DAIKIRI
Diagnostische KI für industrielle Daten

DAIKIRI
Erklärbare Diagnostische KI für industrielle Daten

Project Number: 01IS19085 **Start Date of Project:** 01/01/2020 **Duration:** 30 months

Deliverable 4.3
Procedure for explaining black-box approaches

Dissemination Level	Public
Due Date of Deliverable	Month 24, 31/12/2021
Actual Submission Date	Month 24, 29/08/2022
Work Package	WP4 — Explainable machine learning
Task	T4.3
Type	Report
Approval Status	Final
Version	1.0
Number of Pages	22

The information in this document reflects only the author's views, and the Federal Ministry of Education and Research (BMBF) is not liable for any use that may be made of the information contained therein. The information in this document is provided "as is" without guarantee or warranty of any kind, express or implied, including but not limited to the fitness of the information for a particular purpose. The user thereof uses the information at his/ her sole risk and liability.

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

This project has received funding from the Federal Ministry of Education and Research (BMBF) within the project DAIKIRI under the grant no 01IS19085.

History

Version	Date	Reason	Revised by
0.0	28/06/2022	First draft created	Carolin Walter
0.1	11/07/2022	Draft revised	Fabian Witter
0.2	02/08/2022	Final version created	Carolin Walter
0.3	12/08/2022	Peer reviewed	Dr.-Ing. Stefan Balke
0.4	29/08/2022	Final version submitted	Carolin Walter

Author List

Organization	Name	Contact Information
USU	Carolin Walter	carolin.walter@usu.com
pmOne	Fabian Witter	Fabian.Witter@pmOne.com

Executive Summary

In this deliverable a short summary on state-of-the-art methods to explain black box models is given. We show an example for explaining through visualization and we took a deeper look at the post-hoc explainer SHAP. Therefore, the model and explainer are analyzes. Afterwards we show how black and white box approach work together in DAIKIRI.

Contents

1 Introduction

Black-box methods are the current state of the art in industrial applications. Therefore, this deliverable discusses the procedures for explaining black-box models. First, we give a brief overview of the various existing methods to interpret black-box procedures (Section 2). We have tried and evaluated several of these methods on different data sets relevant to the project. In this deliverable, we present the interpretability through visualization using synthetic oscillation data (Section 3) and the use of SHAP on the data of the logistic use case (Section 4). Here we show on the one hand the explanations that SHAP provides, which due to their complexity rather address data scientists and on the other hand an approach developed by us, which takes the most important arguments from the SHAP values and verbalizes them for a domain expert. Afterwards, we show how the predictions of a black-box model can be explained using the white-box method of the DAIKIRI platform (Section 6). We conclude with a short discussion and outlook (Section 7).

2 Interpreting Black-box Approaches

In the following, we distinguish between white-box and black-box models. The strengths and weaknesses of white-box versus black-box models are visualized in Figure 1. White-box models are highly interpretable machine learning models. The decisions of white-box models are easy to understand for humans. In contrast, black-box approaches cannot be understood directly by their model parameters. However, they are more frequently used due to their (usually) better performance, flexibility, and their large range of algorithms and supported data spaces. Since black-box models are not explainable on their own, additional post-hoc explainers are necessary to provide explanations to humans on black-box decisions. However, additional explainers increase the complexity of the solution and post-hoc explainers may not be consistent with the model.

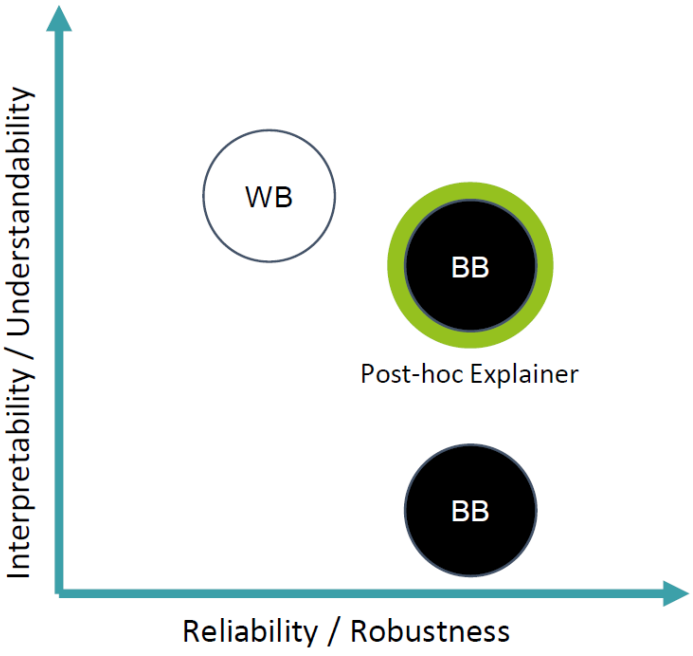


Figure 1: The strengths and weaknesses for black-box (BB) versus white-box (WB) models.

Getting insights on how the model behaves is crucial in case of all real-world problems with incomplete problem formalization [Doshi-Velez and Kim, 2017]. Here, explainers help to detect bias in models and increase trust and social acceptance. Explainers are also used to gain knowledge or learn more about the problem: By interpreting black-box models, relevant knowledge on relationships can be extracted, either contained in the data or learned by the model [Murdoch et al., 2019]. Although procedures for explainability have been used for some time, there is little consensus on what interpretability is and no explicit definition exists what an explainer needs to include [Doshi-Velez and Kim, 2017, Molnar, 2020]. In the following, the difference of some major explainability techniques are represented.

2.1 Intrinsically Explainable Models

Intrinsic explainable models are also called white-box models. They are self-explanatory. This means that the models are mostly short decision trees or slim linear models. So mainly shallow models can be used here. The explanations are always model-specific. For example, a decision tree is a white box model. After learning the parameters, each decision can be explained by traversing upwards through the tree (from leaf to root).

2.2 Example-based Explanations

Example-based explanations [Aamodt and Plaza, 1994] use individual instances to explain the behavior of the model or the underlying data structure. This allows analogies to be found in the data, but the data must be easily representable to humans, e.g., as is the case for image or audio data.

Examples of example-based explanations are counterfactual explanations [Wachter et al., 2017, Dandl et al., 2020], which indicate under which circumstances a prediction would not have been received. A counterfactual explanation of a prediction describes the smallest change to the feature values that changes the prediction to a predefined output. Furthermore, the adversarial examples method [Goodfellow et al., 2014] disturbs the features of a sample a little bit to provoke a wrong prediction. This helps to identify weaknesses in an ML model to prevent later fraud on the learned model. When examining influential data points [Koh and Liang, 2017], it is found out what influence the removal of some data points has on the model parameters. This procedure is used to debug models and understand their behavior.

2.3 Model-agnostic Methods

In the case of model-agnostic methods, the explanations are separated from the ML model. These are post-hoc methods that analyze the model after training. The methods are usually very flexible, i.e., one method can be applied to a variety of ML models, allowing to compare ML models. Usually, feature input and output pairs are analyzed. The methods do not have access to model internals such as the model weights. Model-agnostic methods usually provide global explanation, which explains the behavior of the entire model and local explanation, which explains the prediction of a single data point.

Examples of global model-agnostic methods are Partial Dependence Plots (PDP, Friedman [2001]), ICE (Individual Conditional Expectation, Goldstein et al. [2015]), and ALE (Accumulated Local Effects, Apley and Zhu [2020]) plots. Partial Dependence Plots (PDP) indicate the (partial) effect of one or two features on the prediction and the nature of the relationship. Individual Conditional Expectation Plots indicate by one line per data point how the prediction would behave if the feature were

changed. With ALE plots, the means of predictions with the same feature value is shown.

Another possibility is the creation of a global surrogate model. This is an interpretable model, which is trained with the data points and the predictions of the black box model. This surrogate model must approximate the prediction function as well as possible, and needs to be interpretable. In this way, conclusions can then be drawn from the interpretable model about the black-box model.

Local surrogate models explain individual predictions of a black-box model. Lime [Ribeiro et al., 2016], for example, tests what happens to the predictions for a perturbed input data set and trains an interpretable model on it [Ribeiro et al., 2016]. This works with tabular data, text, and images. Similarly, the scoped rules (anchors) [Ribeiro et al., 2018] provide explanations for classification problems by finding rules that sufficiently anchor a prediction. The output here consists of if-then rules. SHAP (SHapley Additive exPlanations, Lundberg and Lee [2017]) provides locally and globally consistent explanations and is suitable for tabular data, images and NLP in classification and regression tasks.

2.4 Neural Network Explanations

For neural networks, the model-agnostic methods such as local models or partial dependence plots can be used. However, model agnostic methods that create their explanations from the outside are not particularly computationally efficient. In addition, neural networks learn features and concepts from their hidden layers. For this purpose, in addition to the feature visualization there are special tools to uncover them like Pixel Attribution (Saliency Maps, Simonyan et al. [2013]), Detecting Concepts [Kim et al., 2018], Integrated Gradients [Sundararajan et al., 2017], and Grand-CAM [Selvaraju et al., 2017].

2.5 Goals for Explainability within DAIKIRI

Up to now, methods for interpreting black-box models are mostly used by developers and data scientists to further improve their models. Our goal within Workpackage 4.3 of DAIKIRI is to make these methods accessible for users like mechanical engineers or machine operators.

The problem with common “raw” explainers is that some need configuration to generate adequate results. For other explainers, some knowledge is needed on how the explainer works to interpret its findings. Overall, there is often too much information and too many explanations. This could further confuse humans and does not increase reliability and trust in the models. In the following, we show examples on how explanations could be easily accessed.

3 Explanations through Visualization

Due to the difficulty of getting real data from industry, we decided to generate oscillation data. The advantage of synthetic data is that we know the data, there are no unknown artifacts, and therefore we can better assess our approach, model, and explanations. The downside is that the data is simply a theoretical playground and real data would pose additional unknown challenges. Since we have some background in the field of machine tools and condition monitoring and additionally, predictive maintenance of machine tools are sought-after applications, we have decided to generate vibration data in analogy to Seemuang et al. [2016].

3.1 Definition of the Use Case

The aim of the use case is to monitor the condition of a milling machine and to detect anomalies. The vibrations caused by anomalous behaviour of the machine should be visually highlighted and presented in context with the other vibrations of the processing steps.

Milling machines use rotating cutting tools to remove material from a workpiece by machining. At least three feed lines are available. In this way, complex shapes can be produced automatically with high precision. Early detection of tool breakage or wear at the cutting edge promotes the quality of the product and reduces rejects. Spectrograms contain information about the condition of machines. Figure 2 shows the normal state of a cutting operation. The different work steps of the machine tool cause vibrations in different frequency ranges (y-axis) with different intensities (marked by the color) during processing (x-axis).

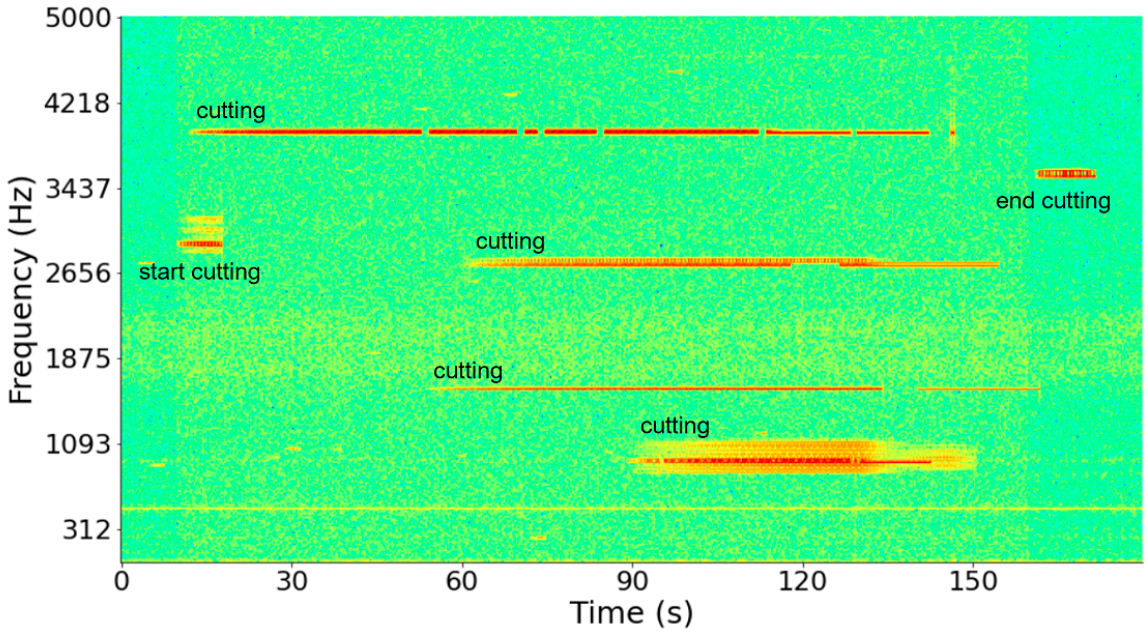


Figure 2: Spectrogram showing a normal state of a cutting operation.

3.2 Experimental Setup

First, we generated the vibration data. A subset of the data contains anomalies, which have connections with other features. Then a model with the data of the normal state was learned. The events in the normal state are classified and labeled. For this purpose, we use a network of Convolutional Neural Networks (CNN) for the feature recognition in the input data in conjunction with a Long Short-Term Memory (LSTM), which supports sequential predictions, i.e., can recognize features over time steps [Goodfellow et al., 2016]. The CNN LSTM learns features in these spectrograms that occur again and again and represent a normal state of the machine. For an unseen spectrogram, the features can be labeled. The detected features are given a bounding box, which specifies the label for the class that is predicted, as well as the accuracy (Figure 3).

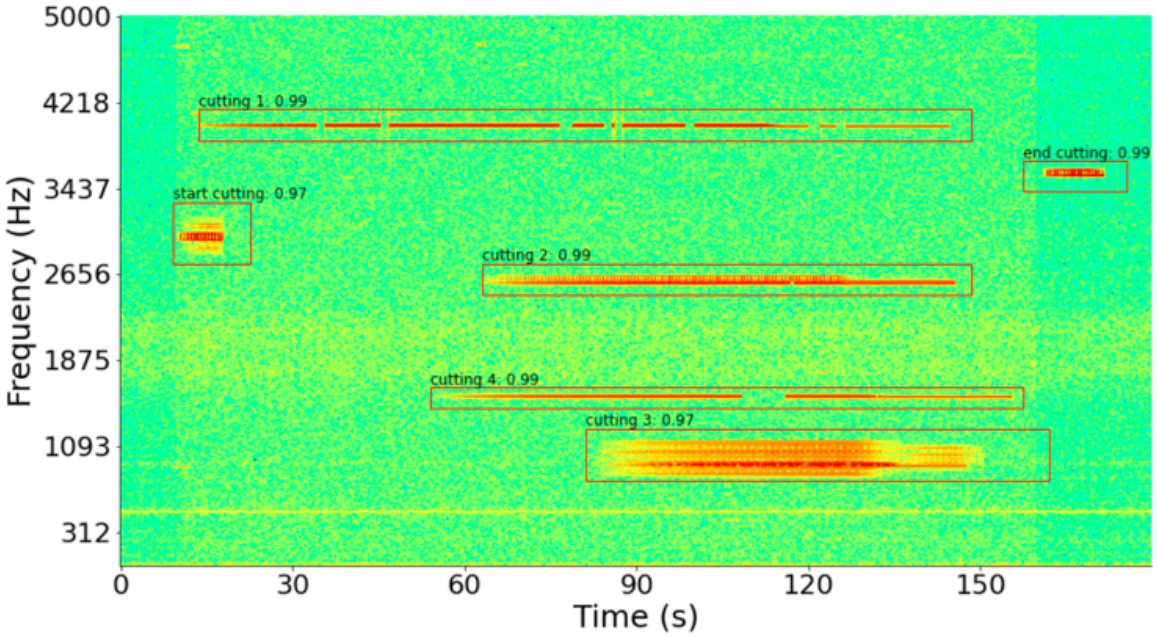


Figure 3: Spectrogram showing a normal state of a cutting operation. Events are recognized and surrounded with a bounding box. The bounding boxes are labeled for the known classes/processes.

3.3 Results: Explanations by Visualization

Running the model for unseen spectrograms containing anomalies, the network will not find a proper class for the anomaly feature and thus output a very poor accuracy for the prediction. The domain expert is thus given a direct visual illustration of where the model detects an anomaly and in which context it occurs. (Figure 4).

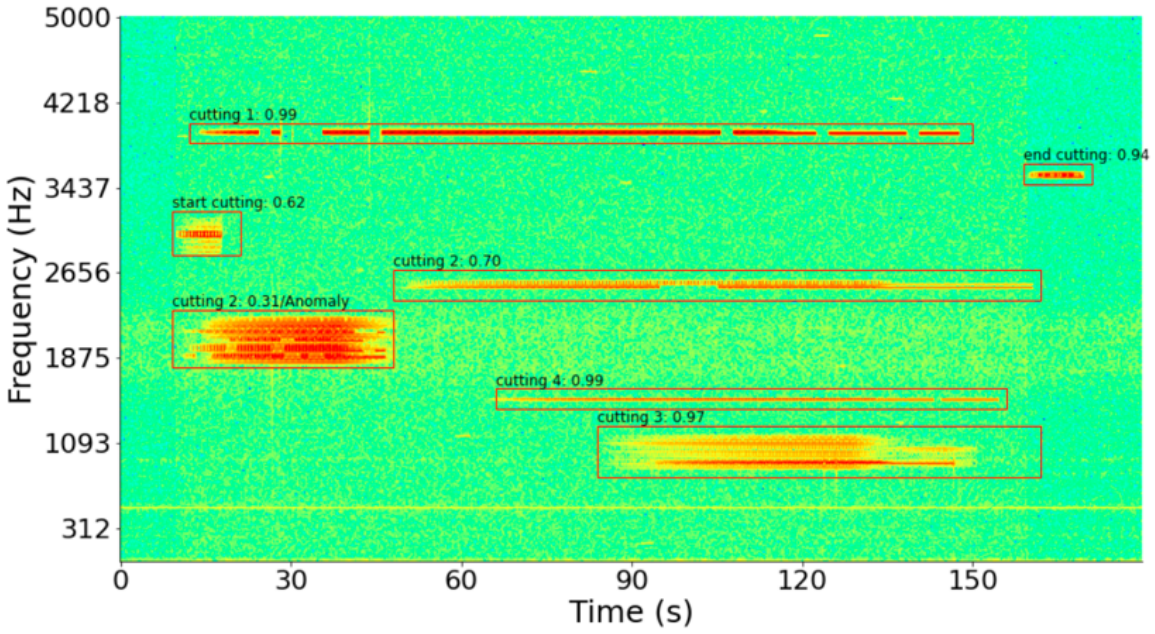


Figure 4: Spectrogram showing an anomalous behavior of a cutting operation.

4 Case Study: Black-box Explainer for Anomaly Detection

Before implementing a black-box model with post-hoc explainer for XIA in practice, we need to investigate this approach for possible downsides and its general behavior. Therefore, we design a case study in industrial context. For the data in this case study, we draw on the "Smart Logistics" use case (cf. Deliverable 6.1). The data set contains three anomaly types (cf. Deliverable 6.1, Section 3.1) which have been labeled using hand-crafted rules. In contrast to most real-world scenarios, the rules are known. This allows us to define an anomaly detection application within which we can analyze the behavior of the black-box model and post-hoc explainer and their interaction in detail. In the following, we describe methodology and results of this case study.

4.1 Definition of an XAI Application for Smart Logistics

In this section, we motivate the use of XAI in a "Smart Logistics" use case and specify requirements for the XAI approach based on the properties of the existing rule-based system. Afterwards, we introduce the architecture of our XAI application.

In the current setup, stock anomalies in the small-parts warehouse are detected by hand-crafted rules. Creating such rules requires the knowledge of logistics experts, or even knowledge of the processes in the specific warehouse the anomaly detection system should be deployed in. Moreover, such anomaly detection systems can only detect anomalies that are defined and covered by the rules, i.e. new or unknown anomalies cannot be detected. Combining both characteristics above leads to high maintenance costs (both monetarily and time-wise) when processes in the warehouse change or new rules are required.

Furthermore, the existing rule-based approach for detecting anomalies has the following additional properties:

1. The detection performance is high according to the hand-crafted rules. Nevertheless, the rules might not cover some edge cases and, hence, the rule-based approach might miss some anomalies.
2. Applications of rule-based approaches to data are always explainable due to the nature of hand-crafted rules. Moreover, the explanations are always correct according to the rules.

These latter properties are favorable and, therefore, a machine learning replacement for the rule-based approach should comply with them. Yet, the fulfillment of high quality in detection performance and explanations is sophisticated.

Therefore, our approach for replacing the rule-based system with an XAI setup implements a hierarchical two-layer architecture consisting of a black-box classifier and a black-box explainer. This architecture with a distinct explanation component allows the usage of state-of-the-art anomaly detection models and, hence, promises higher detection performance compared to a holistic white-box model Seliya et al. [2021]. For this use case, we restrict the classifier to the class of One-Class Classification (OCC) models which learn the normal system behavior within the small-parts warehouse and, accordingly, is able to detect new types of anomalies.

The downside of using black-box classifiers for anomaly detection is their lack of interpretability. To make the model's predictions transparent and interpretable, we add a black-box explainer on top of the classifier, which infers explanations for the classifications of the black-box model.

In the following, we aim to apply and evaluate our approach using the collected data from the

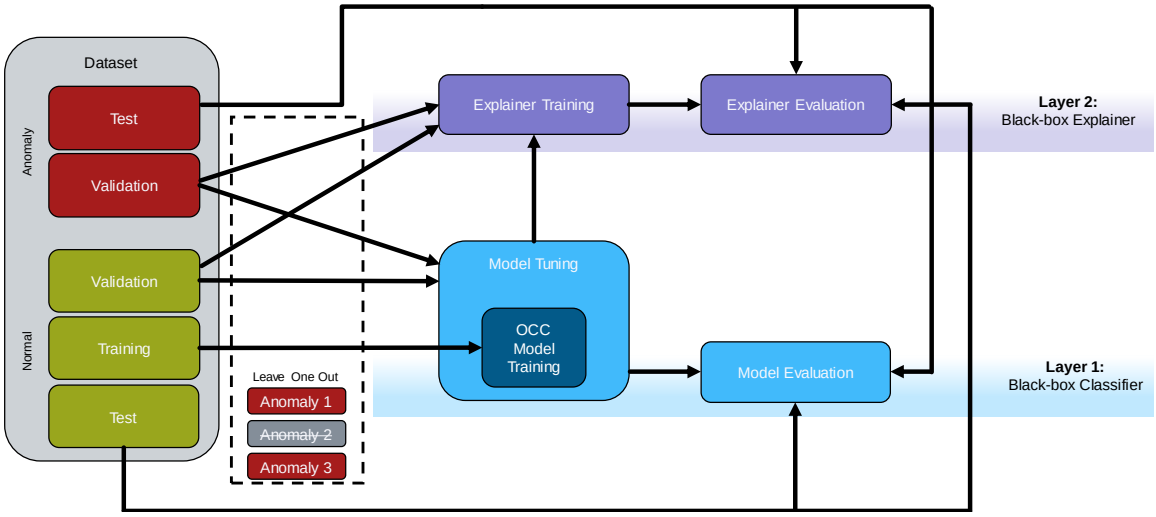


Figure 5: Experimental setup for the smart logistics use case. Experiment steps for the black-box One-Class Classifier (OCC) are located in Layer 1 (blue) of the hierarchy and consist of model training, tuning, and evaluation. Layer 2 (purple) contains explainer training and evaluation steps, which depend on the OCC. The datasets to use are split into training, validation and test set and are build from normal and anomaly samples. The arrows display the data usage or dependencies in the setup, respectively. For training the OCC and explainer, we will remove each anomaly type consecutively from the datasets such that we can evaluate the models for unseen anomaly types.

small-parts warehouse. To this end, we designed appropriate experiments, which are described in the section below.

4.2 Experimental Setup

In this section, we define the procedure for running experiments given the hierarchical model architecture introduced above. The experiments do not only focus on performance of classifier and explainer, but also on the ability of detecting novel anomaly types. We show the experimental setup in Figure 5 and describe the procedure in the following.

For our experiment, we split the data into a training, validation and test set. The training set only contains normal data since we have an OCC setup, i.e. it does not contain any anomalies. The validation set consists of both normal and anomalous records. During our experiments, we do not use all anomalous validation data but remove one of the three anomaly types in turns to evaluate our approach against the ability of dealing with unseen anomalies. This ability is evaluated on the test set, which also contains both data types, including the anomaly type that we removed from the validation set.

Each experiment starts with training the black-box One-Class Classifier (OCC) using the training set (c.f. Layer 1, Figure 5). On top of the training, we perform a model selection via hyperparameter tuning with the validation set, which is missing one anomaly type. Finally, the OCC's anomaly detection performance is evaluated using the test set. The OCC model we used in these experiments is an Isolation Forrest [Liu et al., 2008].

The subsequent explainer procedure of the experiment is visualized within Layer 2 in Figure 5. The specific explainer approach we used is the SHAP framework [Lundberg and Lee, 2017]. The explainer is trained on the validation set, which still is missing one anomaly type, and the OCC's predictions.

The explainer performance is then evaluated on the test set.

In the following section, we describe the construction of the three data sets. Moreover, we name the sample features that we provide for the experiments.

4.3 Data Preprocessing

The preprocessing of the raw time series data is as follows: First, we flatten and aggregate the history in the time series per instance to gain a plain tabular dataset. Within this step, we also generate a specific feature per anomaly type. Second, we sample balanced datasets for training, validation, and testing.

4.3.1 Anomaly Features

We include specific features per anomaly type in the dataset. The features are based on the hand-crafted decision rules which are used to create the ground-truth labels. Therefore, the black-box model should be able to harness these features for detecting anomalies. Moreover, the black-box explainer should identify these features as the most important features.

Below, we list the features for each anomaly type. Each feature is given a name based on its meaning in context of the use case, which is also given.

Anomaly 1

`l_rel_ref_diff`: The relative absolute difference in quantity of the latest stock change compared to the reorder reference

`l_rel_ref_diff_delta`: The difference between the current and the previous relative absolute reference difference

Anomaly 2

`is_order_1`: Whether the second last transaction is an order or a stock change event

Anomaly 3

`g_rel_noise_sgn`: The relative absolute difference in quantity of the latest stock change compared to the moving average of the last n transactions with the same sign

We investigate the use of these features by model and explainer in the evaluation section.

4.3.2 Dataset Generation

For the experiments, we generate three datasets for training, validation, and testing. The distribution over time of the instances in each dataset and per type (normal / anomaly type) is shown in Figure 6.

The datasets are balanced with respect to normal / anomalous data, and each anomaly type is represented equally often. In addition, we respect the time-related dependency of the instances and sample the three datasets with increasing timestamps, i.e., all samples in the validation set have a higher timestamp than in the training set and the test set contains the samples with the highest timestamps. Table 1 contains the exact number of instances per dataset resulting from this sampling strategy.

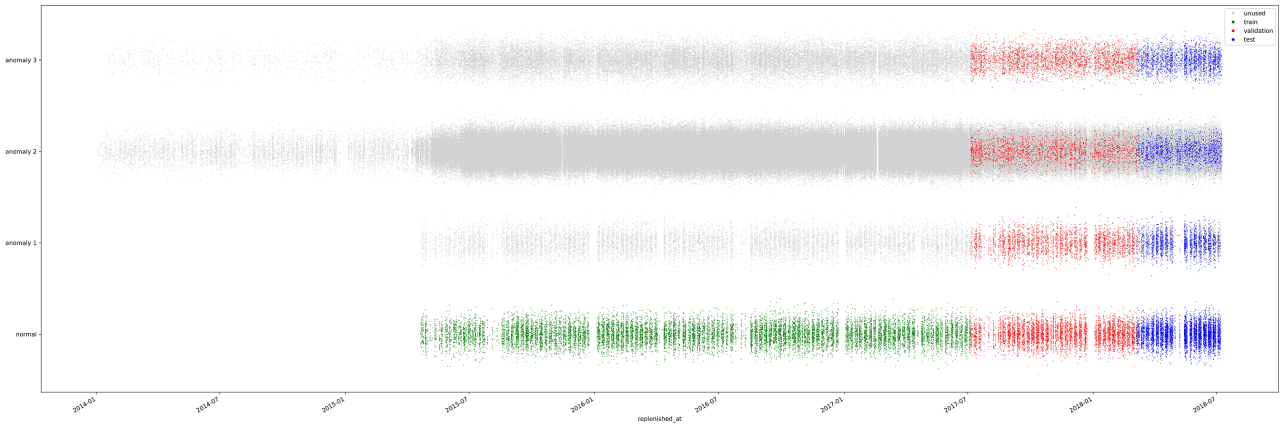


Figure 6: Sample distribution of train, validation, and test set over time. The timestamps of the samples range from 2015 to 2018. Each point represents a data instance to which we add some jitter to the y-axis to visualize data density. The coloring indicates the affiliation to a dataset: Green = train set, red = validation set, blue = test set, and gray = unused.

Table 1: Number of normal instances and number of instances per anomaly type contained in each dataset. Instances are sampled from the smart logistics data.

	Normal	Anomaly 1	Anomaly 2	Anomaly 3
Training	15,761	-	-	-
Validation	5,254	2,627	2,627	2,627
Test	5,254	1,751	1,751	1,751

Table 2: Detection performances of each OCC model (rows) originating from the experiments where a single anomaly type is left out during training. We consider the detection rates per anomaly type and the False Positive Rates (FPR) of the models.

		Detection Performance			
		Anomaly 1	Anomaly 2	Anomaly 3	FPR
Left Out	Anomaly 1	13 %	100 %	96 %	4 %
	Anomaly 2	53 %	100 %	97 %	26 %
	Anomaly 3	53 %	100 %	97 %	26 %

The training dataset only contains normal data, since we perform an anomaly detection task using an OCC model. The validation and test datasets both contain equally many normal data, but we increase the amount of anomalous data in the validation set compared to the test set. This enables us to remove a single anomaly type from the validation set while it remains balanced.

4.4 Results

In this section, we present the results of the experiments that we conducted according to the setup described above. With the experiments, we want to answer the following questions regarding the black-box classifier and the black-box explainer:

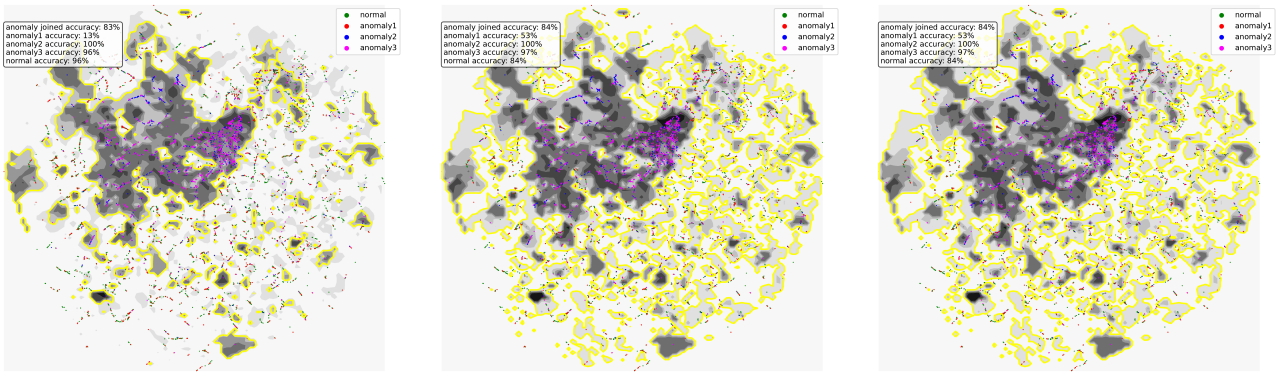
- For the black-box classifier:
  - How well can the classifier detect each of the three anomaly types in general?
  - How well can the classifier detect anomalies of types it has not seen during training?
- For the black-box explainer:
  - How close are the explanations to the hand-crafted rule for each of the three anomaly types in general?
  - How close are the explanations to the hand-crafted rule for anomalies of types the classifier and the explainer have not seen during training?

The questions need to be answered in a given order due to the hierarchical nature of black-box explainer approaches. We evaluate the classifier first since the explainer performance depends on the performance of the classifier, i.e. if the classifier is unable to detect an anomaly well, the explainer will not be able to infer correct explanations.

4.4.1 Classification Performance

We analyze the OCC models with respect to their detection performance for each anomaly type. In addition, we compare the three models that we created with our experiments by leaving each anomaly type out consecutively. Within this analysis, we focus on the models' performances for those anomaly types which have been left out during training.

The resulting detection performances are summarized in Table 2. For the model, which has not seen Anomaly 1 during training (c.f. Table 2, Row 1), we see expected results: The model shows a



(a) Model space w/o Anomaly 1 (b) Model space w/o Anomaly 2 (c) Model space w/o Anomaly 3

Figure 7: Projections of the model spaces created using UMAP [McInnes et al., 2018]. The yellow line represents the decision border of the OCC. Darker areas indicate anomalous space, while lighter areas indicate normal space. Instances of the test set are added as single points to the projection.

strong performance on instances of types Anomaly 2 and Anomaly 3, while it is able to detect only 13 % of unseen Anomaly 1 instances. Furthermore, its False Positive Rate (FPR) is at 4 %, which attests that the decision boundary of the model is placed well for Anomaly 2 and Anomaly 3.

The remaining two models show performances, which differ from the expectation induced by the performance of the first model. Excluding either Anomaly 2 or Anomaly 3 from training does not have any impact on the high detection performance of the model for those anomaly types. Hence, the black-box model is robust against these kinds of anomaly types. By investigating the properties of Anomaly 2 and Anomaly 3, we recognize an extensive overlap in properties of the two anomaly types, i.e. if one of the types is present, the trained model is always able to detect both types. This happens, although different rules are used to manually label each type.

Anomaly 1 is present during training for both models, and we see a significant increase in detection performance compared to the first model. Nevertheless, we expected the performance to be close to 100 % and not at the observed 53 %. Furthermore, we observe an increase in the FPR. From this, we conclude that all models are struggling with detecting Anomaly 1 and that an increase in detection rate for this anomaly type also causes an unwanted increase in FPR.

Finally, we observe that the models trained without Anomaly 2 and Anomaly 3 perform the same. This leads us to the hypothesis that both experiments result in the same model, although each model has been selected using different validation data. The hypothesis can be confirmed when observing the projections of the model spaces shown in Figure 7. The projections for the models trained without Anomaly 2 (cf. Figure 7b) and Anomaly 3 (cf. Figure 7c) are identical, while the projection for the model trained without Anomaly 1 (cf. Figure 7a) shows minor differences. We interpret these differences as changes to the decision boundary between Anomaly 1 and normal instances, since this reflects the differences in performance metrics in Table 2.

4.4.2 Explainer Performance

To complete our case study, we evaluate the performances of the black-box explainers and relate the results to the detection performances of the classifiers. Each explainer is trained using the validation set and the predictions of the OCC model for that data. Like with the black-box model training, we remove a single anomaly type from the validation set and use the classifier that has also not seen

Table 3: Explanation performances of each black-box explainer (rows) originating from the experiments where a single anomaly type is left out during training. We consider the detection rates of the correct feature as the most important feature per anomaly type.

		Explanation Performance		
		Anomaly 1	Anomaly 2	Anomaly 3
Left Out	Anomaly 1	20 %	89 %	0 %
	Anomaly 2	7 %	85 %	0 %
	Anomaly 3	7 %	85 %	0 %

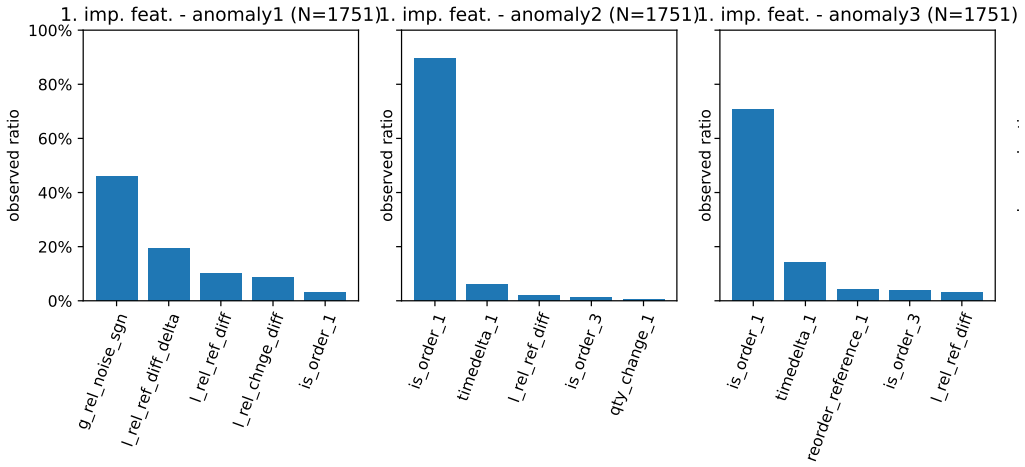
the same anomaly type during training. Therefore, the explainer has no knowledge about the missing anomaly type, either directly nor indirectly.

We measure the performance of each explainer by observing the relative frequency the explainer identifies the specific features that we included for each anomaly type (cf. Section 4.3.1) as the most important feature for the decision of the black-box model. The overall results are shown in Table 3 and Figure 8 gives a more detailed view on the explanations found by the explainers.

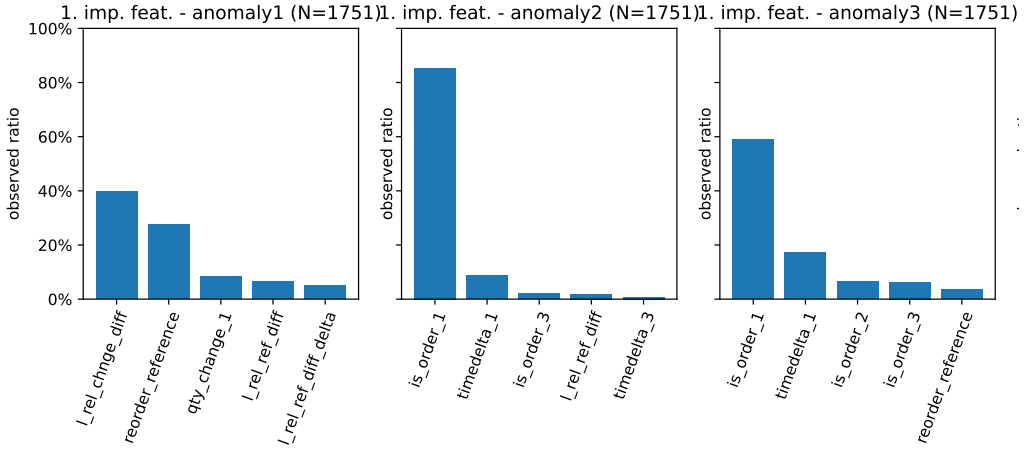
The performance for Anomaly 2 over all experiments (Column 2) correlates with the detection performance of the classifier, although there are some instances that the explainer did not explain correctly. For Anomaly 3, the explainer was not able to identify the according specific feature `g_rel_noise_sgn` as the most important one. Nevertheless, all explainers identifies the specific feature of Anomaly 2 `is_order_1` in 70 % and respectively 59 % of the cases as the most important one (cf. Figure 8). These explanations are also valid, since we discovered an overlap between the properties of Anomaly 2 and Anomaly 3 instances. Therefore, Anomaly 2 and Anomaly 3 are not clearly distinguishable.

The behavior of the explainers for Anomaly 1 is inconsistent with respect to the behavior of the black-box model. When leaving Anomaly 1 out during training (Row 1, Column 1), the explainer performs better than the classifier. This case demonstrates the inconsistencies between black-box explainer and the underlying black-box model, i.e., for some cases the explainer uses the correct explanation to justify a wrong decision of the classifier. In practice, this can increase the distrust against any decision support system.

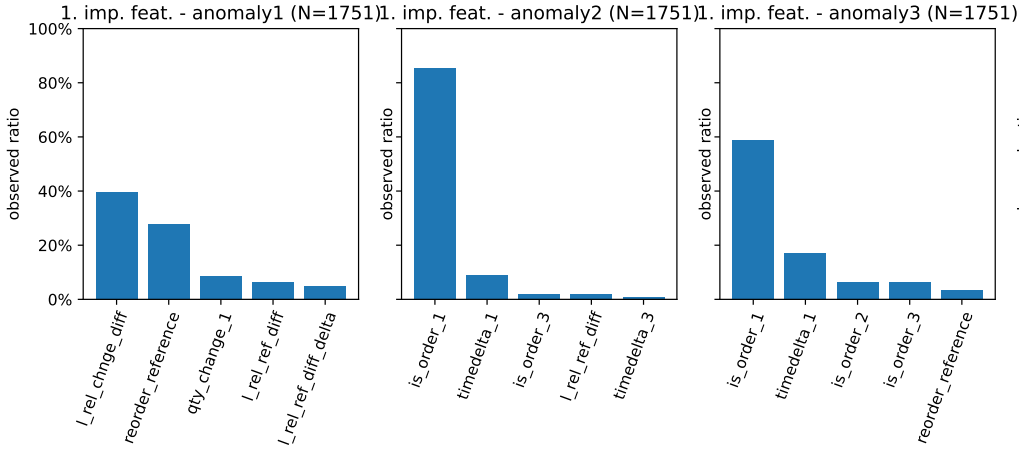
In contrast to the first experiment, the explainers' performances drop significantly for Anomaly 1 when leaving Anomaly 2 and Anomaly 3 out, although the performances of the classifiers increased. This behavior underlines the challenge of finding correct explanations when the underlying model is unable to define a sharp decision boundary.



(a) Most important features when Anomaly 1 is left out.



(b) Most important features when Anomaly 2 is left out.



(c) Most important features when Anomaly 3 is left out.

Figure 8: Top five of the features to be most important for the decision of the black-box classifier as returned by the explainer. The results are given as relative frequencies for the test set per anomaly type from left to right: Anomaly 1 (specific features: `l_rel_ref_diff`, `l_rel_ref_diff_delta`), Anomaly 2 (specific feature: `is_order_1`), and Anomaly 3 (specific feature: `g_rel_noise_sgn`).

5 Simple Verbalization

Since the output of the explanations themselves are not easy to understand at first glance, we have also integrated a simple natural language verbalization for the black box explainer. The path via an ontology and verbalization from WP 4.2 was deliberately not chosen here in order to keep the black box explainer independent and to be able to use it more easily in industrial applications that do not run via the white box model.

With this simple verbalizer, individual data points, i.e., a local explanation for the current prediction, are verbalized. We only explain the anomalies, as the normal state is assumed to be trivial.

If the normal state is recognized for the current data portion, the following announcement is made to the human by verbalization:

```
This data portion shows normal behavior.
```

If the prediction for the data portion yields an anomaly, the output is:

```
This data portion is an anomaly.
```

In addition, further information about the features that led to the anomaly prediction follows. The most important two, respectively the three most important, if the second and third most important features are assigned a similar meaning, are named. Here, the feature is named in combination with the value it had in the current data portion. For example, this will look like this:

```
This data portion is an anomaly.
This was mainly influenced by is_order_1 is 0.0 and timedelta_1 is
32576.000000000004.
Additionally, l_rel_chnge_diff is 2.0.
```

6 White-box explanations for black-box models

Another form of explainability of black-box models is developed as a central result of DAIKIRI via the white-box approach.

First, a black-box model is learned. It creates predictions, for example, on the test data set. These predictions are the positive and negative examples used in the SML approach. In addition, the SML method uses the ontology generated via the white-box approach through vectorization, embedding, clustering and labeling in LabEnt. The SML approach learns about the ontology and the examples of the black-box approach, and can therefore be explained in itself. Here, the white-box approach provides the explanations of the black-box model. In addition, this combination of white and black-box approach has the advantage that the white-box approach is no longer dependent on labeled data if the examples are generated by an unsupervised black-box model.

The pipeline of white-box explanations for black-box models is shown in Figure 9.

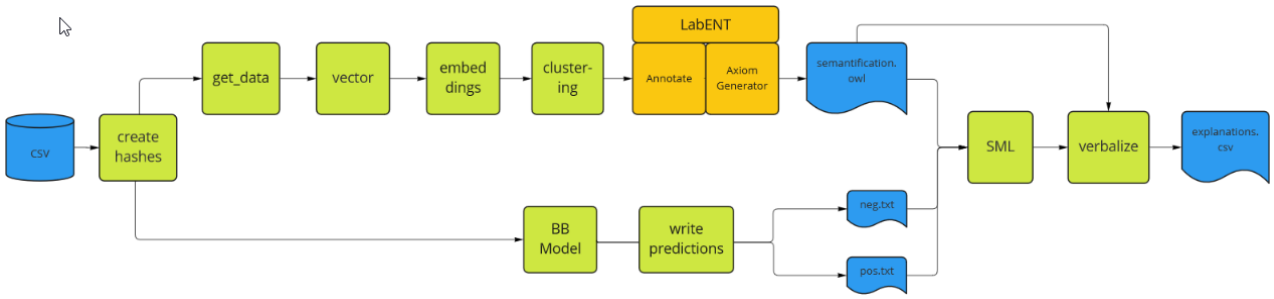


Figure 9: The figure shows a DAIKIRI pipeline with the combination of black-box and white-box approach. Input is a CSV file. For each event, a hash is created over all features of the event. This helps us later to refer to the event i.e. data point. From here, the white-box approach begins at the top. In the get_data, the data (split) is read. After that, the data is vectorized, the embeddings are calculated and is clustering performed. Now comes the human-in-the-loop. In LabEnt, the clusters can be annotated manually. The Axiom Generator then creates the ontology. The black-box approach also starts with the hashed data. A black-box model is learned on the training data set. For the test data set, the predictions are made and the events i.e. hashes of the positive examples are stored in pos.txt. The negative examples are stored in neg.txt. The ontology and these examples are the input for the SML method ontolearn. The results are verbalized and stored in natural language.

7 Discussion and Outlook

At the beginning of this document we gave a short overview how black-box approaches could be interpreted.
In section 3 we generated oscillation data and showed how black box models generate explanations through visualization. So explanations through visualization can be used beyond classical image tasks.
In Section 4, we conducted a case study to exemplarily evaluate the interaction between a black-box model and a post-hoc explainer in a scenario of anomaly detection. We used the anomalies from the Smart Logistics use case and designed three experiments for which we treated each of the three anomaly types as unknown during training of the model and explainer by temporarily removing according data. We discovered an overlap in the properties of two anomaly types from which the model benefits, but makes both types not clearly distinguishable for the explainer. The third anomaly type is hard to detect for the model, which leads to inconsistencies between model and explainer or imprecise explanations in general. In summary, the case study showed potential risks of using a post-hoc explainer to explain a black-box model. To avoid this risks, a white-box approach can be used, but then other possible limitations need to be accepted.
In section 5 we developed a simple verbalizer which converts the outputs of SHAP explainer to natural language.
The meaning of the black-box model for white-box explanation and vice versa the use of the white-box approach to explain the black-box model are discussed in section 6.

References

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

Daniel W Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4): 1059–1086, 2020.

Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In *International Conference on Parallel Problem Solving from Nature*, pages 448–469. Springer, 2020.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008. doi: 10.1109/ICDM.2008.17.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf`.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. URL `https://arxiv.org/abs/1802.03426`.

Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019.

Marco Tulio 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*, pages 1135–1144, 2016.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

N Seemuang, T McLeay, and T Slatter. Using spindle noise to monitor tool wear in a turning process. *The International Journal of Advanced Manufacturing Technology*, 86(9):2781–2790, 2016.

Naeem Seliya, Azadeh Abdollah Zadeh, and Taghi M. Khoshgoftaar. A literature review on one-class classification and its potential applications in big data. *Journal of Big Data*, 8(1), 2021. doi: 10.1186/s40537-021-00514-x.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.