The Quantitative Analysis of Explainable AI for Network Anomaly Detection

Master Thesis

Sinie van der Ben

Faculty of Science - Graduate School of Natural Sciences

Under the supervision of:
Hanna Hauptmann (1rst), Utrecht University
Kate Labunets (2nd), Utrecht University
Mennatallah El-Assady, ETH Zürich

Abstract

In the current digital society, network sizes continue to expand to meet the demand for digitalisation. This expansion is not without security risks. Network Anomaly detection (NAD) is concerned with the detection of malicious traffic in a network. Within the field of NAD, applications that use machine learning to protect the network are on the rise. This rise is accompanied by a request for explanations about the model's decisions. Currently, there are studies that apply Explainable Artificial Intelligence in NAD, but they often lack proper evaluations of proposed explanations. This thesis contributes to the gap by investigating objective properties for explanations in the field of NAD.

Three models are trained on the UNSW-NB15 dataset: Random Forests (RF), Explainable Boosting Machine (EBM) and Autoencoder (AE). Different versions of the dataset are created, and hyperparameter tuning is performed to find the optimal version of a binary and multiclass classifier. Several models are selected and combined with explanations. LIME and SHAP are chosen as model-agnostic explanation methods, while the EBM is inherently explainable. Predictions of the AE are explained by a personalised method, as there is no universal way of explaining the model yet. The explanations are evaluated on two objective properties, namely sensitivity and fidelity.

The results show that the RF outperforms the other two models in binary classification, given model performances only. The binary EBM shows the highest fidelity metrics. For the multiclass classification problem, the RFs trained on a balanced dataset show the best performance, although the values of the objective properties are not optimal.

The final combination of a model with an explanation depends on the importance placed on model performance and each objective explanation property. Aspects such as the format of the dataset or the model hyperparameters influence the model performance and can affect the explanation quality. Explanation quality appears to depend on the dataset, confirming earlier research.

Future research should incorporate different models, explanations and objective properties to extend this research and generate more insights in the objective properties for explanations in the field of NAD.

Table of Contents

List of Abbreviations

AE Autoencoder.

AI Artificial Intelligence.

CNN Convolutional Neural Network.

DT Decision Trees.

EBM Explainable Boosting Machine.

GAM Generalised Additive Model.

GAN Generative Adversarial Network.

kd-Tree kd-Tree.

LR Logistic Regression.

NAD Network Anomaly Detection.

OHE One Hot Encoding.

RF Random Forest.

SVM Support Vector Machine.

XAI Explainable Artificial Intelligence.

Chapter 1

Introduction

In September 2022 two large cyberattacks made headlines: the attack on Uber and the attack on Rockstar Games. Both attacks were carried out by the hacking group Lapsus$. In the case of Uber, the company informed the public that no serious harm was caused by the intrusion [15]. With the hack of Rockstar Games, early development-stage footage of the new game Grand Theft Auto 6 was leaked. Although the leakage was unfortunate, Rockstar Games reported that the hack will not impact the development of the game [54].

These two examples demonstrate attacks on larger private businesses with high stakes, but other common targets include governments, universities and critical infrastructures. Russia's invasion in Ukraine reveals the influence of cyberattacks as a support of warfare [55]. In recent years, there has been a steep increase in the number of cyberattacks as well as the impact per attack. Especially, the occurrence of ransomware and threats against availability, such as Denial of Service (DoS), have increased. The amount of malware attacks also continues to rise [17, 55].

This increase in attacks is a response to digitalisation and corresponding network expansion [55]. The COVID19 pandemic requested a rapid digital transition of the society. Facilitation of remote work and education resulted in additional endpoints in networks. Expanded networks allow for a greater surface attack, as attackers have more vulnerabilities to exploit [55]. Another reason for network expansion is the Internet of Thing (IoT). With IoT, multiple technologies, such as sensors or cloud-based applications, are connected within a network. This leads to a larger network size [16].

Networks are part of cyberspace. Cyberspace denotes the collection of networks, databases and other sources of information that interact with each other [78]. Cyber security is the practice of making the cyberspace more secure. It refers to all activities and measures that protect cyberspace and its communications, both technical and non-technical [11].

As mentioned earlier, a larger network size leads to a larger surface attack. A larger surface attack requires more protection and security. A subset of cyber security is Network Anomaly Detection (NAD). NAD is the detection of malicious traffic by observing patterns that are abnormal [25]. The prevention of attacks and detection of anomalies is the job of security analysts and security engineers. Analysts have to examine data and events from network traffic to spot potential attacks, while engineers design security measures to decrease the risk of successful attacks. Detection is still a partially manual task, having defense systems designed that combine human-crafted rules with automated detection models. Currently, studies are investigating automatic detection of anomalies [12, 25].

Automated detection models can incorporate Artificial Intelligence (AI). In general, AI makes use of analytical models to generate predictions and make detections [39]. AI models are often based on machine learning. Machine learning models are able to generate predictions based on prior experience or previously seen patterns, using labeled or unlabeled data. A subset of machine learning is deep learning. Deep learning models are capable of handling problems of increased complexity and have shown to exceed human performance on several tasks [76]. They have shown to outperform shallow machine learning models on most tasks that require high-dimensional data [39, 43]. Deep learning models often require more training data than other machine learning models.

Nowadays, detection models often include a form of AI, machine learning or deep learning. They allow for automated detection of anomalies. The use of these techniques removes the burden on humans to explicitly formulate knowledge or perform manual classification [39]. Model performance is crucial for detection systems, as missing intruders can have harmful outcomes. In addition to model performance, users of AI models are challenged with model interpretability.

1.1 Problem Statement

Interpretability or explainability is the ability of a model to be understood by a human [20]. In this thesis, the term 'explainability' is used. Explainability can influence the decision-making process of users or impact the trust they have in a model's prediction. It allows users to understand the underlying reasoning behind a decision or find biases incorporated by the model [67, 70]. A problem with complex models is that they are often not inherently explainable. This means that a human is not able to understand the model without additional methods. Lack of explainability can influence the interaction between the human and the model, such as the trust users have in a mode. Users must be able to follow the reasoning of the model to gain trust [7]. Lack of trust influences the adoption of the model's decisions [39]. Aside from trust, if users do not understand a model, they have trouble creating an accurate mental model of the system. This makes it difficult to improve a system [45].

Explainable AI (XAI) is the field within AI that tries to facilitate the understanding of models and their decisions for humans [72]. XAI can provide explanations or insights about the decisions made by the model.

Explanations are ways to explain the decisions made by the model [44, 57]. The quality of explanations can be measured subjectively and objectively. Subjective measures include a user to evaluate the explanation, while objective explanations evaluate the explanation without a user [82]. Objective measures investigate the quality of an explanation [70], by examining certain properties of the explanation method. Measuring a property that reflects quality allows for quality assessment of that explanation prior to presenting it to the user. Examples of such properties are accuracy, fidelity, consistency and sensitivity [70]. Both subjective and objective measures of explanations are important in XAI, as they share the final goal of providing a good explanation.

As the use of deep learning models in any field has increased in recent years, it has also increased in the field of NAD [26]. In general, machine learning and deep learning models are still fallible. In the field of NAD, it is important to have accurate models to decrease the number of false negatives. Missclassification of intrusions can have harmful results. With the goal of helping to improve the detection and understanding of the models and attacks, explainability is a desiderata for models in the field of NAD.

Although there have been applications of XAI in the field of cybersecurity, the contribution of XAI in the field of NAD is not satisfied. Currently, explanations for NAD are often not evaluated on objective properties. Although several properties can contribute to improving explanation quality, this thesis focuses on fidelity and sensitivity. Fidelity reflects the truthfulness of the explanation [82]. An explanation is fidel if it accurately demonstrates the decision process of the underlying model. For example, if an explanation notes certain features as important for a prediction, perturbing them would lead to a different prediction. In the field of NAD, explanations have to reflect the truth about decisions; otherwise they are of no use to analysts and engineers [30]. Although fidelity is an important property, it is often not included in the evaluation of explanations for NAD.

Sensitivity, often called robustness, is a property of an explanation that reflects the ability to handle small perturbations without large consequences [82]. If small, non-significant perturbations lead to large differences in the explanation, the explanation does not show stable behaviour. The lack of stable behaviour makes it difficult to know which features are truly important for the prediction. Previous work of XAI for NAD includes robustness, but mostly as a tool to improve explanations with the aim of improving the model [5, 29].

Given the current state of research on objective properties to examine explanations in the field of NAD, there is still a gap to fill. Mentioned before, the objective properties of explanations can contribute to the quality of an explanation, before presenting it to the target user. Past studies on NAD have shown that there are several ways to include XAI, but the next step is evaluating their quality. The current research contributes to this field by focusing on fidelity and sensitivity in evaluating explanations, to ensure explanations adhere to these properties before being presented to target users.

1.2 Research Question

The goal of this thesis is to evaluate the quality of explanations through objective properties in the field of NAD. In earlier studies, explanation methods such as LIME and SHAP are applied to models that classify anomalies, but their explanations lack objective measures [2, 12, 73].

Following the goal of this study, the research question is formulated:

How can network anomaly detection be improved by including objective properties of explanations as quality indicators?.

To answer the research question, different explanation methods are evaluated. Explanations need a trained model to explain predictions. Part of this research is devoted to training models for NAD. Although model performance is not the main focus of this study, a model that performs accurately is preferred. To examine differences between models, three different models will be compared: a Random Forest (RF), an Explainable Boosting Machine (EBM) and an Autoencoder (AE).

After model evaluation, explanations will be evaluated using the two objective properties described earlier, fidelity and sensitivity.

In the final stage, model performances and explanation evaluations will be combined. Aside from objective metrics, there are other concerns to consider for a final decision, such as the choices of the data set and the computational complexity of the models and explanations.

This approach leads to the following three subquestions:

1. R1: Which model shows the best performance with respect to the model metrics?

2. R2: Which explanation shows the best performance regarding objective explanation metrics fidelity and sensitivity?

3. R3: Which combination of model and explanation creates the best compromise between explainability and model performance?

Chapter 2

Background and Related Work

This chapter provides background knowledge on Explainable AI, Network Anomaly detection, Machine Learning and a combination of the three. The first section covers how the literature study was conducted.

2.1 Literature search

The search for papers was conducted through Google Scholar and IEEE XPlore. Review or survey papers were used to find specific topics through the papers cited in reviews and surveys.

Regarding the quality of the papers, arXiv preprints were excluded, except the paper of Doshi-Velez and Kim [20] and InterpretML [65]. Another paper used is under review, but accepted by the NeurIPS 2022 Benchmarks and Datasets chair on September 16, 2022 [27]. If the journal in which the paper was published was unknown, it was checked that the papers were peer-reviewed before acceptance.

During the search, there was no selection criteria for the year of publication, but recent articles were preferred.

The news articles from the introduction had to come from a known objective newspaper. They do not state research facts, but rather events around the world. The facts and numbers stated about cybersecurity risks in the introduction come from reports that were written by larger tech companies and objective institutions.

Below, an overview of the terms used in the search process is given. Terms were combined to create queries for different topics or create similar queries with different words.

- 'explainability', 'explainable AI', 'XAI', 'interpretability'
- 'properties', 'desiderata', 'characteristics', 'factors', 'requirements', 'objectives'
- 'survey', 'review'
- 'network anomaly detection', 'anomaly detection', 'NAD', 'network intrusion detection', 'NID'
- 'cybersecurity', 'cyberattacks', 'cyberspace'
- 'objective', 'subjective', 'human-centered', 'human evaluation', 'application-grounded', 'human-grounded', 'functional-grounded'
- 'machine learning', 'deep learning', 'neural networks'
- 'comprehensibility', 'robustness', 'stability', 'fidelity', 'sensitivity', 'truthfulness'
- 'Autoencoder', 'Random forest', 'CNN', 'RNN', 'LSTM', 'EBM', 'GAN'

2.2 Explainable AI

In recent years, machine learning and deep learning have been applied in many domains. These methods have shown to be able to handle complex problems. Despite their performance, it is difficult for humans to understand how these models work exactly and they are often considered *black boxes*. Black-box models are uninterpretable by humans [28, 44, 70], which can be problematic in situations where understanding decisions is important. Examples of black boxes are Random Forest (RF) and Support Vector Machines (SVM), as well as different neural networks, such as Convolutional Neural Networks (CNN) and Auto-Encoders (AE). They are opposed to *white-boxes*, which are inherently interpretable models [44, 60]. Logistic Regression (LR) and Decision Trees (DTs) are white boxes. The field concerned with the explainability of models is called *explainable AI* (XAI). Explainability means the model's decision can be explained in a human-understandable way [20]. Explainers are algorithms or methods that provide insights about the model and these insights can be called explanations [14, 44]. As there are different models, the dimensions of the explanation can differ.

Model agnostic vs. Model specific Explanation methods can be model-agnostic or model-specific [28, 48]. Model-agnostic explanations do not depend on the inner structure of the model, which means that they are applicable on different kinds of models. An example is LIME, short for Local Interpretable Model-agnostic Explanations [69]. LIME generates feature importances by understanding how the output is affected by perturbations of the input. Opposed to model-agnostic methods are the model-specific methods. Model-specific explanation methods are usable on one type of model because they access the interior of the model. An example is DeepLIFT (Deep Learning Important FeaTures) [75], a method that compares the activation of each neuron in a deep neural network to a reference score. It is a specific explanation method for neural networks. The method redefines how gradients are calculated, which means that implementation requires insight into the architecture of a convolutional neural network.

Ante-hoc vs. Post-hoc Ante-hoc explanations are techniques that incorporate explainability into the structure of the model. The model is directly explainable after training. These models can also be called *self-interpretable* [67]. Examples of self-interpretable models are DTs and regression models. Ante-hoc methods are opposed to post-hoc methods. Post-hoc explainability techniques aim to interpret models already trained, without having any knowledge of the inner workings of the model[48, 67]. Post-hoc methods include counterfactuals. A counterfactual describes the smallest change necessary in a feature value of the current input to change the output prediction [48, 60]. Another example is SHAP (SHapely Additive Explanations) [52]. This explanation method shows the contribution of each feature to the prediction made by a model.

Local vs. Global Post-hoc explanations can be local or global [28, 48]. A local explanation helps the user understand how a model makes a decision for a single data point. LIME, SHAP and counterfactuals are examples of local explanations. On the other hand, a global explanation supports the user in understanding how the model makes decisions in general. Accumulated Local Effects (ALE) is a global explanation method that shows the average effect of features on the prediction of a model [6].

Scope and target Another important consideration for the choice of the explanation method is the scope of the explanation and the purpose of the explanation. The scope can be considered as the field of application of the model. It can be general, explaining the model regardless of the field, or specific, such as explaining a model created for medical applications. The target is the intended user of the model. The level of expertise of the target user influences the preferences and format of the explanations [28]. Using a medical example, explainability can request the expertise of the target user within a specific scope. Explanations can contribute to establishing the importance of features for medical detections and classifications. False predictions must be checked manually due to the high-risk consequences of false negatives, highlighting the need for medical expertise of the target user [4].

Summary Although there are different dimensions of explanations, this thesis mainly focuses on model-agnostic post-hoc local explanations within the scope of network anomaly detection. The investigation focuses on explanations of a single prediction of an individual instance to be informed on the contributing features that make a model classify an attack as such.

2.3 Evaluation of Explanations

As explanations have the goal of providing explainability, evaluations are needed to show if they do and what the quality of an explanation is. An evaluation can be done with or without humans. Doshi-Velez and Kim propose three ways to evaluate explanations, of which the first two include humans: application-grounded and human-grounded. The last is functionally grounded, which can be done without humans [20].

The application-grounded evaluation focuses on how well an explanation can support users. It tries to assess how an explanation can improve the performance of a user on a task in comparison to a situation without the explanation. The general approach for this evaluation is to let domain experts perform specific tasks. The performance on a task can be measured and, with an additional questionnaire, researchers can assess the impact of the explanation on the task [40]. By approaching evaluations in this way, studies measure the influence of explanations on decision making, trust and task performance [59].

The second way is human-ground evaluation. Like application-grounded, human-grounded evaluations include a user in the evaluation, but the task is simpler. It does not require the user to be a domain expert. The evaluation assesses the quality of the explanations itself, rather than the impact of the explanation on the decision-making process. An example of a human-grounded evaluation is asking users through a questionnaire or survey which explanation method is preferred for understanding model decisions [41]. Instead of asking which explanations the user prefers, cognitive load analysis can assess the impact of an explanation on the performance and confidence of the users by performing simple, nondomain specific tasks [37].

Application- and human-grounded evaluations can investigate properties such as trust [47, 59], understandability [20, 47, 59, 67] and usability of the explanation [59]. These properties require subjective experience of the user.

The last evaluation of Doshi-Velez and Kim does not include users but tries to evaluate an explanation through a formal definition of interpretability. This is called the functionality-grounded evaluation [20]. A formal definition of interpretability depends on the quality properties of the explanation that can be quantified. The specific quantification depends on the property itself. Through objective metrics, the goodness of an explanation can be quantified explanation [8].

An example of an objective property for the quality of an explanation is fidelity [22, 28, 70, 80]. Fidelity represents how well the explanations reflect the behaviour of the model. If an explanation deems a certain feature to be important for a prediction, removing or altering the feature would alter the prediction. There is a difference in local and global fidelity. Globally fidel explanations reflect the behaviour of the whole model, while local fidelity represents the behaviour for a certain datapoint. An explanation that is locally fidel does not have to be globally fidel [69].

Sensitivity is another quality property. This property shows how the explanations differ for similar instances [70]. Other words for this property can be 'robustness' or 'stability'. It measures how sensitive an explanation is to small changes in the input. A way to assess sensitivity is through a perturbation-based approach. Perturbing the input and observing the changes in the output explanation give information about the sensitivity of the original explanation [82].

Other examples of quality properties of explanations are consistency and certainty. The degree to which similar explanations are generated for different models trained on the same dataset by the same explanation method is measured by consistency [70]. Different models produce a similar prediction. Consistency measures the similarity of the explanations of each prediction that are provided by the same explanation method. Inconsistency can thereby show that the models might use different ways to arrive at the same prediction. Consistency as a quality property can also be used to perform feature selection, by choosing a model trained a minimal subset that shows consistency it is explanation with a model trained on a larger dataset. To quantify consistency, the shared knowledge between smaller and larger feature sets can be used [22].

Machine learning methods can show how certain they are of a prediction. But this certainty can be incorporated by the explanation method. Certainty is a quality property of the explanation. If a model is uncertain about a prediction, the explanation method can reflect this uncertainty [70]. For example, if a model predicts something abnormal for two different instances, one instance being 100% certain and the other 60%, the difference can be reflected in the explanation.

Each explanation method has advantages and disadvantages, and this is reflected through the explanation properties. Based on objective desiderata and the properties of the explanation, it can differ by classification problem, model and data set to which explanation method is suitable [8, 23, 46]. An explanation method can show differences in its objective properties if it has to explain from different models trained on different datasets [8].

Summary The scope of this thesis focuses on objective properties of explanations. The appearance of objective properties is dependent on the dataset and the model. This paper uses fidelity and sensitivity as objective properties. Fidelity is an objective measure used often to assess quality of explanations without a user. Sensitivity is a desired property because of the need to be able to handle small differences in anomalous data.

2.4 Explainable AI for Machine Learning and Deep Learning models

There is a relationship between the complexity of the model and the explainability of a model [28]. More complex models are more difficult to explain and understand compared to less complex models.

Within machine learning, there are shallow and deep models. Shallow models are models such as SVM, RF and Gradient Boosting Machines. Small Multilayer Perceptrons (MLP) are considered shallow as well. These models use predefined features for learning. Deep models, such as CNNs and AEs, require large amounts of data to learn features directly from the data. However, shallow models are not, by definition, more inherently explainable than deep models.

In recent years, several methods have been proposed to explain a variety of machine and deep learning models. One of such methods is DiCE (Diverse Counterfactual Explanations) [61]. DiCE shows counterfactuals as perturbed versions of the instance to be explained. The authors stress the importance of providing a set of counterfactuals with diversity. DiCE is not limited to binary classification; it works for multiclass classifiers. The generation of counterfactuals can be done through model-agnostic methods as well as gradient-methods, allowing DiCE to be used on neural networks as well.

Similarly to counterfactuals, there is Contrastive Explanation Method (CEM) [18]. The method has Pertinent Positives and Pertinent Negatives. For Pertinent Positives, the method finds the minimal features that must be present to predict the same class as the given instance. The Pertinent Negatives are the opposites: the minimal features that have to be absent in order to keep the same class of a given instance. The method is applicable to neural networks and shallow machine learning models.

There are other explanation methods that use a feature attribution or importance approach. Mentioned briefly already, LIME is a local model-agnostic explanation method [69]. Given input data and a black-box model, LIME generate a new dataset based on input perturbations and the corresponding output of the model. Based on this dataset a new local surrogate model is trained. The explanations contain the feature effects on a specific prediction of interest. With this approach, LIME does not need any information about the model internals, making the method widely applicable on different models [7, 36]. On the other hand, LIME uses a surrogate model to explain instances. The quality of the instances relies on the quality of the surrogate model.

Mentioned often together with LIME, another model-agnostic method is SHAP [52]. SHAP is based on shapely values that are used to calculate feature contributions to a prediction. SHAP has been used in a variety of tasks, using tabular, textual and image data. There are variations of SHAP, such as KernelSHAP and TreeSHAP. KernelSHAP is based on ideas from LIME and Shapely values [52], which is the default implementation of SHAP. As mentioned above, LIME trains a local model on perturbed inputs and their outputs. If this model is a linear regression model together with a weighting kernel, KernelSHAP uses the regression coefficients of this local model to estimate the SHAP values. TreeSHAP is a variation of SHAP intended for tree-based methods, such as Decision Trees, RF and Gradient boosted Trees [53]. The improvement of TreeSHAP over normal SHAP is the increased speed of calculating local explanations.

A difference from LIME and a drawback of SHAP is that the computational complexity lies higher due to Shapely values [36]. Although both LIME and SHAP are model-agnostic methods, their explanations can differ [52].

Previously mentioned, deep learning models are inherently unexplainable. There have been attempts to create explanation methods for deep learning models specifically. DeepLIFT [75] makes an attempt to explain deep learning models. DeepLIFT looks at changes in inputs through differences between inputs and reference scores for neuron activations. The difference in output is explained by the difference in inputs and reference scores.

Based on SHAP and DeepLIFT, DeepSHAP was developed[52]. Although KernelSHAP can also be used on deep models, DeepSHAP was developed for computational performance. The rules used in DeepLIFT can be chosen to approximate Shapely values.

Not only the type of model, but the type of learning requires different explanation methods. Within machine learning and deep learning, there are supervised and unsupervised models. Supervised models require labelled input during training, whereas unsupervised models do not. Some explanation methods require labelled input as training data to learn, which indicates that they are only suitable for supervised problems. Most methods mentioned before, DeepLIFT, DiCE and SHAP (and its variations), create explanations for supervised models. They require labelled output data to form an explanation.

Unsupervised models such as AEs and Generative Adversarial Networks (GANs) often require a different approach to provide an explanation. A key limitation of unsupervised models, such as AE, is their lack of interpretability and the difficulty of explainability. The latent space representation of an AE acts as a bottleneck for explainability as feature representations become uninterpretable.

Although explanations for AEs are difficult, there have been attempts to explain them. Attempts range from inherently changing the structure of the AE to model-agnostic models. One such method is GEE: a **G**radient-based **E**xplainable variational auto-**E**ncoder [64]. GEE uses gradients in the explanation of a model. The rationale is that anomalies can be clustered based on their gradients. Although GEE is trained in an unsupervised fashion, for the explanation, it does need labelled data on the type of attacks. This method is called semi-supervised.

Counterfactuals can be used to explain anomalies detected by AEs [31]. They mention the importance of reliability and add this as a constraint to the generation of counterfactuals. Reliability is based on robustness. The outcome of their explanation is a tuple with reliable counterfactuals.

Aside from explanation methods for models, supervised and unsupervised, there has been research to create inherently explainable models that show similar performances to their complex counterparts. One of such models is Explainable Boosting Machine (EBM) [65]. An EBM is a tree-based Generalised Additive Model (GAM) that is considered to be a white-box model. An explanation shows the contribution of each feature to the prediction. Although the model is highly interpretable, the authors claim it is just as accurate as black-box complex models.

Summary Methods such as LIME and SHAP are the model-agnostic post-hoc local explanations for the predictions of network anomaly detection with respect to this thesis. They are easy to implement and explain feature contributions. As an exception to model-agnostic methods, the EBM is an interpretable model that claims to have similar performances as more complex models, but with inherent explainability. Models such as AE show good performance without the need to use predefined features, but are still difficult to explain.

2.5 Network Anomaly Detection Machine Learning Techniques

Detection of network intrusions is often the second line of defence, after a firewall, and are detected by an Intrusion Detection System (IDS) [25]. IDSs are automated defence systems. They do not automatically protect the system, but can recognise the moment of an intrusion or after the intrusion has already happened. An IDS can be classified as a Network Intrusion Detection System (NIDS) [32], if it is an IDS for a network. A NIDS can be anomaly-based; then it is called network anomaly detection (NAD). NAD is the field of detecting anomalous traffic within a network that deviates from baseline behaviour. In principle, it can be any network, varying from industrial applications to private networks. Detection of anomalies is derived from behaviour on the network that is not normal. As NAD systems are often trained with normal behaviour only, anomaly-based systems have the advantage of being able to detect novel attacks [32].

NAD systems can be based on several techniques, such as rule-based, statistical, clustering and machine learning. Statistical methods include wavelet analysis and PCA [25]. KNN with clusters or K-means are clustering techniques that are used to cluster anomalous behaviour [25]. In the past, most NAD systems were rule-based or used statistical methods, but these options are outperformed by machine learning anomaly detection, especially on larger datasets. Using machine learning for anomaly detection is now state of the art [34]. Supervised and unsupervised machine learning techniques have been applied for NAD [32]. Shallow machine learning models, such as SVM and RF, have been examined for their use in anomaly detection [34].

Using shallow models alone, such as an SVM, might be insufficient for good performance in NAD [81]. An SVM is combined with a data transformation method called logarithmic marginal density ratio transformation (LMDRT). It shows improved performance over using the SVM alone, together with faster training speed.

Intrusion detection can happen as a two-stage approach [42]. In the first stage, the most representative train and test samples for the dataset are selected. In the second stage, selected samples are used to train a Least Square SVM (LS-SVM), which is a modified version of the SVM. Its performance shows an improvement over other two-stage approaches in the detection of four types of attacks.

Class imbalance can be a problem within network intrusion datasets. Difficulty Set Sampling Techniques (DSST) is proposed to deal with class imbalance [49]. This method is combined with a classifier. The performance between different class imbalance algorithms and models is compared in two datasets. RF in combination with multiple class imbalance methods shows better performance over other methods in almost all settings on the CSE-CCIC-IDS2018 dataset.

Although shallow methods perform well for the task of network anomaly detection, these approaches alone show false alarms and low detection rates [43]. Aside from shallow machine learning models, anomaly detection by deep learning models has been investigated.

Seven deep learning models are analysed on different intrusion datasets, together with RF, Naive Bayes (NB), SVM and Artificial Neural Network (ANN) [26]. The deep learning models are recurrent neural networks (RNN), deep neural networks (DNN), restricted Boltzmann machines (RBM), deep belief networks (DBN), CNNs, deep Boltzmann machines (DBM) and deep AE. The deep learning models outperform the shallow models, although the SVM and RF show comparable performance.

There are two-stage approaches that include deep learning. An example is a hybrid intrusion detection system consisting of two phases [43]. The first phase is normal anomaly detection that separates normal and attack traffic. The first phase was tested on various shallow classifiers such as RF and RF, with RF showing the best performance. This classifier detects normal and attack network packets. Attack traffic goes into the second phase, where a CNN+LSTM network further categorises the data. The strength of CNNs to find locally important features is combined with the ability of LSTM to capture long-term dependencies.

A problem with supervised anomaly detection is often the lack of labelled data. Labelling requires time-consuming manual labour. Another problem of a supervised approach is the continuous development of new attacks. New attacks create new anomalous patterns in the network, which the model has not seen before. Reliance on labels of previously known attacks can result in difficulties with real-time detection [77]. There have been attempts to tackle the need for labels. DevNet [66] is a neural network that tries to solve the lack of labelled data with weakly supervised learning. The model is able to learn to generalise and learn the representation of both normality and abnormality from a few labelled anomalies, with complete absence of labelled or unlabelled.

LSTMs are commonly used in anomaly detection, and they can be supervised, but also unsupervised. LSTMs can combined with One Class SVM and Support Vector Data Descriptin (SVDD) [24]. The advantage of the use of LSTMs is the ability of the method to process variable-length data sequences. This means that the method is suited for time series data or network packets collected from a longer period. The approach can be extended to work with GRU or RNN.

AE have shown promising results over the past few years in terms of anomaly detection. As labelling can be a labour-intensive task, the AE does not require labels. AEs can be trained on normal samples only. The rationale behind this is the low reconstruction loss for normal samples, but a high reconstruction loss for anomalous samples because the AE has not seen this kind of data before. An advantage of this approach is the ability to handle unseen data. As attackers evolve, the features or characteristics of attacks also evolve. AEs have been used in various ways, as the main model or as the dimensionality reduction method for feature representation[77].

The AE can be used for feature representation in combination with a SVM [68]. The idea behind this structure is to improve the classification of shallow learners with the help of deep learners. Deep learners can learn efficient representations of features that improve the classification of the SVM. The AE can be combined with an RF as well. The AE is used for the feature representation that is used as input for the RF. Anomaly detection is separated into two stages for multimodal anomaly detection. Two AE are used in the first stage, followed by a RF in the second stage [9]. Another approach contains two AEs and two detectors [83]. The loss of the first AE is normalised, followed by a global detector. The first detector detects whether there is an anomaly or not. If so, there is another AE with a global classifier to classify the type of threat and its properties. The AEs have LSTM layers for both the encoder and the decoder.

An early use of AE for anomaly detection is within a power transmission network [85]. They use the Stacked Denoising Autoencoder (SDAE), which is a deep neural network that consists of multiple DAEs. The input of one DAE is the first layer of the previous DAE. In this way, the whole SDAE can learn feature representations from different levels of abstraction. They evaluate their approach against KNN, RF and LR. All methods are outperformed by SDAE, although the RF is the closest in terms of accuracy.

An ensemble of AEs can be used to detect anomalous behaviour in network traffic. An example is Kitsune [58]. One of the important aims was to create a NAD system which was unsupervised because the model had to be able to be trained fast and perform online processing.

Summary Research on NAD models has evolved with the increasing popularity of deep learning. From statistical and rule-based to shallow and deep learning models for NAD. Although shallow machine learning models are an improvement over statistical and rule-based models, alone they are often outperformed by complex deep learning models such as the AE. The AE has been a good fit for network anomaly detection due to its ability to learn the notion of anomalies without labelled data. For multiclass classification, the AE can be combined with shallow models.

2.6 Explainable AI for Network Anomaly Detection

With the rise of machine learning methods in the field of cybersecurity, the demand for explainable models has risen equally. The importance of XAI for cybersecurity is for several reasons, such as transparency and trust [86], improving the performance of classifiers and explaining errors [12]. By explaining errors, understanding missclassifications can help experts understand the model and implement necessary changes to overcome these errors, with the goal of improving model performance. Relevance of features as an explanation most commonly used in detection of network anomalies [12, 79]. There are studies that propose new explanation methods [1, 30, 86] and studies that incorporate already existing methods into their approach [2, 12].

TRUST XAI (Transparancy Relying Upon Statistical Theory XAI) is proposed by distinguishing the primary AI model and the underlying XAI model. The primary model is in charge of classification while the XAI model is responsible for providing an explanation of the primary model's behaviour. They can be separate models and the same model (using an interpretable model). TRUST modifies the feature values to remove redundancy and interdependencies. Using mutual information, the key features per class label are determined and called 'representatives'. The statistical behaviour of the representatives is modelled with their density functions. The method is compared to LIME and shows to be faster and more accurate. A limitation of the approach are the representatives picked by mutual information, which might lead to overfitting on the train data. Another limitation is the lack of users. They claim that their approach outperforms LIME, but they do not validate the actual perceived difference in their target users, who are security analysts.

LEMNA (Local Explanation Method using Nonlinear Approximations) was designed specifically for security applications and is an explanation method for black boxes [30]. The authors of LEMNA aimed to create a method with high fidelity, as they note that existing methods suffer from low fidelity.The method provides a set of key features for the classification result. To improve fidelity and allow for flexibility, LEMNA can be used for linear and nonlinear decision boundaries. The use case shows the intention of the authors to provide a method that helps security analysts to understand classification errors and fix the errors. They measure fidelity with 3 self-made fidelity approaches.

Many approaches within XAI for NAD incorporate SHAP [12], with most of these studies aiming to find feature importances. The end goal of finding the important features differs. Information about features themselves can be the goal, but insights in the features is used to improve model performance. Another approach uses SHAP to investigate the features three different models (RF, XGBoost and Keras Sequential algorithms) use to classify five different attack types [2]. The main focus is to find out if all models are suited for the task of NAD.

SHAP is used to evaluate the classification results of RF and a feedforward neural network on multiple intrusion detection datasets [73]. The aim was to find the key features that contribute to the prediction results. The authors conclude that per dataset similar features have different contributions to the same attack type, but there are similarities between the key features.

Mentioned in chapter 2.4, AEs are difficult to explain due to their inherent reduction in dimensionality of the features, but are frequently used in the field of NAD [77]. Current available methods try to explain an AE by alternating the architecture of the model itself, which can be seen as the state of the art for explaining AE. Previously discussed in chapter 2.4, the authors of GEE try to create an inherently explainable AE [64]. Aguilar et al. [1] created an AE based on a decision tree, making the model more explainable. Although the model was originally designed to work for the detection of visual anomalies of breast cancer, the model is able to handle categorical and numerical data.

In [5], they propose an approach using SHAP in an AE that is based on the robustness of the explanation methodology. They aim to reduce the reconstruction error through feature perturbations. The study shows that the reconstruction error is reduced more when SHAP is incorporated, instead of LIME.

Although the methods claim to provide explanation or improve trust, explanations are rarely tested subjectively or objectively. Many studies claim to support or increase user trust or explainability, but often no metrics are provided to support the claim. Explanations and explanation methods are often compared with others in terms of speed, visuals or ease of computation [86]. Model performance can also be an indicator that studies use to validate their explanations [2]. This lack of both subjective and objective evaluations is not specific to the domain of NAD [50]

Although subjective evaluations do not occur often in this field, but they do happen. In [51], the authors conduct a human-grounded evaluation with experts to investigate the influence on trust of their designed visual explanation. A more interactive approach of XAI for NAD is approached through the creation of an agent that functions as a junior analyst, to support a senior analyst [35] . Through interaction, the junior analyst agent can support the senior analyst task. The evaluation was carried out through feedback sessions with expert users.

Similarly to subjective evaluations, objective property evaluations do occur in XAI for NAD, but are rare. LEMNA uses fidelity to measure its explanation performance [30]. Robustness is one of the most used quality properties in evaluating XAI for NAD [12]. Mentioned before, [5] use robustness as a quality measure of an explanation method to decrease the reconstruction error. Similarly, robustness is used to verify the proposed frameworks to extract the most dominant features to improve the performance of the model in fingerprinting of the website [29].

Summary There are studies on explainability within the field of NAD, but many of them lack metrics that support their claims. Model improvement or feature investigation are two of the most common goals of XAI in the field of NAD. Although there are methods that incorporate quality properties, this is a rare stand-alone approach to explanation methods within the field of NAD.

Chapter 3

Methodology

The proposed method aims to investigate the right balance between model performance and model explainability. To examine the performance differences between the interpretable and uninterpretable models, three different models are compared. Two black boxes are implemented and one white box: the Random Forest (RF), the Autoencoder (AE) and the Explainable Boosting Machine (EBM). The models are trained on versions of the UNSW-NB15 dataset. The different versions of the dataset are the result of feature analysis or upsampling techniques. Model performance is examined through selected metrics. For each model, there is a different explanation method. For the Random Forest, LIME and SHAP are examined as model-agnostic explanation methods. The Autoencoder is explained through a method using neighbourhood differences translated into feature importances. The Explainable Boosting Machine is inherently explainable and does not need an additional explanation method. The explanations are examined through objective properties, namely sensitivity and fidelity. A final conclusion is drawn by combining the information of both steps, as well as general information about the dataset, models and methods. An overview of the process can be seen in Figure 3.1.

3.1 Dataset

The dataset of interest is the UNSW-NB15 dataset, which contains data on network traffic [63]. It was created as a response to other datasets (e.g. KDD-99) that were often used as benchmarks for NAD, but were no longer

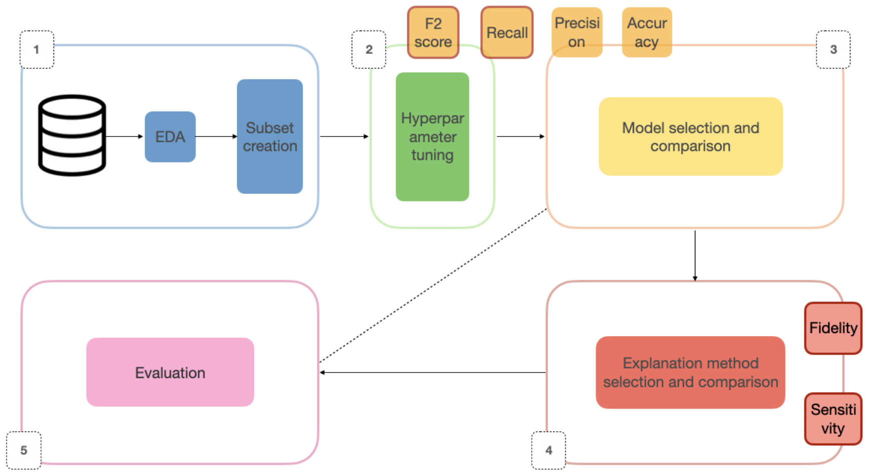


Fig. 3.1. Visualisation of the process in separate steps. **1.** The dataset phase, with exploratory data analysis and subset creation. **2.** The hyperparameter tuning stage to choose the optimal model settings. **3.** Final models performance evaluation and comparison. **4.** Given the models from phase 3, implementation and evaluation of explanation methods. **5.** Evaluation phase, where the results from model and explanations separately are combined

representative of current network traffic. The dataset contains data that was collected during 15 hours of network traffic of three networks. It is publicly available in CSV format. The dataset contains 49 features of which 47 can be used for classification. The other two are the label and the attack category. The normal samples in the dataset have label 0, while the abnormal or malicious samples have label 1. The column *attack_cat* gives specific information about the kind of attack. There are 9 attack types: Analysis, Backdoors, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode and Worms. Although the dataset contains normal and abnormal samples, there are more normal samples (2,218,761) than anomalous samples (321,283).

This dataset is used for this thesis because it contains a rich feature set for network anomaly detection and consists of up-to-date network traffic information. The full list of feature meanings can be found in the original article of the dataset [63]. This dataset has been used to evaluate the performance of NAD models [66, 84], as well as in comparisons to other datasets [74].

To work with the selected models, the dataset has to be prepared. Missing values have to be filled in and categorical features have to be converted to numerical. To find the optimal dataset, different versions are examined, using different numbers of features or upsampling techniques to balance normal and abnormal samples.

3.1.1 Attack Types

There are 9 attack types in the dataset. They are briefly explained below.

- **Analysis**. An attacker tries to gain access to a network to listen to and capture data on the network. Through analysis, the attacker can find valuable information.
- **Backdoor**. With a backdoor attack, the attacker tries to bypass security to gain unauthorised access to a system.
- **DoS**. DoS stands for 'denial-of-service'. The attacker tries to make the network unavailable to its users by disrupting the server.
- **Exploits**. The attacker is aware of a vulnerability in the software of the system and tries to exploit this in its attack.
- **Fuzzers**. The attacker injects semirandom data into a programme to detect the output. With this, the attacker tries to find implementation bugs to exploit.
- **Generic Attack**. Data storage and communication are often encrypted, which the attacker tries to decode.
- **Recoinnaissance**. With a recoinnaissance attack, the attacker tries to collect as much information about the system as possible, to enlarge the attack surface.
- **Shellcode**. Shellcode is a set of instructions in code format that executes a command in software to take over control of the software or the machine.
- **Worms** A worm is a form of malware that can replicate itself from one computer to another. A worm in a network tries to find computers connected to the network to replicate itself. A worm attack can steal sensitive information by installing a backdoor.

3.2 Models

The models used during this study are a Random Forest, an Autoencoder and an Explainable Boosting Machine. The RF and the EBM are shallow machine learning models, while the AE is a deep learning model. Using the three different models, the balance between model complexity and explainability is evaluated.

3.2.1 Random Forest

A Random Forest Classifier is a supervised machine learning method consisting of an ensemble of decision trees [10]. Individual trees in the ensemble differ through the use of bagging. Bagging (Bootstrap Aggregating) means that each tree samples N instances with replacement. This allows each tree to have a different training set, forcing diversity among the trees in the forest. In addition to bagging, each tree is allowed to select the optimal feature for a split from a subset of features instead of the entire feature set, which equally enforces diversity. The diversity of both methods contributes to a reduction in the risk of overfitting the whole forest. Each individual tree makes a prediction, and a majority vote is taken for the final prediction. The approach of RF makes it a stable method that is not sensitive to outliers.

When comparing shallow machine learning models for NAD, the RF often outperforms other methods [43, 49]. Additionally, compared to neural networks, the RF can outperform them in tabular data [27]. The RF is a shallow machine learning model that needs model-agnostic explanation methods. It is essentially more explainable than AE because it does not compress feature representations. This is why RF is included in model selection.

3.2.2 Explainable Boosting Machine

The Explainable Boosting Machine [65] is not a black box but a transparent model. It can provide local and global explanations. EBM is an improved version of a Generalised Additive Model (GAM). A GAM learns feature functions per feature and adds the results together.

The EBM uses gradient boosting and bagging. With gradient boosting, multiple simple models are created to form a stronger ensemble. Each simple model is trained using one feature at a time. For each K features, T simple models are trained, resulting in $K * T$ simple models. Including bagging, each model is trained on a subset of the samples. Combining all T simple models for one feature, the EBM learns the best feature function per feature to show the contribution per feature to the output. Aside from separate feature functions, the EBM can learn pairwise feature functions. The final prediction consists of additive feature values, and the impact of features on the prediction can be visualised as an explanation.

The EBM balances accuracy and model complexity. It is less complex than gradient boosting or Random Forest, but the authors claim that EBM performance is similar to more complex models and directly interpretable.

Because the explanation of an EBM relies on the internal structure of the model, inference is fast. However, training an EBM can be computationally expensive, due to the additive nature.

EBM shows a feature impact, which falls in a category similar to LIME and SHAP. The difference between EBM and the two model-agnostic explanation methods is that the feature impacts are not generated through a surrogate model. To investigate the differences in model performance and explainability between an inherently explainable model and a black box with an additional explanation method, the EBM is one of the three models in this research.

After the onset of this study, an article was published investigating the EBM for network intrusion detection. The results show that the EBM outperforms a Support Vector Machine [56].

3.2.3 Autoencoder

An Autoencoder is a neural network with the goal of reconstructing the output from compressed representation of the input in an unsupervised manner [71]. The network is composed of an encoder and a decoder. The layer between these parts is called the 'latent space' or 'bottleneck'. The encoder maps the input to a representation in the latent space. The decoder reconstructs the input based on the representation in the latent space. The reconstruction is evaluated with a reconstruction loss that measures the difference between the input and the reconstructed output. To ensure that the encoder learns a compressed representation, the dimensions of the latent space are often smaller than the input. If the number of dimensions in the latent space is too large, there is a risk of directly copying the input to the output without learning representations. The encoder is responsible for encoding the input features into representations with lower dimensions, which is why a trained encoder can be used separately for dimensionality reduction [77].

For binary classification, the reconstruction error of the AE can serve as a threshold. The threshold can be a set value or an adaptive value. In this thesis, an adaptive value is chosen as a threshold. The threshold for the reconstruction error is calculated by taking the 90th percentile value of the reconstruction errors. If the reconstruction error exceeds this value, a sample is marked as malicious. An adaptive value is preferred over a static value, as a static value requires prior knowledge about the distribution of the reconstruction error. The 90th percentile is specifically the result of adhering to the default implementation of the AE used.

For the AE, hyperparameters have to be tuned to work optimally on the dataset. However, tuning hyperparameters for a neural network is a difficult task in general [19].

The AE is chosen as a model in this thesis as it has shown 1) to outperform other machine learning and deep learning approaches for NAD and 2) the ability to adapt to new cyberattacks [77].

3.2.4 Autoencoder + Random Forest

The fourth model framework is a proof of concept. Two models are combined: the Autoencoder and the Random Forest. The combination of two models in the field of NAD has been done before [43]. More specifically, the AE has been combined previously with a simpler model such as the RF or SVM [9, 83].

Both the AE and the RF have their advantages. The AE is able to predict unseen data through a heightened reconstruction error. This is beneficial in the field of NAD, as it does not require a dataset to contain labelled malicious samples. The RF performs well on tabular data [27] and can outperform other shallow machine learning models [43, 49].

Combining AE with RF allows training the AE on a dataset with normal samples only and the RF on a smaller dataset with malicious samples.

Figure 3.2 shows the approach. Both the AE and the RF are trained individually. The AE is trained on normal samples only, while the RF is trained on malicious samples from the same dataset. This dataset has to contain enough samples for the RF to be able to correctly predict malicious classes.

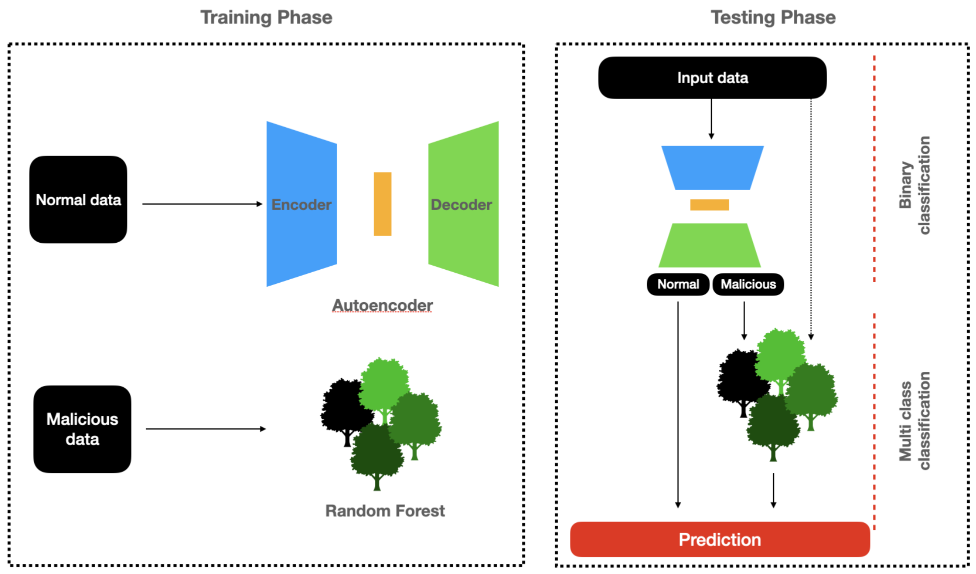


Fig. 3.2. The AE-RF setup of the train and test phase. In the train phase, both models are trained on separate datasets. In the testing phase, the AE makes the first binary prediction. If the prediction is malicious, the RF predicts the specific class.

In the testing phase, the AE receives a sample. It predicts if the sample is normal or malicious. If the sample is predicted as normal, this is the final prediction. If the sample is malicious, the RF uses the original features of the sample for a second prediction. The original features could be used for explanation purposes.

Although the complete model is implemented as a proof of concept, only the RF trained on the malicious samples is used for explanation purposes.

3.2.5 Model evaluation

There are two model selection stages in this process: the hyperparameter phase and the final model selection and evaluation phase. In both phases, the same performance metrics are used. The metrics used for model performance are accuracy (Eq. 3.1), recall (Eq. 3.2), precision (Eq. 3.3) and F2-score (Eq. 3.4). Recall and F2-score are noted to be more important than accuracy and precision. Recall is a score to determine how many of the true instances are actually detected. It can be called the *detection rate*. Damage done by intruders can have disastrous effects and therefore the classifier has to be sensitive for anomalies. This is why high recall is preferred over high precision. However, high recall in combination with a too low precision is not desired, therefore the F2-score is chosen. The F2-score accounts weighs recall more heavily than precision, but a precision that is too low will affect the score as well.

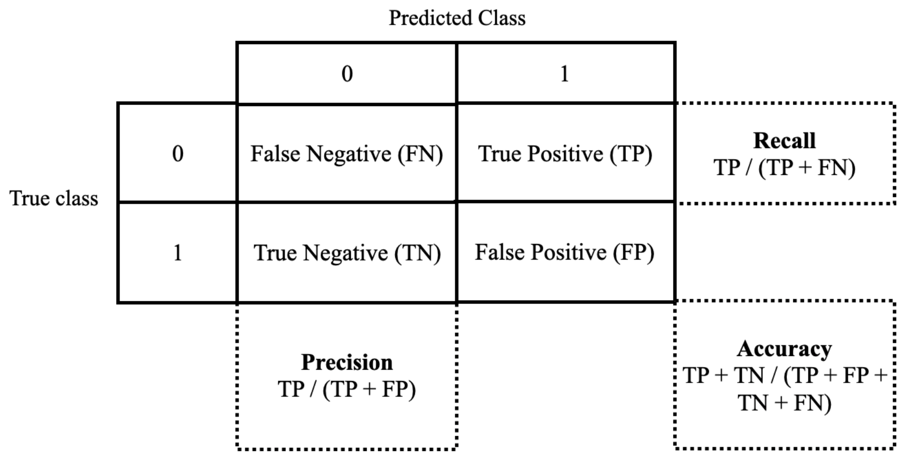


Fig. 3.3. Confusion matrix overview. 'Sensitivity' is a different name for 'recall'

$$Accuracy : TP + TN/(TP + TN + FP + FN) \tag{3.1}$$

$$Recall : TP/TP + FN \tag{3.2}$$

$$Precision : TP/TP + FP \tag{3.3}$$

$$F2 - score : 5 * \frac{(Precision * Recall)}{(4 * Precision + Recall)} \tag{3.4}$$

The metrics are based on values that can be extracted from a confusion matrix. The confusion matrix and three of the metrics are shown in Figure 3.3.

3.3 Explanation Methods

There are two model-agnostic explanation methods in this study: LIME and SHAP. The EBM is inherently explainable, which indicates that it does not require an external explanation method. The three explanation approaches show feature importances, which is the contribution or importance of each feature for a prediction. To stay consistent, the explanation of AE tries to show the importance of features. This explanation is not based on a surrogate model as for the other two.

3.3.1 SHAP

SHAP is a model-agnostic explanation method based on the idea of a surrogate model [52]. SHAP can be used locally or globally, but this study focuses on local use. SHAP itself is based on the notion of Shapely values, which originates from Game Theory. Shapely values consider every subsets of combinations of features to determine the contribution of a single feature to the outcome of the model. The difference between two subsets of features, where one set has one additional feature over the other set, determines the marginal contribution of one feature. Marginal contributions can be added together, as SHAP is an additive feature attribution method to create a weighted average, similar to a *weighted linear regression*. The final result shows the impact of the feature on the prediction of the model compared to the absence of the feature. The SHAP explanation must adhere to local accuracy and consistency.

A disadvantage of SHAP is that it can be computationally intensive, especially for large datasets or complex models. Another disadvantage is the lack of precision, due to the restrictive properties [21].

To evaluate possible explanation methods for models in the field of NAD, SHAP is one of the explanation methods used. This study uses TreeSHAP, which is an optimised version of SHAP for tree-based methods. SHAP is a method that is easy to implement, model-agnostic and well-known, which is why it is used as one of the explanation methods.

3.3.2 LIME

LIME is a model-agnostic explanation method for local explainability [69]. Similar to SHAP, LIME uses a surrogate model to explain a black box. The idea behind LIME is based on perturbations of the input. The instance to be explained is first converted to a binary vector, since the authors note that this is an interpretable representation. Around this instance, samples are drawn. Using small perturbations, they generate outputs from the original black-box model. With small perturbations as input, they collect the outputs of the original model and use the weighted collected samples to train a linear interpretable model to estimate the decision boundary locally. The explanation has to adhere to local fidelity.

Similarly to SHAP, LIME can suffer from lack of precision due to restrictive criteria [21]. Another known downside of LIME is that the explanations can be sensitive to small perturbations in the input data [21], although it can depend on the model if it is more or less robust than SHAP [46].

Both SHAP and LIME are feature attribution explanations, which means that the explanation provides information about the contribution of the features [14]. Although both aim to provide feature attributions, the two can give significantly different values [52]. Equal to SHAP, LIME is easy to implement, model-agnostic and well-known. The two are often compared, and this thesis follows that trend.

3.3.3 Explaining the Autoencoder

The AE cannot be explained in a similar way as the RF or the EBM. The AE reduces the feature dimensions of the input feature as part of the encoder-decoder structure. The latent dimension represents the reduced dimensionality of the features. Research is still ongoing to find a universal approach to map each feature to a latent dimension and back. However, following the approach of feature importance as explanations, this study tries to approximate feature importance of the AE predictions.

The feature importances are based on a perturbation-based method using neighbourhood differences of output points. The idea is based on the thought that altering an important feature leads to larger neighbourhood differences. Figure 3.4 shows the approach.

A Kd-Tree is used to organise the original output points in space. These are multiple points from a test set. The original instance is the instance to be explained. The original neighbourhood is the neighbourhood of that point in the output space. The number of neighbours is determined by combining the elbow method and the silhouette score, shown in Figure 3.5. Using both methods together, the number of neighbours is set at 50. Every feature

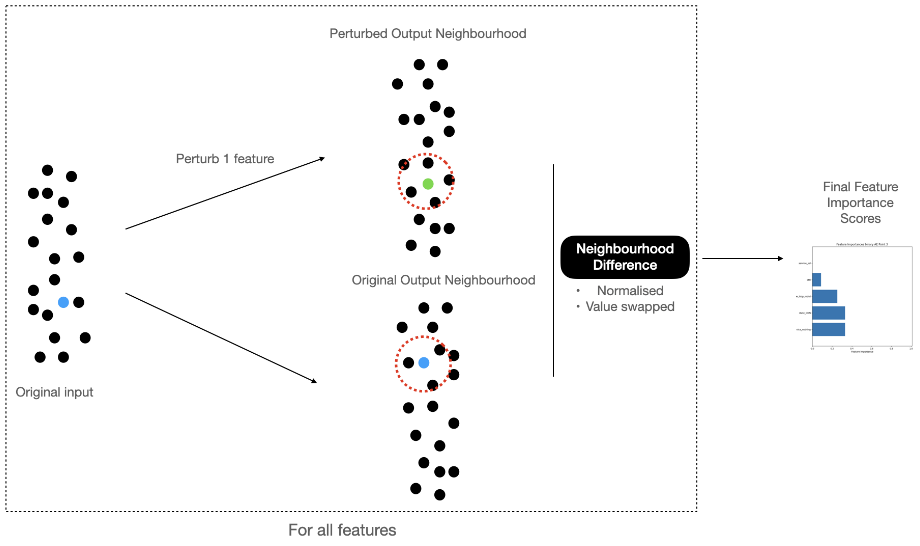


Fig. 3.4. The process of generating feature importances for the Autoencoder. Per feature, difference between the original output neighbourhood and perturbed output neighbourhood is calculated. After this, the feature importances are based on normalised and swapped Jaccard Similarity.

of the original instance is perturbed, and the new output neighbourhood is calculated. The original neighbourhood is compared with the perturbed neighbourhood through Jaccard Similarity and this gives a similarity score. The Jaccard Similarity is a metric of overlap two lists of points, which returns 1 if the two are completely similar and 0 if they are completely dissimilar. The Jaccard values are normalised and swapped, giving an important feature a high value instead of a low value. This value is the importance of the feature used for the AE.

3.3.4 Objective Explanation Evaluation

Explanations will be evaluated on fidelity and robustness properties. Both properties are measured quantitatively, because they do not depend on the subjective experience of the target user.

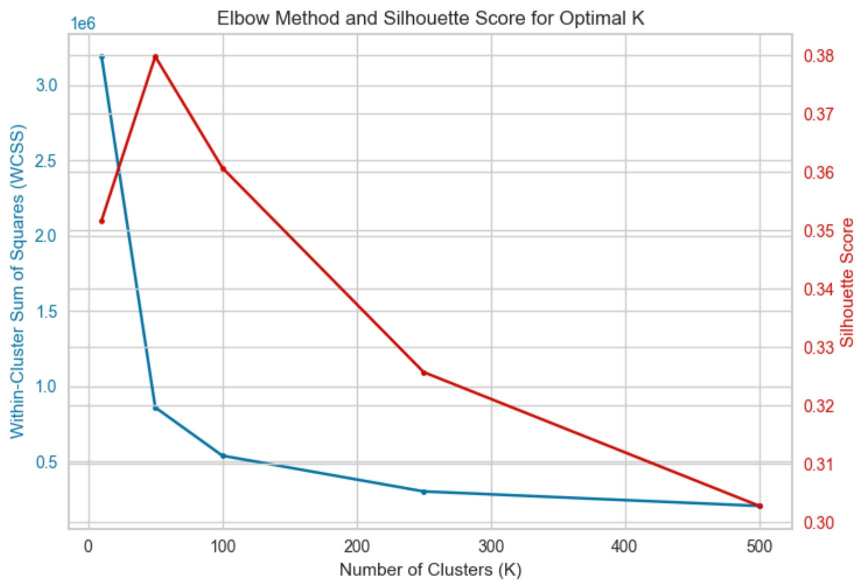


Fig. 3.5. The Elbow-Silhouette combination to determine the value of K neighbours

Sensitivity

Sensitivity, often referred to as robustness, is the property of an explanation that reflects the output explanation behaviour resulting from variations in the input [8, 12, 20].

Sensitivity can be evaluated by examining the similarity between an original explanation and an explanation in which inputs are slightly perturbed. Sensitivity can be seen as the stability of the output after small, non-significant perturbations [82]. The stability of the output can be examined by similarity, through a distance metric, for example.

In this study, the sensitivity is measured using the Euclidean distance in the feature space. With feature space, the feature importances are meant. Take Random Forest and SHAP as an example. There is an original sample x with output y and explanation e. This sample has a top 5 of features. For every feature f of sample x, the input is perturbed separately, creating $x_{perturbed}$. This provides a perturbed output $y_{perturbed}$ with a new explanation $e_{perturbed}$. Then the Euclidean distance is used to determine the distance between the top 5 features of x and the same features of $x_{perturbed}$. The distance is divided by the difference between x and $x_{perturbed}$ and this gives the sensitivity for the explanation of one sample. The equation is shown in Eq. 3.3.4. This process is repeated for every of the top features of the explanation. To calculate the average sensitivity value over multiple explanations, the sensitivity values are averaged.

$$Sensitivity = \frac{(\mathbf{d}(x, x\_perturbed))}{(|x - x\_perturbed|)} \tag{3.5}$$

A low value means a low sensitivity as the distance between features in the feature space is small. It shows that a change in the input did not lead to a large change in the output. A high value means a high sensitivity.

Sensitivity is a property that can be measured quantitatively and supports objective determination of the quality of the explanation [8]. Therefore, the sensitivity is used for the evaluation of the explanation method. It is specifically used in this study, as the assumption is that stable explanations are valuable in NAD. If an explanation is not stable, it is difficult to follow a model's decision. This makes it difficult to use the results of the explanation to understand or improve anomaly detection.

The expectation is that LIME is more sensitive than SHAP, due to results from previous studies [8, 21]. The EBM is expected to be comparable to SHAP. The sensitivity of the AE explanation is difficult to forecast because it is not a method that has been studied before. It is expected to be more sensitive than the other methods, as the method was not developed with low sensitivity as desiderata.

Fidelity

Fidelity is a measure of the correctness of the explanation method in generating a true explanation for the model's decision [59, 82]. In other words, can be seen as the ability of the explanation to capture the underlying reasoning of the model [14]. Fidelity is desirable, as good explanations give insights in the true reasoning, otherwise users can be misleaded when they use the model [14]. In this thesis the following notion of fidelity is adapted: fidelity shows model trustworthiness [82].

Fidelity and accuracy are related under the notion correctness. If an explanation is highly fidel and the model's accuracy is high, then the explanation is highly accurate [60].

Fidelity can be measured through a perturbation-based approach. For every feature f of the top 5 features of the original instance x, a significant perturbation is made [82]. With $x_{perturbed}$, the expectation is that if the feature were truly significant, it would change the prediction. A perturbation can be made based on sampling from the normal distribution of the feature. The strength of the perturbation affects the sampling. A high perturbation strength means sampling values from the outer sides of the distributions. A low perturbation strength means sampling values from the distribution that occur more often.

For the final fidelity score of the local explanation, the number of changed predictions is divided by the number of perturbed features. The equation is shown in Eq. 3.3.4. The fidelity represents a percentage of changed predictions of the total perturbed features. The average fidelity over multiple explanations is calculated in a similar way to the sensitivity. Scores are added and averaged.

$$Fidelity = \frac{changed\ predictions}{total\ perturbed\ instances} \tag{3.6}$$

Similarly to sensitivity, fidelity is a property that can be measured quantitatively [82]. For the selection of the best explanation method in combination with the model, fidelity is used to objectively make this selection. Fidelity is chosen as an important objective property for explanations for NAD for a reason similar to sensitivity. If the explanations are not fidel, the explanation is of no use to understand the model's prediction. It cannot contribute to any improvements in anomaly detection.

As both sensitivity and fidelity measure changes as a result of perturbations, perturbation strengths are included for fidelity. Higher perturbation strengths are expected to lead to higher fidelity scores for all explanation methods. As the EBM is inherently explainable, the expectation is that this method yields the highest fidelity scores. The fidelity of LIME and SHAP are expected to be similar [80], although other research has shown that the fidelity of LIME and SHAP can differ depending on the model or the dataset [8]. As the AE shows feature attributions, the values could lie close to those of LIME and SHAP. On the other hand, the AE explanation is not optimised to adhere to certain properties.

Number of samples

Evaluation of the whole test set is computationally too expensive. Therefore, a sample size is determined for the evaluation of the explanations. A test was done with RF-Basic SHAP and LIME to determine the number of samples to choose between 10, 100 and 1000 samples. Sensitivity and fidelity with perturbation strength 1 are indicated in Figure 3.1. These results show that there are no large performance differences between 100 or 1000 samples. To account for possible fluctuations, 100 samples over 10 are preferred.

Model and Explanation	**10**	**100**	**1000**
Sensitivity			
RF-Basic SHAP	0.0781	0.0613	0.0634
RF-Basic LIME	0.0501	0.0447	0.0419
Fidelity			
RF-Basic SHAP	6.06	7.77	6.26
RF-Basic LIME	4.71	5.87	4.44

Tab. 3.1. Sensitivity and fidelity scores of 10, 100 and 1000 samples on RF-Basic with SHAP and LIME

3.4 Model and Explanation Implementation

The models and the explanation methods are implemented using existing code with adjustments where necessary.

- Autoencoder: The Pyod library [1] is used, which has been used for earlier implementations of AE for anomaly detection [33]. Small alternations are made to make the latent space accessible and extract values.
- Random Forest: the scikit-learn package[2].
- EBM: the EBM is designed by InterpretML [65] and the original implementation is used [3]

[1]https://github.com/yzhao062/pyod
[2]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[3]https://github.com/interpretml/interpret

- SHAP (TreeSHAP): the Alibi library [4]. This library allows for easy extraction of feature importances. It is based on the original paper [52].
- LIME: the LIME package [5] based on the original paper [69].

The Snellius cluster is used for a limited amount of hours to train the models. The rest of the model training is done on a MacBook Air 2020 with the M1 chip.

[4]https://docs.seldon.io/projects/alibi/en/stable/examples/overview.html
[5]https://github.com/marcotcr/lime

Chapter 4

Dataset and Hyperparameter Analysis

To evaluate the model and explanation performances, the dataset has to be pre-processed and the models need to have the optimal hyperparameters. First, the dataset is explored and analysed. With information from the analysis, the dataset is cleaned and different subsets are created. Second, multiple versions of the models are trained on the subsets to find the optimal hyperparameters for the type of model on the task. This results in one version of RF, EBM and AE per dataset. Per dataset, the version of the optimal model can differ.

4.1 Exploratory Data Analysis and Cleaning

The dataset consists of 2,540,043 records stored in 4 separate CSV files. To explore the dataset, the four separate CSV files are concatenated. There are 49 features in the original dataset, which gives the dataset the shape of (2540043, 49). The last two features are *label* and *attack_type*. The full list of feature abbreviations and descriptions can be found in the original article of the dataset [63], but in the text there are supporting notes about the features to understand the exploratory data analysis.

Network traffic travels in packages from source to destination. These packages have specific characteristics, and each packet consists of bytes of data. Many features include information about packages that are transferred between the source and destination. The 's' in front of a feature represents

’source’, while ’d’ represents ’destination’. For example *sbytes* is the name of the feature that represents the bytes that go from the **s**ource to destination, while *dbytes* represents the opposite (**d**estination to source). Additionally, features with ’ct’ in their name count the occurrences of a specific characteristic of a connection over 100 connections.

The dataset is unbalanced, which is visualised in Figure 4.1. In total, there are 2,218,76 normal samples against 321,283 anomalous samples.

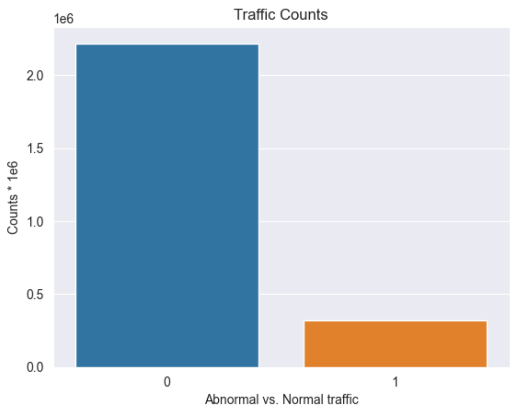


Fig. 4.1. Traffic counts normal (0) vs. abnormal (1) data

4.1.1 Attack types

Within the dataset, there are 9 different attack categories. A small description of each attack type is given in Section 3.1.1. The names and their absolute occurrences are presented in Table 4.1. These numbers are visualised in Figure 4.2.

Table 4.1 and Figure 4.2 show that within the anomalous class, the attack types are not balanced. ’Generic’ is the largest anomalous group with 215,481 instances, while ’worms’ is rarely present in the whole dataset with its 174 instances.

4.1.2 Preprocessing

Before evaluating the features, pre-processing was performed on the dataset, such as investigating the presence of missing values (Nan or null). There were

Attack name	Numbers
Generic	215,481
Exploits	44,525
Fuzzers	24,246
DoS	16,454
Recoinnaissance	13,987
Analysis	2,677
Backdoor	2,329
Shellcode	1,511
Worms	174
Total	321,283

Tab. 4.1. Overview of the number the different attack types present in the UNSW-NB15 dataset

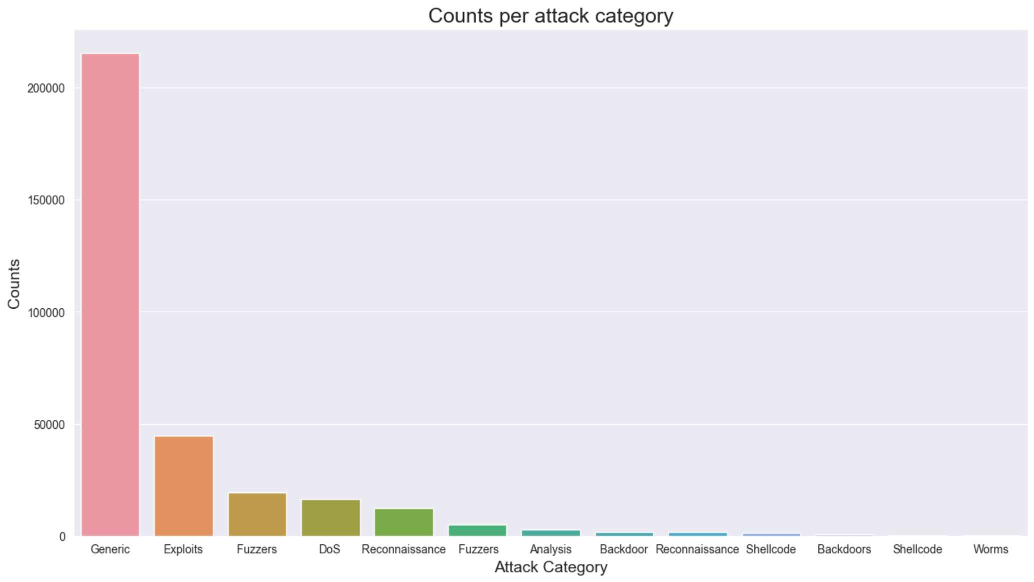


Fig. 4.2. Ordered counts per attack category. The last three categories are difficult to visualise, because they have less occurrences compared to the largest class 'Generic'

three columns where Nan or null values appeared: *attack_cat*, *ct_flw_http_mthd*, *is_ftp_login*.

For *attack_category*, Nan appeared in the attack category for every normal instance. This was replaced by the string 'normal'.

The *ct_flw_http_mthd* had Nan values in the rows where a 0 originally appeared. This is probably a result of conversion. The Nan values were replaced with 0.

The other feature, *is_ftp_login*, is a binary column. However, Nan values appeared in the rows where a 0 was supposed to be. The Nan values were replaced with 0.

Regarding other categorical columns, *service* feature could have the value '-' when there was no service. This was replaced by the word 'nothing'.

The feature *ctp_ftp_cmd* had missing values where the original 0 appeared. It is an integer column, so missing values were replaced with 0.

4.1.3 Feature exploration

To explore the features in the dataset a correlation heatmap was created, displayed in Figure 4.3. The heatmap is not mirrored. The heatmap uses the Pearson correlation as default. The values can range from 1 to -1. A high positive value means a positive correlation, while a low negative value means a negative correlation.

From the heatmap, there are several observations made. The first observation is that features that are highly correlated with other features are often related to each other. An example is *sbytes* with *sloss* or *sttl* and *ct_state_ttl*, which are shown in Figure 4.4 and 4.5.

Secondly, the features *sttl* and *ct_state_ttl* show a high correlation with the label. Related to these features is *dttl*, which does not show a high correlation with the label or with the other two TTL features. The TTL features will be explored in detail later.

Lastly, most features with 'ct' in their name appear to have a higher correlation with the label compared to other features. These features capture serial data over 100 connections, which could be more informative than one record alone.

The following sections highlight some feature correlations. Not all features that were explored are shown, as they show similar patterns as other features or show no pattern at all. Other articles have done more extensive research on the features of the dataset, which are used for the choices made [62, 74, 73]

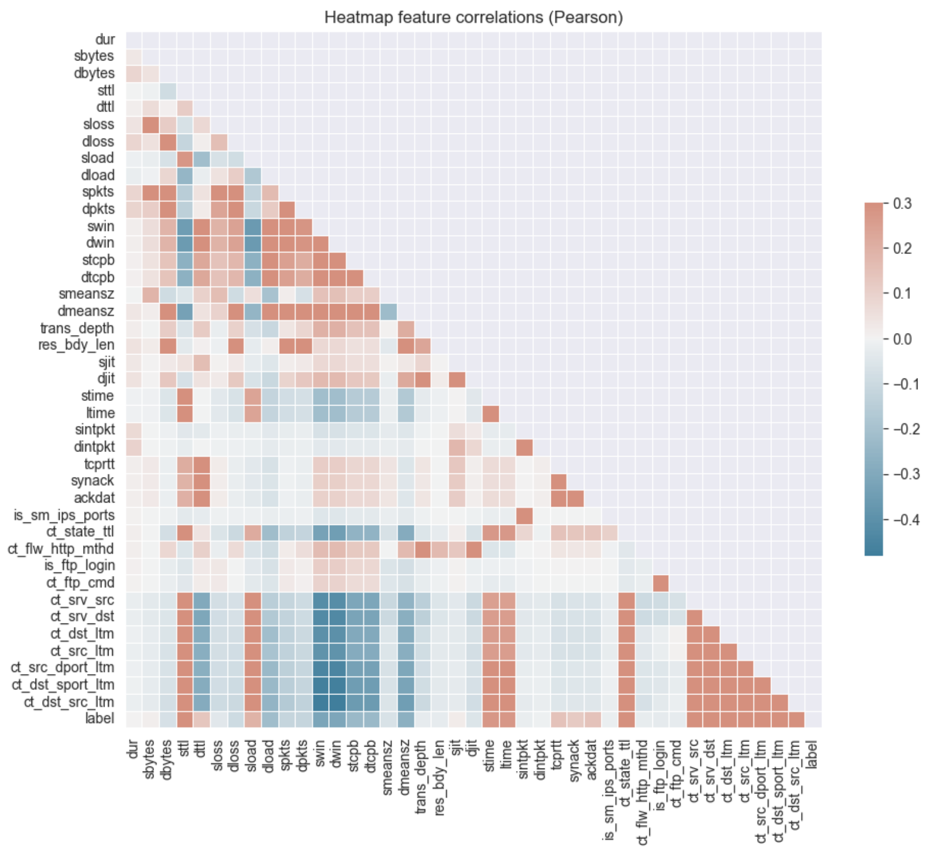


Fig. 4.3. Heatmap of feature correlations using Pearson. Positively correlated features lean towards the orange colour, while negatively correlated features are coloured blue.

Bytes and Loss feature correlations

Using a pairplot, correlation is explored between two features that are highly orange in Figure 4.3 and are related to each other. 'bytes' features include the source to destination bytes of packets or vice versa, while 'loss' represents the loss of packets dropped from source to destination or vice versa. The number of lost packages influences the number of bytes, as lost packages decrease the number of bytes.

sbytes and *sloss* show a high correlation in Figure 4.3, similarly for *dbytes* and *dloss*. The correlation for both combinations is represented in Figure 4.4. The first graph in Figure 4.4 appears to show a linear relationship between the *sloss* and *bytes*, beside the additional points at the beginning of both graphs. A similar pattern can be found for *dbytes* and *dloss* in the second graph of the figure.

Given the correlations in Figure 4.3, other related features were explored and show a similar pattern, such as *sloss* and *spkts* (packet count) or *dmeansz* (mean of flow packet size) and *dbytes*.

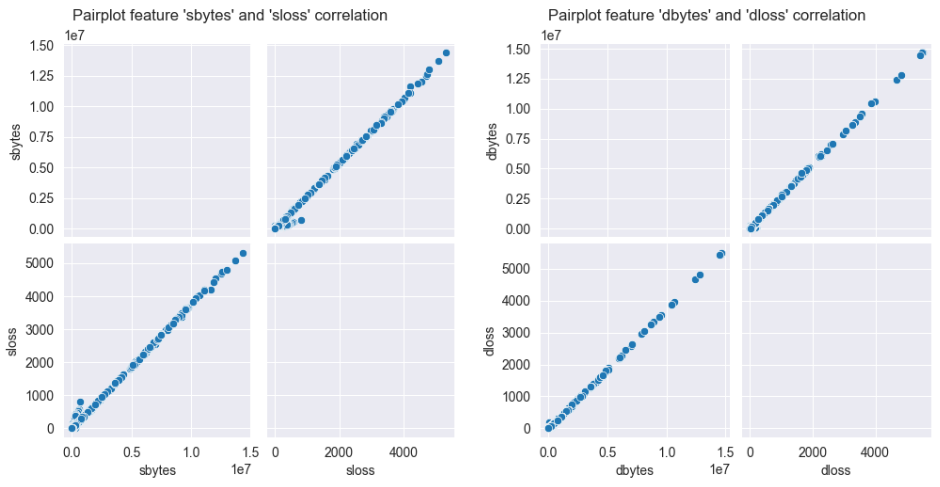


Fig. 4.4. Pairplots for **1.** *sloss* and *sbytes* and **2.** *dloss* and *dbytes*

TTL Features

There are multiple TTL features. 'TTL' stands for Time To Live, which is the time that a package can live in a network. It is also called a 'hoplimit', because it prevents a package from circulating or 'hopping' in a network infinitely. The number works as a counter, where each time a package passes a router, it loses a number until it is 0.

There are three TTL features in the UNSW-NB dataset: *sttl*, *dttl* and *ct_state_ttl*. From Figure 4.3, it was visible that there was a high correlation between *sttl* and *ct_state_ttl*. As shown in the first graph of Figure 4.5, this is not a linear relationship, but it is possible that some values of both features are correlated. For example, the high values of *sttl* seem to correspond to all values (total of 6) of *ct_state_ttl*. Between *sttl* and *dttl* there was a negative correlation value in Figure 4.3. Shown in the second graph of Figure 4.5, the specific values of both characteristics correspond to each other. For example, instances with value 0 for both features.

To be complete, *ct_state_ttl* and *dttl* are shown in the third plot of Figure 4.5. In Figure 4.3, the colour of the block was light, indicating almost no correlation. It is difficult to find a pattern in the figure, which can be explained by the lack of correlation inferred from the heatmap.

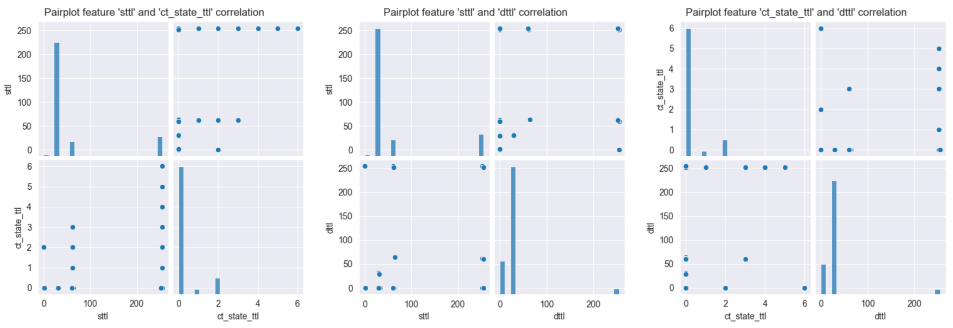


Fig. 4.5. Pairplots for **1.** *sttl* and *ct_state_ttl*, **2.** *sttl* and *dttl* and **3.** *ct_state_ttl* and *sttl*

Correlation with the label

Besides the correlations between features, correlations between features and the label are examined. From Figure 4.3, the 'ct' features showed a correlation with the label, similar to *sttl*.

There are features that display patterns when plotted against the normal versus the abnormal label. An example is *sttl*. The data for the normal and malicious labels are separated and plotted in different plots in Figure 4.6, because the absolute values differ in magnitude. This difference is seen by the difference in the values on the y-axis. From this plot, the values of *sttl* for normal data lie below 75, while there is a clear peak at 250 for malicious data.

As *sttl* and *ct_state_ttl* were correlated, the feature values of the latter are plotted for normal and malicious data separately in Figure 4.7. In this figure, there is a clear peak at the value 0 for normal data, while there is a peak at 2 for malicious data. Again, the y-axes differ because the absolute values of normal and malicious data differ.

Another feature that shows a larger variety of peaks when plotted against the malicious label is *ct_dst_sport_ltm*, shown in Figure 4.8. This feature

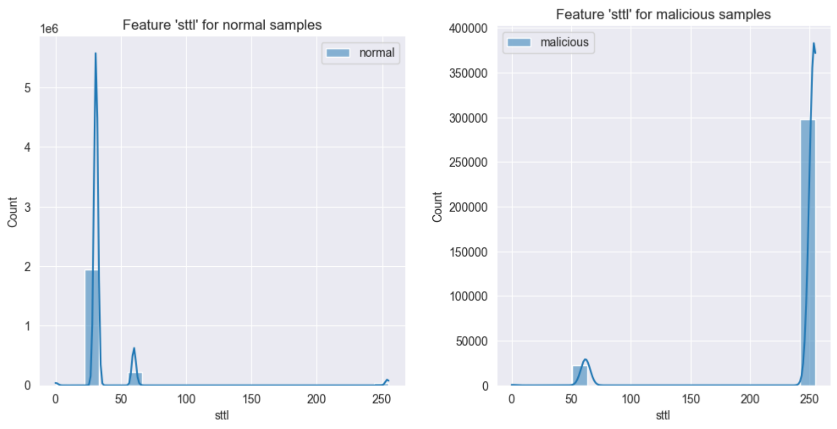


Fig. 4.6. Feature values of *sttl* for normal and abnormal samples

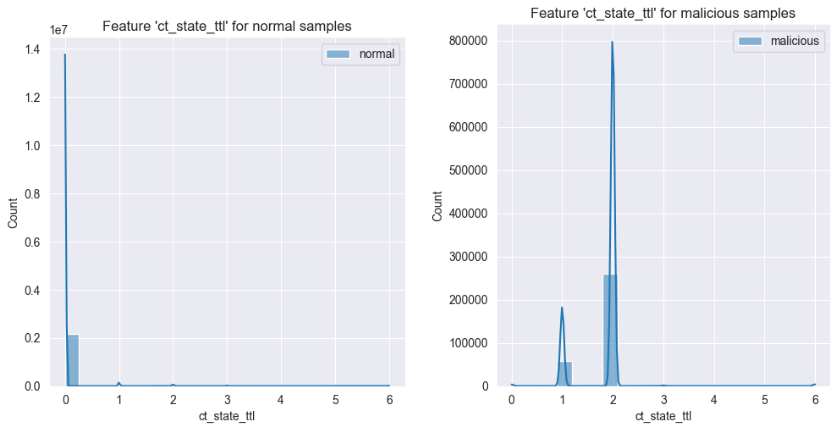


Fig. 4.7. Feature values of *ct_state_ttl* for normal and abnormal samples

represents the number of connections from the same destination address to the source port over 100 connections. Again, the values of the y-axes differ, but the values of the feature for malicious data show a spread until the value of 25. Both normal and malicious data show a peak at 0.

Both *ltime* and *stime* have a high correlation with the label in Figure 4.3. These features represent the last time and the start time of the package, and thus a time frame. Figure 4.9 shows the *ltime* for both normal and malicious data, again with different y axes. There is a clear peak for normal data at lower values. A similar pattern is found for *stime*, which is why its figure is not included in the report.

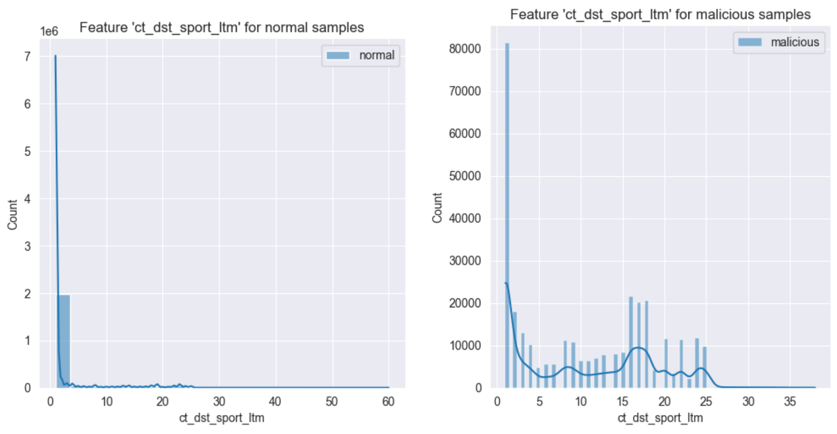


Fig. 4.8. Feature values of *ct_dst_sport_ltm* for normal and abnormal samples

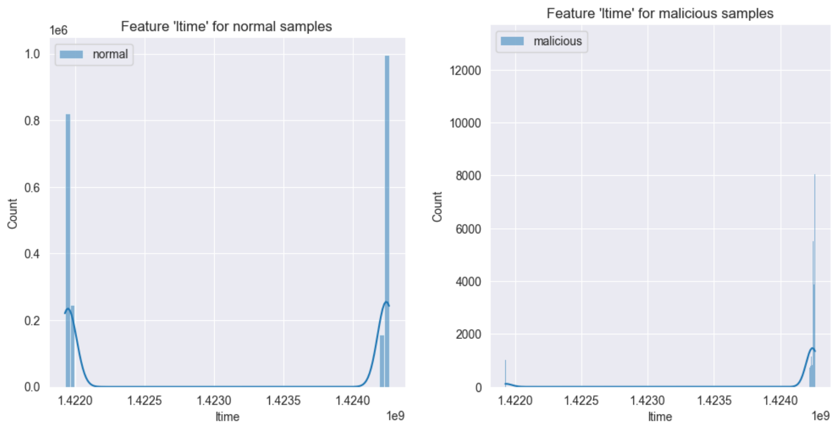


Fig. 4.9. Feature values of *ltime* for normal and abnormal samples

It is beyond the scope of this thesis to include all graphs with the values of a feature for the different values of the label. The plots shown are the ones that displayed the clearest differences when plotted separately for normal and malicious data. Other features have either similar patterns as they are related to the features shown, or they do not show interesting patterns with the label.

4.2 Different Datasets

After cleaning and exploring the data set, different subsets were created. All subsets consist of a train, validation and test set, that are distributed by 80-10-10 of the total records. The mentioned final size of the dataset includes the attack category and the label column.

All numerical features in the sets are normalised and scaled using the scikit-learn library. All categorical features are One Hot Encoded (OHE), using the OneHotEncoder of the scikit-learn library. OHE is chosen over Label Encoding because Label Encoding (or Categorical Labeling) imposes a rank. Hash Encoding was not chosen because it might induce problems in explainability, as not all hashed values can be traced back to their original values. A downside of OHE is that it can increase the dimensionality of the dataset.

The creation of every dataset is explained in the following sections.

4.2.1 Basic set

The Basic data set is formed through the findings of other papers in combination with the exploratory data analysis.

Highly correlated features that represent similar characteristics are combined and the individual features are removed if they did not have a high correlation with the label. This was the case for:

- *sbytes* and *dbytes* to *total_bytes*
- *spkts* and *dpkts* to *total_pkts*
- *sloss* and *dloss* to *total_loss*

Both *ltime* and *stime* were not combined, as both showed a high correlation with the label.

Earlier research has been done to find the optimal feature subset of the dataset, but they do not agree on all features [38, 62, 74]. The results of these papers are combined with the results of the current data analysis.

The three TTL features showed a correlation with the label and it has been observed that they could represent the label [74]. Therefore, one of the features is removed, namely *sttl.* This feature is removed as it had a positive correlation with *ct_state_ttl* and a negative correlation with *dttl*. The other two were not correlated.

Other features were removed that did not show clear correlations with the label during exploration. This is to reduce the number of dimensions of the dataset: *ct_src_dport_ltm*, *ct_dst_src_ltm*, *ct_srv_src* and *sload*.

Others have removed other features related to ports and IP addresses [38, 74] and this approach is followed. These were: *srcip* and *dstip* and *sport* and *dsport*.

The categorical feature*proto* has 135 different values in total, of which 9 occur in the normal category and 129 in the malicious category. To decrease the number of columns as a result from OHE, a new proto category is created that only contains 5 of the values instead of the 9 in the normal set or the 129 in the malicious set. These values are 'tcp', 'udp', 'unas', 'arp', 'ospf'. All other values are named 'others'. In Figure 4.10, more different values of the feature occur for the malicious category than for the normal category, with some having almost 0 percentage presence. The original column of *proto* is removed.

The final size of the Basic set after normalisation and One Hot Encoding is (2,540,043, 63).

4.2.2 Whole set

The Whole set includes all features of the dataset, except for *sport* and *dsport*. *sport* has 100,343 total categorical values and *dsport* 128,310. These values represent the ports, which means that they are categorical and have to be One Hot Encoded. These features were removed, otherwise the dimensionality of the dataset would have increased with at least 228,653 features. They were also removed in other studies [38, 74] and do not show a high correlation with the label in Figure 4.3.

The Whole set includes the normal values for *proto*, which means that there are 135 different categorical values. All values are normalised and One Hot Encoded.

The total size of the Whole set is (2,540,043, 296).

4.2.3 SMOTE set

As the distribution of the normal and malicious samples is imbalanced, a method can be applied to reduce this imbalance. One of such methods is

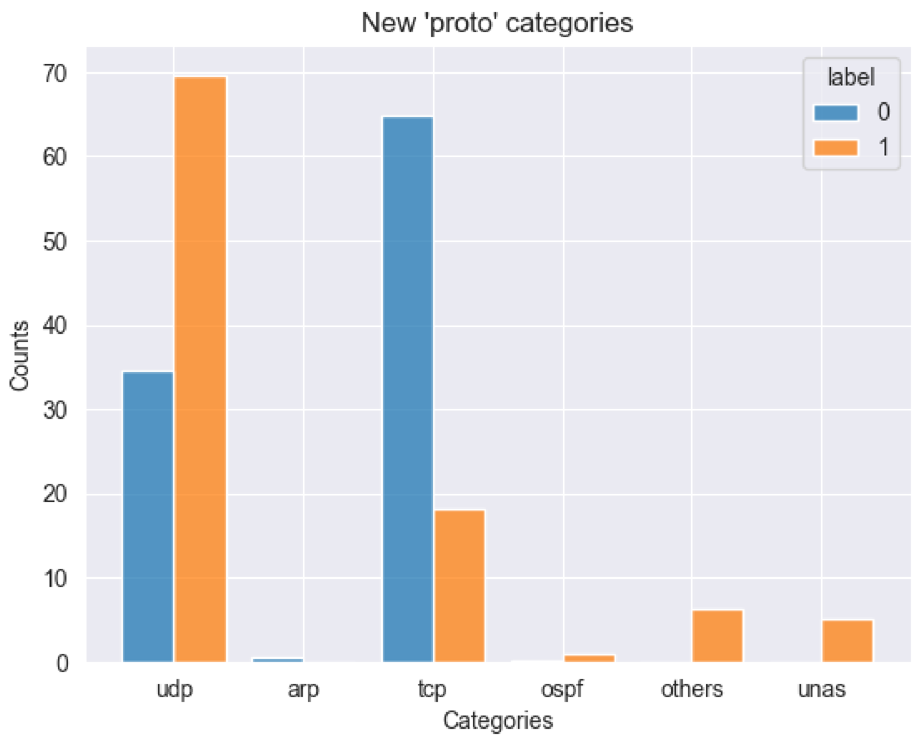


Fig. 4.10. The values for the newly created *proto* column

SMOTE [13]. SMOTE is a minority oversampling technique. It generates synthetic training samples that are similar to the data samples from the minority class.

Within the malicious part of the dataset, the distribution is also imbalanced. Category ‘worms’ occurs much less frequently than ’generic’. It is possible to apply SMOTE on normal vs. malicious traffic or normal vs. all separate categories of traffic. Both approaches are used. SMOTE is applied on the basic set for both binary and multiclass classification, to limit the size of both datasets. For binary classification, the labels for normal and malicious data are balanced. For multiclass classification, all classes are balanced (e.g. ’worms’ occur as much as ’normal’). The size of the set for binary classification is (4,437,520, 63) and for the multiclass classification is (22,187,600, 63). Notice how the latter is roughly 10 times as large as the basic dataset.

Oversampling was chosen over undersampling, as the latter would have decreased the dataset size. With oversampling, samples from the majority classes are deleted. The smallest class has 174 instances, which is why oversampling would have made the dataset too small.

4.2.4 Normal only set

The Normal set is based on the basic set and is of equal size. The major difference is the absence of malicious samples in the train set. The train set consists of only normal samples, while all malicious samples are equally divided over the validation and test set. There are two reasons behind this approach: 1) the absence of labelled data in real-life situations and 2) the ability of the AE to be trained on normal data only and detect anomalies through the reconstruction error. The AE is expected to be trained on only normal data and classify abnormal data in the test set correctly, as it uses a greater reconstruction error to detect anomalies. For the RF and the EBM, the expectation is that they will not be able to classify incorrect samples if they are not in the train set.

Similarly to the other datasets, the Normal set is scaled and One Hot Encoded, which leads to a final size of (2,540,043, 63)

4.2.5 Latent set

The Latent set are extracted latent dimensions of the AE. The Basic set is used as input for a trained AE and the values of the 8 Latent dimensions are extracted. These eight latent dimensions each give a value for each input sample, translating into a new dataset of eight features. Each feature is a latent dimension. Figure 4.11 shows the aim of this approach. Both an AE and a RF are trained on the Basic set or Normal set and a second RF is trained on the latent dimensions extracted from the AE. As the AE can function as a dimensionality reduction method, this approach is explored.

There are two possible latent sets: one in which the AE is trained on normal data and the other in which the AE is trained on the basic set. For convenience, the first set is called Latent-Normal and the second set is called Latent-Basic.

As the input is already scaled and encoded, the latent dimension is not. The sizes of both sets are (2,540,043, 10), because they include the original label and attack category.

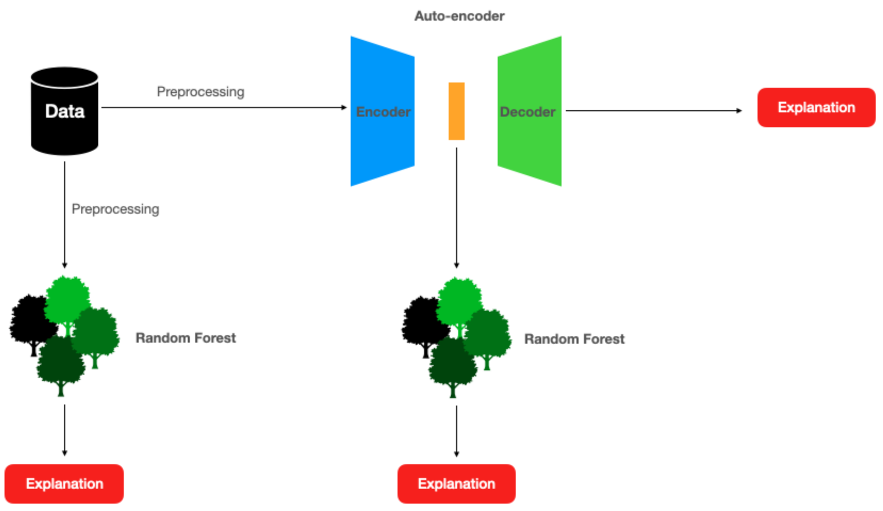


Fig. 4.11. Idea of the usage of the Latent dataset with a RF

4.2.6 Minority set

The original dataset is unbalanced. The use of SMOTE to balance the dataset increases the size of the dataset, which is not always a desired outcome. To create a dataset that contains every class, but is smaller in size than the SMOTE dataset, the minority set is created. It contains the total available instances of the four attack categories with the lowest numbers: analysis, backdoor, shellcode and worms. The other five attack categories are also included by dividing the total of the four minority attack categories by five to get a number. With this number, the remaining categories are sampled and added. Overall performance of a model should not depend on dataset size but on how well the train set represents the distribution of the original dataset [3].

The size of the total minority set is (13381, 63), including 'label' and 'attack_cat'. For the train, validation and test split, there is still the 80-10-10 ratio. The Minority set is used to train the AE-RF combination.

4.3 Model Hyperparameter tuning

In the following sections, the results of hyperparameter tuning are discussed. Hyperparameter tuning is performed to obtain the optimal version per model per dataset. For every model, various hyperparameters are examined on the validation set. Although for each model more hyperparameters can be tuned or larger ranges of numbers can be included, a selection is made that allows for a diversity of models using the available hardware. Additional to model performance, the hyperparameter tuning allows for the examination of performance differences due to the dataset. The aim of this thesis is not to find the optimal model for NAD, but to find a model with a dataset that performs accurately enough to support the explanation methods.

The choice of values is based on earlier observed applications of the models. The options for the RF are in Table 4.2, for the EBM in Table 4.3 and for the AE in Table 4.4. The number of hyperparameters for the AE is lower due to hardware limitations. Per model, different versions are selected with the highest train and validation score to further analyse. Depending on the model type, the number of versions to compare differs, because there are more different versions of the RF than of the AE. In general, there are four versions compared for the RF, three for the EBM and two for the AE. The chosen version is the model used in the explanation evaluation and comparison.

Hyperparameter	**Values**	**Description**
n_estimators	100, 200, 400, 600	Number of trees in the forest
max_features	"sqrt", "log2"	Number of features per split
max_depth	None, 10, 20	Maximum depth per tree
max_samples	None, 0.5	Maximum samples per split

Tab. 4.2. Hyperparameters and their values for the Random Forest. 48 combinations in total

The hyperparameters chosen for the RF are shown in Table 4.2. The four hyperparameters to tune are the number of estimators (trees), the maximum number of features considered per split, the maximum depth per estimator (tree) and the maximum samples to consider per split. The default number of estimators is 400. The numbers above and below are included. The default maximum number of features is 'sqrt'. According to the scikit-learn documentation, 'log2' and 'None' are other options. To prevent overfitting, "None" is excluded. For the maximum depth, "None" is the default. This means that the nodes are expanded until all leaves have fewer than two samples. To see if a strain on the depth is beneficial, the numbers 10 and 20 are included. The maximum number of samples reflects the bootstrap process. The default version is 'None', which means that all samples are considered. To see how it affects the performance, 0.5 is included. This value means that half of the samples are considered per split.

Hyperparameter	**Values**	**Description**
max_bins	256, 128	Bins for the feature processing step
interactions	10, 5, 0	Interaction terms
max_rounds	5000, 3000, 1000	Rounds for boosting

Tab. 4.3. Hyperparameters and their values for the Explainable Boosting Machine. 27 combinations in total

The hyperparameters evaluated for the EBM are shown in Table 4.3. Similar to the RF, four hyperparameters are selected to tune. These are the maximum number of bins for the discretisation of continuous features, the interaction terms between features and the maximum rounds for boosting. The default maximum number of bins is 256. This number controls the smoothness of the function. As higher numbers make the feature processing step more complex, 128 is added. It may result in the loss of information. For the interaction terms, 10 is the default. The value 0 makes the model truly additive. Number 5 is included as a bridge between 10 and 0. The maximum rounds for boosting has 5000 as its default value. Higher numbers are not included to prevent an increase in train time, while lower numbers are explored to see if less complex models behave accurately.

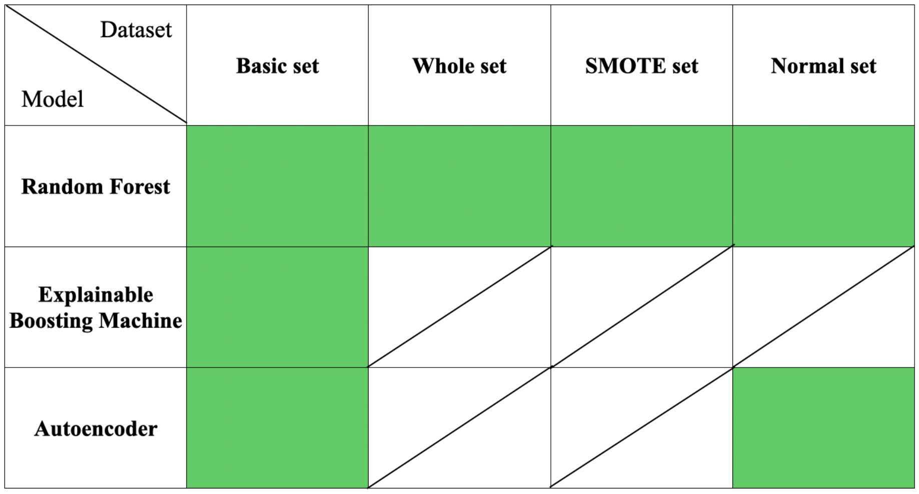
Hyperparameter	**Values**	**Description**
hidden_neurons	[32, 16], [32, 16, 8], [64, 32, 16]	The layers and neurons of the encoder and the decoder
learning_rate	1e-2, 1e-3, 1e-4	The learning rate of the algorithm
weight_decay	1e-5, 1e-4	Weight decay for the Adam optimiser

Tab. 4.4. Hyperparameters and their values for the Autoencoder. 18 combinations in total

For the AE, three hyperparameters are tuned: hidden neurons and layers, the learning rate and the weight decay. The Pyod package does not allow for a separation of hidden layers and neurons per layer, which is why these two are in the same hyperparameter. The default version is [32, 16], which is expanded with one larger hidden layer and one smaller hidden layer. The default learning rate is 1e-3 and two learning rates around this value are chosen. The default weight decay for the Adam optimiser is 1e-5. A value higher than this one is added to see if a larger weight decay speeds up training but keeping model performance accurate.

4.3.1 Binary Datasets

An overview of the models evaluated is shown in Table 4.12. Some evaluations are excluded, as previous evaluations with other models show either poor performance or no difference between the datasets. If train and validation recall and F2-scores are equal or close to equal for the models, other metrics are considered. If these are equal, confusion matrices can support the decision, or simpler models are preferred.



Dataset Model	**Basic set**	**Whole set**	**SMOTE set**	**Normal set**
Random Forest	🟩	🟩	🟩	🟩
Explainable Boosting Machine	🟩	╱	╱	╱
Autoencoder	🟩	╱	╱	🟩

Fig. 4.12. The evaluated models per dataset for binary classification. Green indicates versions on that dataset are evaluated and the results are discussed. A line means that this is not the case.

In addition to Figure 4.12, the Latent set versions are evaluated for the RF and the EBM.

The RF is evaluated on the Basic set, Whole set, SMOTE set and Latent set to investigate if the dataset size or the number of features improves performance. The results of the RF on the datasets are used to determine on which sets the EBM and the AE are trained. The expectation is that the AE is able to perform well on the Normal set, compared to the RF and the EBM. The RF is trained and validated on the Normal set for exclusion of this set.

All evaluations are done to find the optimal version of a model for binary classification. Additionally, a dataset can be selected that balances performance and dataset characteristics, such as the number of features or the amount of samples.

Random Forest

Basic set The range of all metrics for different versions of the Basic validation set goes from 0.9904 to 0.9960. Of all versions of the RF trained in the Basic validation set, several have a recall and F2-score greater than 0.995. This is the highest possible value on these two metrics. The versions with these scores vary in their hyperparameters. All of them (except one) have a "None" maximum depth of a tree in common. Not placing a constraint on the maximum depth could lead to overfitting. Another observation is that models without a maximum depth and without a constraint on the maximum number of samples all score above 0.9950 on all metrics, regardless of the number of trees.

Considering model complexity, two versions are selected with accuracy, F2-score, precision and recall above 0.995 on the validation set. The F2-score and recall are 0.9960. Two other versions are selected with scores below these, but with a constraint on the maximum depth. The versions are shown in Table 4.5.

The selected versions do not contain a version with log2 as setting. An observation is that versions with similar settings that differ only in *max_feat*, perform slightly worse if they have log2 than with sqrt.

As the results are close, the confusion matrices of the four versions are examined in Figure A.1. The confusion matrices of the first, second and fourth versions show similar distributions, while the third version shows a heightened number of false positives opposed to false negatives. The first version has the lowest number of false positives and false negatives. Version 1 is chosen as the final version and is referred to as RF-Basic from now on.

Version	Estimators	Max Features	Max Depth	Max Samples
1	200	Sqrt	None	None
2	100	Sqrt	None	None
3	100	Sqrt	20	0.5
4	200	Sqrt	10	0.5

Tab. 4.5. Four selected versions of the hyperparameter tuning process of the RF on the Basic set

Whole set In comparison to the Basic dataset, the range of values for the metrics on the Whole validation set is wider. It goes from 0.9906 to 0.9967, which is both higher and lower than on the Basic set. A similar approach for selecting models is used as for the RF-Basic. There are two models with metrics above 0.9960 and two below. For the versions with a score below, two versions with a smaller number of trees, a maximum number of depths, and a constraint on the maximum number of samples are considered.

There are eight versions with all metrics 0.9967. Most versions have log2 as the maximum feature selector, which is interesting in comparison to the basic set, where models with log2 generally scored slightly lower than those with sqrt.

The second version has a constraint on the maximum number of samples considered for a split, although it has performance metrics similar to those in the first version. Hyperparameters for the versions are shown in 4.6.

For the next two versions, versions that score below 0.9960 are considered. Using the complexity constraints, versions of interest are the ones with a lower number of estimators (100, 200), a maximum depth of the tree of 10 or 20 and half of the features considered per split. The hyperparameter combinations of the last two models are shown in Table 4.6.

As the highest scores on the train and validation set differ by 0.0007 between the basic and the whole versions, no version trained on the Whole set is used in further examination. The Basic set has fewer features (61) compared to the Whole set (294), while the performance is close.

Random Forest: SMOTE set The range of performance metrics on the SMOTE set goes from 0.9931 to 0.9976. The score of 0.9976 is the highest found for the recall and the F2-score on the validation set compared to the

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	600	Log2	None	None
2	400	Log2	None	0.5
3	200	Sqrt	20	0.5
3	100	Sqrt	20	0.5

Tab. 4.6. Four selected versions of the hyperparameter tuning process of the RF on the Whole set

Basic and Whole versions.

There are three versions that score equal on all metrics; therefore, the simplest version is chosen, with 200 estimators. There are two models that have 200 estimators and score 0.9976 on all metrics, and they only differ in the selection for the maximum number of features. The version is selected with the default parameters of the scikit-learn implementation. The chosen versions are shown in Table 4.7.

Similarly to the Basic set best versions, the best performing versions have a low number of estimators, a constrained depth of the tree, and using half of the samples in a split. Furthermore, considering versions that only differ between sqrt and log2 for the feature selection approach, the versions with sqrt appear to perform better. This pattern was found at the Basic set as well.

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	200	Sqrt	None	None
2	100	Sqrt	None	None
3	100	Sqrt	20	None
3	200	Sqrt	10	0.5

Tab. 4.7. Four selected versions of the hyperparameter tuning process of the RF on the SMOTE set

SMOTE is an upsampling technique to balance the samples in the dataset, by increasing the minority class. In the binary case, SMOTE almost doubles the original dataset size. Although the SMOTE set has an equal number of samples for both the normal and malicious class, the performance difference with the Basic set is 0.016 on recall and F2-score. It is possible that the Basic set is balanced enough for binary classification, which is why the performance of versions trained on the SMOTE increases with 0.016 only. Therefore, no SMOTE model for binary classification will be investigated further.

Normal set As the train set of the Normal set consists of only normal samples, it is expected that the RF is not able to classify malicious samples in the validation set. The performance on the train set is 1.0 for every metric for every version. However, all other metrics are below 0.4 (accuracy = 0.5, F2-score = 0.3685, precision = 0.2744 and recall = 0.3685). The versions are able to correctly classify normal samples, of which there are more than anomalous samples, but this is a random guess.

These versions are not evaluated further or included in further research.

Latent set There are two latent sets: Latent-Normal and Latent-Basic. For both datasets, the RF versions are evaluated.

Version	Estimators	Max Features	Max Depth	Max Samples
1	100	log2	10	0.5
2	400	log2	10	0.5
3	400	Sqrt	10	0.5
3	200	Sqrt	10	None

Tab. 4.8. Four selected versions of the hyperparameter tuning process of the RF on Latent-Normal

For the Latent-Normal set, the value of the F2-scores and recall ranges from 0.9386 to 0.9486. The highest value for precision is 0.9587 and for accuracy 0.9481. Compared to the Basic set, these values are lower.

For the Latent-Basic set, the F2-score and the recall range from 0.9821 to 0.9869. Accuracy follows this range. The precision has the same lower limit, but an upper limit of 0.9873. These values are higher compared to the versions of the Latent-Normal set. It is still lower than the Basic set, but the Latent-Basic set has 8 features compared to the 61 of the Basic set.

For the final version of the Latent set, models of Latent-Normal are not considered. Given the metrics and the simplicity of the first version of Latent-Basic, this version is chosen as the final model. This is now called RF-Latent.

Version	Estimators	Max Features	Max Depth	Max Samples
1	200	Sqrt	20	0.5
2	400	log2	20	None
3	100	Sqrt	10	0.5
3	200	log2	10	0.5

Tab. 4.9. Four selected versions of the hyperparameter tuning process of the RF on Latent-Basic

Summary Of the five initial dataset, versions trained on four datasets were evaluated. Versions of the Normal set were excluded. Although the performance metrics of all versions on the different datasets is above 0.93, there are other aspects of to take into account about the model and the dataset. About the model, the complexity of the versions played a role in selecting the final version. Aspects of the dataset, such as number of features or size, impacted the decision to use a dataset further in this study.

The RF trained on the Basic dataset showed lower performance metrics than the RF trained in the Whole set, but the difference was 0.0007. However, the Whole set contains more features. This shows that more features do not lead to improved performance. Therefore, no model selected trained on the Whole set for further investigation.

Secondly, the RF trained on SMOTE showed a small increase in performance over the RF trained on basic. However, the SMOTE dataset is almost twice as big as the Basic dataset. Using dataset size as a factor in the decision, the binary versions trained on the SMOTE dataset were not investigated further.

Based on the performance and characteristics of the dataset, the final model and dataset selected is the RF trained on the basic dataset. From now on it is referred to as RF-Basic.

Aside from RF-Basic, RF-Latent was selected from different RF trained on Latent-Basic or Latent-Normal. The Latent-Basic versions showed better performance than the Latent-Normal versions. This model is selected for further investigation, as it has a low feature number and represents the latent space of the AE.

Explainable Boosting Machine

The Whole set, SMOTE set and the Normal set are excluded for the evaluation of the hyperparameters of the EBM. The RF does not show large performance differences between the Basic, the Whole set or the SMOTE set, and it does not work on the Normal set. The expectation is that the EBM performs similar to the RF on the datasets, which is why they are excluded. As the EBM is also supervised, they are excluded from the EBM evaluation.

Three versions are chosen and not four, as there are fewer EBM versions available from hyperparameter tuning than for the RF.

Basic set The values for the metrics of the binary versions of the EBM are similar for several combinations of hyperparameters. Values range between 0.9875 and 0.9933 on all metrics. The recall or F2-score is never lower than 0.9879.

There are versions that show 0.9933 on all metrics of the validation set. All of these versions include interaction terms (5 or 10) and have a set value of 256 for the maximum number of bins for the feature preprocessing stage. The different number of rounds for boosting (1000, 3000 and 5000) are all present with these best versions.

Similarly to the RF, two versions are selected with the highest performance metrics. A third model is selected because it does not have the highest performance metrics but is simpler in its hyperparameters.

Version	**Bins**	**Interactions**	**Rounds**
1	256	10	5000
2	256	5	1000
3	128	5	5000

Tab. 4.10. Three selected versions of the hyperparameter tuning process of the EBM on the Basic dataset

The first version is the default implementation of the EBM. The second version has fewer interaction terms and fewer rounds for boosting, but scores similar on the validation set. The third version has fewer bins than the first two versions but is equal in the number of rounds as version 1 and interaction terms as version 2. This version has performance metrics of 0.9931, as opposed to 0.9933 of all metrics of the first two versions.

Looking at the confusion matrices (Figure A.2), all versions have higher false negatives than false positives. The first two versions have the same number of false negatives and the third version has the highest number. The first version has the lowest false positives of all three versions. The false positives of the second and third version differ only 1.

The first version is selected based on its validation scores and confusion matrix. From now on, this is referred to as EBM-Basic.

Latent set As the RF-Latent showed comparable performances as the RF-Basic, the EBM was trained on this dataset as well. The RF-Latent-Basic showed better performance than the RF-Latent-Normal. Therefore, EBM is evaluated only on the Latent-Basic set. As RF-Latent for binary classification did not show large performance differences from the RF-Basic, this is expected for the EBM trained on the latent dimensions as well.

The values of the metrics of the EBM Latent versions are lower than the binary EBM-Basic, similar to the difference between RF-Basic and RF-Latent. The values range between 0.9735 and 0.9793 on all metrics.

Two versions are chosen with 0.9793 on all metrics and one version with metrics below that. The first version has the same parameters as the default implementation. As the second version already has constraints on the number of bins and rounds of boosting, the third version is chosen based on a constraint on the interaction terms.

Version	**Bins**	**Interactions**	**Rounds**
1	256	10	5000
2	128	10	1000
3	256	5	3000

Tab. 4.11. Three selected versions of the hyperparameter tuning process of the EBM on the Latent dataset

Figure A.3 shows the confusion matrices of all three models. The first version has the lowest number of false negatives and false positives, compared to the other two version. Therefore, the first version is selected as the final version for the Latent set and is referred to as EBM-Latent.

Summary Inferred from the performance of RF trained on Whole, SMOTE and Normal sets, the EBM was not evaluated on these sets. From the results on the performance metrics and the confusion matrices, a final version was selected of the EBM trained on the basic set. This version is now called EBM-Basic. The versions showed a comparable performance to the RF-Basic versions.

Aside from the Basic set, the EBM was trained on the Latent dataset. Similar to the RF-Basic and RF-Latent, the EBM-Latent showed a slight decrease in performance compared to the EBM-Basic.

Autoencoder

The AE is evaluated on the Basic set and on the Normal set. The AE is expected to perform well on the Basic set, but better on the Normal set. The validation set of the Normal set contains malicious samples that are absent in the train set and this increases the reconstruction error.

The AE trained on Whole and SMOTE sets are not included in further evaluations, as the Basic set is evaluated to investigate if a train set with both normal and malicious samples works for the AE.

Only two versions are selected for further comparison, as the hyperparameter tuning process of the AE is limited due to hardware and time constraints.

Basic set For the Basic set, the recall on the validation set ranges from 0.7958 to 0.8478 for all versions of the AE with different hyperparameters. The best recall is 0.8478. The F2-score ranges from 0.7917 to 0.8446. The model that scores the highest recall also scores the highest F2-score. The precision of this model is highest compared to other models.

However, examining the AE trained on the Normal set, the AE trained on the Basic set does not show equally good performance. Therefore, the AE trained on Basic is not used further.

Normal set With the Normal set, the AE is trained on the Normal data only, but tested on both normal and malicious data. The expectation is that the AE performs well on this set, and this is confirmed by results. All AE versions trained on the Normal set only show higher values for all metrics compared to the AE versions trained on the Basic data.

Recall scores range from 0.9624 to 0.7901 for all versions, while the F2-score ranges from 0.9619 to 0.7879. The other two metris follow this last range. Compared to the other binary versions of the RF and EBM, the range of metrics is wider.

Two of the best versions of the AE are selected, and their hyperparameters are shown in Figure 4.12.

The best version has a recall of 0.9624 and an F2-score of 0.9619. The second version has a recall of 0.9623 and an F2-score of 0.9618. As AE is a neural network, the trainable parameters increase with the number of neurons and number of layers. Version 1 has 5557 trainable parameters, while version 2 has 13613.

Version	**Hidden layers**	**Learning Rate**	**Weight Decay**
1	[32,16,8]	1e-4	1e-5
2	[64,32,16]	1e-3	1e-5

Tab. 4.12. Two selected versions of the hyperparameter process of the AE

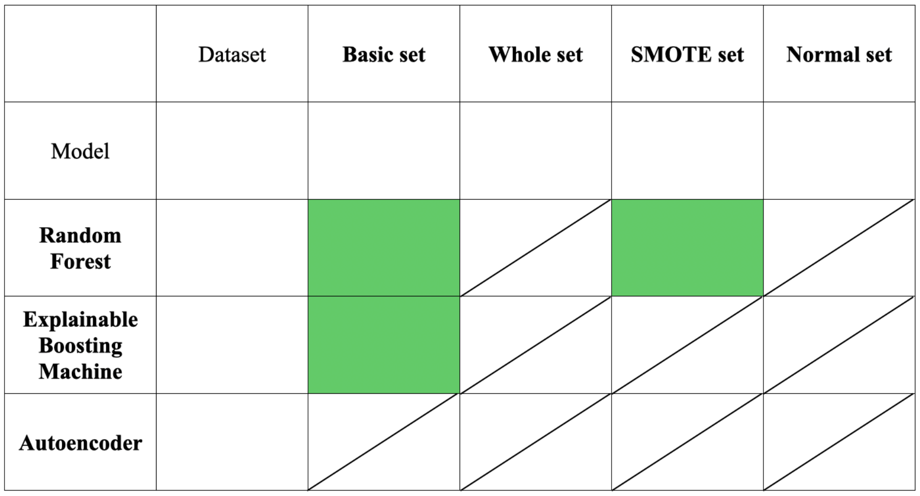
Given the performance as well as the complexity, version 1 is chosen as the final model. This will be called AE-Normal.

Summary As expected, the AE performed well on the Normal validation set. The AE showed a decrease in performance on the Basic dataset. The difference between the highest values on the metrics of the validation set of the Normal and Basic versions is 0.1146. Because of this difference, the AE trained on the Basic set is not included in further investigation, and the AE-Normal is the main AE used in this thesis.

4.3.2 Multiclass Datasets

Figure 4.13 shows the models and datasets evaluated for the multiclass hyperparameter tuning. Additionally, the RF is evaluated for the Latent-Basic set.

For the multiclass evaluation, the Whole dataset is not evaluated for any model, because of the minimal performance difference between the binary versions of the RF. The SMOTE set for multiclass classification balances all classes equally. It is a different set from the SMOTE for binary classification and is investigated for the RF. An EBM is not trained on the SMOTE dataset, due to hardware limitations.



	Dataset	**Basic set**	**Whole set**	**SMOTE set**	**Normal set**
Model					
Random Forest		🟩	╱	🟩	╱
Explainable Boosting Machine		🟩	╱	╱	╱
Autoencoder		╱	╱	╱	╱

Fig. 4.13. The evaluated models per dataset for multiclass classification

The AE is excluded from the multiclass evaluation, as the implementation was not suited for multiclass performances. Therefore, the Normal set is not evaluated for any model in the multiclass case, as training the RF and EBM on this set did not yield good performance.

Again, all evaluations are done to find the optimal version of a model for a dataset. As the dataset can be unbalanced, the results can show if certain datasets are not suited for the multiclass classification task.

Random Forest

Below, the results of the multiclass RF hyperparameter tuning on the Basic, SMOTE, Latent and Minority set are shown.

Basic set Contrary to the binary versions of the Basic dataset, none of these versions scored 1.0 on metrics during training. The highest training accuracy, F2score, precision and recall are 0.9998.

The scores differ more between metrics than between the metrics of binary versions. Validation metrics vary for accuracy and recall from 0.9773 to 0.9820. The F2-score values range from 0.9750 to 0.9820. The precision ranges from 0.9750 to 0.9815.

Similarly to the previous approach, two versions with the highest metrics are selected and two versions with lower scores on the metrics, but simpler if possible. The four versions are shown in Table 4.13. The first version has the highest recall (0.9820) and F2-score (0.9820), which are combined with an accuracy of 0.9820 and a precision of 0.9815. The second version has a slightly lower F2-score of 0.9811 and a recall of 0.9817. The precision is 0.9810 and the accuracy 0.9817. The second version is simpler than version 1, as it has restrictions on depth and number of samples to consider per split.

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	200	Sqrt	None	None
2	200	Sqrt	20	0.5
3	200	Sqrt	10	0.5
4	100	Sqrt	10	0.5

Tab. 4.13. Three selected versions of the hyperparameter tuning process of the RF on Basic for multiclass

As the two previous models already have a low number of estimators, the other two considered versions have constraints on the depth and the maximum samples considered per split. The best performing version with constraints has an accuracy of 0.9778, an F2-score of 0.9730, a precision of 0.9781 and a recall of 0.9778. It has a smaller depth than the second version. The last and fourth version scores the lowest of all versions on all metrics, but it is the most constrained tree. It has the lowest number of estimators, lowest depth and constraint on samples per split.

Taking into account the classification reports for the four versions, not all versions predict all classes (Figure A.1, A.2, A.3 and A.4). Version 1 and version 2 predict the class 'worms', while the other two models do not. The classification reports of the third and fourth versions show clear absence of values, having precision, recall and F1-scores of zero. This means that the model has not predicted these classes.

The behaviour of the versions resembles that of the training data. The training data is unbalanced with respect to certain classes, such as ‘worms’ or ‘shellcode’. The models predict poorly on minority classes, but are able to predict majority classes correctly. The overall accuracy is high because it is weighted. The validation set has a large number of normal samples, so these predictions are weighed more heavily. However, the macro-average shows a lower number.

Comparing the scores between the binary and multiclass classifications, the model appears to be able to predict correctly whether the data is normal or anomalous, but not what kind of anomalous data it is. This can be a result of an unbalanced data set.

Although all versions are not performing optimally for multiclass classification, the first version is chosen. It has the highest averaged metrics and tries to predict all classes, not leaving any values in the classification report empty. From now on its called RF-Basic MC.

SMOTE set Between the binary versions of the Basic set and the SMOTE set there were no large performance differences, even though the SMOTE set was almost twice as large. The SMOTE set for binary balanced normal and malicious labels. The SMOTE set for multiclass classification balances the differences in the classes instead of labels. For instance, the class 'worms' occurs as often as the 'normal' samples. The size of the SMOTE set multiclass is almost ten times larger compared to the Basic set. This increase in dataset size leads to an increased train time. The train time on the Basic set for multiclass classification took between 2 and 7 minutes, depending on the hyperparameters. A high number of estimators lead to a longer train time. The train time on the SMOTE set for multiclass classification went up from 40 minutes to several hours, when the number of estimators was large. Originally, there are 48 combinations of hyperparameters to compare. To narrow down the number and decrease the overall train time, only 100 and 200 trees are used in the hyperparameter search. This leads to 24 combinations.

As there are 24 combinations instead of 48, two models are examined in depth. The best performing version has a score on all metrics of 0.9970. The second-best model scores above 0.9818 on all metrics. An overview of the hyperparameters is shown in Table 4.14.

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	100	Sqrt	10	0.5
2	200	Sqrt	20	None

Tab. 4.14. Two selected versions of the hyperparameter tuning process of the RF on SMOTE for multiclass

The classification report shows high scores on precision, recall and accuracy for all individual classes in the classification report of version 1 (Table A.5 and of version 2 (Table A.6).

For version 1, all metrics for the individual classes are above 0.99. For version 2, all metrics are above 0.92, except for the precision of ‘dos’. This is 0.8863. The average scores of the metrics are all above 0.98 and this is the same for the macro average.

In comparison to the results of the Basic multiclass versions, the two SMOTE versions have high scores on the metrics of all individual classes.

The confusion matrices show a clear dominance in the diagonal of the categories, which means that most classes are predicted correctly (Figures A.4 and A.5). From the matrices it is clear that the class 'dos' is predicted often when it is actually another class. This is visible with high values in the confusion matrix of version 2 (Figure A.5). The second version also shows higher values outside the diagonal. It is possible that there are classes that show overlap in their feature values, but the classification reports indicate that most of the time these classes are predicted correctly.

Version 1 shows higher scores on the metrics than version 2, which is why version 1 is chosen as the version to continue with for the rest of the thesis. It is called RF-SMOTE MC.

Latent-Basic set Due to the results of the RF-Basic multiclass and the minor performance differences between the RF-Basic and RF-Latent-Basic for the binary case, the expectation is that the RF-Latent-Basic resembles the performance of the RF-Basic multiclass. The multiclass Latent set servers as a proof of concept, therefore two versions were created and trained.

One of the versions is the default version of the RF implementation of scikit-learn. The second version is one with restrictions on the number of estimators, depth and number of samples to consider. Their hyperparameters are shown in Table 4.15.

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	400	Sqrt	20	None
2	100	Sqrt	20	0.5

Tab. 4.15. Two selected versions of the hyperparameter tuning process of the RF on Latent-Basic for multiclass

Both versions have the same F2-score of 0.9628. The recall of the first version is 0.0001 lower than that of the second version (0.9647 vs 0.9648). The precision of the first model (0.9588) is lower than that of the second model (0.9585), while the accuracy of the second model is higher (0.9648 as opposed to 0.9647). However, the differences between the metrics are at most 0.0003. The versions perform almost equally. The scores are lower compared to the RF-Basic MC versions, which shows a resemblance to the difference in binary classification scores of both models on the dataset.

The classification report for the first version is shown in Figure A.7 and the second version in Figure A.8. In comparison to the RF-Basic for multiclass, not all Basic versions predicted every class, but the Latent versions did. Even with the lower number of features, the Latent versions are able to make a prediction for each class.

Both versions show similar scores on all individual classes, with some minor differences. As the two versions show similar performance, the simplicity factor is used to make the final decision. Version 2 is selected as the final model, which is now called RF-Latent MC.

Minority set The Minority set serves as a proof of concept. The performance of the dataset is tested on one RF with default implementation. The AE is excluded from hyperparameter tuning. The hyperparameters can be found in Table 4.16.

Version	**Estimators**	**Max Features**	**Max Depth**	**Max Samples**
1	400	Sqrt	20	None

Tab. 4.16. The default implementation of the RF of the minority set

Overall, the model scores 0.7638 on the recall and 0.7636 on F2-score. The precision is 0.7633 and the accuracy of 0.7638. All metrics lie close to each other.

Although most classes are predicted correctly, with scores of 1.0 or slightly lower, the RF appears to have difficulties distinguishing 'backdoor' and 'analysis'. This can be seen in the classification report (Table B.4) and in the confusion matrix (Figure A.6). The scores of both 'backdoor' and 'analysis' are lower compared to all other classes. The scores for 'backdoor' are lower than for 'analysis' in the confusion matrix.

It is possible that the model is not able to distinguish these classes and needs more samples to learn the decision boundary. The lack of samples could be the reason for the model's inability, although the class 'worms' is predicted correctly all the time.

There is only one version tested for the Minority set. From now on this model is referred to as RF-Minority.

Summary A general observation for the multiclass classification versions is that the metrics differ more between versions, compared to the binary classification versions.

For the multiclass hyperparameter tuning, four datasets with different versions of the RF were evaluated. The RF-Basic MC and RF-Latent MC showed similar performances as their binary counterparts. The differences between the multiclass models show similarity to the differences between the binary models.

Although most versions on the datasets showed comparable average performance, the performance on the minority classes was lower than on the majority classes. The SMOTE versions showed the best performance on all individual classes, followed by the Minority versions. The latter had difficulty distinguishing between two individual classes, which lowered the average scores for the metrics.

As a result, there are four RF for multiclass classification, trained on four different datasets.

Explainable Boosting Machine

The following section contains the results of EBM hyperparameter tuning for multiclass classification. For the multiclass EBM, only the Basic set is evaluated as other sets served as proof of concept or were excluded for the RF as well. The Minority sets serves as a proof of concept and are only validated on the RF. As the Latent set did not show a good performance on the individual classes for the RF, it is not included for examination with the EBM. Due to the size of the SMOTE set, it was not possible to train an EBM locally on this dataset.

Basic set The values of the multiclass EBM versions range between 0.9501 and 0.9659 for all metrics.

There are several versions that show the same scores on the validation set. All of these versions include interaction terms (5 or 10). Compared to the versions for binary classification, the versions include 128 or 256 bins for the feature preprocessing stage and all different numbers for boosting.

Using a similar approach as for the Random Forest, two versions are selected from the highest scoring versions. They have a recall of 0.9659, an F2-score of 0.9610, a precision of 0.9551 and an accuracy of 0.9659. As a third version, a simpler version with high scores on performance metrics is selected.

Version	**Learning Rate**	**Bins**	**Interactions**
Rounds			
1	128	10	5000
2	128	10	1000
3	128	0	3000

Tab. 4.17. Three selected versions of the hyperparameter tuning process of the EBM

The first model resembles the default values for the EBM, in addition to the bins for continuous features. The second model has fewer rounds for boosting but shows equal scores.

Secondly, the third version does not include interaction terms, while the two other versions do. The first two versions score higher on all performance metrics, which could indicate that interaction terms are of importance for the training models on this dataset. The third model scores a similar recall of 0.9659, but a lower F2-score (0.9555), which is due to the lower precision (0.9519).

The classification reports show similar patterns compared to each other. They can be found for version 1, 2 and 3 in Figure A.9, A.10 and A.11. In all versions, the classes ‘worms’, ‘shellcode’, ‘analysis’ and ‘backdoor’ are not predicted once. The third version does not predict the class ’dos’ either.

Interestingly, the class ’recoinnaissance’ belongs to the five classes with the lowest instances in the dataset, but its precision value is the highest compared to all other individual classes.

Versions 1 and 2 show similar classification reports for the validation set. Both models have similar hyperparameters, with the only difference in the number of rounds. Version 2 is chosen as the final model because of its simplicity. This version is now referred to as EBM-Basic MC.

Summary For the EBM, only the Basic set is evaluated. These versions show similar performance on average and weighted, compared to each other and to the RF-Basic multiclass versions. They have high average scores on metrics, but often low scores for individual classes. This can be a result of the unbalanced dataset.

Although the performance of the EBM-Basic for the multiclass case is not optimal, one version is chosen to investigate further. This version is called EBM-Basic MC.

Chapter 5

Results

5.1 Model Performance

Table 5.1 shows all models that are compared and their performance metrics on the test set. The test set per model can differ, but is always based on the UNSW-NB15 dataset.

Model	**Accuracy**	**F2-score**	**Precision**	**Recall**
Binary classification				
RF-Basic	0.9960	0.9960	0.9960	0.9960
RF-Latent	0.9872	0.9872	0.9875	0.9872
EBM-Basic	0.9913	0.9913	0.9916	0.9913
EBM-Latent	0.9864	0.9865	0.9867	0.9864
AE-Normal	0.9617	0.9612	0.9635	0.9617
multiclass classification				
RF-Basic	0.9819	0.9818	0.9814	0.9819
RF-Latent	0.9655	0.9637	0.9591	0.9655
RF-SMOTE	0.9819	0.9818	0.9832	0.9819
RF-Minority	0.7638	0.7578	0.7623	0.7638
EBM-Basic	0.9629	0.9574	0.9479	0.9629

Tab. 5.1. Scores of models on datasets for binary and multiclass classification

In general, the aim is to find optimal models for binary and multiclass classification. This is done through five comparisons, of which two binary and three multiclass. Models differ in structure or train set used. The comparisons show model performances but allow for the inclusion of other aspects that influence the performance, such as hyperparameters or dataset format. This presents reasons for the exclusion or inclusion of models for the explanation evaluation. Models can be excluded if their performances are too low and possibly influence the objective properties for the explanations. On the other hand, models can be included, although their performances are lower compared to similar models for reasons beyond mere performance.

As a final result, the model performances of the AE+RF model are shown. The results are not included in Table 5.1, because this model is explored as a proof of concept. Comparison of the multiclass results of the other models is made briefly, but not explored into depth.

5.1.1 Binary: RF-Basic vs. EBM-Basic vs. AE-Normal

The first binary comparison is between the RF-Basic, EBM-Basic and AE-Normal. This comparison evaluates the balance between model complexity and model performance for the UNSW-NB dataset. Although the train set differs for the AE model, it contains the same features and is sampled from the same original dataset. The RF and the EBM are less complex compared to the AE and the evaluation is done to see if a complex model outperforms less complex models on the same dataset.

Three models are compared for binary classification. Of these three models, two are trained on the same dataset. The AE-Normal is trained on a different dataset. A consequence of this is the difference in the test sets. This could affect the numbers in the confusion matrices.

Based on the metrics on the test set shown in Figure 5.1, the RF-Basic outperforms the other models. The EBM-Basic outperforms the AE-Normal.

The RF-Basic and EBM-Basic both show a higher number of false negatives than false positive, shown in the confusion matrices in Figure B.1 and Figure B.2. These models predict normal behaviour more often when the true class is malicious than when the class is normal. In the case of the AE-Normal it is the opposite: the number of false positives is larger than the number of false negatives (shown in Figure B.3). The number of false negatives of the AE-Normal is the lowest of all three models.

The Normal set has only normal samples in the train set. It does not contain any malicious samples, and all malicious samples are placed in the test set. Therefore, the AE test set contains more malicious samples that the other dataset. This can be seen in the true positive class of the confusion matrix of Figure B.3. The test set of the Normal set contains more than 160.000 correctly classified malicious samples compared to 30.000 samples for the other two models.

Taking into account Table 5.1, the difference in F2-score and recall of RF-Basic and AE-Normal is smaller than 0.035. This holds for the other metrics as well. The difference in scores on the test set is even smaller for RF-Basic and EBM-Basic, as it is below 0.005.

Based on the performance metrics, the RF-Basic outperforms the other two models. A possible explanation for the performance of the RF-Basic is the dataset. The dataset is of tabular format, which is a preferred format for Random Forests in general.

However, for a classification model in the field of NAD a high detection rate is preferred, which is why the F2-score and the recall are the two important scores. This indicates that the model has to be sensitive to anomalies.

Given that the AE is trained on normal data only, the AE appears to be more sensitive for anomalous data. This could be a possible explanation for the number of the false positives in the confusion matrix in Figure B.3, aside from the increased number of malicious samples in the test set.

The percentage represents the number of false positives out of the total number of malicious samples in the test sets. There are 32,037 malicious samples in the Basic set and 160,868 in the Normal set.

- RF: 1,451% false positives
- EBM: 2,400% false positives
- AE: 5,848% false positives

Although the Normal test set contains more malicious samples, the AE predicts more samples to be malicious compared to the other two models.

All models are combined with explanations and evaluated in the next section. Although the RF-Basic outperforms the other two models, there are other reasons to include the EBM-Basic and the AE-Normal. The EBM-Basic shows similar performance, but is inherently explainable. The AE-Normal shows comparable performance but is more difficult to explain than the other two models. Investigating the explanation differences together with the model differences can provide insight into the differences in explanations.

5.1.2 Binary: RF-Basic vs. RF-Latent vs. EBM-Basic vs. EBM-Latent vs. AE-Normal

The second binary comparison is between two models trained on the Basic set, two models trained on the Latent set and the AE-Normal. The whole comparison of the three models investigates whether there are differences in performance, as all models capture similar information of the dataset in a different manner. The Latent dataset is extracted from the AE latent dimensions. The Basic dataset has 61 features for classification while the Latent has 8. The comparison investigates the differences between the RF and EBM trained on a dataset with a higher dimensionality and on a compressed version of the dataset. The RF-Latent and EBM-Latent are essentially trained on the same latent dimensions as the AE-Normal can construct. The RF-Latent, the EBM-Latent and the AE-Normal are compared to see if there are differences between them in performance.

In this comparison, there are three different test sets. The Basic set, the Latent set and the Normal set. All test sets are equal in size. The test set of the Latent set has an equal number of malicious samples as the Basic set. In the previous subsection 5.1.1 the difference between the test set of the Normal and the Basic dataset was mentioned, as well as the differences in false positives and false negatives on the test set of the RF-Basic, EBM-Basic and the AE-Normal. The RF-Basic has higher scores on the performance metrics and a lower number of false positives, while the AE-Normal has a lower number of false negatives. The EBM shows a performance that is in between the RF-Basic and the AE-Normal.

The values of the RF-Latent metrics are in between the RF-Basic and AE-Normal (Table 5.1. The F2-score and recall are both 0.0088 lower than that of the RF-Basic. Compared to the AE-Normal, the F2-score is 0.0260 higher and the recall is 0.0255 higher of the RF-Latent. All scores are above 0.98.

The confusion matrix of the RF-Latent (Figure B.4 shows higher values of false negatives compared to the RF-Basic and the AE-Normal. The absolute number of false positives is higher than those of the RF-Basic, as well as the relative percentage (7,142%). The percentage is higher compared to the percentage of AE-Normal.

Similarly to RF-Latent, the values of the metrics of the EBM-Latent are in between the EBM-Basic and the AE-Normal, shown in Table 5.1. Comparing the EBM-Latent with the EBM-Basic, the F2-score is 0.0048 lower and the recall is 0.0047 lower. Compared to the AE-Normal, the F2-score is 0.0253 higher and the recall is 0.0247 higher. Again, all scores are above 0.98.

The number of false negatives and false positives is higher compared to the EBM-Basic, shown in Figure B.5. The number of false negatives is higher compared to the AE-Normal, while the false positives are lower. The percentage of false positives is higher than the percentage of both the EBM-Basic and AE-Normal (7,338%).

There are minor differences in the performances of the RF-Latent and EBM-Latent. They show similar patterns: their performance metrics are in between the same model trained on the Basic set and the AE-Normal and the percentages of false positives are higher than the AE-Normal.

All models are evaluated with an explanation in the next section. The RF-Basic, EBM-Basic and the AE-Normal were already included by the previous comparison, but the RF-Latent and EBM-Latent show comparable performances. As they differ in the dataset from the RF-Basic and the EBM-Basic, this can show the influence of a dataset on the explanation performances.

5.1.3 Multiclass: RF-Basic vs. EBM-Basic

The third comparison is a multiclass comparison between the RF-Basic and the EBM-Basic. Although hyperparameter tuning for both models showed their inability to correctly classify minority samples (Section 4.3.2 and Section 4.3.2), this comparison tries to find if one model outperforms the other model to possibly include in the explanation evaluation.

Based on metrics only in Table 5.1, the RF-Basic performs better than the EBM-Basic for multiclass classification. The recall of the EBM-Basic is 0.0190 lower compared to the RF-Basic. The F2-score of the EBM-Basic is 0.0244 lower compared to the RF-Basic. The accuracy difference is similar to the recall difference, while the precision difference is the largest of all metrics.

From the confusion matrix in Figure B.7 it shows that the EBM-Basic does not predict every class. Five of nine classes are not predicted by the model once. This is confirmed by the classification report in Table B.3. Using the confusion matrix in Figure B.8 Although not always correct, shown in the confusion matrix (Figure B.6) and the classification report (Table B.2), the RF-Basic predicts every class at least once. It appears to be able to capture the diversity of the dataset better than the EBM-Basic.

The performance on individual classes and on average shows that the RF-Basic outperforms the EBM-Basic for the multiclass classification, but the RF-Basic is not able to classify most of the minority classes correctly.

Both models are not combined with an explanation for further evaluations. They do not show accurate model performance on minority classes, which might affect the explanation performance.

5.1.4 Multiclass: RF-Minority vs. RF-SMOTE

The fourth comparison is made between RF-Minority and RF-SMOTE. This is a multiclass comparison. Both datasets are balanced and have shown during hyperparameter tuning that they are able to classify most individual classes correctly. The datasets differ in other aspects, such as size. The comparison is made to see how differences between the datasets might influence performance and what the specific differences in performance are.

RF-SMOTE appears to outperform RF-Minority, given the performance metrics in Table 5.1.

Examining the confusion matrices (RF-Minority: Figure B.8, RF-SMOTE: Figure B.9), the minority model shows difficulties in distinguishing between the classes 'analysis' and 'backdoor'. This was shown earlier in Section 4.3.2. The classification report of the RF-Minority (Table B.4) confirms the observation of the confusion matrix and shows that the model has the lowest metrics for the class 'analysis' and 'backdoor'. It does not include the 'normal' class because the purpose of the model is to distinguish between malicious classes.

Aside from the two classes mentioned above, most classes have a precision, recall, and F-score comparable to the SMOTE set. There is no direct explanation for RF-Minority performance on these two classes. Knowing that the model has difficulties predicting these classes, one can consider adding more samples of these classes. But the prediction mistakes are not necessarily due to having not a lot of samples, because 'worms', as the lowest occurring class, are predicted correctly almost every time. The performance on 'analysis' and 'backdoor' can indicate the model is not able to generalise between these classes with the given number of samples, while other classes are different enough that they only need a few samples to capture the diversity within the class.

Although the RF-SMOTE shows high metrics for all classes, the confusion matrix in Figure B.9 shows that the model makes mistakes in all classes. The observation is that, with more data, the model appears to make mistakes for all classes, in general.

Performance-wise, the RF-SMOTE outperforms the minority model. However, the SMOTE set is approximately 150 times bigger than the minority set.

A possible downside of SMOTE is that it can introduce noise and create more overlap between classes. It does not take into consideration neighbouring examples can be from other classes. In other words, both are close, and the decision boundary is vague. This can increase the overall difference between classes and introduce additional noise, but this is not reflected by the scores in the classification report nor the confusion matrix (Table B.5 and Figure B.9)

Using SMOTE for multiclass, it appears that a RF works better in the multiclass case if the dataset is balanced. The RF-Minority model confirms this observation, in addition to predictions of 'analysis' and 'backdoor'. To capture the diversity of the classes for the multiclass classification, the dataset needs a different number of occurrences of each class.

Both models are evaluated in combination with an explanation. The RF-SMOTE outperforms the RF-Minority, but it is clear where the RF-Minority falls short. Using explanations, these shortcomings can be highlighted and compared with similar model predictions and explanations of the RF-SMOTE. Also, the RF-SMOTE shows accurate performance on multiclass classification, which is why this model is included to be evaluated further with an explanation.

5.1.5 Multiclass: RF-Basic vs. RF-Latent

The final multiclass comparison is made between the RF-Basic and RF-Latent. Mentioned above, the RF-Basic did not show the ability to classify minority classes correctly. The RF-Latent for multiclass showed similar behaviour, during hyperparameter tuning (Section 4.3.2). Through the comparison, both models are evaluated in depth to see if there are performance differences due to the dataset and if the performance is good enough to continue to the explanation evaluation stage.

Looking at Table 5.1, the performance of the RF-Basic multiclass is better than the performance of RF-Latent multiclass.

Including confusion matrices (RF-Basic: Figure B.1 and RF-Latent: B.10, both models show similar behaviour. They are able to predict the majority classes, but not the minority classes. Although both models show this behaviour, the metrics of the RF-Basic appear to be higher for medium-sized classes, such as 'fuzzers' and 'exploits', compared to the RF-Latent. Additionally, RF-Latent does not predict the class 'worms' once. This is clearly shown in the confusion matrix in Figure B.10.

The difference in performance for individual classes between the two models may be due to a lower number of features for the Rf-Latent. A lower number of features might provide less information to distinguish between classes.

Both models are not included for further evaluation with an explanation. Both models do not show accurate performance on most of the minority classes, which might affect the explanation.

5.1.6 Additional Multiclass: AE+RF

As the AE+RF is a proof of concept, no hyperparameter tuning was performed. The AE-Normal was chosen as the AE and an RF with default parameters. Table 5.2 shows the performance metrics of the AE+RF framework.

Model	**Accuracy**	**F2-Score**	**Precision**	**Recall**
AE-RF	0.9770	0.9789	0.9869	0.9770

Tab. 5.2. The model performance scores of the AE-RF model for multiclass classification.

The AE model correctly predicts 214590 samples correctly and 418 samples incorrectly as 'normal', which leaves 38997 samples for the RF to classify. Figure B.11 shows that the model still makes mistakes for the minority classes. From the confusion matrix it appears that the model incorrectly predicts many instances to be 'analysis' or 'exploits'. Another mistakes is made for 'fuzzers', which is often predicted while the class is 'normal'. The difficulty with the 'analysis' class was seen previously for the RF-Minority, so this could be expected. However, the difficulty for the other classes was not expected.

The average performances of the model are displayed in Table 5.2. The average performances are lower compared to the other multiclass models, but higher than the RF-Minority. The values of the individual classes are shown in Table B.7. Although performances on individual classes are higher compared to the RF-Basic, RF-Latent, EBM-Basic and the RF-Latent, the performances are lower compared to the RF-SMOTE and the RF-Minority model.

As the model serves as a proof of concept and the performance metrics are lower compared to other multiclass models, the model is not included for further investigation. The isolated RF-Minority model is used for explanation purposes in the next phase.

5.1.7 Summary

Given model performance on the binary classification problem, the RF-Basic outperformed the EBM-Basic ad the AE-Normal based on the performance metrics. Given the confusion matrix, the AE-Normal seems to be more sensitive to false positives than the other two models. The RF and EBM trained on the Latent set show comparable performances to the RF and EBM trained on the Basic set, even though this model has 8 features instead of 62.

For the multiclass case, the RF-SMOTE is the only model that is able to classify all individual classes correctly. The RF-Minority performance comes close for most individual classes, but not all. The other multiclass models show good performance on the individual classes with high occurrences in the dataset, but poor performance on the minority cases.

Given the results on the model performance, certain models are selected to be combined with an explanation and examined in the next section.

The following models and explanations are compared:

- Binary: RF-Basic+LIME, RF-Basic+SHAP, EBM-Basic, AE-Normal
- Binary: RF-Basic+LIME, RF-Basic+SHAP, RF-Latent+LIME, RF-Latent+SHAP,
- multiclass: RF-SMOTE+LIME, RF-SMOTE+SHAP, RF-Minority+LIME, RF-Minority+SHAP.

5.2 Explanation Performance

After the model evaluation, the explanations were evaluated for sensitivity and fidelity. The results are shown in Table 5.3. The sensitivity scores are 0 at best and can theoretically be infinite. The fidelity scores are percentages, where a high percentage is preferred. The fidelity scores are included for three perturbation levels.

Explanation	**Sensitivity**	**Fidelity 1**	**Fidelity 2**	**Fidelity 3**
Binary classification				
RF-Basic SHAP	0.0611	4.13	7.81	10.65
RF-Basic LIME	0.0339	2.03	5.90	9.78
RF-Latent SHAP	0.1139	23.48	28.97	31.13
RF-Latent LIME	0.0641	10.87	17.78	21.65
EBM-Basic	0.0743	9.42	12.42	15.09
EBM-Latent	0.0623	13.55	18.65	22.61
AE-Normal	0.3512	6.79	8.05	9.61
multiclass classification				
RF-SMOTE SHAP	0.0692	9.94	12.0	13.0
RF-SMOTE LIME	0.0447	3.84	3.61	2.84
RF-Minority SHAP	0.0489	29.71	29.03	29.29
RF-Minority LIME	0.0683	3.87	4.23	5.16

Tab. 5.3. Scores of explanation methods of models on datasets for binary and multiclass classification

In the following subsections, explanations are compared. There are three comparisons, of which two are on binary classification and one is on multiclass classification. The goal of these comparisons is to find differences between explanations of models of different complexity. Additionally, several RF trained on different datasets are evaluated in combination with LIME and SHAP. This provides insights into the difference a dataset might induce in explanation methods.

Comparisons are based on the results of Table 5.3 and supporting figures. For most comparisons, individual explanations are added to show the difference in the explanation methods. In order to exclude the visual differences, a barplot with feature importance is used for every explanation. The feature importances shown in the figures are normalised. A general note about the feature importance of the AE is that it often shows clear feature importances for a few features, while others are low. This affects visibility.

There are three general cases of explanations for individual points. For the first two cases, all models correctly predict the label. The first case is the correct prediction of the normal label and the second case is the prediction of the malicious label. The third case is an explanation of a point where the model is incorrect. To make a fair comparison between explanations by different methods, ideally the point occurs in all test sets of the trained models. This is not always possible because the point has to adhere to the case requirement and occur in all test sets. It is mentioned if the point does not occur in all three test sets.

The RF-Minority is examined without the attachment of the AE for explanation purposes.

5.2.1 Binary: RF + SHAP vs. RF + LIME vs. EBM vs. AE

The first comparison is made between three different models that all have a different explanation method. The RF uses two model-agnostic methods, the AE uses a model-specific personalised method, and the EBM is inherently explainable. All models differ in complexity, with the AE being the most complex and the EBM the least. Their performances were comparable, and this comparison investigates if their explanations are too.

Table 5.3 includes the sensitivity and fidelity scores of the explanations of the RF-Basic, EBM-Basic and AE-Normal for binary classification. For the RF, both LIME and SHAP are included. The EBM is inherently explainable and the AE uses a personalised method.

First, the sensitivity score is compared. A low sensitivity score is preferred, as this indicates the explanation is not sensitive to small perturbations that do not alter the prediction. The RF-Basic+LIME has the lowest sensitivity for the binary classification. This is followed by RF-Basic+SHAP and EBM-Basic. The sensitivity of the AE is the highest compared to the other three.

Secondly, the fidelity scores are compared. The EBM-Basic shows the highest fidelity scores for all perturbation levels, while the RF-Basic+LIME shows the lowest fidelity scores for all levels. RF-Basic+SHAP and AE-Normal show comparable fidelity scores. All models show an increase in fidelity scores with increasing perturbation strength.

The sensitivity is plotted against the fidelity to show the spread of the values through the dataset. Figures C.1, C.2 and C.3 show the spreads for the explanations of RF-Basic, EBM-Basic and AE-Normal.

All explanations show an increase in the spread of fidelity values when the perturbation strength increases noted above about the results of Table 5.3.

RF-Basic+LIME shows less spread in sensitivity and fidelity values at lower perturbation strengths than the RF-Basic+SHAP. The EBM-Basic shows a wide spread of values at perturbation strength 1 already, but the number of number of explanations per fidelity value appears to be more divided at higher perturbation strengths. The EBM-Basic shows a similar spread of sensitivity values as the RF-Basic+LIME, which is a smaller spread of the sensitivity values than the RF-Basic+SHAP or the AE-Normal.

Similarly to the RF-Basic explanations and the EBM-Basic, the spread of the AE increases as the perturbation strength increases, but the spread appears to be more strictly separated. There are many explanations with a fidelity of 0 or close to 0, followed by a jump to a fidelity around 35-40. This can be a result of the method of calculation of the feature importances.

Figure 5.1 shows four explanations for point 1, where all models correctly predict the label to be normal. Except for the AE, all other three explanations have *ct_state_ttl* in their top 5 features. Both RF explanations have that feature as the most important feature. The AE has *dttl* as its most important feature, while this is the second most important feature for the RF-Basic+LIME. This does not occur in the other explanations.

The RF-Basic+SHAP and the EBM-Basic explanations have the most overlap in their top 5, as three features occur in both explanations.

For point 1, it looks as the feature importances of the EBM-Basic explanation are gradually increasing. For the AE, there is one high peak, two smaller peaks and two values too small to show in the figure. The RF-Basic+LIME shows a similar high peak for its two most important features, followed by lower importances. The RF-Basic+SHAP shows a high peak for its most important feature, but the features after that shows similar feature importances that are lower.

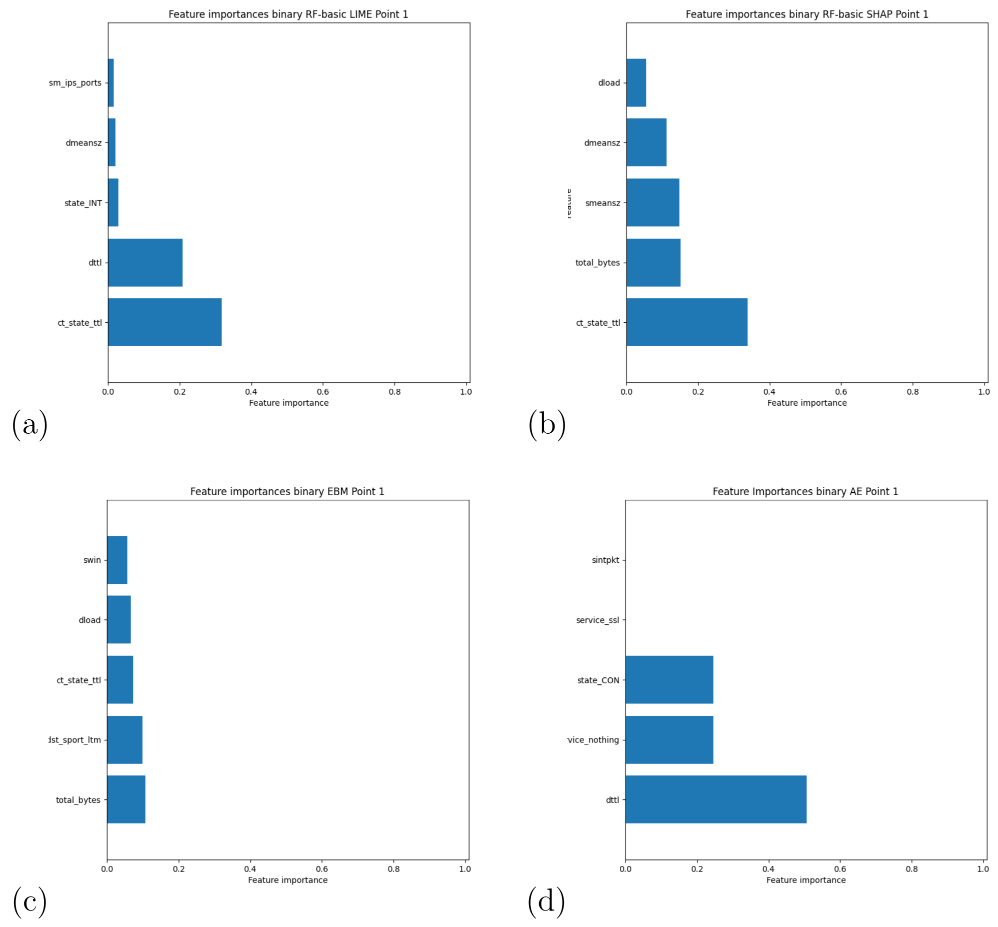


Fig. 5.1. Feature importances of different explanations with all models correctly predicting the label ('0') of the same point. *(a)* RF-Basic + LIME with strength 1 *(b)* RF-Basic + SHAP with strength 1 *(c)* EBM-Basic with strength 1 *(d)* AE-Normal with strength 1

For the second point, the explanations are shown in Figure 5.2. For this point, the models correctly predict the malicious sample. The two explanations of RF-Basic and the EBM share the most important feature, which is again *ct_state_ttl*. That feature does not occur in the AE explanation. The RF-Basic explanations share four out of five features, although the order differs. The EBM-Basic shares 3 features with RF-Basic+SHAP and 2 with the RF-Basic+LIME. The AE does not share any features with the other explanations. It also shows two high peaks, followed by values too small to be visual.

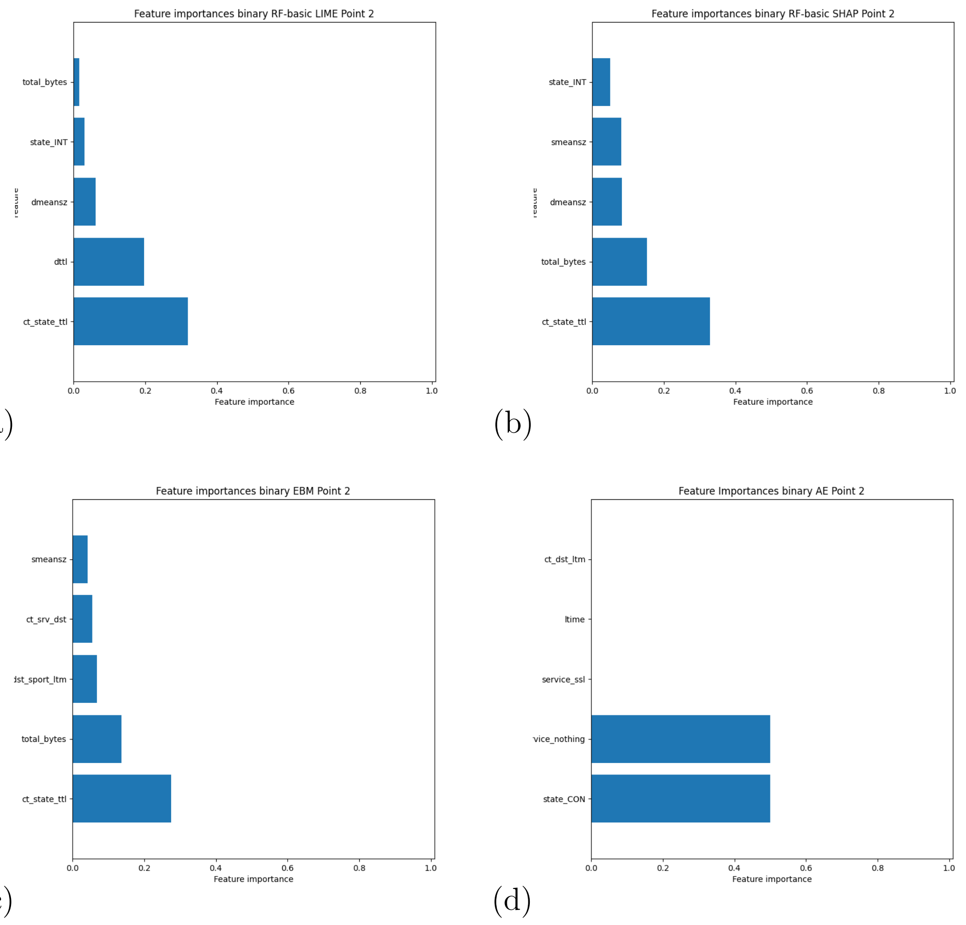
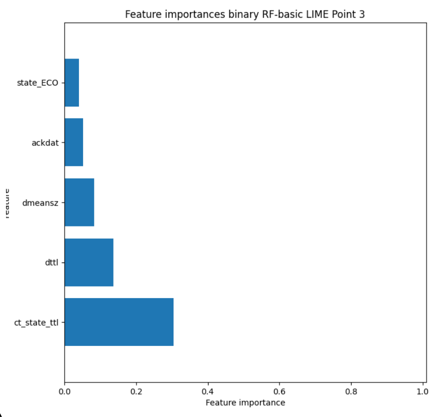
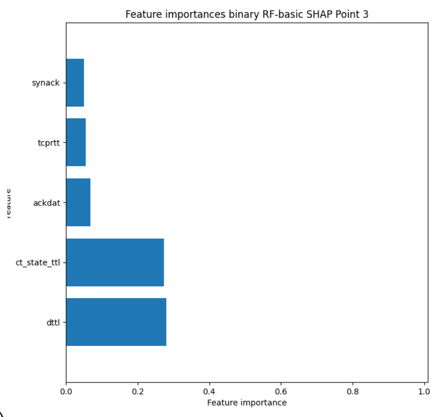


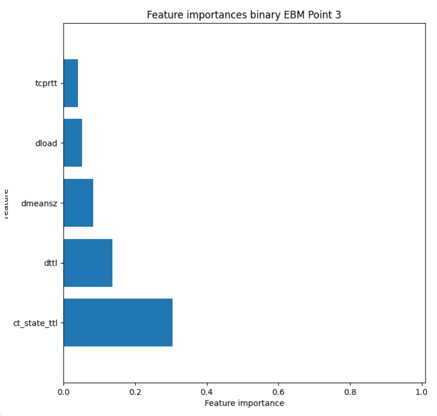
Fig. 5.2. Feature importances of different explanations with all models predicting the label ('1') correctly. The prediction is '1', but the label is '0' (a) RF-Basic + LIME with strength 1 (b) RF-Basic + SHAP with strength 1 (c) EBM-Basic with strength 1 (d) AE-Normal with strength 1



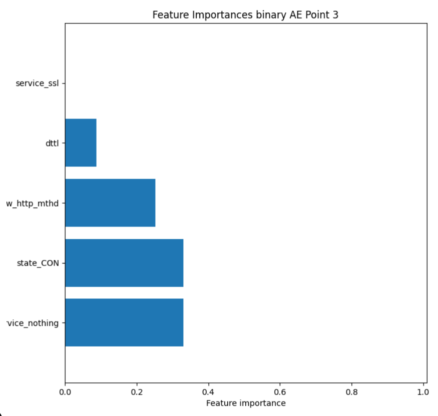
(a)



(b)



(c)



(d)

Fig. 5.3. Feature importances of different explanations with all models predicting the label incorrectly. The prediction is '1', but the label is '0' (a) RF-Basic + LIME with strength 1 (b) RF-Basic + SHAP with strength 1 (c) EBM-Basic with strength 1 (d) AE-Normal with strength 1

Figure 5.3 shows explanations for the third point. All models incorrectly predict the label to be malicious, while it is normal. Again, EBM and both RF explanations have *ct_state_ttl* in their top features. A difference now is that all explanations have *dttl* in their top features. The EBM-Basic and the RF-Basic+LIME share the top 3 features in the same order. The RF-Basic+SHAP shares two features with the RF-Basic+LIME and the EBM-Basic. Aside from one shared feature, the AE does not share more features with the other explanations.

5.2.2 Binary: RF and EBM Basic vs. RF and EBM Latent

The second comparison is made between two models trained on the Basic set and two models trained on the Latent set. Both the RF-Basic and RF-Latent use LIME and SHAP as explanations. The Basic and Latent differ in the train set. The comparison can provide information on differences in the explanation properties as a result of different datasets. The focus lies on the comparison between the Basic models and the Latent models, as the Basic models were already compared in the previous section. Although the latent dimensions are uninterpretable by humans, this comparison is made because the Latent model showed comparable model performance and the interest lies with the influence of the dataset on the explanation. A comparison is made between the RF-Latent and EBM-Latent explanations are well, because the explanations could provide information on which latent dimensions are important for predictions.

Shown in Table 5.3, the RF-Latent explanations have a higher sensitivity value and a higher fidelity value for every perturbation strength, compared to the RF-Basic explanations. The sensitivity score of the RF-Latent+SHAP is the highest of all four explanations. RF-Basic+LIME has the lowest sensitivity score, followed by RF-Basic+SHAP. The RF-Latent+SHAP has the highest fidelity scores of the explanations for the binary classification models for all three perturbation strengths. The RF-Latent+LIME has lower fidelity scores compared to RF-Latent+SHAP, but these scores are still higher than the fidelity scores of both explanation methods for RF-Basic. All explanations show an increase in fidelity scores as the perturbation strength increases.

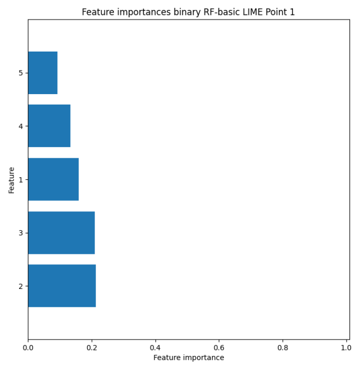
Figures C.4 and C.1 show the spread of sensitivity and fidelity of the RF-Latent and RF-Basic explanations. Similar to earlier findings, the fidelity spread increases with the strength of the perturbation. The fidelity spread of RF-Latent is wider than that of RF-Basic, which was also noted in Table 5.3. However, Figure C.4 shows that the spread of the sensitivity is larger for both explanations of RF-Latent compared to the RF-Basic explanations in Figure C.1.

From Table 5.3, the EBM-Latent explanations show a lower sensitivity score and higher fidelity scores compared to the EBM-Basic. The EBM-Latent explanations have a lower sensitivity compared to both LIME and SHAP explanations of the RF-Latent, while the fidelity scores are in between.

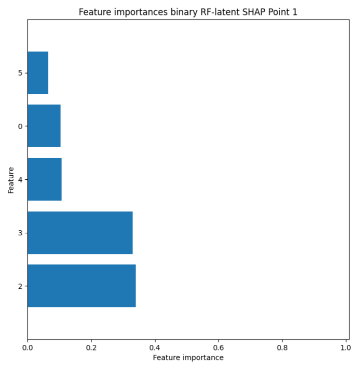
The fidelity scores do show the increment that goes with the increase in perturbation strength, as other models have shown before.

Figures C.5 and C.2 show the spread of sensitivity and fidelity of the EBM-Latent and EBM-Basic explanations. The fidelity spread for both models increases with the strength of the perturbation. From the Figures in C.5, the spread of the sensitivity value is smaller. This opposite is visible in the comparison between the RF-Latent and the RF-Basic.

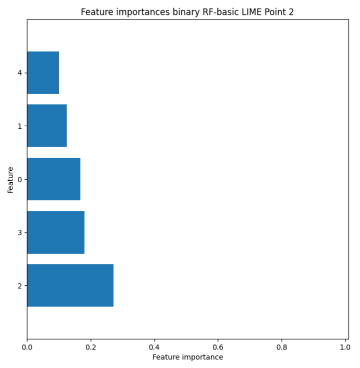
As it is unclear what each latent dimension represents, explanations for specific points cannot be compared to the models trained on the Basic set. To see which latent dimensions are of importance for predictions, explanations of individual points of the Latent set are included. The three points compared are the same three points for the RF and the EBM explanations.



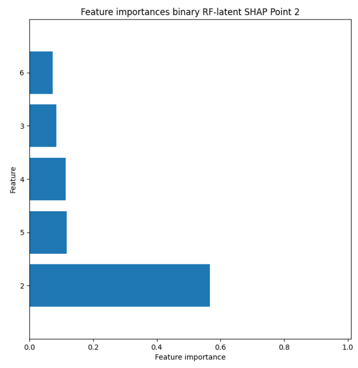
(a)



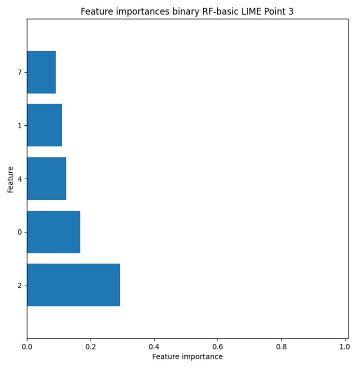
(b)



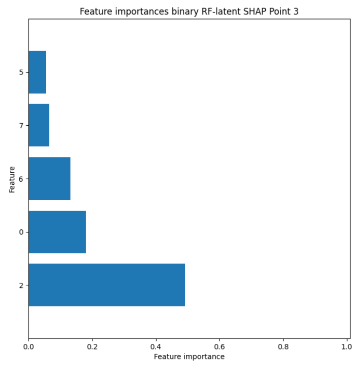
(c)



(d)

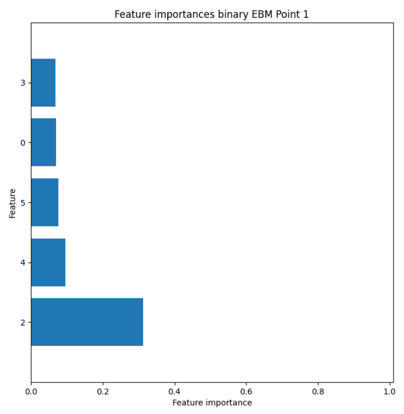


(e)

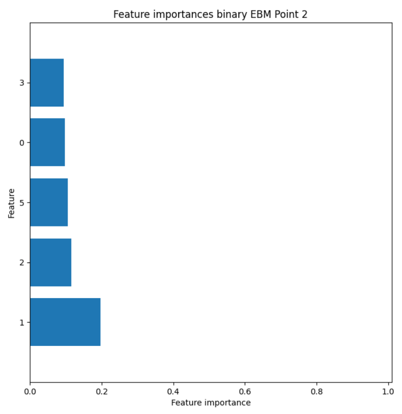


(f)

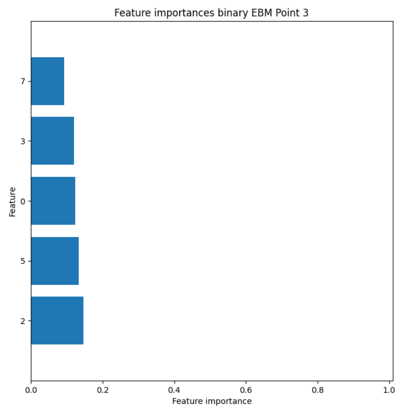
Fig. 5.4. Explanations for RF-Latent points (a) RF-Latent + LIME for case 1 (b) RF-Latent + SHAP for case 1 (c) RF-Latent + LIME for case 2 (d) RF-Latent + SHAP for case 2 (e) RF-Latent + LIME for case 3 (f) RF-Latent + SHAP for case 3



(a)



(b)



(c)

Fig. 5.5. Explanations for EBM-Latent points (a) EBM-Latent for case 1 (b) EBM-Latent for case 2 (c) EBM-Latent for case 3

The three points of the RF-Latent are explored first. For point 1, where the RF correctly predicts the normal sample, the latent dimension *3* is the second most important for both LIME and SHAP. Latent dimension *3* is also important for the LIME explanation of point 2, where the model correctly predicts an anomaly, but less important for SHAP. For the SHAP explanation of point 2, aside from latent dimension *2*, other latent dimensions show similar feature importances. For the LIME explanation, the feature importances decrease gradually. For the third point, where the RF predicts anomaly but the label is normal, latent dimension *0* is the second most important feature for LIME and SHAP. In the top 5 of both explanations, latent dimension *7* occurs, which has not occurred in the explanations for other points. For all three points, every latent dimension occurs at least once in an explanation.

Compared to the RF-Basic explanations, the *ct_state_ttl* feature occurred in every explanation as the most important feature, except for SHAP for point 3. Not every feature occurred at least once in every explanation, but the model has 62 features instead of 8.

For the three explanations of the EBM, latent dimensions *0, 2, 3* and *5* occur in every explanation. All explanations differ with one latent dimesion. Interestingly, latent dimension *4* is the second most important for point 1 but occurs nowhere else. Similarly, latent dimension *1* is the most important point for point 2, but occurs in no other explanation of the EBM.

For the EBM-Basic, the *ct_state_ttl* feature occurred in every explanation. There is no other feature of the Basic set that occurs in every explanation.

Between the RF-Latent and the EBM-Latent explanations, it differs per point if they show overlap in the latent dimensions. Latent dimension *2* the most or second-most important features for all explanations. In general, the explanations of the EBM-Latent show more overlap in the explanations of different points compared to the explanations of the RF-Latent. For the first point, the EBM-Latent shows overlap with the explanation of RF-Latent+SHAP. Although the order of the features differ, features occur in both explanations. The explanation of the EBM-Latent differs one feature with RF-Latent+LIME. The second point differs more between the explanations of the two models. Both explanations of the RF-Latent have latent dimension *2* as the most important feature, while this is latent dimension *1* for the EBM-Latent. Also, latent dimension *4* is absent from the EBM-Latent explanation, while present for the two other explanations. The third point shows a clear feature importance for latent dimension *2* for the explanations of LIME and SHAP, but this is absent for the explanation of the EBM. It does have latent dimension *2* as the most important, but other importances are close.

Compared to the Basic set, Latent set has fewer features than the Basic set, which is a possible explanation for the increased sensitivity and fidelity scores. Perturbing 1 of 8 total features might affect the explanation or prediction faster than perturbing 1 of 62 features. Considering sensitivity, a small perturbation might change the explanation more, as the explanation depends more heavily on each feature. Looking at fidelity, perturbing a feature if the total number of features is low might push a point over a decision boundary more easily. The explanations of RF-Latent and EBM-Latent showed a clear presence of latent dimension *2* and overlap in the important latent dimensions for the first point. However, other explanations lacked overlap. There were similar differences between RF-Basic and EBM-Basic explanations. They often shared the most important feature, but the other features and the amount of importances were different.

A downside of the latent dimension is the lack of explainability. Although the explanations can show the importance of each latent dimension for the prediction, it is unclear which input features map to the latent dimensions. Aside from this downside, the Latent explanations display the difference in objective properties compared to Basic explanations. The difference between the explanations is the dataset, which shows an influence of the dataset on the explanation properties.

5.2.3 Multiclass: RF-SMOTE vs. RF-Minority, both LIME and SHAP

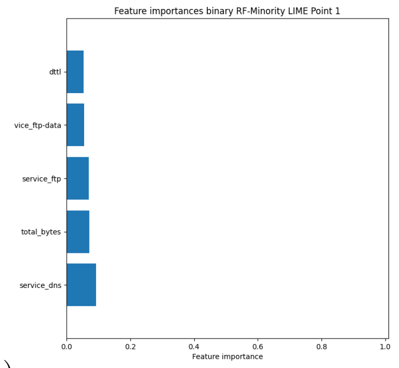
The final comparison is made between the model-agnostic explanations of RF-SMOTE and RF-Minority. Both models use the same explanation methods. This provides insights on the effect of different datasets on the explanation methods for multiclass classification. The RF minority showed difficulty in predicting two classes, although both classes were equally represented in the train data. This comparison might provide insight into the inability to correctly predict these classes.

Given the results in Table 5.3 for the RF-SMOTE and RF-Minority explanations, the sensitivity values lie close, while the fidelity scores do not. Both models adhere to earlier findings, where SHAP has a higher sensitivity than LIME, as well as a higher fidelity.

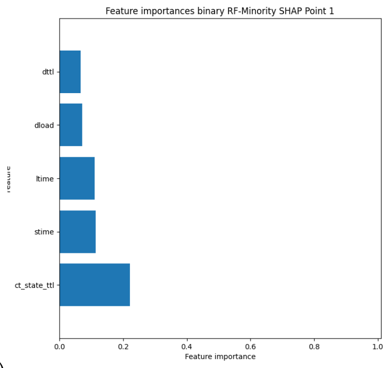
Contrary to earlier findings, the fidelity of explanations of both models do not or rarely increase with the perturbation strength. The fidelity values look stable compared to earlier explanations. From Figure C.6 for the RF-Minority explanations, it appears that the width of the spread for fidelity for both explanations does not increase, but the samples appear to be more spread over the whole range when the perturbation strength increases. The spread of the explanations of RF-SMOTE in Figure C.7 shows similar behaviour, but the samples appear to be packed more closely.

There are no overlapping points between the two test sets of SMOTE and

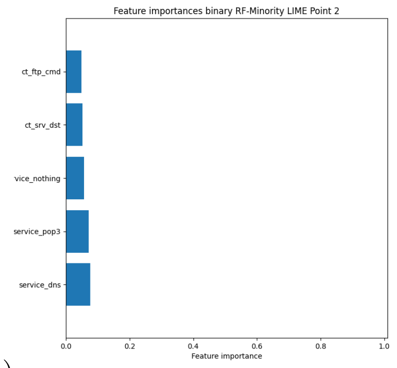
Minority. Therefore, the points discussed are not the same points. However, all points adhere to the same cases. The first case is predicting the normal sample correctly, the second case is predicting a malicious sample correctly, and the third case is wrongfully predicting a 'backdoor' sample to be 'analysis'. The latter is chosen, as earlier results from Section 5.1.4 have shown that the Minority model has difficulty distinguishing between these two classes.



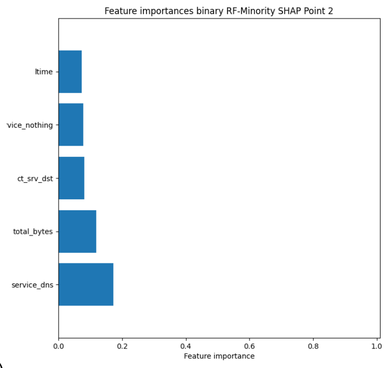
(a)



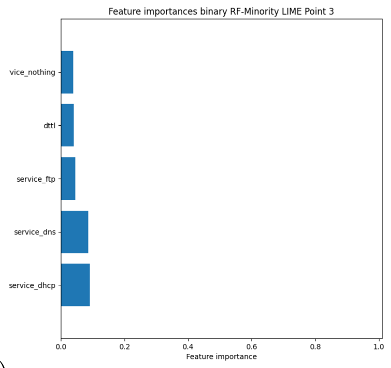
(b)



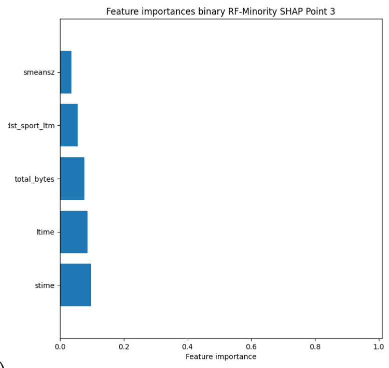
(c)



(d)



(e)



(f)

Fig. 5.6. Explanations for RF-Minority points (a) RF-Minority + LIME for case 1 (b) RF-Minority + SHAP for case 1 (c) RF-Minority + LIME for case 2 (d) RF-Minority + SHAP for case 2 (e) RF-Minority + LIME for case 3 (f) RF-Minority + SHAP for case 3

The thing to note about the explanation the points of RF-Minority, *ct_state_ttl* is absent from the top 5 features, except in the SHAP explanation of point 1. In general, it appears that the important features are different between points and between explanations. The features *stime*, *ltime*, *vice_nothing*, *service_dns* and *dttl* appear 3 times in 6 explanations, but other features occur less often. The 'time' features do not occur in the explanations of LIME. For all points and explanations, there appears to be diversity among the feature importances.

In the first case, both explanations of the RF-Minority show the feature importances for the correct prediction of a normal sample. The only overlapping feature is *dttl*, which is the fifth most important feature in both explanations. For RF-SMOTE, there is no overlap between features of the LIME and SHAP explanations. The LIME explanations of both models are similar, as they have four overlapping features. The SHAP explanation of the RF-Minority is similar to the SHAP explanation of the RF-SMOTE. There are three overlapping features, the most important feature being *ct_state_ttl*.

In the second case, the label is normal data, but the prediction is anomalous ('exploits'). Between the LIME and SHAP explanations of the RF-Minority, there are 3 overlapping features: *service_dns*, *vice_nothing* and *ct_srv_dst*. The RF-SMOTE correctly predicts 'exploits' for the second case as well. Aside from *total_bytes*, there are no overlapping features between LIME and SHAP of RF-SMOTE. There is one overlapping feature for RF-Minority LIME and RF-SMOTE LIME, while there are three overlapping features between RF-Minority SHAP and RF-SMOTE SHAP.

For the third and final case, the label is 'analysis' but the prediction is 'backdoor'. Section 5.1.4 showed that the RF-Minority has difficulties with distinguishing between these two classes. There are no overlapping features between the LIME and SHAP explanations. It could indicate that because the model has difficulties predicting the correct class, there are no clear features that play an important role. Although the RF-SMOTE model did not show poor performance on both classes, there is still a case where the model predicts 'analysis' while the label is 'backdoor'. There is one overlapping feature between LIME and SHAP: *total_bytes*. There is one overlapping feature between RF-Minority and RF-SMOTE LIME, while there are three overlapping features between RF-Minority and RF-SMOTE SHAP.

In general, the explanations made by the same explanation method for points with similar requirements of two different models show overlap in the features. The models are trained on different datasets, but appear to use similar features for predictions, although it depends on the explanation method which features are chosen. There appears to be more overlap in the features chosen by SHAP than by LIME.

5.2.4 Summary

In total, three comparisons were made to evaluate the explanations of the selected models. Overall, the LIME for the RF showed the lowest sensitivity values on the Basic set. The EBM-Basic had the highest fidelity values on the Basic set. The RF-Latent+SHAP showed the highest fidelity values, but the latent dimensions are uninterpretable. The fidelity values appeared to be dataset dependent, as the RF-Basic explanations had different fidelity values compared to the RF-Latent explanations.

For the multiclass classification, the RF-Minority+SHAP had the highest values for fidelity and the second lowest sensitivity value. Again, from the observations of the multiclass cases, the fidelity appeared dataset-dependent, as the results differed between RF-Minority and RF-SMOTE explanations.

The binary explanations showed an increase in fidelity values with the increase in perturbation strength, while this increase was often absent at the multiclass explanations. The individual points showed some overlap in feature importances between different explanations.

In the multiclass case, SHAP explanations showed more overlap between different models compared to LIME explanations. However, this finding is based on one comparison only.

5.3 Explanation and Model Evaluation Summary

For the binary case, the RF-Basic outperformed the EBM-Basic and the AE-Normal, considering the performance metrics. The RF-Latent showed performance lower compared to the RF-Basic. Considering the explanation performances, the RF-Basic+LIME had the lowest sensitivity values. The RF-Latent+SHAP had the highest fidelity values. Although the explanations for the models trained on the Latent set showed the highest fidelity scores, they are not considered as a final combination, as the Latent dimensions are not explainable. If they are excluded, the EBM-Basic had the highest fidelity values. The fidelity values appeared to be dataset-dependent, as the fidelity values of RF-Basic+SHAP and EBM-Basic are lower than RF-Latent+SHAP and EBM-Latent.

For the multiclass classification problem, the RF-SMOTE and the RF-Minority were the only two models that showed accurate performance on individual classes. The RF-SMOTE outperformed the minority model. The RF-Minority model showed similar performance to the RF-SMOTE in predicting individual classes but had difficulties distinguishing between two individual classes, which lowered its average performance metrics. Other models showed a high value for the average performance metrics, but low values for individual class performance metrics. Although RF-SMOTE outperformed RF-Minority, this was not the case for the explanations. Sensitivity values appeared equal, while the fidelity values for the RF-Minority explanations were higher compared to the RF-SMOTE explanations.

Combining the results from the binary Basic and Latent explanations of RF and EBM with the multiclass RF SMOTE and Minority explanations, the differences indicate that there might be a dataset-dependence of not only model performance, but objective explanation properties as well. This confirms previous research [23, 46].

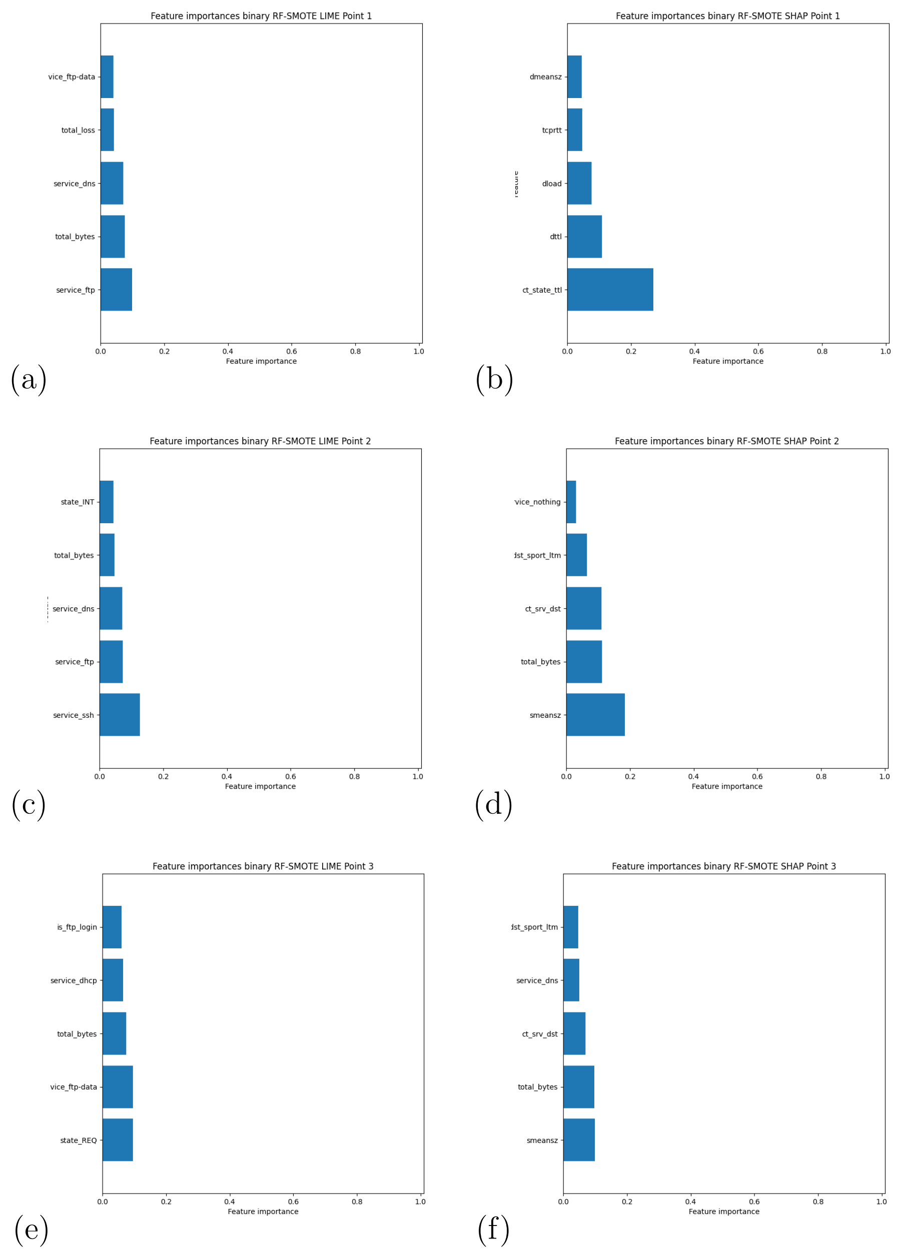


Fig. 5.7. Explanations for RF-SMOTE points (a) RF-SMOTE+LIME for case 1 (b) RF-SMOTE+SHAP for case 1 (c) RF-SMOTE+LIME for case 2 (d) RF-SMOTE+SHAP for case 2 (e) RF-SMOTE+LIME for case 3 (f) RF-SMOTE+SHAP for case 3

Chapter 6

Discussion

This study examined the objective properties of explanations with the aim to fill the gap identified for the objective assessment of explanations in the field of NAD. The contribution of this research is the evaluation of explanation methods through quantification of the objective properties of explanations.

In the following sections, the research conducted and the results are discussed. Each section of the result tries to answer one of the subquestions:

1. R1: Which model shows the best performance with respect to model metrics?
2. R2: Which explanation shows the best performance regarding objective explanation metrics fidelity and sensitivity?
3. R3: Which combination of model and explanation creates the best compromise between explainability and model performance?

First, the results are of the model and explanation evaluation are discussed. The results are followed by the limitations of this study. Finally, suggestions for future work are presented.

6.1 Model Performance

Taking into account model performance metrics only, this thesis showed that the RF outperforms the other two models on binary classification. The RF-Basic scored highest on the performance metrics. Although the RF had the highest absolute scores, all other binary models scored above 0.95 on all metrics. This indicates that their performances lie close and that other factors could influence the final performance. Examples of factors are the format of the dataset or the selection of hyperparameters.

Interestingly, the RF trained on 8 features showed a performance comparable to that of the RF trained on a dataset with 61 or 294 features. The 8 latent dimensions appear to capture enough information for the model to decide correctly whether a sample is normal or malicious. Similar performance patterns were shown by the EBM-Basic and EBM-Latent. This strengthens the assumption that the format of the dataset influences model performance [27], as well as confirms previous research that AE is suitable as a dimensionality reduction method [77].

Both the RF and the EBM perform well on tabular data, indicating that the difference between the RF and the EBM may be due to choices in feature engineering with a small influence. As mentioned earlier, tree-based models outperform deep learning models on tabular data [27]. This is a possible explanation for the performance gap between the RF and the EBM with the AE. As the AE is a neural network, it is not an optimal model for tabular data. Although the dataset is not in the optimal format for the AE, the model scored above 0.96 on all metrics.

Hyperparameter tuning was performed, but the number of values and combinations was limited by hardware. Training the RF was computationally inexpensive compared to the EBM and the AE, which resulted in the opportunity to try more hyperparameters. The performances of the EBM on the test set lie close to the RF, with a difference below 0.001. This may be due to hyperparameter settings. Training the AE took longer than the other two models on an equally sized dataset, which is why less hyperparameters were tuned. It is possible that the settings are not optimal for the binary classification problem, on top of the dataset format. Tuning hyperparameters for a neural network is a difficult task by itself [19]. Although less hyperparameters were tried, AE did show increased sensitivity for malicious samples compared to the other two models. This is favoured in the field of NAD, as a high detection rate lowers the possibility of network invasion.

The selection of a binary model purely based on metrics would be the RF-Basic. It has the highest scores on the Basic dataset. The EBM-Basic follows closely. Depending on preferences, such as a high detection rate, the AE-Normal is a suitable option. Although the percentage of false positives is higher compared to the RF and EBM, it shows sensitivity for malicious samples, which could be a desirable property.

For multiclass classification, different models showed equal performance on the average performance metrics but lacked individual class performances. RFs trained on a balanced dataset showed equal performances on the average metrics as on the individual classes. The RF-SMOTE had the highest individual class scores. Much of the training was restricted due to hardware limitations. Training a model on a larger dataset requires more computational expenses than training on a smaller dataset. This makes it difficult to compare the RF-SMOTE to other models on the same dataset, as training them was not feasible. However, the RF-Minority showed performances comparable to those of the RF-SMOTE on individual classes, despite the Minority set having a smaller size than the SMOTE set.

The RF-Minority showed difficulty in the separation of two classes, despite the fact that these classes had more samples compared to other classes. The confusion matrix showed that these classes were mostly predicted as each other, not as other classes. Although dataset size does not contribute to model performance, the amount of variation captured by the train set does [3]. It is possible that there are too few samples of the two classes to capture the distribution of the two classes completely and that the model needs more samples to learn the differences that determine the separation between these classes. The RF-SMOTE did not show this behaviour. This strengthens the idea that the model needs more samples to distinguish between some individual classes.

Although there are no other comparisons of models to support this claim, both the RF-Minority and RF-SMOTE show that a balanced dataset is important for the multiclass classification. The RF-Basic and EBM-Basic showed poor performances for multiclass classification. The size of the dataset has to be tuned for the model to be able to generalise all individual classes. This could indicate that not every class requires an equal number of samples in the train set.

An unexpected result is the poor performance on individual classes of the AE+RF combination. Although the RF was trained on a balanced dataset, it was not able to correctly predict the minority classes. It was expected that this combination of models showed improved performance over models trained on unbalanced datasets, but this was not the case. The results on this model could strengthen the idea that the Minority set does not capture the distribution of all classes correctly and need more samples. This behaviour was expected for two individual classes, but more classes were predicted incorrectly. The setup did not work as expected, but as it was a proof of concept, no effort was made to optimise it. Optimisation of the setup could improve the performance, but the distribution of the Minority set appears to play a role in performance. Previous research has shown that a similar setup can work for NAD [9], thus optimisation could improve performance.

Given model performances only, the RF-SMOTE would be selected as the multiclass model. It has the highest performance scores on average and on individual classes. A downside of RF-SMOTE, and a reason to choose the RF-Minority, is the size of the SMOTE dataset. This increases train time. The RF-Minority could be chosen, although the missclassifications for two individual classes have to be fixed.

6.2 Explanation Performance

The explanations were evaluated on sensitivity and fidelity. A low sensitivity and a high fidelity are optimal. For the binary classification, RF-Basic explanations had the lowest sensitivity scores, while the AE-normal explanation showed the highest score. The highest fidelity scores were reached by the RF-Latent+SHAP explanations and EBM-Latent explanations. This was followed by the RF-Latent+LIME and the EBM-Basic. The RF-Basic+LIME showed the lowest fidelity performance. Although RF-Latent+SHAP and EBM-Latent showed high fidelity scores, the explanations do not provide information about the direct importance of the original features. The explanations show which latent dimension is important for the decision, but they do not show which features are mapped to that latent dimension. This requires an additional indirect method to explain the latent dimensions.

For the multiclass explanations, the RF-Minority+SHAP had the highest fidelity values and the second lowest sensitivity values. While the fidelity values of RF-SMOTE+LIME and RF-Minority+LIME are the lowest of the multiclass explanations, their sensitivity values are comparable to the other binary and multiclass explanations. The explanations of the RF-Minority were the only ones that confirm that SHAP is less sensitive than LIME, while others did not confirm earlier research [21]. The SHAP fidelity scores were generally higher than those of LIME for both binary and multiclass, contrary to earlier findings [8, 80].

Fidelity appears to be dependent on the dataset, confirming earlier research that explanation properties can depend on datasets [8, 23]. While the binary RF-Latent+LIME had the second highest fidelity scores, the binary

RF-Basic + LIME had the lowest fidelity scores. A similar pattern is found between the EBM-Basic and the EBM-Latent. The differences between the Latent and the Basic set are the number and meaning of features. The Latent set has 8 features obtained from the latent dimensions of the AE, while the Basic set has 62 original features. Given the difference in the number of features, it is possible that perturbing 1 feature out of 8 influences the final prediction more than perturbing 1 out of 62 features. A reason for this could be a prediction's reliance on each feature. If a model has to infer a prediction based on 8 features, perturbing one feature could have a larger impact on the prediction compared to a prediction given 62 features. A feature with a larger impact on the prediction can overturn the prediction faster compared to a feature with a low impact on the prediction. The fidelity score measures changes in prediction based on the perturbation of the most important features. The fidelity score appears to increase if the dataset contains more features have an impact on the model's prediction.

Given the influence of the meaning of features, the latent dimensions capture information about an unknown number of input features. It is possible that they capture combined feature information from features that are not individually important. If a latent dimension is perturbed, the prediction could be overturned faster compared to the perturbation of one individual feature. This a possible explanation for the increase in fidelity score of the Latent models compared to the Basic models.

A similar difference in fidelity scores can be found for the multiclass SHAP explanations of RF-SMOTE and RF-Minority. Both models were trained on different datasets and showed differences in fidelity scores. However, a similar explanation as for the Basic and Latent sets does not hold, as the SMOTE and Minority have the same number of features. A possible explanation could be the size of both sets. The Minority set has fewer samples compared to the SMOTE set. It is able to classify most individual classes correctly, but the low number of samples might affect the strength of the decision boundaries. As fidelity is based on changed predictions after perturbing the most important features, it is possible that a decision is overturned faster for the Minority set. RF-Minority has seen fewer samples to be able to generalise the classes and is not able to capture the complete distribution correctly [3]. However, there are no other multiclass models trained on the SMOTE and Minority set to strengthen this explanation of fidelity differences for the same explanation method.

The sensitivity scores showed less dependence on the dataset. This is in line with earlier findings, that certain properties of explanation methods can differ less when applied on models with different datasets [8, 22]. The heightened sensitivity score of the AE could be a result of the Normal set or the implementation of the explanation. As there is no universal method for explaining the AE, the current method was created to show that the importance of features to be comparable to the other methods. This study investigates the objective properties of the explanations and does not aim to create the optimal explanation method for AE. The explanation method is not optimised to adhere to certain properties, unlike LIME and SHAP.

As a final note on sensitivity and fidelity, it is possible that both properties request different behaviour from the explanation and underlying model. Previous research has shown that scores on explanation properties can differ when evaluated on different models trained on different datasets [8]. Sensitivity requires the explanation to be robust against small changes in feature values, while fidelity tests how the model prediction changes in response to perturbation in the most important features. Sensitivity measures the changes in the feature importances of the explanation, but if the model relies heavily on a different feature after a small perturbation in another feature, the explanation might change. For fidelity, the model has to respond to changes in the important features, while sensitivity requires the explanation (and model) to not respond to changes perturbations [8, 82]. In this study, different perturbation levels for fidelity were included to investigate this. The expectation was that perturbation levels would produce higher fidelity scores as they are more significant [82]. All binary explanations showed an increase in fidelity scores, with differences between model with explanation combinations. However, for multiclass explanations, this was not the case. The increase in fidelity score was lower or absent. Multiclass classifiers must predict 9 or 10 different classes. As more outcomes are possible compared to the binary case, there could be a lower threshold for a change in a feature value that overturns the prediction. It was expected that a higher perturbation strength would change the prediction more often compared to smaller perturbations. A possible explanation for the absence of increase in fidelity scores for the multiclass explanations could be that a lower perturbation strength already yields a high fidelity score. As a result, increasing the perturbation strength does not affect the fidelity as much as for the binary class explanations. However, considering this explanation, the following expectation would be a higher value for fidelity with perturbation strength 1 if a binary and multiclass model's explanation are compared. This expectation cannot be answered, as there are no models that perform well on the same dataset for binary and multiclass classification.

Both sensitivity and fidelity appear to request different behaviour from the underlying model and to show data-dependency. This is in line with earlier research, that shows properties of an explanation method are model and dataset-dependent [8].

The explanation for binary models with the best performance with respect to objective explanation metrics is the EBM-Latent. However, similarly to RF-Latent, the explanations are not directly interpretable. Given the performance metrics and explanability, EBM-Basic is the best explanation. Although it does not have the lowest absolute sensitivity score, it does have higher fidelity scores compared to the other explanation methods. The final choice of an explanation method can depend on the preferences placed on one of the two metrics, but the EBM-Basic is the option if both sensitivity and fidelity are weighed equally.

For multiclass model explanations, the RF-Minority+SHAP has the best performance. It has the second-lowest sensitivity scores but the highest fidelity scores. Taking into account the behaviour of the classifier, the RF-SMOTE+SHAP might be a better option, as it is able to correctly predict all classes.

6.3 Model and Explanation Combined

Following the results of model and explanation performances on the binary classification task, there is not one model with one explanation method that outperforms all others.

Of the binary models, the RF-Basic outperformed the other two models on metrics, but the performance of the other models was close. Considering aspects such as dataset format, hyperparameter tuning or detection preferences, one could opt for a different model. If a model is preferred that is close in performance but inherently explainable, the EBM-Basic is an option. If a high detection rate for malicious samples is important, the AE-Normal can be suitable.

The chosen objective properties were sensitivity and fidelity. Depending on the purpose of the explanation, more weight can be placed on the importance of one property. If sensitivity and fidelity are weighed equally, concessions have to be made as there is not one explanation method that had the highest scores on both.

If sensitivity is weighed more heavily over fidelity, one could consider the explanations of binary models with LIME. These showed the lowest sensitivity scores compared to other explanations for all binary explanations, contrary to the literature.

If fidelity is preferred, the EBM-Basic explanations can be chosen. These explanations showed the highest fidelity scores on the binary task. Additionally, if both sensitivity and fidelity are weighed equally, the EBM-Basic can be chosen as well. Although the RF-Latent and EBM-Latent explanations showed a balance between low sensitivity and high fidelity with good model performance, the explanations are not useful without further investigating the feature mapping to the latent dimensions. Contrary to the RF-Latent and EBM-Latent explanations, EBM-Basic does show the feature importance of the input features for binary classification. After all the Latent explanations, the EBM-Basic has the highest scores on fidelity. The sensitivity values are comparable to those of LIME and SHAP.

Considering model performance together with both objective properties, a suggestion is the EBM-Basic. This model has the second highest score on model performance. As discussed above, the EBM-Basic is a suitable option if both sensitivity and fidelity scores are equally important. Although there is no model with optimal scores on all metrics, the EBM-Basic balances model and explanation performance for the binary classification task.

Taking into account the model performance for the multiclass classification task, the RF-SMOTE outperforms the other models in terms of average performance and individual class performance. The SMOTE dataset brings considerations, such as a large dataset size that increases train time. The performance of other models on the Basic set showed the inability to predict minority classes correctly, indicating the need for a balanced dataset. The RF-Minority was trained on fewer samples and showed performance comparable to that of RF-SMOTE on most individual classes. The ideal model for multiclass classification is trained on a dataset size smaller than SMOTE, but larger than Minority.

Again, for multiclass explanations, sensitivity or fidelity can be preferred for a final chosen explanation. For the sensitivity values, the RF-SMOTE LIME or RF-Minority SHAP can be chosen as explanations. Both have comparable low values for sensitivity. For the fidelity scores, the RF-Minority SHAP showed the highest values. No other explanation for the multiclass classifiers scores similar fidelity values.

If sensitivity and fidelity are weighed equally, the RF-Minority SHAP explanations showed a balance between these scores. It had the second-lowest sensitivity scores and the highest fidelity scores.

Considering model performance together with explanation metrics, there is no optimal model for the multiclass case. Although the RF-Minority model showed high performance on most individual classes and good performance on the objective properties, it was unable to perform well on two individual classes. This makes the performance of the model questionable. The RF-SMOTE scored high on all model performance metrics for individual classes as the only multiclass model. However, the objective explanation properties were considerably lower than those of the RF-Minority. If model performance and objective explanation metrics are weighed equally and hardware is not an issue, the RF-SMOTE+SHAP can be the final decision. It has higher fidelity metrics and a low sensitivity metric compared to the RF-SMOTE+LIME.

To summarise, the final combination of model and explanation depends on the weights attached to model performance and the objective explanation metrics. If model performance is preferred over high scores on objective metrics, it would result in a different choice of model compared to preference of objective metrics equal to or over high performance.

From the results of the binary and multiclass explanations, the quality metrics of the explanation appear to depend on the dataset [22] and the possible reliance on underlying model behaviour. This implies that a different model or a changed dataset format can be chosen to optimise the properties of the explanation method. Given the suggestions for model and explanation combinations, there is room for improvement through the decision of the model structure or the dataset formats, aside from the explanation methods themselves. As the quality metrics are measured for the explanations, but they depend on the underlying model behaviour and dataset, improvement of the quality metrics could require adjustments in the model or the dataset. The objective properties of explanations could guide towards a different decision for a final model or a different dataset format. Thereby, it is important that the objective properties of interest are chosen to match the purpose of the explanation, as it could steer decisions for explanations, models, and datasets in a different direction if other properties were chosen.

6.4 Limitations

The following section contains the limitations of this study. Four limitations are discussed.

The first limitation is the dataset. There are two smaller limitations related to this: the format of the dataset and the lack of temporal information. First, although the dataset itself contains up-to-date and valuable information about network traffic, the format of the dataset could have influenced the model performance. The dataset is available online in tabular format and this can affect model performance. Previous research often showed the effectiveness of the AE, but neural networks are outperformed by other models on tabular data [27]. Secondly, the dataset does contain temporal information about network traffic, but is not available as a time series dataset. In this way, it is difficult for models to incorporate information about subsequent events, aside from the time-based features or the manual crafted features that capture information over multiple connections. Temporal information from the dataset could not be fully incorporated in the classification task.

A second limitation was the selection of models and explanation methods. Three models and two model-agnostic explanation methods were selected for this research. Although this study tried to optimise each model for the classification task, it is possible that there is another model that is able to outperform the selected models. This is similar for the explanation methods. Both LIME and SHAP are feature attribution methods, and the EBM displays feature attributions as an explanation. However, other explanations might be more suitable for the field of NAD.

A third limitation concerns the hardware restrictions. The process of training the models or tuning the hyperparameters was limited by the available hardware. Temporary access was granted on a high performance cluster, but this was limited to several hours. Training the AE and the EBM took more time compared to the RF and most of this training was done on the cluster. The remaining versions and all versions of the RF were trained locally. Training times differed per model and per hyperparameter combination, but training one AE for binary classification took approximately 20 times longer than training one RF. Not only did the hyperparameters of the models strain the training time, but the size of the dataset did as well. The SMOTE dataset for multiclass classification contained more than 22,000,000 samples. The size of this dataset increased the training time for all models, which resulted in difficulties of tuning the hyperparameters of the RF and training the EBM at all.

A fourth limitation is the fact that the AE did not work for the multiclass implementation. Two attempts were taken. The first attempt took the binary AE implementation and adjusted it for multiclass classification, but it did not show promising results. The other attempt combined the AE+RF. The results did not show performances comparable to RF-SMOTE. Due to time restrictions, no other AE was implemented for multiclass classification. The other models were compared on the multiclass case, but this was not possible for the AE.

6.5 Future work

There are not many studies in the field of XAI for NAD that include objective evaluations. This study contributes to incorporating objective properties in evaluations of explanations for NAD, but there is room to expand on these results.

As a beginning, future work could incorporate more models and explanation methods to compare. There might be a model or an explanation method that is a better fit than the current models and explanations. To support optimal performance of the dataset, adjusting the dataset per model would be beneficial. A larger pool of comparisons could provide more insight into the best-suited explanations for the task of NAD.

Aside from incorporating more models or explanation methods, an idea for future work is to expand the objective properties. This thesis focuses sensitivity and fidelity, as these were deemed important for the general task of NAD. In future work, one could evaluate more specifically which objective properties or combinations of properties are valuable in the field of NAD.

Another direction could be to look at results from articles that claim to provide explainability in the field of NAD but do not provide objective or subjective measures. Articles provide new models or new explanation methods for NAD that can be evaluated with the approach of this thesis. This could provide new insights in the quality of the proposed approaches and possible improvements.

Lastly, this study focused on the objective properties of explanations. The idea behind this is that the quality of explanations can be measured objectively before explanations are presented to potential targets. A step further could be the inclusion of a user study to test the results found.

Chapter 7

Conclusion

The gap identified in this thesis was the minimum number of studies that used the objective properties of explanations to examine explanations in the field of NAD. Although research has been done to include XAI for NAD, it lacks an objective evaluation of the proposed methods. The current research attempted to contribute to the objective evaluation of explanations in the field of NAD by evaluating several explanation methods through two objective properties. The main question was: *How can network anomaly detection be improved by including objective properties of explanations as quality indicators?*. To answer this question, explanations for trained models were evaluated using selected objective properties.

As a starting point, three models were selected that all needed different explanation methods. The three models had different performance on the datasets. Evaluation of model performance showed that the performances between different models were close. A selection for models could be made purely based on performance metrics, but there are other aspects to consider when selecting the optimal model, such as sensitivity to anomalies and the size of the dataset. The explanation methods used to extract explanations from the models all provided feature importances. Through sensitivity and fidelity, the explanations were evaluated. Not one explanation had the lowest sensitivity and highest fidelity scores, thus a decision on the explanation method depends on preferences or requirements placed on the explanation. Combining the performance of the model and the explanation method, there is no model for the binary or multiclass classification task with an explanation that has both the highest model performance and the optimal objective metrics.

One of the main findings of this study is that the final decision on the usage of an explanation method depends on preferences for model performance and objective properties. As not one explanation method showed optimal scores on the quality metrics, compromises must be made given the importance placed on the objective properties.

A second finding from this study is that the dataset appears to influence the objective properties of the explanation. Additionally, objective properties can request different behaviour of the underlying model. Both of these can affect the results on the objective properties.

Combining these findings, objective properties can contribute to the quality of the explanation because they allow for the investigation of the optimal model and explanation combination. The objective property can help provide insights into the differences between the explanations of different models trained on different datasets. A combination of a model and an explanation can be chosen based on the quality properties.

Appendix A

Model Hyperparameter Tuning

A.1 Binary classification

A.1.1 RF: Basic set

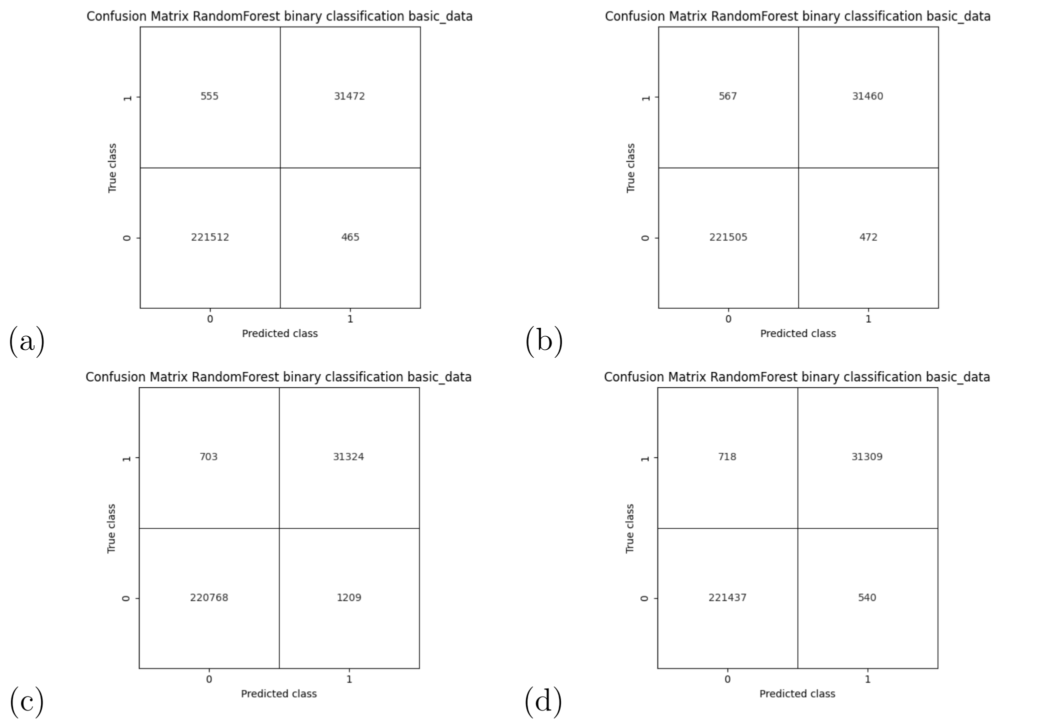


Fig. A.1. Confusion matrices of binary RF on the Basic set. (a) Version 1 (b) Version 2 (c) Version 3 (d) Version 4

A.1.2 EBM: Basic set

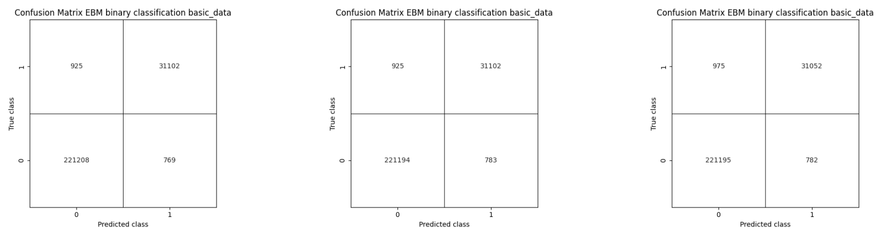


Fig. A.2. Confusion matrices of binary EBM on the Basic set. **Left**: version 1. **Middle**: version 2. **Right**: version 3.

A.1.3 EBM: Latent set

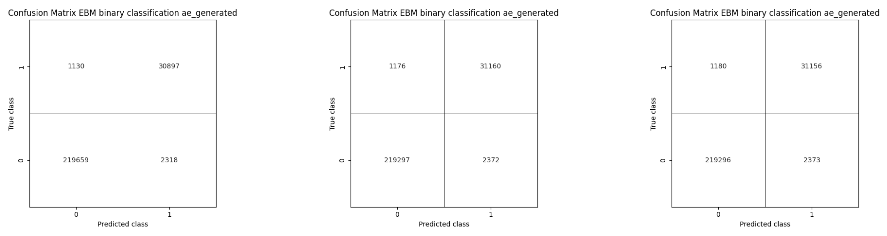


Fig. A.3. Confusion matrices of binary EBM on the Latent set. **Left**: version 1. **Middle**: version 2. **Right**: version 3.

A.2 Multiclass classification

A.2.1 RF: Basic set

Class	Precision	Recall	F-score	Support
analysis	0.1116	0.1148	0.1131	244
backdoor	0.1092	0.1029	0.1059	243
dos	0.3510	0.3057	0.3268	1665
exploits	0.7066	0.7547	0.7299	4477
fuzzers	0.7427	0.6844	0.7124	2468
generic	0.9926	0.9831	0.9878	21364
normal	0.9968	0.9985	0.9976	221977
recoinnaissance	0.7929	0.7758	0.7843	1387
shellcode	0.6500	0.5723	0.6087	159
worms	0.2000	0.0500	0.0800	20
accuracy				0.9820
macro avg	0.5653	0.5342	0.5447	254004
avg	0.9815	0.9820	0.9817	254004

Tab. A.1. Classification report RF Basic multiclass version 1

Class	Precision	Recall	F-score	Support
analysis	0.6000	0.0615	0.1115	244
backdoor	0.5312	0.0700	0.1236	243
dos	0.3559	0.2276	0.2777	1665
exploits	0.6251	0.8381	0.7161	4477
fuzzers	0.7516	0.5859	0.6585	2468
generic	0.9980	0.9806	0.9893	21364
normal	0.9954	0.9985	0.9970	221977
recoinnaissance	0.9149	0.7678	0.8350	1387
shellcode	0.6565	0.5409	0.5931	159
worms	0.2500	0.0500	0.0833	20
accuracy				0.9817
macro avg	0.6679	0.5121	0.5385	254004
avg	0.9810	0.9817	0.9805	254004

Tab. A.2. Classification report RF Basic multiclass version 2

Class	Precision	Recall	F-score	Support
analysis	0.5000	0.0041	0.0081	244.0
backdoor	0.0	0.0	0.0	243.0
dos	0.7241	0.0126	0.0248	1665
exploits	0.5586	0.9232	0.6960	4477
fuzzers	0.7110	0.3339	0.4544	2468
generic	0.9998	0.9733	0.9863	21364
normal	0.9912	0.9987	0.9950	221977
recoinnaissance	0.9353	0.6467	0.7647	1387
shellcode	1.0	0.0189	0.0370	159
worms	0.0	0.0	0.0	20
accuracy				0.9778
macro avg	0.6420	0.3911	0.3966	254004
avg	0.9781	0.9778	0.9735	254004

Tab. A.3. Classification report RF Basic multiclass version 3

Class	Precision	Recall	F-score	Support
analysis	0.0	0.0	0.0	244
backdoor	0.0	0.0	0.0	243
dos	0.6250	0.0120	0.0236	1665
exploits	0.5551	0.9249	0.6938	4477
fuzzers	0.7340	0.3355	0.4605	2468
generic	0.9999	0.9730	0.9863	21364
normal	0.9913	0.9986	0.9949	221977
recoinnaissance	0.9010	0.6431	0.7505	1387
shellcode	1.0	0.0126	0.0248	159
worms	0.0	0.0	0.0	20
accuracy				0.9776
macro avg	0.5806	0.3900	0.3934	254004
avg	0.9770	0.9776	0.9734	254004.0

Tab. A.4. Classification report RF Basic multiclass version 4

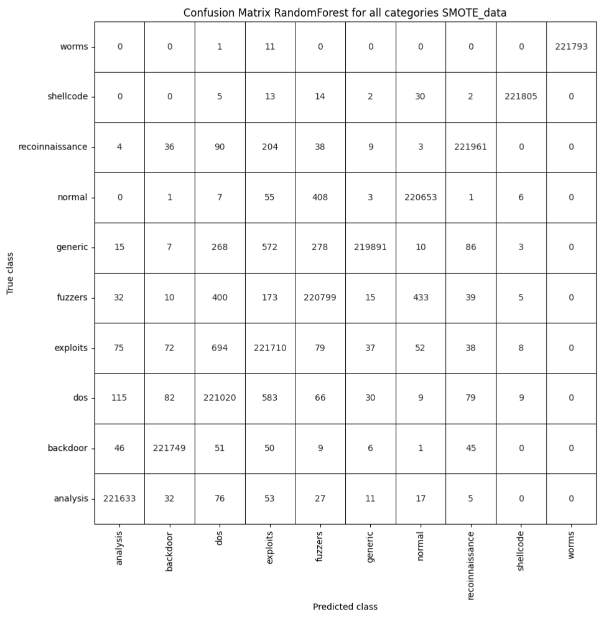
A.2.2 RF: SMOTE set

Class	Precision	Recall	F-score	Support
analysis	0.9987	0.9990	0.9989	221854
backdoor	0.9989	0.9991	0.9990	221957
dos	0.9928	0.9956	0.9942	221993
exploits	0.9923	0.9953	0.9938	222765
fuzzers	0.9959	0.9950	0.9954	221906
generic	0.9995	0.9944	0.9969	221130
normal	0.9975	0.9978	0.9977	221134
recoinnaissance	0.9987	0.9983	0.9985	222345
shellcode	1.0	0.9999	0.9998	221871
worms	1.0	0.9999	1.0	221805
accuracy				0.9974
macro avg	0.9974	0.9974	0.9974	2218760
avg	0.9974	0.9974	0.9974	2218760

Tab. A.5. Classification report RF SMOTE multiclass version 1

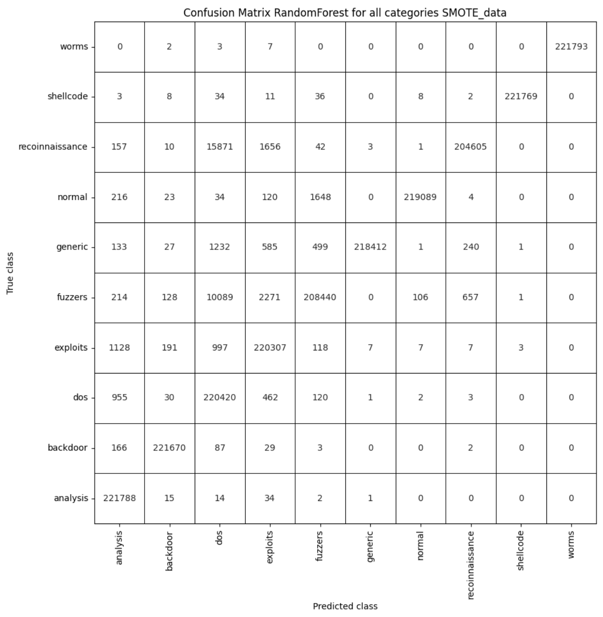
Class	Precision	Recall	F-score	Support
analysis	0.9868	0.9997	0.9932	221854
backdoor	0.998	0.9987	0.9984	221957
dos	0.8860	0.9929	0.9364	221993
exploits	0.9770	0.9890	0.9830	222765
fuzzers	0.9883	0.9393	0.9632	221906
generic	0.9999	0.9877	0.9938	221130
normal	0.9994	0.9908	0.9951	221134
recoinnaissance	0.9955	0.9202	0.9564	222345
shellcode	1.0	0.9995	0.9998	221871
worms	1.0	0.9999	1.0	221805
accuracy				0.9818
macro avg	0.9831	0.9818	0.9819	2218760
avg	0.9831	0.9818	0.9819	2218760

Tab. A.6. Classification report RF SMOTE multiclass version 2


Confusion Matrix RandomForest for all categories SMOTE_data

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	recoinnaissance	shellcode	worms
worms	0	0	1	11	0	0	0	0	0	221793
shellcode	0	0	5	13	14	2	30	2	221805	0
recoinnaissance	4	36	90	204	38	9	3	221961	0	0
normal	0	1	7	55	408	3	220653	1	6	0
generic	15	7	268	572	278	219891	10	86	3	0
fuzzers	32	10	400	173	220799	15	433	39	5	0
exploits	75	72	694	221710	79	37	52	38	8	0
dos	115	82	221020	583	66	30	9	79	9	0
backdoor	46	221749	51	50	9	6	1	45	0	0
analysis	221633	32	76	53	27	11	17	5	0	0

Fig. A.4. Confusion matrix for version 1 of RF trained on SMOTE for multiclass


Confusion Matrix RandomForest for all categories SMOTE_data

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	recoinnaissance	shellcode	worms
worms	0	2	3	7	0	0	0	0	0	221793
shellcode	3	8	34	11	36	0	8	2	221769	0
recoinnaissance	157	10	15871	1656	42	3	1	204605	0	0
normal	216	23	34	120	1648	0	219089	4	0	0
generic	133	27	1232	585	499	218412	1	240	1	0
fuzzers	214	128	10089	2271	208440	0	106	657	1	0
exploits	1128	191	997	220307	118	7	7	7	3	0
dos	955	30	220420	462	120	1	2	3	0	0
backdoor	166	221670	87	29	3	0	0	2	0	0
analysis	221788	15	14	34	2	1	0	0	0	0

Fig. A.5. Confusion matrix for version 2 of RF trained on SMOTE for multiclass

A.2.3 RF: Latent-Basic

Class	Precision	Recall	F-score	Support
analysis	0.7500	0.0369	0.0703	244
backdoor	0.1818	0.0082	0.0157	243
dos	0.2456	0.0252	0.0458	1665
exploits	0.5115	0.6918	0.5881	4477
fuzzers	0.3242	0.2164	0.2595	2468
generic	0.9649	0.9552	0.9600	21364
normal	0.9852	0.9938	0.9894	221977
recoinnaissance	0.3392	0.2516	0.2889	1387
shellcode	0.2500	0.0063	0.0123	159
worms	0.2500	0.0500	0.0833	20
accuracy				0.9647
macro avg	0.4802	0.3235	0.3313	254004
avg	0.9588	0.9647	0.9603	254004

Tab. A.7. Classification report RF Latent-Basic multiclass version 1

Class	Precision	Recall	F-score	Support
analysis	0.5500	0.0451	0.0833	244
backdoor	0.1667	0.0041	0.0080	243
dos	0.2393	0.0234	0.0427	1665
exploits	0.5120	0.6918	0.5884	4477
fuzzers	0.3342	0.2208	0.2659	2468
generic	0.9651	0.9562	0.9606	21364
normal	0.9848	0.9937	0.9893	221977
recoinnaissance	0.3531	0.2495	0.2924	1387
shellcode	0.3000	0.0189	0.0355	159
worms	0.2500	0.0500	0.0833	20
accuracy				0.9648
macro avg	0.4655	0.3253	0.3349	254004
avg	0.9585	0.9648	0.9603	254004

Tab. A.8. Classification report RF Latent-Basic multiclass version 2

A.2.4 RF: Minority

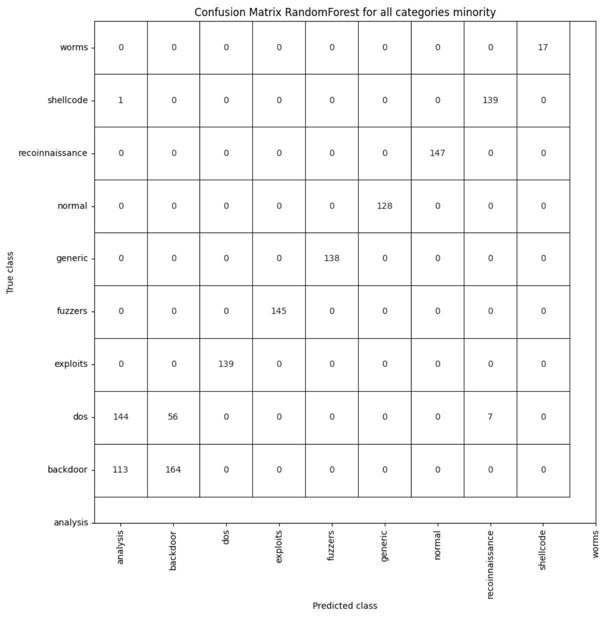


Fig. A.6. Confusion matrix for RF-Minority multiclass

A.2.5 EBM: Basic

Class	Precision	Recall	F-score	Support
analysis	0.0	0.0	0.0	244
backdoor	0.0	0.0	0.0	243
dos	0.4177	0.0823	0.1375	1665
exploits	0.6919	0.4764	0.5643	4477
fuzzers	0.5204	0.0567	0.1023	2468
generic	0.9435	0.9755	0.9592	21364
normal	0.9730	0.9991	0.9859	221977
recoinnaissance	0.9875	0.2286	0.3712	1387
shellcode	0.0	0.0	0.0	159
worms	0.0	0.0	0.0	20
accuracy				0.9659
macro avg	0.4534	0.2819	0.312	254004
avg	0.9551	0.9659	0.9561	254004

Tab. A.9. Classification report EBM Basic multiclass version 1

Class	Precision	Recall	F-score	Support
analysis	0.0	0.0	0.0	244
backdoor	0.0	0.0	0.0	243
dos	0.0	0.0	0.0	1665
exploits	0.6998	0.4956	0.5803	4477
fuzzers	0.5078	0.066	0.1169	2468
generic	0.9318	0.9761	0.9535	21364
normal	0.9736	0.999	0.9862	221977
recoinnaissance	0.9861	0.2559	0.4064	1387
shellcode	0.0	0.0	0.0	159
worms	0.0	0.0	0.0	20
accuracy				0.9659
macro avg	0.4099	0.2793	0.3043	254004
avg	0.9519	0.9659	0.9556	254004

Tab. A.10. Classification report EBM Basic multiclass version 2

Class	Precision	Recall	F-score	Support
analysis	0.0	0.0	0.0	244
backdoor	0.0	0.0	0.0	243
dos	0.0	0.0	0.0	1665
exploits	0.6998	0.4956	0.5803	4477
fuzzers	0.5078	0.066	0.1169	2468
generic	0.9318	0.9761	0.9535	21364
normal	0.9736	0.9990	0.9862	221977
recoinnaissance	0.9861	0.2559	0.4064	1387
shellcode	0.0	0.0	0.0	159
worms	0.0	0.0	0.0	20
accuracy				0.9659
macro avg	0.4099	0.2793	0.3043	254004
avg	0.9519	0.9659	0.9556	254004

Tab. A.11. Classification report EBM Basic multiclass version 3

Appendix B

Model performance results

B.1 Binary: RF-Basic

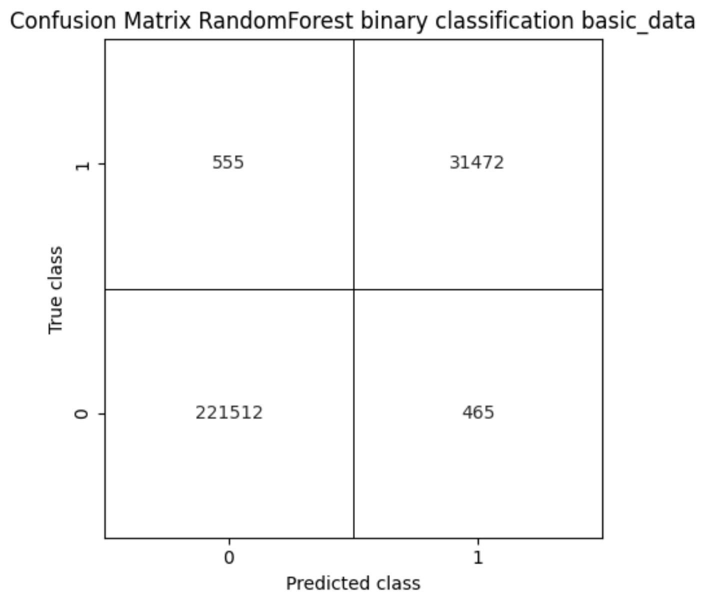


Fig. B.1. Confusion matrix for RF-Basic binary on the test set

B.2 Binary: EBM-Basic

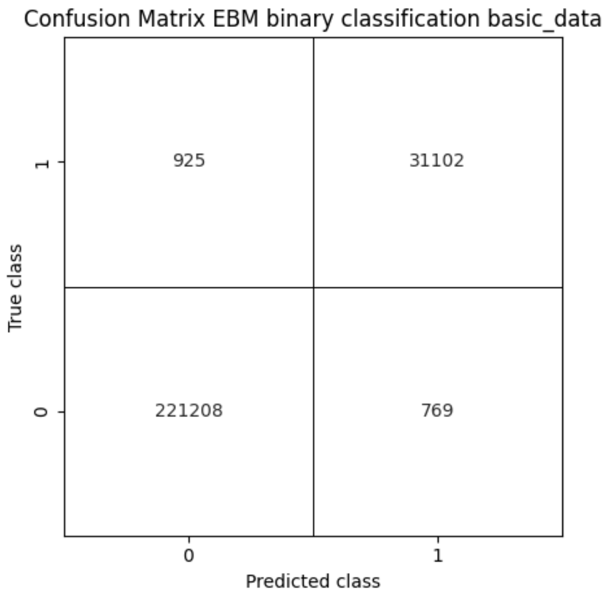


Fig. B.2. Confusion matrix for EBM-Basic binary on the test set

B.3 Binary: AE-Normal

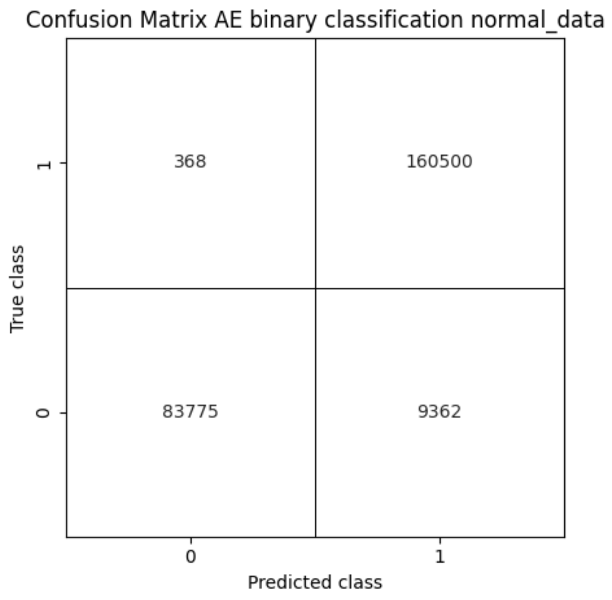


Fig. B.3. Confusion matrix for AE-Normal binary on the test set

B.4 Binary: RF-Latent

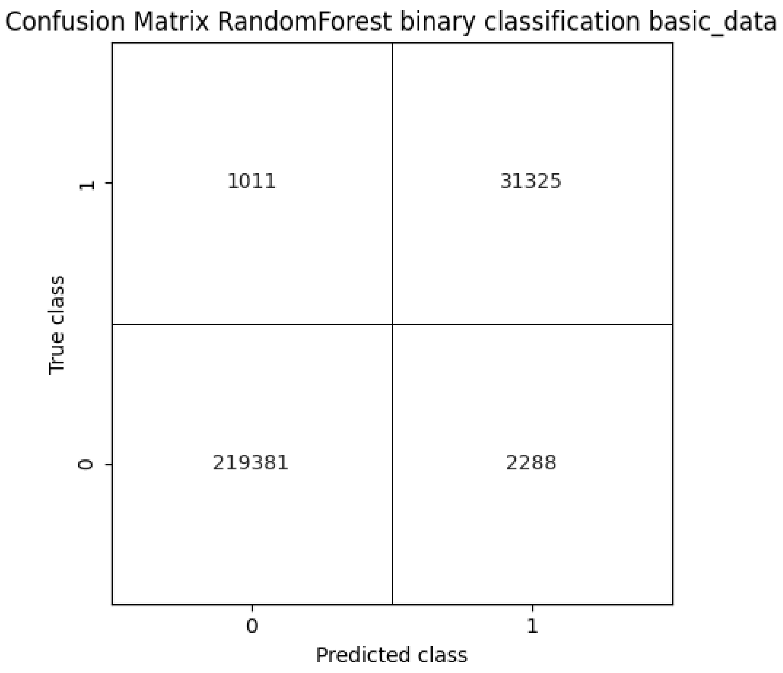


Fig. B.4. Confusion matrix for RF-Latent binary on the test set. Note: the title in this graph is wrong. It should be *ae_generated* or *latent*

B.5 Binary: EBM-Latent

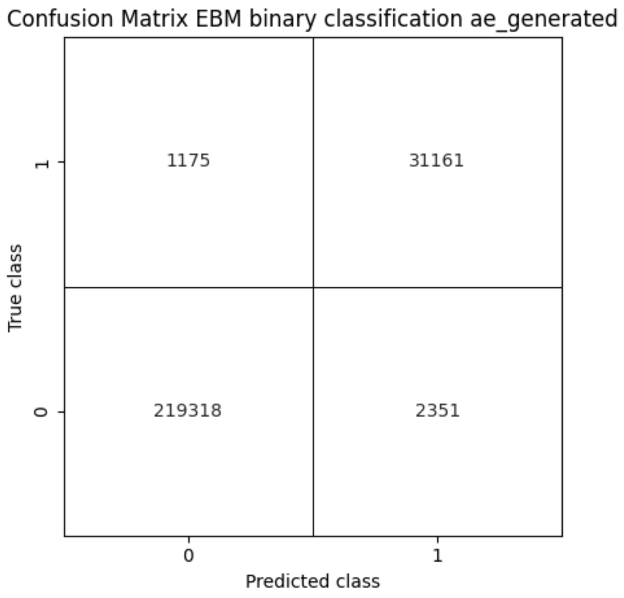
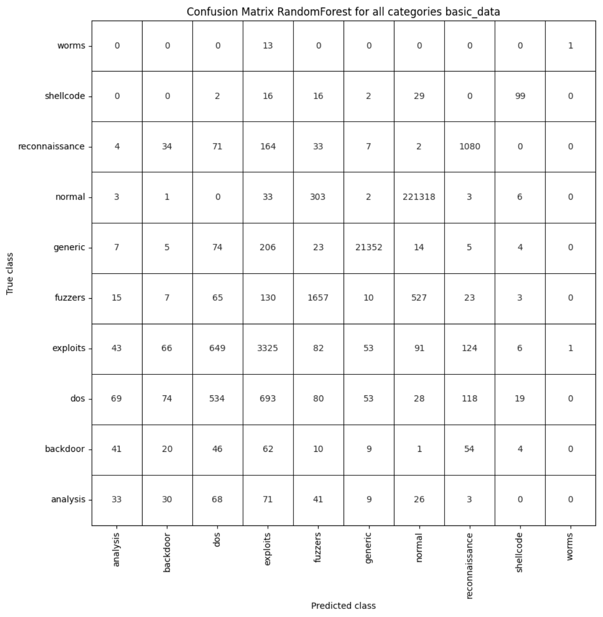


Fig. B.5. Confusion matrix for EBM-Latent binary on the test set

B.6 Multiclass: RF-Basic


Confusion Matrix RandomForest for all categories basic_data

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	reconnaissance	shellcode	worms
worms	0	0	0	13	0	0	0	0	0	1
shellcode	0	0	2	16	16	2	29	0	99	0
reconnaissance	4	34	71	164	33	7	2	1080	0	0
normal	3	1	0	33	303	2	221318	3	6	0
generic	7	5	74	206	23	21352	14	5	4	0
fuzzers	15	7	65	130	1657	10	527	23	3	0
exploits	43	66	649	3325	82	53	91	124	6	1
dos	69	74	534	693	80	53	28	118	19	0
backdoor	41	20	46	62	10	9	1	54	4	0
analysis	33	30	68	71	41	9	26	3	0	0

Fig. B.6. Confusion matrix for RF-Basic multiclass on the test set

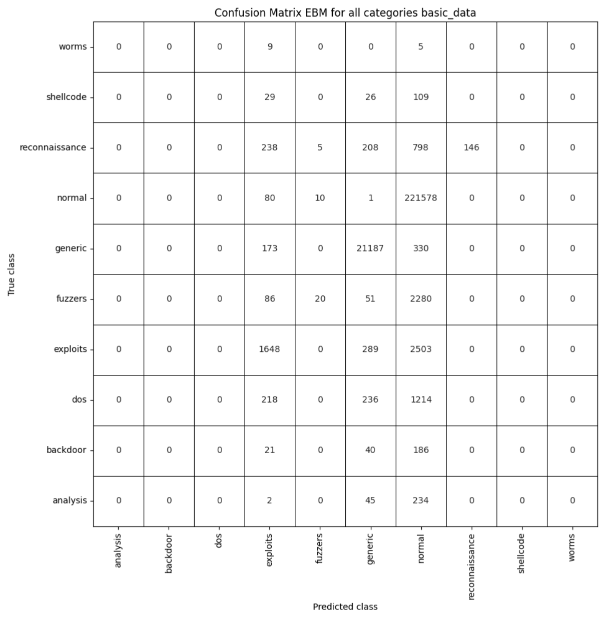
Class	Precision	Recall	F-score	Support
analysis	0.1535	0.1174	0.1331	281
backdoor	0.0844	0.0810	0.0826	247
dos	0.3539	0.3201	0.3362	1668
fuzzers	0.7055	0.7489	0.7265	4440
exploits	0.7381	0.6799	0.7078	2437
generic	0.9933	0.9844	0.9888	21690
normal	0.9968	0.9984	0.9976	221669
recoinnaissance	0.766	0.7742	0.7701	1395
shellcode	0.7021	0.6037	0.6492	164
worms	0.5000	0.0714	0.1250	14
accuracy				0.9819
macro avg	0.5993	0.5379	0.5517	254005
avg	0.9814	0.9819	0.9816	254005

Tab. B.2. Classification report RF-Basic multiclass on the test set

B.7 Multiclass: EBM-Basic

Class	Precision	Recall	F-score	Support
analysis	0.0	0.0	0.0	281
backdoor	0.0	0.0	0.0	247
dos	0.0	0.0	0.0	1668
exploits	0.6581	0.3712	0.4747	4440
fuzzers	0.5714	0.0082	0.0162	2437
generic	0.9594	0.9768	0.9680	21690
normal	0.9666	0.9996	0.9828	221669
recoinnaissance	1.0	0.1047	0.1895	1395
shellcode	0.0	0.0	0.0	164
worms	0.0	0.0	0.0	14
accuracy				0.9629
macro avg	0.4156	0.2460	0.2631	254005
avg	0.9479	0.9629	0.9499	254005

Tab. B.3. Classification report EBM-Basic multiclass on the test set


Confusion Matrix EBM for all categories basic_data

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	reconnaissance	shellcode	worms
worms	0	0	0	9	0	0	5	0	0	0
shellcode	0	0	0	29	0	26	109	0	0	0
reconnaissance	0	0	0	238	5	208	798	146	0	0
normal	0	0	0	80	10	1	221578	0	0	0
generic	0	0	0	173	0	21187	330	0	0	0
fuzzers	0	0	0	86	20	51	2280	0	0	0
exploits	0	0	0	1648	0	289	2503	0	0	0
dos	0	0	0	218	0	236	1214	0	0	0
backdoor	0	0	0	21	0	40	186	0	0	0
analysis	0	0	0	2	0	45	234	0	0	0

Fig. B.7. Confusion matrix for EBM-Basic multiclass on the test set

B.8 Multiclass: RF-Minority

Class	Precision	Recall	F-score	Support
analysis	0.4380	0.4079	0.4224	277
backdoor	0.2545	0.2705	0.2623	207
dos	1.0	1.0	1.0	139
exploits	1.0	1.0	1.0	145
fuzzers	1.0	1.0	1.0	138
generic	1.0	1.0	1.0	128
recoinnaissance	1.0	1.0	1.0	147
shellcode	0.9521	0.9929	0.9720	140
worms	1.0	1.0	1.0	17
accuracy				0.7638
macro avg	0.8494	0.8524	0.8508	1338
avg	0.7633	0.7638	0.7634	1338

Tab. B.4. Classification report RF-Minority multiclass on test data

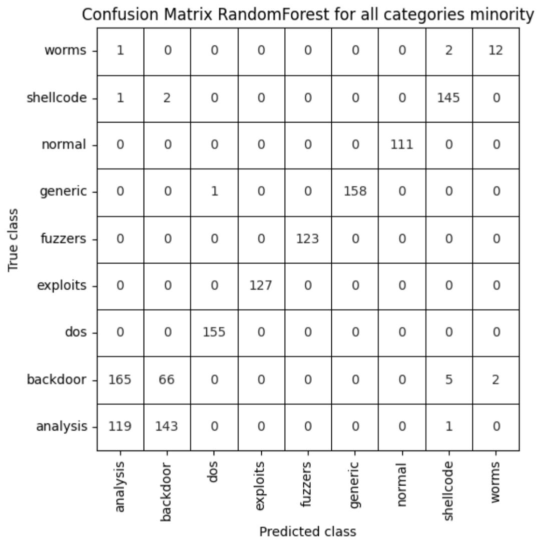
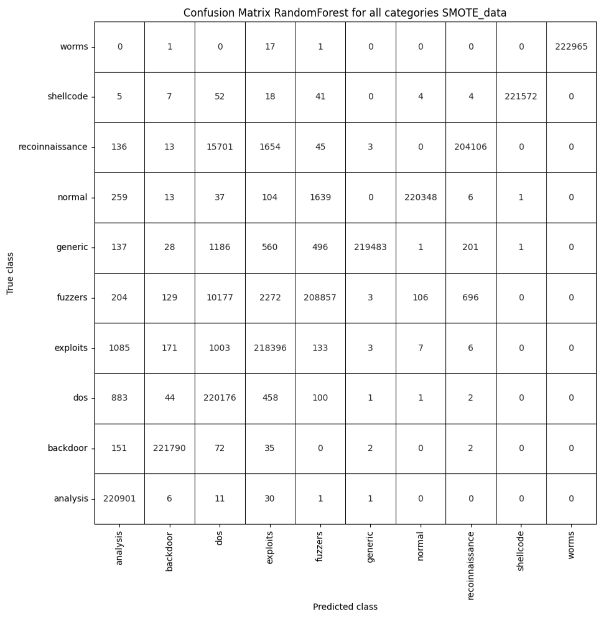


Fig. B.8. Confusion matrix RF-Minority multiclass on the test set

B.9 Multiclass: RF-SMOTE

Class	Precision	Recall	F-score	Support
analysis	0.9872	0.9998	0.9935	220950
backdoor	0.9981	0.9988	0.9985	222052
dos	0.8863	0.9933	0.9368	221665
exploits	0.9770	0.9891	0.9830	220804
fuzzers	0.9884	0.9389	0.9630	222444
generic	0.9999	0.9882	0.9941	222093
normal	0.9995	0.9907	0.9951	222407
recoinnaissance	0.9955	0.9208	0.9567	221658
shellcode	1.0	0.9994	0.9997	221703
worms	1.0	0.9999	1.0	222984
accuracy				0.9819
macro avg	0.9832	0.9819	0.982	2218760
avg	0.9832	0.9819	0.982	2218760

Tab. B.5. Classification report of RF SMOTE MC on the test set

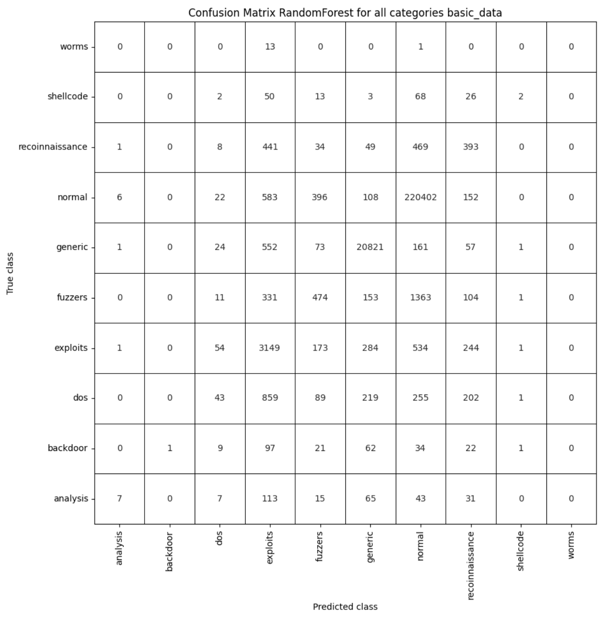


Confusion Matrix RandomForest for all categories SMOTE_data

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	recoinnaissance	shellcode	worms
worms	0	1	0	17	1	0	0	0	0	222965
shellcode	5	7	52	18	41	0	4	4	221572	0
recoinnaissance	136	13	15701	1654	45	3	0	204106	0	0
normal	259	13	37	104	1639	0	220348	6	1	0
generic	137	28	1186	560	496	219483	1	201	1	0
fuzzers	204	129	10177	2272	208857	3	106	696	0	0
exploits	1085	171	1003	218396	133	3	7	6	0	0
dos	883	44	220176	458	100	1	1	2	0	0
backdoor	151	221790	72	35	0	2	0	2	0	0
analysis	220901	6	11	30	1	1	0	0	0	0

Fig. B.9. Confusion matrix RF-SMOTE on the test set

B.10 Multiclass: RF-Latent


Confusion Matrix RandomForest for all categories basic_data

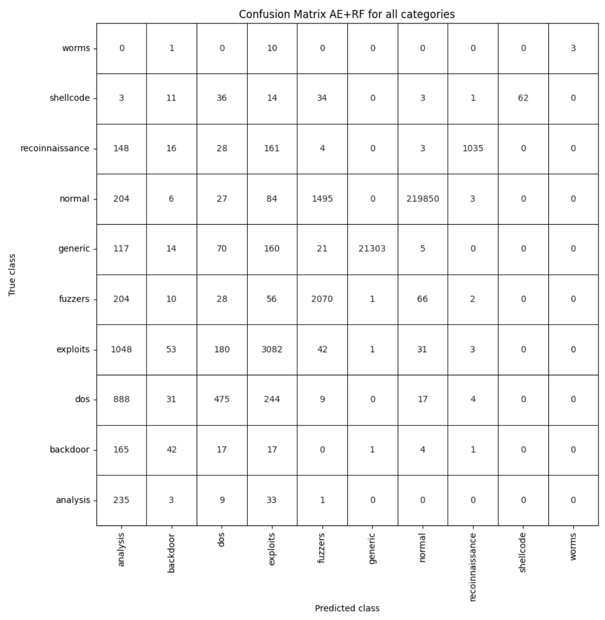
True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	recoinnaissance	shellcode	worms
worms	0	0	0	13	0	0	1	0	0	0
shellcode	0	0	2	50	13	3	68	26	2	0
recoinnaissance	1	0	8	441	34	49	469	393	0	0
normal	6	0	22	583	396	108	220402	152	0	0
generic	1	0	24	552	73	20821	161	57	1	0
fuzzers	0	0	11	331	474	153	1363	104	1	0
exploits	1	0	54	3149	173	284	534	244	1	0
dos	0	0	43	859	89	219	255	202	1	0
backdoor	0	1	9	97	21	62	34	22	1	0
analysis	7	0	7	113	15	65	43	31	0	0

Fig. B.10. Confusion matrix for RF-Latent multiclass on the test set

Class	Precision	Recall	F-score	Support
analysis	0.3684	0.0249	0.0467	281
backdoor	0.2500	0.0040	0.0080	247
dos	0.1833	0.0132	0.0246	1668
exploits	0.5055	0.7081	0.5899	4440
fuzzers	0.3577	0.1986	0.2554	2437
generic	0.9574	0.9606	0.9590	21690
normal	0.9871	0.9941	0.9906	221669
recoinnaissance	0.3065	0.2810	0.2932	1395
shellcode	0.2500	0.0122	0.0233	164
worms	0.0	0.0	0.0	14.0
accuracy				0.9655
macro avg	0.4166	0.3197	0.3191	254005
avg	0.9591	0.9655	0.961	254005

Tab. B.6. Classification report of RF-Latent multiclass on the test set

B.11 Multiclass: AE+RF


Confusion Matrix AE+RF for all categories
True class
Predicted class

True class \ Predicted class	analysis	backdoor	dos	exploits	fuzzers	generic	normal	recoinnaissance	shellcode	worms
worms	0	1	0	10	0	0	0	0	0	3
shellcode	3	11	36	14	34	0	3	1	62	0
recoinnaissance	148	16	28	161	4	0	3	1035	0	0
normal	204	6	27	84	1495	0	219850	3	0	0
generic	117	14	70	160	21	21303	5	0	0	0
fuzzers	204	10	28	56	2070	1	66	2	0	0
exploits	1048	53	180	3082	42	1	31	3	0	0
dos	888	31	475	244	9	0	17	4	0	0
backdoor	165	42	17	17	0	1	4	1	0	0
analysis	235	3	9	33	1	0	0	0	0	0

Fig. B.11. Confusion matrix for AE+RF multiclass on the test set

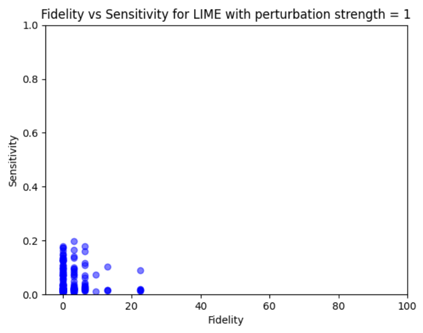
Class	Precision	Recall	F-score	Support
analysis	0.0780	0.8363	0.1427	281
backdoor	0.2246	0.1700	0.1935	247
dos	0.5460	0.2848	0.3743	1668
exploits	0.7982	0.6941	0.7426	4440
fuzzers	0.5631	0.8494	0.6772	2437
generic	0.9999	0.9822	0.9909	21690
normal	0.9994	0.9918	0.9956	221669
recoinnaissance	0.9867	0.7419	0.847	1395
shellcode	1.0	0.3780	0.5487	164
worms	1.0	0.2143	0.3529	14
accuracy				0.9770
macro avg	0.7196	0.6143	0.5865	254005
avg	0.9869	0.9770	0.9808	254005

Tab. B.7. Classification report of AE+RF on the test set

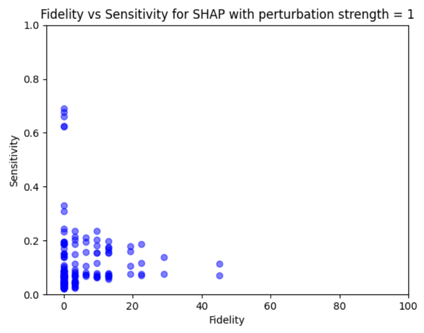
Appendix C

Explanation results

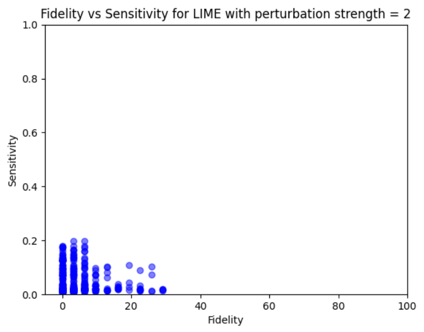
C.1 Binary: RF-Basic



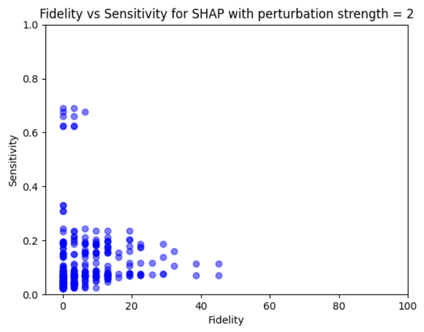
(a)



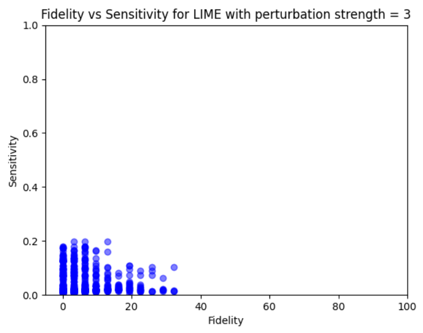
(b)



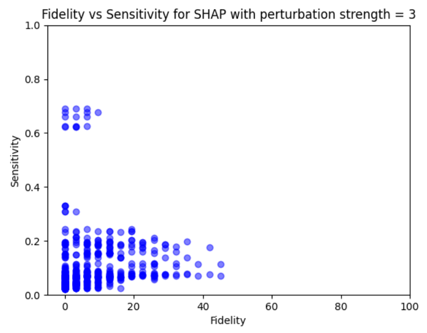
(c)



(d)



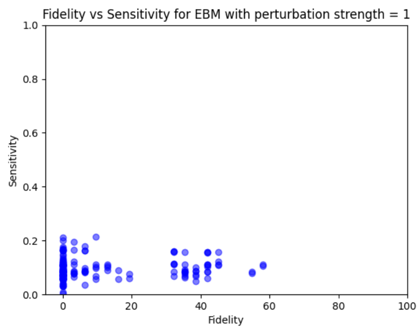
(e)



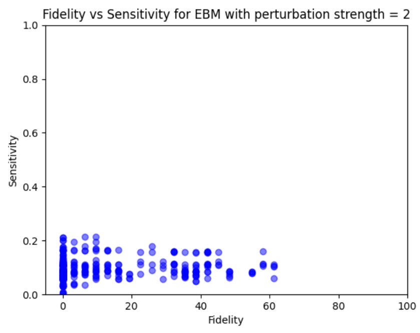
(f)

Fig. C.1. Sensitivity and fidelity with different perturbation strengths. (a) RF-Basic + LIME with strength 1 (b) RF-Basic + SHAP with strength 1 (c) RF-Basic + LIME with strength 2 (d) RF-Basic + SHAP with strength 2 (e) RF-Basic + LIME with strength 3 (f) RF-Basic + SHAP with strength 3

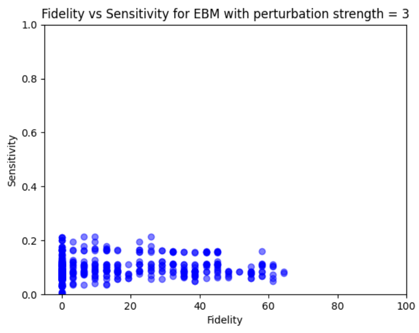
C.2 Binary: EBM-Basic



(a)



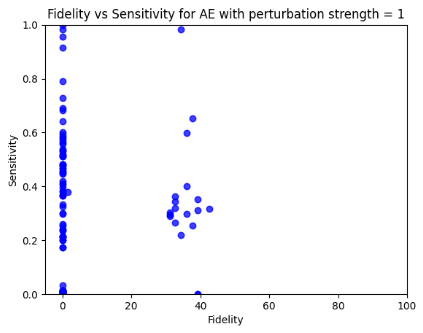
(b)



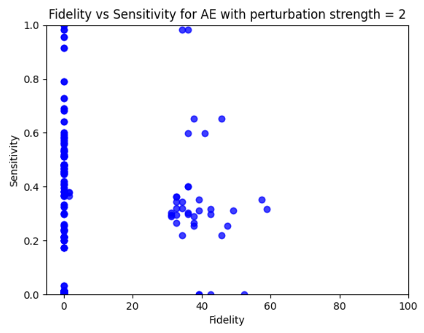
(c)

Fig. C.2. Sensitivity and fidelity with different perturbation strengths. (a) EBM-Basic with strength 1 (b) EBM-Basic with strength 2 (c) EBM-Basic with strength 3

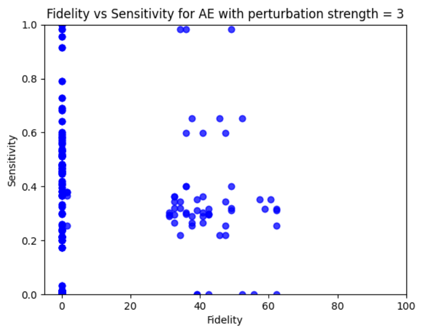
C.3 Binary: AE-Normal



(a)



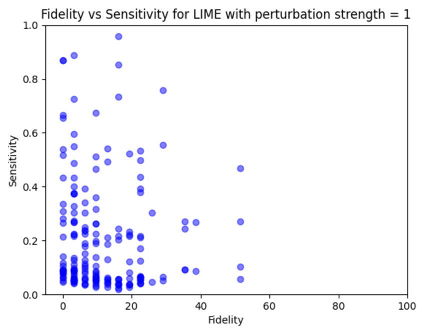
(b)



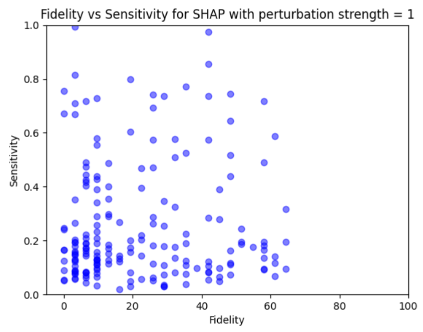
(c)

Fig. C.3. Sensitivity and fidelity with different perturbation strengths. (a) AE-Normal with strength 1 (b) AE-Normal with strength 2 (c) AE-Normal with strength 3

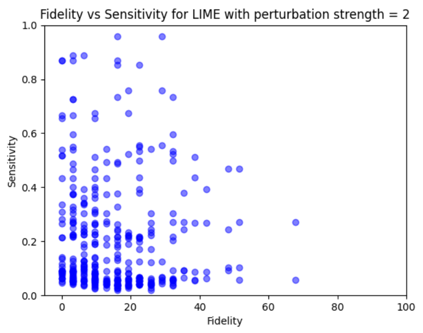
C.4 Binary: RF-Latent



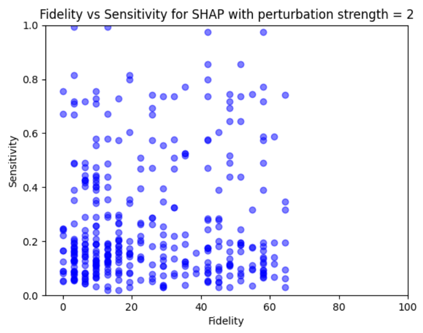
(a)



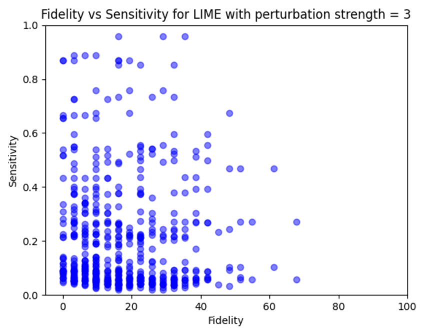
(b)



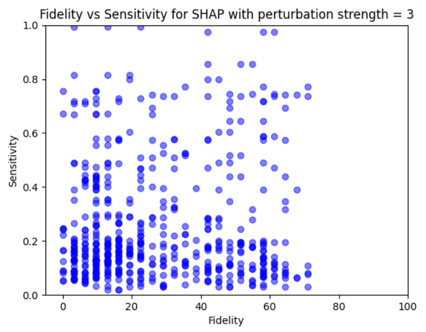
(c)



(d)



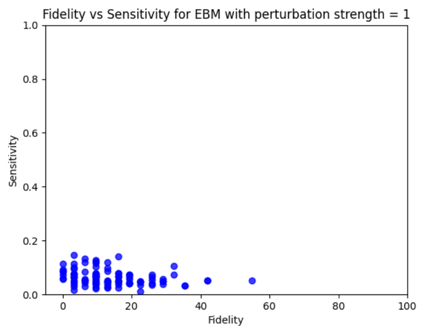
(e)

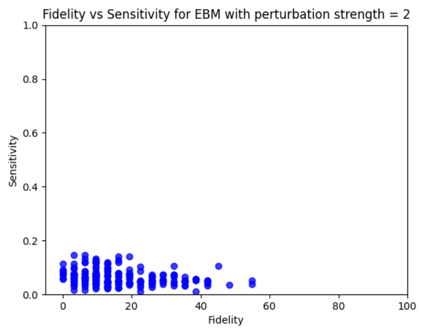


(f)

Fig. C.4. Sensitivity and fidelity with different perturbation strengths. (a) RF-Latent + LIME with strength 1 (b) RF-Latent + SHAP with strength 1 (c) RF-Latent + LIME with strength 2 (d) RF-Latent + SHAP with strength 2 (e) RF-Latent + LIME with strength 3 (f) RF-Latent + SHAP with strength 3

C.5 Binary: EBM-Latent

(a) 

(b) 

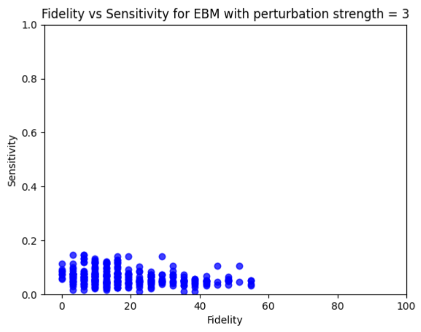
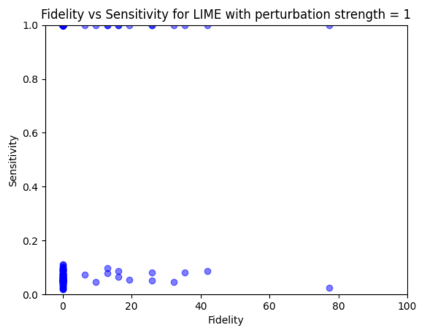
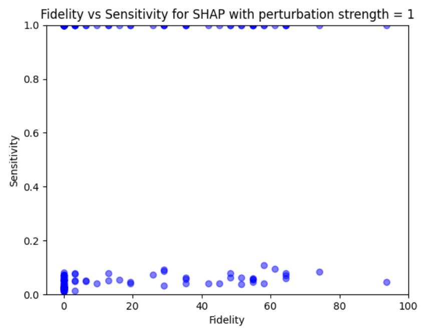
(c) 

Fig. C.5. Sensitivity and fidelity with different perturbation strengths. (a) EBM-Latent with strength 1 (b) EBM-Latent with strength 2 (c) EBM-Latent with strength 3

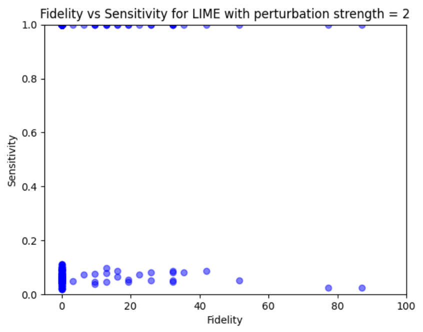
C.6 Multiclass: RF-Minority



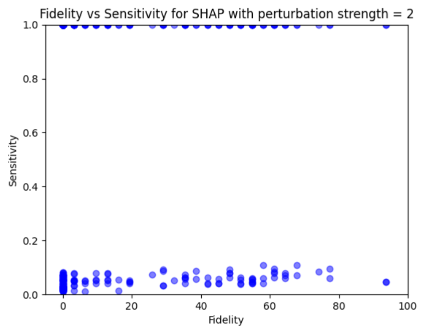
(a)



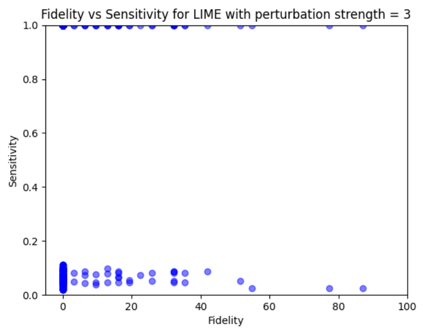
(b)



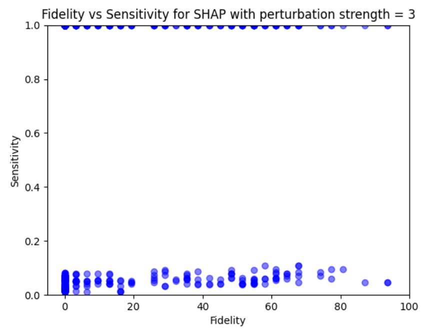
(c)



(d)



(e)



(f)

Fig. C.6. Sensitivity and fidelity with different perturbation strengths. (a) RF-Minority + LIME with strength 1 (b) RF-Minority + SHAP with strength 1 (c) RF-Minority + LIME with strength 2 (d) RF-Minority + SHAP with strength 2 (e) RF-Minority + LIME with strength 3 (f) RF-Minority + SHAP with strength 3

C.7 Multiclass: RF-SMOTE

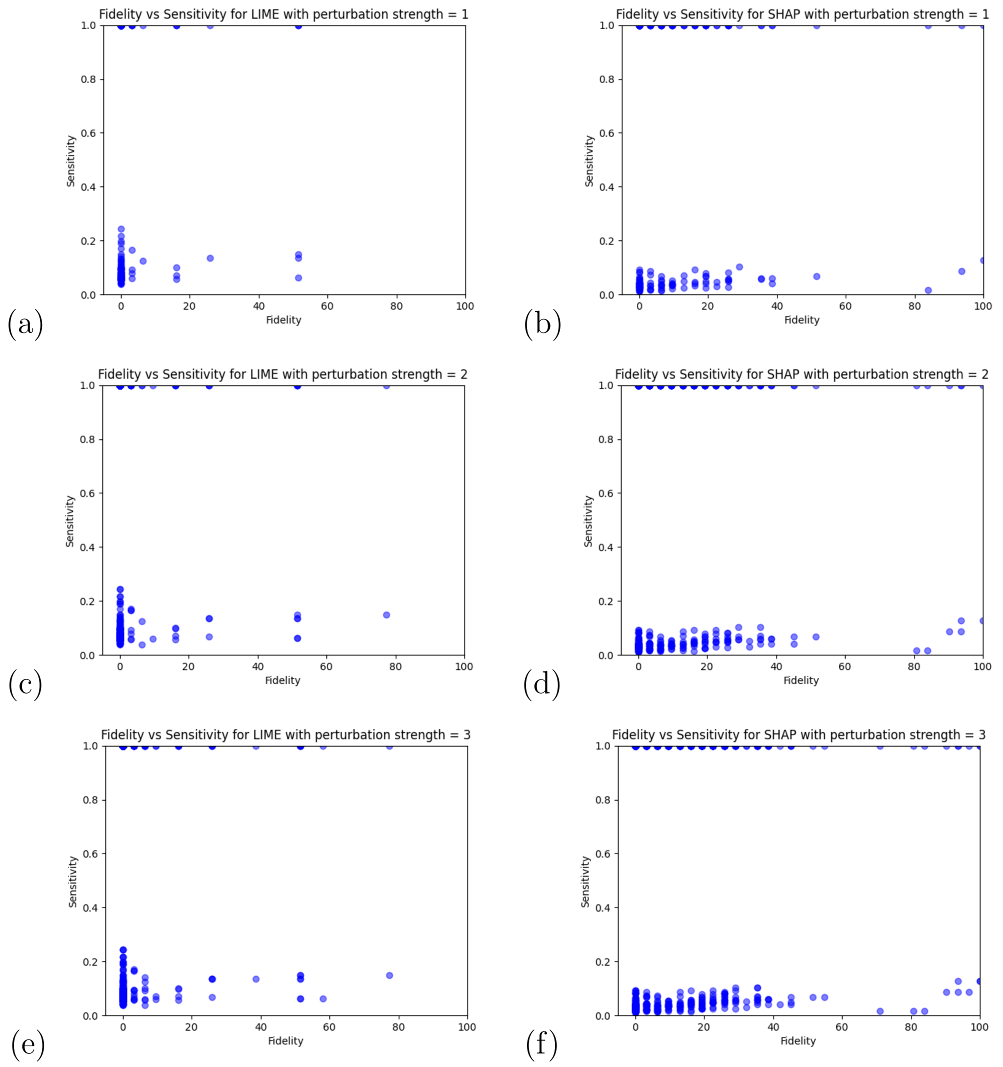


Fig. C.7. Sensitivity and fidelity with different perturbation strengths. (a) RF-SMOTE + LIME with strength 1 (b) RF-SMOTE + SHAP with strength 1 (c) RF-SMOTE + LIME with strength 2 (d) RF-SMOTE + SHAP with strength 2 (e) RF-SMOTE + LIME with strength 3 (f) RF-SMOTE + SHAP with strength 3

Bibliography

[1] Diana L. Aguilar et al. "Towards an Interpretable Autoencoder: A Decision-Tree-Based Autoencoder and its Application in Anomaly Detection". In: *IEEE Transactions on Dependable and Secure Computing* 20.2 (2023), pp. 1048–1059.

[2] Rafa Alenezi and Simone A Ludwig. "Explainability of Cybersecurity Threats Data Using SHAP". In: *IEEE Symposium Series on Computational Intelligence*. 2021, pp. 1–10.

[3] Alhanoof Althnian et al. "Impact of dataset size on classification performance: an empirical evaluation in the medical domain". In: *Applied Sciences* 11.2 (2021), p. 796.

[4] Julia Amann et al. "Explainability for artificial intelligence in healthcare: a multidisciplinary perspective". In: *BMC Medical Informatics and Decision Making* 20.1 (2020), pp. 1–9.

[5] Liat Antwarg et al. "Explaining anomalies detected by autoencoders using Shapley Additive Explanations". In: *Expert systems with applications* 186 (2021), p. 115736.

[6] Daniel W. Apley and Jingyu Zhu. "Visualizing the effects of predictor variables in black box supervised learning models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 82.4 (2020), pp. 1059–1086.

[7] Alejandro B. Arrieta et al. "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Information fusion* 58 (2020), pp. 82–115.

[8] Francesco Bodria et al. "Benchmarking and survey of explanation methods for black box models". In: *Data Mining and Knowledge Discovery* (2023), pp. 1–60.

[9] Giampaolo Bovenzi et al. "A hierarchical hybrid intrusion detection approach in IoT scenarios". In: *IEEE Global Communications Conference*. IEEE. 2020, pp. 1–7.

[10] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[11] Myriam D. Cavelty. "Cyber-security". In: *The Routledge handbook of new security studies*. Routledge, 2010, pp. 154–162.

[12] Fabien Charmet et al. "Explainable artificial intelligence for cybersecurity: a literature survey". In: *Annals of Telecommunications* (2022), pp. 1–24.

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

[14] Zixi Chen et al. "What Makes a Good Explanation? A Harmonized View of Properties of Explanations". In: *Workshop on Trustworthy and Socially Responsible Machine Learning*. NeurIPS. 2022.

[15] Kate Conger and Kevin Roose. "Uber Investigating Breach of Its Computer Systems". In: *The New York Times* (Sept. 2022). [Online; accessed 5 Dec 2022]. URL: `https://www.nytimes.com/2022/09/15/technology/uber-hacking-breach.html`.

[16] Angelo Corallo et al. "Cybersecurity awareness in the context of the Industrial Internet of Things: A systematic literature review". In: *Computers in Industry* 137 (2022), p. 103614.

[17] European Union Agency for Cybersecurity (ENISA). [Online; accessed 5 Dec 2022]. Nov. 2022. URL: `https://www.enisa.europa.eu/publications/enisa-threat-landscape-2022`.

[18] Amit Dhurandhar et al. "Explanations based on the missing: Towards contrastive explanations with pertinent negatives". In: *Advances in Neural Information Processing Systems* 31 (2018).

[19] Gonzalo I. Diaz et al. "An effective algorithm for hyperparameter optimization of neural networks". In: *IBM Journal of Research and Development* 61.4/5 (2017), pp. 9–1.

[20] Finale Doshi-Velez and Been Kim. "Towards a rigorous science of interpretable machine learning". In: *arXiv preprint arXiv:1702.08608* (2017).

[21] Emmanuel Doumard et al. "A quantitative approach for the comparison of additive local explanation methods". In: *Information Systems* 114 (2023), p. 102162.

[22] Ghada Elkhawaga et al. "Evaluating Explainable Artificial Intelligence Methods Based on Feature Elimination: A Functionality-Grounded Approach". In: *Electronics* 12.7 (2023), p. 1670.

[23] Radwa ElShawi et al. "Interpretability in healthcare: A comparative study of local machine learning interpretability techniques". In: *Computational Intelligence* 37.4 (2021), pp. 1633–1650.

[24] Tolga Ergen and Suleyman Serdar Kozat. "Unsupervised anomaly detection with LSTM neural networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 31.8 (2019), pp. 3127–3141.

[25] Gilberto Fernandes et al. "A comprehensive survey on network anomaly detection". In: *Telecommunication Systems* 70.3 (2019), pp. 447–489.

[26] Mohamed A. Ferrag et al. "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study". In: *Journal of Information Security and Applications* 50 (2020), p. 102419.

[27] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. "Why do tree-based models still outperform deep learning on tabular data?" In: *arXiv preprint arXiv:2207.08815* (2022).

[28] Riccardo Guidotti et al. "A survey of methods for explaining black box models". In: *ACM Computing Surveys (CSUR)* 51.5 (2018), pp. 1–42.

[29] Berk Gulmezoglu. "XAI-based microarchitectural side-channel analysis for website fingerprinting attacks and defenses". In: *IEEE Transactions on dependable and secure computing* 19.6 (2021), pp. 4039–4051.

[30] Wenbo Guo et al. "Lemna: Explaining deep learning based security applications". In: *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security.* (2018), pp. 364–379.

[31] Swastik Haldar, Philips G. John, and Diptikalyan Saha. "Reliable Counterfactual Explanations for Autoencoder Based Anomalies". In: *Proceedings of the 3rd ACM India Joint International Conference on Data Science Management of Data*. 2021, pp. 83–91.

[32] Yasir Hamid et al. "Benchmark Datasets for Network Intrusion Detection: A Review." In: *International Journal of Network Security* 20.4 (2018), pp. 645–654.

[33] Songqiao Han et al. "Adbench: Anomaly detection benchmark". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 32142–32159.

[34] Swetha Hariharan et al. "XAI for intrusion detection system: comparing explanations based on global and local scope". In: *Journal of Computer Virology and Hacking Techniques* (2022), pp. 1–23.

[35] Eric Holder and Ning Wang. "Explainable artificial intelligence (XAI) interactively working with humans as a junior cyber analyst". In: *Human-Intelligent Systems Integration* 3.2 (2021), pp. 139–153.

[36] Andreas Holzinger et al. "Explainable AI methods-a brief overview". In: *xxAI-Beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020.* Springer. (2022), pp. 13–38.

[37] Antoine Hudon et al. "Explainable artificial intelligence (XAI): how the visualization of AI predictions affects user cognitive load and confidence". In: *Information Systems and Neuroscience: NeuroIS Retreat 2021.* Springer. (2021), pp. 237–246.

[38] Tharmini Janarthanan and Shahrzad Zargari. "Feature selection in UNSW-NB15 and KDDCUP'99 datasets". In: *Proceddings IEEE 26th International Symposium on Industrial Electronics (ISIE).* IEEE. 2017, pp. 1881–1886.

[39] Christian Janiesch, Patrick Zschech, and Kai Heinrich. "Machine learning and deep learning". In: *Electronic Markets* 31.3 (2021), pp. 685–695.

[40] Sérgio Jesus et al. "How can I choose an explainer? An application-grounded evaluation of post-hoc explanations". In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency.* 2021, pp. 805–815.

[41] Jeya V. Jeyakumar et al. “How can i explain this to you? an empirical study of deep neural network explanation methods”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4211–4222.

[42] Enamul Kabir et al. “A novel statistical technique for intrusion detection systems”. In: *Future Generation Computer Systems* 79 (2018), pp. 303–318.

[43] Muhammad A. Khan, Md Rezaul Karim, and Yangwoo Kim. “A scalable and hybrid intrusion detection system based on the convolutional-LSTM network”. In: *Symmetry* 11.4 (2019), p. 583.

[44] Todd Kulesza et al. “Principles of Explanatory Debugging to Personalize Interactive Machine Learning”. In: *Proceedings of the 20th International Conference on Intelligent User Interfaces.* ACM, 2015, pp. 126–137.

[45] Todd Kulesza et al. “Tell me more? The effects of mental model soundness on personalizing an intelligent agent”. In: *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems.* 2012, pp. 1–10.

[46] Himabindu Lakkaraju, Nino Arsov, and Osbert Bastani. “Robust and stable black box explanations”. In: *International Conference on Machine Learning.* PMLR. 2020, pp. 5628–5638.

[47] Benedikt Leichtmann et al. “Effects of Explainable Artificial Intelligence on trust and human behavior in a high-risk decision task”. In: *Computers in Human Behavior* 139 (2023), p. 107539.

[48] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. “Explainable ai: A review of machine learning interpretability methods”. In: *Entropy* 23.1 (2020), p. 18.

[49] Lan Liu et al. “Intrusion detection of imbalanced network traffic based on machine learning and deep learning”. In: *IEEE Access* 9 (2020), pp. 7550–7563.

[50] Pedro Lopes et al. “XAI Systems Evaluation: A Review of Human and Computer-Centred Methods”. In: *Applied Sciences* 12.19 (2022), p. 9423.

[51] Hampus Lundberg et al. “Experimental Analysis of Trustworthy In-Vehicle Intrusion Detection System Using eXplainable Artificial Intelligence (XAI)”. In: *IEEE Access* 10 (2022), pp. 102831–102841.

[52] Scott M. Lundberg and Su-In Lee. "A unified approach to interpreting model predictions". In: *Advances in Neural Information Processing Systems* 30 (2017).

[53] Scott M. Lundberg et al. "From local explanations to global understanding with explainable AI for trees". In: *Nature Machine Intelligence* 2.1 (2020), pp. 2522–5839.

[54] Keza MacDonald, Keith Stuart, and Alex Hern. "This article is more than 2 months old Grand Theft Auto 6 leak: who hacked Rockstar and what was stolen?" In: *The Guardian* (Sept. 2022). URL: `https://www.theguardian.com/games/2022/sep/19/grand-theft-auto-6-leak-who-hacked-rockstar-and-what-was-stolen`.

[55] Microsoft. Nov. 2022. URL: `https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE5bUvv?culture=en-us&amp;country=us`.

[56] Tarek A. El-Mihoub, Lars Nolle, and Frederic Stahl. "Explainable Boosting Machines for Network Intrusion Detection with Features Reduction". In: *Artificial Intelligence XXXIX: Proceedings of 42nd SGAI International Conference on Artificial Intelligence*. Springer. 2022, pp. 280–294.

[57] Tim Miller. "Explanation in artificial intelligence: Insights from the social sciences". In: *Artificial intelligence* 267 (2019), pp. 1–38.

[58] Yisroel Mirsky et al. "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection". In: *Proceedings of Network and Distributed Systems Security (NDSS) Symposium*. 2018.

[59] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. "A multidisciplinary survey and framework for design and evaluation of explainable AI systems". In: *ACM Transactions on Interactive Intelligent Systems (TiiS)* 11.3-4 (2021), pp. 1–45.

[60] Christoph Molnar. *Interpretable machine learning*. [Online; accessed 21 Nov 2022]. 2022.

[61] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. "Explaining machine learning classifiers through diverse counterfactual explanations". In: *Proceedings of the 2020 conference on Fairness, Accountability, and Transparency*. 2020, pp. 607–617.

[62] Nour Moustafa and Jill Slay. "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set". In: *Information Security Journal: A Global Perspective* 25.1-3 (2016), pp. 18–31.

[63] Nour Moustafa and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)". In: *Proceedings of Military Communications and Information Systems Conference (MilCIS)*. IEEE. 2015, pp. 1–6.

[64] Quoc P. Nguyen et al. "Gee: A gradient-based explainable variational autoencoder for network anomaly detection". In: *Proceedings of IEEE Conference on Communications and Network Security (CNS)*. IEEE. 2019, pp. 91–99.

[65] Harsha Nori et al. "InterpretML: A Unified Framework for Machine Learning Interpretability". In: *arXiv preprint arXiv:1909.09223* (2019).

[66] Guansong Pang, Chunhua Shen, and Anton van den Hengel. "Deep anomaly detection with deviation networks". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019, pp. 353–362.

[67] Jonathon P. Phillips et al. "Four principles of explainable artificial intelligence". In: *Gaithersburg, Maryland* (2020).

[68] Majjed Al-Qatf et al. "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection". In: *IEEE Access* 6 (2018), pp. 52843–52856.

[69] Marco T. Ribeiro, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 1135–1144.

[70] Marko Robnik-Šikonja and Marko Bohanec. "Perturbation-based explanations of prediction models". In: *Human and Machine Learning: Visible, Explainable, Trustworthy and Transparent* (2018), pp. 159–175.

[71] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Cambridge, MA, USA: MIT Press, 1986, pp. 318–362. ISBN: 026268053X.

[72] Wojciech Samek et al. “Explaining deep neural networks and beyond: A review of methods and applications”. In: *Proceedings of the IEEE* 109.3 (2021), pp. 247–278.

[73] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. “Evaluating standard feature sets towards increased generalisability and explainability of ML-based network intrusion detection”. In: *Big Data Research* 30 (2022), p. 100359.

[74] Mohanad Sarhan et al. “Netflow datasets for machine learning-based network intrusion detection systems”. In: *Big Data Technologies and Applications*. Springer, 2020, pp. 117–135.

[75] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning important features through propagating activation differences”. In: *Proceedings of International Conference on Machine Learning*. PMLR. 2017, pp. 3145–3153.

[76] David Silver et al. “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play”. In: *Science* 362.6419 (2018), pp. 1140–1144.

[77] Youngrok Song, Sangwon Hyun, and Yun-Gyung Cheong. “Analysis of autoencoders for network intrusion detection”. In: *Sensors* 21.13 (2021), p. 4294.

[78] Yu I Starodubtsev et al. “Cyberspace: terminology, properties, problems of operation”. In: *Proceedings of International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*. IEEE. 2020, pp. 1–3.

[79] Julian Tritscher, Anna Krause, and Andreas Hotho. “Feature relevance XAI in anomaly detection: Reviewing approaches and challenges”. In: *Frontiers in Artificial Intelligence* 6 (2023), p. 7.

[80] Mythreyi Velmurugan et al. “Evaluating fidelity of explainable methods for predictive process analytics”. In: *Intelligent Information Systems: CAiSE Forum 2021, Melbourne, VIC, Australia, June 28–July 2, 2021, Proceedings*. Springer. 2021, pp. 64–72.

[81] Huiwen Wang, Jie Gu, and Shanshan Wang. “An effective intrusion detection framework based on SVM with feature augmentation”. In: *Knowledge-Based Systems* 136 (2017), pp. 130–139.

[82] Chih-Kuan Yeh et al. “On the (in) fidelity and sensitivity of explanations”. In: *Advances in Neural Information Processing Systems* 32 (2019).

[83] Aviv Yehezkel, Eyal Elyashiv, and Or Soffer. “Network anomaly detection using transfer learning based on auto-encoders loss normalization”. In: *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*. 2021, pp. 61–71.

[84] Hongpo Zhang et al. “An effective deep learning based scheme for network intrusion detection”. In: *Proceedings of the 24th International Conference on Pattern Recognition*. IEEE. 2018, pp. 682–687.

[85] Liang Zhou et al. “Cyber-attack classification in smart grid via deep neural network”. In: *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*. 2018, pp. 1–5.

[86] Maede Zolanvari et al. “TRUST XAI: Model-Agnostic Explanations for AI With a Case Study on IIoT Security”. In: *IEEE Internet of Things Journal* 10.4 (2023), pp. 2967–2978.