaling stronger resiliency against binning anonymization.

To answer the third subquestion, *which features are most important for the detection of malicious traffic in an IoT environment, and how do anonymization techniques impact the importance of these features*? we can conclude that in the binary as the multi-class case, the ports were most impacted using the binning approach and generally have the most impact when anonymized on the classification performance. When using black-marker on all sensitive features, we saw that the `TCP_FLAGS` became the most essential feature in the classification performance. When using binning in the binary setting, the `L4_DST_PORT` kept its position as the most impactful feature. However, when binning in the multi-class approach, the order of the most impactful features changes significantly over the baseline, with `TCP_FLAGS` becoming more critical than `L4_DST_PORT`. We hypothesized that for good classification between multiple classes, the distinction between ephemeral and service ports is insufficient, and more precise bins are needed.

Now that we have concluded our subquestions, we can conclude with our main research question, *To what extent can less sensitive flow data detect cyber-attack traffic in an IoT environment, and how does it compare to methods with privacy-sensitive, unanonymized data?*. As we saw, binning algorithms perform most positively in the binary and multi-class settings. Especially in the binary class setting, binning all sensitive features scored relatively well compared to the raw baseline using the KNN and SVM models. However, in the multi-class setting, the binning anonymization had more impact on the performance than in the binary class setting. We can thus conclude that cyberattack traffic can be detected positively in a binary class setting using a anonymization scheme with binning techniques for the sensitive features.

6.2 Limitations

One of the limitations we ran into when conducting the research was the size of intrusion detection datasets. As most datasets available are captures of several days, these datasets can get up to gigabytes of data [37]. With sparse compute power, training on this data size can become cumbersome. Because of this limitation, we choose to conduct our research on the first, more compact, NF-ToN-IoT dataset instead of the more recent NF-ToN-IoT-v2 dataset. The main difference between these two sets is the amount of flow-based features they contain, and the number of flows that are contained, the attack types in both sets are identical. This limited compute power was also the reason for only training on classical machine learning models in our work instead of the deep learning approaches we saw in many of the related works.

7 Conclusions and Further Work

This paper has looked at how privacy impacts an IoT IDS' performance and the performance-privacy trade-off using various anonymization techniques. We also looked at how the performance was impacted by the anonymization of specific features using various metrics. We concluded by answering our main research questions that a private IoT IDS can still be performant in the binary classification task with sometimes showing almost equal performance compared to its non-anonymized counterpart, especially using binning anonymization techniques. However, black-marker anonymization can greatly impact binary classification performance, especially when taking maximum precaution with black-marker anonymization of all privacy-sensitive features. The multi-class classification performance of the tested models was sub-optimal, even on unanonymized flows. The anonymization of flows in the multi-class setting made the models even more unstable.

While this thesis provides a reasonable basis for evaluating the performance of a private IoT IDS, we still would like to mention some shortcomings we think are present in the scope of this paper and conclude in what directions future work could be taken. Firstly, we tested on a dataset that, while captured in a real-world environment, will probably not have a one-to-one mapping of the situation in a live setting. It is thus essential to look at the performance-privacy trade-off in a live setting, possibly by using a similar experimental setup as seen in our experiment sections. Secondly, while this work evaluated the most common flow-based features, many of the features that could be exported in a flow exporter setup were not considered. Future work could also look into features like the removed `L7_PROTO` or IP addresses and conclude on their privacy sensitivity and anonymization techniques while conducting similar experiments as done in this work. Third, some work can be done utilizing different machine learning approaches. This work only focused on a supervised learning environment using a labeled dataset, and unsupervised machine learning approaches were not considered in the scope of this work. As most IDS want to stay performant even in a changing environment, it is essential to look at the privacy-performance trade-off in an unsupervised way. The experiment could be set up using the dataset considered in this work, where the labels can act as ground truth for the unsupervised created clusters.

References

[1] "State of Cybersecurity resilience," tech. rep., Accenture, 2021. https://www.accenture.com/us-en/insights/security/invest-cyber-resilience,.

[2] M. van Haastrecht, G. Golpur, G. Tzismadia, R. Kab, C. Priboi, D. David, A. Răcătăian, L. Baumgartner, S. Fricker, J. F. Ruiz, E. Armas, M. Brinkhuis, and M. Spruit, "A Shared Cyber Threat Intelligence Solution for SMEs," *Electronics*, vol. 10, no. 23, 2021.

[3] "ENISA threat landscape," tech. rep., European Union Agency for Cybersecurity, 2021. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2021.

[4] M. Pogkas, A. Martin, and M. Benhamou, ""Major Hacks Have Cooled, But No One Is Celebrating Yet"," *Bloomberg*, 2022.

[5] M. Abomhara and G. M. Køien, "Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks," *Journal of Cyber Security and Mobility*, vol. 4, no. 1, pp. 65–88, 2015.

[6] L. Bliss, "Las Vegas Gambles on a 'Smart City' Technology Makeover," *Bloomberg*, 2019.

[7] "Unit42 IoT threat report," tech. rep., Palo Alto Networks, 2020. https://unit42.paloaltonetworks.com/iot-threat-report-2020/.

[8] F. House, ""Zero care about zero days"," *FireEye*, 2021. https://www.trellix.com/en-us/about/newsroom/stories/perspectives/zero-care-about-zero-days.html,.

[9] C. J. Fung, "Collaborative intrusion detection networks and insider attacks," *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 2, pp. 63–74, 2011.

[10] E. Boschi and B. Trammell, "IP Flow Anonymization Support," RFC 6235, RFC Editor, May 2011. http://www.rfc-editor.org/rfc/rfc6235.txt.

[11] "General Data Protection Regulations: Article 4." https://gdpr-info.eu/art-4-gdpr/.

[12] R. Hofstede, P. Čeleda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras, "Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 2037–2064, 2014.

[13] "IoT Signals Report," tech. rep., Microsoft, 2021. https://azure.microsoft.com/en-us/resources/iot-signals/.

[14] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Computers Security*, vol. 70, pp. 238–254, 2017.

[15] J. Quittek, T. Zseby, B. Claise, and S. Zander, "Requirements for IP Flow Information Export (IPFIX)," RFC 3917, RFC Editor, October 2004.

[16] R. Hofstede, A. Pras, A. Sperotto, and G. D. Rodosek, "Flow-Based Compromise Detection: Lessons Learned," *IEEE Security Privacy*, vol. 16, no. 1, pp. 82–89, 2018.

[17] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, and Y. Elovici, "Detection of Unauthorized IoT Devices Using Machine Learning Techniques," *CoRR*, vol. abs/1709.04647, 2017.

[18] J. Hou, P. Fu, Z. Cao, and A. Xu, "Machine Learning Based DDos Detection Through NetFlow Analysis," in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*.

[19] Q. A. Tran, F. Jiang, and J. Hu, "A Real-Time NetFlow-based Intrusion Detection System with Improved BBNN and High-Frequency Field Programmable Gate Arrays," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 201–208, 2012.

[20] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

[21] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[22] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[23] I. Ullah and Q. H. Mahmoud, "Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks," *IEEE Access*, vol. 9, pp. 103906–103926, 2021.

[24] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "Netflow datasets for machine learning-based network intrusion detection systems," in *Big Data Technologies and Applications* (Z. Deze, H. Huang, R. Hou, S. Rho, and N. Chilamkurti, eds.), (Cham), pp. 117–135, Springer International Publishing, 2021.

[25] K. Tan, J. Yeo, M. E. Locasto, and D. Kotz, "Catch, Clean, and Release: A Survey of Obstacles and Opportunities for Network Trace Sanitization," in *Privacy-Aware Knowledge Discovery: Novel Applications and New Techniques* (F. Bonchi and E. Ferrari, eds.), ch. 5, pp. 111–141, Chapman and Hall/CRC Press, January 2011.

[26] A. Aleroud, F. Yang, S. C. Pallaprolu, Z. Chen, and G. Karabatis, "Anonymization of Network Traces Data through Condensation-Based Differential Privacy," *Digital Threats*, vol. 2, oct 2021.

[27] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. d. Hartog, "Ton_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 485–496, 2022.

[28] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_IoT Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020.

[29] L. Deri, "nProbe: an Open Source NetFlow Probe for Gigabit Networks," in *In Proc. of Terena TNC 2003*, 2003.

[30] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, p. 6, 2020.

[31] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," *CoRR*, vol. abs/1705.07874, 2017.

[32] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101, RFC Editor, January 2008. http://www.rfc-editor.org/rfc/rfc5101.txt.

[33] "Pycaret Library." https://pycaret.org/.

[34] S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," *Informatica (Slovenia)*, vol. 31, pp. 249–268, 2007.

[35] V. Sapra and A. Bhardwaj, "8.1.2.1: Volumetric DDoS Attacks," in *Security Incidents & Response Against Cyber Attacks*, Springer, 2021.

[36] "Mitre, Technique T1571: Non-Standard Port." https://attack.mitre.org/techniques/T1571/.

[37] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A Survey of Network-based Intrusion Detection Data Sets," *CoRR*, vol. abs/1903.02460, 2019.

A All Models

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.9971	0.9983	0.9977	0.9987	0.9982	0.9908	0.9908	65.5000
gbc	Gradient Boosting Classifier	0.9804	0.9982	0.9990	0.9772	0.9880	0.9357	0.9373	44.5690
nb	Naive Bayes	0.9770	0.9626	0.9984	0.9736	0.9859	0.9240	0.9262	3.4160
dt	Decision Tree Classifier	0.9765	0.9454	0.9966	0.9748	0.9856	0.9228	0.9244	4.3540
et	Extra Trees Classifier	0.9756	0.9984	0.9757	0.9939	0.9847	0.9251	0.9263	20.7350
ridge	Ridge Classifier	0.9719	0.0000	0.9777	0.9872	0.9824	0.9122	0.9124	3.4040
lda	Linear Discriminant Analysis	0.9719	0.9964	0.9777	0.9872	0.9824	0.9122	0.9124	3.9400
lr	Logistic Regression	0.9706	0.9972	0.9705	0.9928	0.9815	0.9097	0.9111	22.9690
ada	Ada Boost Classifier	0.9619	0.9982	0.9566	0.9959	0.9758	0.8872	0.8919	13.0130
lightgbm	Light Gradient Boosting Machine	0.9588	0.9966	0.9563	0.9925	0.9738	0.8775	0.8831	5.2540
rf	Random Forest Classifier	0.9554	0.9989	0.9455	0.9990	0.9715	0.8695	0.8765	24.0450
svm	SVM - Linear Kernel	0.9539	0.0000	0.9454	0.9972	0.9706	0.8647	0.8713	3.5950
qda	Quadratic Discriminant Analysis	0.8640	0.9262	0.8478	0.9777	0.8869	0.7204	0.7455	3.6800
dummy	Dummy Classifier	0.1960	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	3.5400

Figure 10: Pycaret `compare_models()`, binary classification (baselines)

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.6137	0.8410	0.6137	0.6599	0.6315	0.5148	0.5179	29.9520
et	Extra Trees Classifier	0.6067	0.8164	0.6067	0.5386	0.5509	0.4575	0.4853	55.2990
lr	Logistic Regression	0.4702	0.8774	0.4702	0.7531	0.5161	0.3913	0.4254	929.1230
rf	Random Forest Classifier	0.5398	0.7880	0.5398	0.5326	0.5156	0.3931	0.4129	63.6500
svm	SVM - Linear Kernel	0.4452	0.0000	0.4452	0.7257	0.4753	0.3602	0.4011	9.1230
gbc	Gradient Boosting Classifier	0.4032	0.7423	0.4032	0.6364	0.4303	0.3124	0.3460	1310.6270
ada	Ada Boost Classifier	0.4242	0.6502	0.4242	0.4787	0.4079	0.3172	0.3576	39.2570
dt	Decision Tree Classifier	0.3546	0.6679	0.3546	0.6734	0.3993	0.2900	0.3920	5.9950
lightgbm	Light Gradient Boosting Machine	0.4049	0.7031	0.4049	0.5367	0.3986	0.3027	0.3398	42.9680
lda	Linear Discriminant Analysis	0.3555	0.8467	0.3555	0.5463	0.3404	0.2824	0.3381	4.9130
ridge	Ridge Classifier	0.3096	0.0000	0.3096	0.4655	0.3141	0.2454	0.2965	4.5600
nb	Naive Bayes	0.2459	0.7935	0.2459	0.5528	0.2244	0.1604	0.1961	4.2380
qda	Quadratic Discriminant Analysis	0.2508	0.6294	0.2508	0.4103	0.2115	0.1418	0.1712	5.1280
dummy	Dummy Classifier	0.1960	0.5000	0.1960	0.0384	0.0642	0.0000	0.0000	3.9840

Figure 11: Pycaret `compare_models()`, multi-class classification (baselines)

B Abbreviations

ANN: Artificial Neural Network
RNN: Recurrent Neural Network
RF: Random Forest
DNN: Deep Neural Network
KNN: K-nearest neighbours
DT: Decision Tree
SVM: Support Vector Machine
GDPR: General Data Protection Regulation
PII: Personally Identifiable Information
IP: Internet Protocol Address
IDS: Intrusion Detection System
MITM: Man-in-the-middle
XSS: Cross site scripting
DoS: Denial of Service
DDoS: Distributed Denial of Service