# Why Are You Weird?
# Infusing Interpretability in Isolation Forest for Anomaly Detection

**Nirmal Sobha Kartha,** [1] **Clement Gautrais,** [2] **and Vincent Vercruyssen** [2]
KU Leuven
[1] nirmal.nsk.kartha@gmail.com, [2] firstname.lastname@kuleuven.be

**Abstract**

Anomaly detection is concerned with identifying examples in a dataset that do not conform to the expected behaviour. While a vast amount of anomaly detection algorithms exist, little attention has been paid to explaining *why* these algorithms flag certain examples as anomalies. However, such an explanation could be extremely useful to anyone interpreting the algorithms' output. This paper develops a method to explain the anomaly predictions of the state-of-the-art Isolation Forest anomaly detection algorithm. The method outputs an *explanation vector* that captures how important each attribute of an example is to identifying it as anomalous. A thorough experimental evaluation on both synthetic and real-world datasets shows that our method is more accurate and more efficient than most contemporary state-of-the-art explainability methods.

## 1 Introduction

Anomaly detection is a crucial data mining task as it deals with identifying examples in a dataset that do not conform to the expected behaviour (also called anomalies). Detecting anomalies is essential in mission-critical domains such as network intrusion detection (García-Teodoro et al. 2009), credit scoring and fraud detection (Thiprungsri and Vasarhelyi 2011; Bolton, Hand, and Others 2001), or patient screening and monitoring (Lin et al. 2005).

Anomaly detection algorithms usually work in a completely *unsupervised* manner (Campos et al. 2016). They do not make use of label information to identify anomalous examples in a dataset. Such algorithms work by assigning each example an anomaly score derived from its attributes. High scores indicate anomalousness while low scores indicate normality. By thresholding these scores, we obtain a binary label (anomaly or normal).

While a variety of algorithms exist to compute different types of anomaly scores, surprisingly little attention has been devoted to developing methods that can explain *why* a particular score is assigned to an example. However, such an *anomaly explanation* could be instrumental for domain experts interpreting the algorithms' output. For instance, an explanation of which factors led to flagging a network activity as anomalous, i.e., an intrusion, can help the expert design a better security system. Informally, model explainability refers to the degree to which the human can understand the cause of a model decision (Miller 2019).

The ability to explain the output of an anomaly detection algorithm will also engender trust in its predictions (Molnar 2019). In this paper, we tackle the challenge of explaining the output of the *Isolation Forest* (IFOREST) anomaly detection algorithm (Liu, Ting, and Zhou 2008). Although this algorithm often achieves state-of-the-art performance (Domingues et al. 2018), it is a black-box model with little to no model interpretability. Concretely, we make the following contributions:

- We contribute an algorithm that exploits the structure of the IFOREST method to explain its predictions.
- We conduct an experimental study to evaluate the quality and speed of the generated explanations and show that our method performs on par with the state-of-the-art while being an order of magnitude faster.

## 2 Preliminaries

We briefly describe how the IFOREST algorithm works (Liu, Ting, and Zhou 2008). This algorithm's main assumption is that anomalies are easier to *isolate* from the rest of the data than normals whereby the isolation is achieved by randomly and recursively splitting the data. The IFOREST algorithm has two steps. Given a dataset $X = \{x_1, \ldots, x_n\}$ with each example $x \in \mathbf{R}^d$, it first builds an ensemble of $T$ isolation trees. Each tree recursively splits a random subsample of size $\psi$ of the data using axis-parallel splits until each example ends up in its own leaf node or the tree depth limit is reached. In each split node, the split attribute and split value for that attribute are chosen randomly. Then, it computes an anomaly score $a(x)$ for each example $x$ based on the path lengths of the example traversing each tree and the size of the nodes it ends up in:

$$a(x) = \frac{1}{T} \sum_{\text{trees}} (x \text{ leaf node depth} + \nu(x \text{ leaf node size}))$$
$$\nu(i) = 2H(i) - 2$$

with $H(i)$ the $i^{th}$ harmonic number. The score can be thresholded to derive a binary label for each example.

## 3 Methodology

The problem we are trying to solve in this paper, is:

**Given:** A dataset $X = \{x_1, \ldots, x_n\}$ and an ensemble of $T$ Isolation trees built using this dataset.

**Do:** Generate an explanation vector $w \in \mathbf{R}^d$ for each example $x$ that quantifies how important each attribute is to the predicted label of $x$.

Our explainability method is uniquely designed for the IFOREST anomaly detection algorithm. The key insight behind our approach is that the importance of an attribute to predicting an example as anomalous, correlates with how instrumental that attribute is towards isolating the example. In an isolation tree, an attribute's power to isolate an example is captured by its ability to shorten the expected depth of the branch the example finally lies in. Intuitively, those attributes contribute more to the explanation of an example's predicted label. The next sections present what shortening the expected branch depth means in greater detail.

For a given example, our method outputs a vector $w \in \mathbf{R}^d$ capturing the importance of each attribute in predicting the example's label. The value in $w$ for each attribute represents the average path length shortening obtained by using this attribute. This average is computed over all the trees in the IFOREST. A value close to 0 means that the attribute had little to no influence in making this data point anomalous. A negative value means that this attribute made the point look like a normal point. A positive value means that the attribute had an influence in making the point anomalous.

### 3.1 Intuitions

Our method leverages three intuitions about the structure of the IFOREST algorithm to explain its predictions. First, the split attributes along an example's path in an isolation tree are instrumental in isolating it from the rest of the data. Thus, they are likely to explain the example's predicted label. Second, anomalous examples are more likely to be isolated closer to the root, which is also the core premise of the IFOREST algorithm. This tells us to assign more weight to split attributes involved with larger parent node sizes, when computing the explanation vector. Finally, the most informative split nodes help shorten an example's path length in an isolation tree. This entails that split attributes and split values are more informative if they split the data in that node in an unbalanced manner and if they allocate the example we are explaining to the smaller child node. Our method employs a weighting scheme that takes into account these three intuitions.

### 3.2 Algorithm

Let $x \in \mathbf{R}^d$ be an example for which we want to explain its anomaly score output by the IFOREST. The IFOREST $= \{t_1, t_2, ..., t_T\}$ consists of an ensemble of $T$ isolation trees trained on the full dataset. Algorithm 1 presents the computation of the importance of each attribute to the anomaly score. Its output is the explanation vector $w$. **Line 2** iterates over all trees in the Isolation Forest. **Line 3** initiates the traversal of the isolation tree by starting at the root. **Lines 6 to 11** compute to which child node the example $x$ should go to. It does so by following the data split of the current node. **Line 12** computes the score of the considered split. It depends on both the parent node size and the current node size. Section 3.3 explains this step in more detail. **Line 13** attributes the score of the current node to the split attribute that is used in this node and adds it to the attribute importance vector $w$. After traversing all trees, the resulting vector $w$ is returned. This vector is in fact the explanation vector we set out to compute.

**Algorithm 1:** Computing the attribute importances

**Input:** An IFOREST=$\{t_1, t_2, ..., t_T\}$ and an example $x = [x_1, x_2, \ldots, x_d]$
**Output:** $w$, the attribute importance vector

1 $w = [0, 0, \ldots, 0] \in \mathbf{R}^d$;
2 **for** $t_i \in IF$ **do**
3  node = root($t_i$);
4  **while** *not node.isLeaf()* **do**
5  f = node.$feature$;
6  value = node.$value$;
7  **if** $x[f] < value$ **then**
8  node = node.$leftChild$;
9  **else**
10  node = node.$rightChild$;
11  **end**
12  $score = \log_2\left(\frac{|node.parent|}{|node|}\right) - 1$;
13  $w[f] = w[f] + score$;
14  **end**
15 **end**
16 **return** $w$

### 3.3 Assigning Weights to Node Splits

A key contribution of our paper is the definition of an appropriate weighting scheme to evaluate the value of each node split in an isolation tree to the anomaly score of an example traversing the tree. It is based on the intuitions outlined in Section 3.1. Concretely, the weighting scheme needs to assign a higher value to the split attribute in a node when:

1. The split attribute shortens the path length of an example traversing the isolation tree through an imbalanced split.
2. The split attribute assigns the example to the smaller child node when there is an imbalanced split.
3. The split attribute occurs in a node with a larger node size. These are the nodes closer to the root and they are more likely to isolate anomalous examples.

**Balanced Splits** Before deriving the weight score, let us briefly consider how a *binary search tree* (BST) would split a dataset (Cormen et al. 2009). The authors of the original IFOREST paper also identified the structural equivalence between an isolation tree and a binary search tree (Liu, Ting, and Zhou 2008).

Consider a BST where all splits are balanced, with a dataset size of 256 examples. Each node split results in

two child nodes of equal sizes, until each example gets isolated to its own leaf node of size 1 at a path length of $log_2(256) = 8$. After the first split, it requires an additional $log_2(128) = 7$ splits - or equivalently, $(pathlength - 1)$ to isolate the example from the second level. In other words, every recursive balanced split takes us *1 unit step* closer to the isolation of the example.

**Imbalanced Splits** In an isolation tree, the random selection of both split attribute and split value entail that anomalous examples get isolated at shorter path lengths. Hence, the usefulness of selecting a specific attribute to split on depends on how much it contributes to isolating the example. Hence, our weighting scheme should quantify and capture precisely how much a split reduces the path length of an example compared to its path length in a BST.

An imbalanced split where the anomalous point gets allocated to the child node with lower size, should assist by *more than 1 unit step* to this example's isolation. For a given example $x$, we compute the weight as:

$$score = \log_2 \left( \frac{|node(x).parent|}{|node(x)|} \right) - 1 \qquad (1)$$

where $node(x)$ is the node example $x$ ends up in after the split. We subtract 1 because this is simply the value obtained when the split is perfectly balanced:

$$\log_2 \left( \frac{|node(x).parent|}{\frac{|node(x).parent|}{2}} \right) = \log_2(2) = 1$$

This weighting scheme rewards imbalanced splits that allocate the anomalous point in question to the smaller child node. A perfectly balanced split (for instance $\{|node.parent| = 256 \Rightarrow |node| = 128\}$) gets a null reward: $\log_2 \frac{256}{128} - 1 = 0$. The worst possible split (for instance $\{|node.parent| = 256 \Rightarrow |node| = 255\}$) gets the lowest weight value of $log_2 \frac{256}{255} - 1 = -0.99$. The best split (for instance, $\{|node.parent| = 256 \Rightarrow |node| = 1\}$) gets the highest weight value $log_2 \frac{256}{1} - 1 = 7$. It is clear that the score of Equation 1 varies between $-1$ and 7.

### 3.4 Time Complexity

Our method iterates over every tree in an IFOREST. For each tree, it follows the path leading to the example $x$, and computes the ratio between each node size and its child size. As described in the original IFOREST paper, the expected length of that path is $\log_2(|S|)$, with $|S|$ the sample size that is used to fit every tree. Hence, the number of operations is in $O(\log_2(|S|) * T)$. Our method is linear in the number of trees, and logarithmic with respect to the sample size.

## 4 Related Work

Model explainability can be *global* or *local*. Global explainability methods seek to demystify the entire machine learning model, while local methods only aim to explain a model's individual predictions. We are interested in local explainability as we seek to generate an explanation for the anomaly score of each example separately. Local explainability methods can either be model-agnostic or designed on a model-by-model basis.

### 4.1 Model-Agnostic Methods

Two popular model-agnostic methods are LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Shapley values) (Lundberg and Lee 2017). LIME uses locally interpretable white-box surrogate models to approximate the decision boundaries around an example of interest. It can explain the output of different classifiers, including - theoretically at least - anomaly detectors. However, LIME often produces unstable explanations that are sensitive to the chosen hyperparameters (Alvarez-Melis and Jaakkola 2018).

SHAP provides both local and global explanations. It assigns a score to each attribute of an example by looking at how the prediction for this example changes when considering only a subset of the attributes. As such, SHAP is able to compute the impact of each feature on a prediction. Although it is often consistent with human explanations (Lundberg and Lee 2017), it suffers from instability (Alvarez-Melis and Jaakkola 2018). In contrast to both SHAP and LIME, our method uses the trees learned by the IFOREST to provide the explanations. Our method does not probe the IFOREST by either sampling similar data points or masking features, but uses the model that is actually learned to provide explanations.

### 4.2 Methods Specific to Anomaly Detection

Few explainability methods for anomaly detection algorithms exist. Most related to our work is the post-hoc DIFFI method (Carletti, Terzi, and Susto 2020). It computes the importance of each attribute in the IFOREST algorithm based on the tree depth. In contrast to DIFFI, our method also takes into account the imbalance of a node split in each tree of the IFOREST ensemble when deriving an explanation.

In local-DIFFI, the attribute importances are also computed in an additive manner by assigning a weight to each split node. However, the node weight is computed as $\frac{1}{h_{t_i}(x)} - \frac{1}{h_{max}}$ where $h_{t_i}(x)$ represents the height of the leaf node of $x$, and $h_{max}$ is the expected leaf depth of balanced splits: $\log_2(|root(t_i)|)$. This weighting scheme therefore assigns a constant weight to all nodes on a given path, based on the depth of that path. This contrasts with our method which assigns a weight to each node based on the imbalance of this specific node. In other words, the local-DIFFI weighting scheme does not capture the intuition that split nodes that shorten the path length of an example should be rewarded. Our method does take into account the contribution of each node to the path length of an example. This is important because an attribute that is instrumental to isolating the example probably also provides an explanation why the example is anomalous.

## 5 Experiments

In this section, we try to answer the following questions:

**Q1:** Does our method provide accurate local explanations for examples flagged as anomalies?

**Q2:** Is our method more efficient than comparable state-of-the-art methods?

**Q3:** What are the key differences between local-DIFFI and our method?

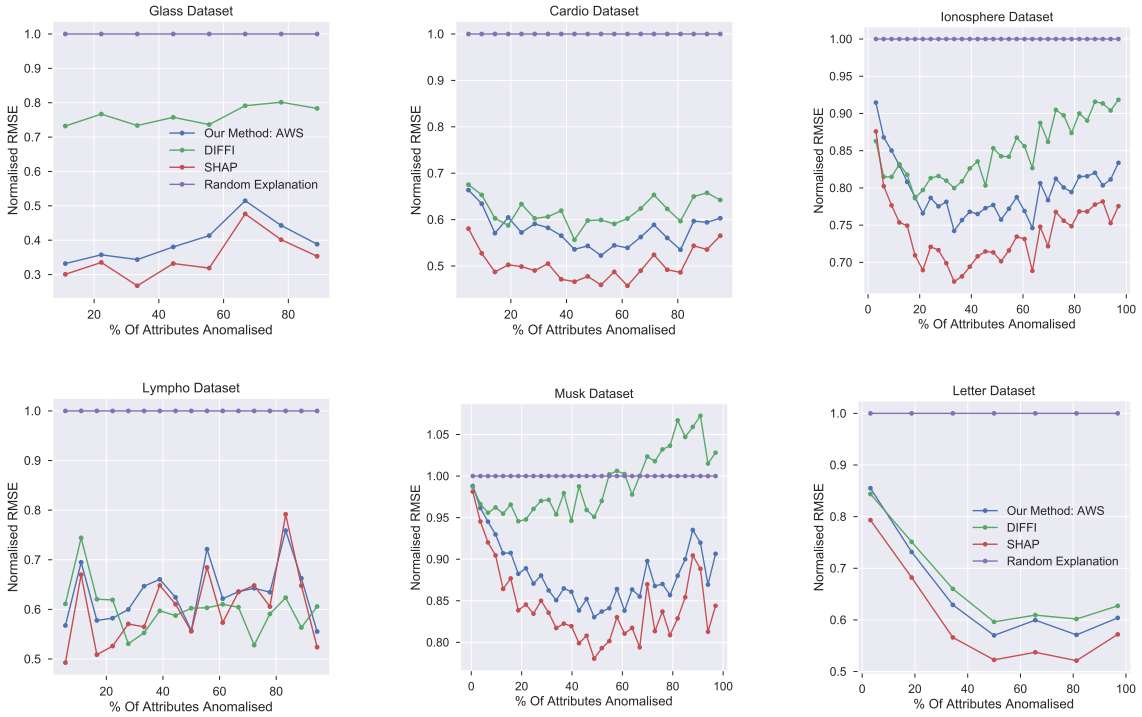



Figure 1: The plots show the evolution of the RMSE against the percentage ($m$) of *anomalized* attributes for each of the six real-world datasets. RMSE are normalized using the random baseline. The plots clearly show that our method performs better than or on par with DIFFI for each dataset. It also matches the performance of SHAP (or is very close to it) in each dataset.

**Methods** We compare the following methods: the model-agnostic local SHAP values (Lundberg and Lee 2017), the local-DIFFI method designed for the IFOREST algorithm (Carletti, Terzi, and Susto 2020), and our method as described in Section 3. The methods do not have specific hyperparameters to set. Each method explains the output of the same constructed IFOREST with $T = 100$ and $\psi = 256$.

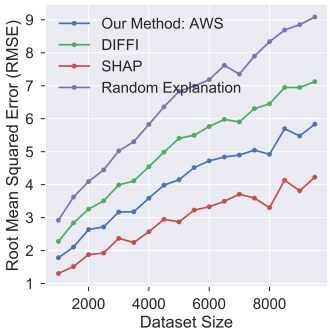
**Data** We run our experiments on both synthetic and real-world data.[1] The synthetic data consist of multiple settings of normally distributed point clusters wherein we systematically vary dataset size (from 1,000 to 10,000), dimensionality (from 2 to 50), and the proportion of attributes randomly chosen examples are made anomalous across (till 100%). The six real-world data are listed in Table 1. For the sake of consistency, these are the same datasets as used in the DIFFI paper (Carletti, Terzi, and Susto 2020).

**Experimental Setup** The fundamental design issue with explainability experiments is the *absence of a ground truth* to evaluate the predictions of the methods. Hence, we design the following procedure that simulates a ground-truth. Given a datasets consisting only of normal examples and an IFOREST trained on these data, we randomly pick $n$ examples with replacement and for each example: (1) obtain an *initial* explanation vector $w_0$, (2) randomly select a coalition of $m\%$ of the example's attributes, (3) for each of those, change its value to the $3*$max value of that particular attribute's full range over all the data, (4) obtain a *new* explanation vector $w_{new}$ for the modified example, (5) compute the change between the initial and new explanations as $w = w_0 - w_{new}$ and normalize $w$. We now expect the change in explanation to occur only for the coalition of attributes that we changed, captured by the expected explanation vector $w_e$. This allows us to compute the *root mean squared error* (RMSE) between vectors $w$ and $w_e$. For instance, if we change the value of only one attribute in steps (2) and (3), we would expect $100\%$ of the change in the explanation to occur for this particular attribute, reflected in vector $w$. Finally, we average the errors over all $n$ randomly picked examples. We also compute the RMSE for a random explanation vector.

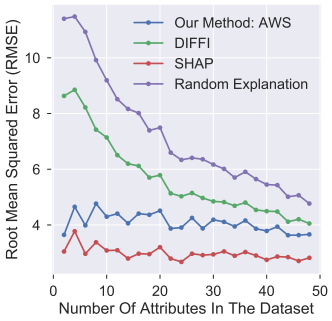
| Dataset ID | Nr. of examples | Nr. of features |
|---|---|---|
| glass | 214 | 10 |
| cardio | 1831 | 21 |
| ionosphere | 351 | 33 |
| lympho | 148 | 18 |
| musk | 3062 | 166 |
| letter | 1600 | 32 |

Table 1: Six real-world datasets from the UCI repository (Dua and Graff 2017).

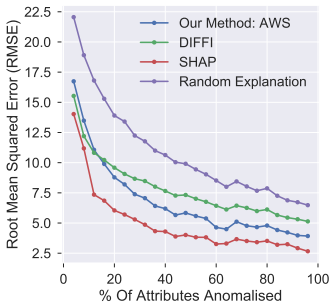
[1] A full implementation of the experiments and methods is provided online: https://github.com/karthajee/AWS-IF-interpretability




(a) RMSE vs dataset size.




(b) RMSE vs dimensionality.




(c) RMSE vs % anomalized attr.

Figure 2: The plots show the results of the experiments on the synthetic data.

| | Execution time per example (sec) with $m = 10\%$ | | | | | | Execution time per example (sec) with $m = 100\%$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | glass | cardio | iono | lympho | musk | letter | glass | cardio | iono | lympho | musk | letter |
| **Our method** | **0.027** | 0.040 | 0.041 | **0.040** | 0.040 | **0.029** | **0.026** | 0.039 | 0.040 | **0.039** | 0.039 | **0.028** |
| DIFFI | 0.029 | **0.037** | **0.038** | 0.045 | **0.039** | 0.030 | 0.030 | **0.037** | **0.038** | 0.040 | **0.037** | 0.029 |
| SHAP | 0.255 | 0.213 | 0.230 | 0.305 | 0.244 | 0.274 | 0.269 | 0.196 | 0.243 | 0.264 | 0.216 | 0.251 |

Table 2: The table shows the execution time per example for different percentages of *anomalized* attributes for each of the six real-world datasets. Our method is an order of magnitude faster than SHAP. It performs on par with DIFFI. The percentage of anomalized attributes does not influence the execution time (i.e., the inference of an example's explanation vector).

## 5.1 Experimental Results

**Q1: Accuracy of the Explanation Methods** In this experiment, we evaluate the accuracy of our explanation method. The latter is defined as the RMSE between the expected explanation $w_e$ and the provided explanation $w$. For instance, if an example is *anomalized* among $k$ attributes, its ideal explanation vector has value $\frac{1}{k}$ for each of these $k$ attributes, and 0 for the others. Figure 1 shows for each of the six real-world datasets how the RMSE values of the compared methods change as a function of $m$. The RMSE values are normalized using the performance of the random baseline. Our method performs better or equal to local-DIFFI on each dataset. Its performance is similar to SHAP on each dataset.

We conduct experiments on synthetic data to measure the impact on accuracy of systematically varying: the dataset size, the dataset dimensionality, and the percentage of anomalized attributes. Figure 2(a) shows the RMSE of the different methods for a varying dataset size. An increase in dataset size increases the RMSE of each method in a similar fashion. Figure 2(b) shows the RMSE of the different methods for a varying number of attributes. In this experiment, $m = 100\%$. The performance of our method and SHAP are not affected by varying dimensionality, in contrast to local-DIFFI. Figure 2(c) shows the RMSE of the different methods for a varying number of anomalized attributes. In both Figure 2(b) and Figure 2(c), the RMSE is decreasing when the number of anomalized attributes is decreasing. The computed RMSE is $||w_e - w||_2 = \sqrt{\sum_{i=1}^{d}(w_e(d) - w(d))^2} = \sqrt{\sum_{i=1}^{d}(w_e(d)^2 + w(d)^2 - 2 * w_e(d) * w(d))}$. When the number of anomalized attributes increases, the number of non zero elements in $w_e$ increases, hence it is more likely than $2 * w_e(d) * w(d)$ is non zero. Moreover, recall that $w_e$ has value $\frac{1}{k}$ for each anomalized attribute, so $\sum_{i=1}^{d} w_e(d)^2 = k * \frac{1}{k^2} = \frac{1}{k}$. Hence, we have RMSE$= \sqrt{\frac{1}{k} + \sum_{i=1}^{d}(w(d)^2 - 2 * w_e(d) * w(d))}$ so the RMSE is expected to resemble the inverse function with respect to the number of anomalized attributes.

**Q2: Efficiency of the Explanation Methods** Table 2 shows the execution time during inference (i.e., computing the explanation vector for an example) for each method on the six datasets for different percentages of anomalized attributes. As both local-DIFFI and our method use similar approaches, it is expected that their execution times are comparable. As noted in Section 3.4, the complexity of our method depends on the number of trees and the sample size. As both of these variables are kept constant, the execution time of our method also remains constant. Because it exploits the structure of the IFOREST, our method is an order of magnitude faster than SHAP.

**Q3: Key differences with local-DIFFI** As explained in Section 4.2, our method differs from local-DIFFI by defining a new scheme to weight node splits. In local-DIFFI, all nodes along a path are allocated a constant weight depending solely on the depth of that path. Our method instead rewards each node based on the imbalance it created towards isolating the considered example. While both methods reward splits that lead to shorter path lengths, our method treats node individually. We illustrate in a toy example how this key difference impacts the explanation provided by both methods, and argue that our method addresses a shortcom-

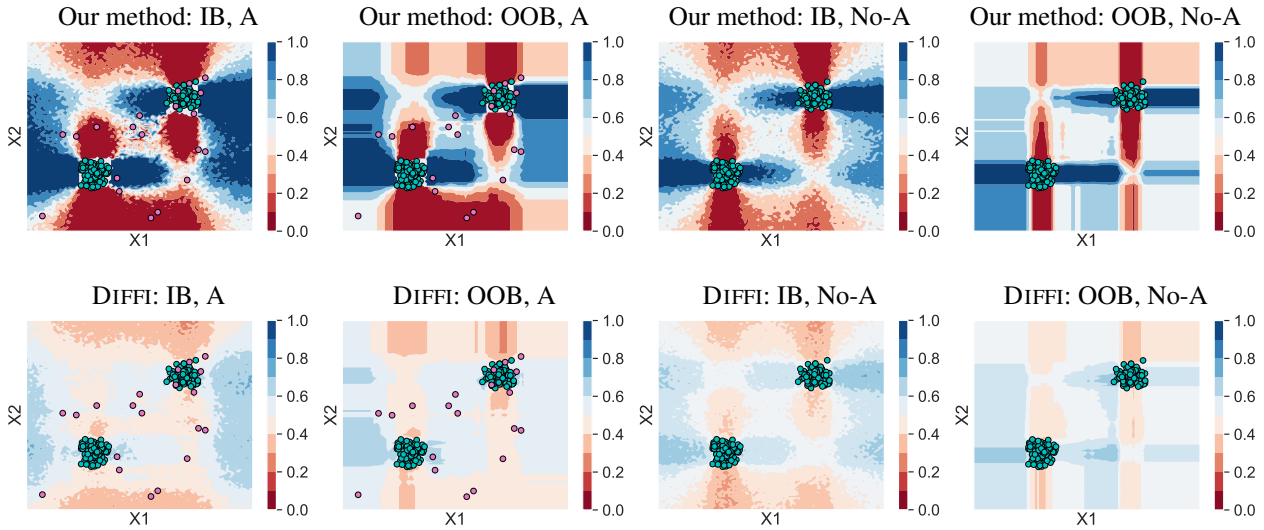



Figure 3: Heatmaps representing the contribution of the each attribute to the anomaly score (blue = $X1$ and red = $X2$). For each plot, one cluster lies on the bottom left and the other in the top right quadrant of the plot. Explanation vectors are normalised so their values sum to 1. The first row depicts heatmaps of our method, the second row depicts heatmaps of local-DIFFI. When the IFOREST is retrained for each heatmap point, we say that the setting is *in-bag* (IB), as the data point is part of the trained IFOREST. When theIFOREST is not retrained, the setting is said to be *out-of-bag* (OOB). When there are anomalies in the training data, the setting is *with anomalies* (A). When there are no anomalies in the training data, the setting is said to be *no anomalies* (No-A). There are therefore a total of 4 settings for each method.

ing of local-DIFFI weighting scheme.

The toy example is composed of a 2D two-cluster configuration and is depicted in Figure 3. The heatmaps depict the contribution of each attribute ($X1$ and $X2$) to the anomaly score. Blue means $X1$ explains the score, while red means that $X2$ explains the score. We consider four different settings when computing the heatmaps. First, the IFOREST can be retrained for every point of the heatmap, so that the explained example is part of the training data. This is the *in-bag* (IB) setting. Second, if the IFOREST is not retrained on the example, the setting is *out-of-bag* (OOB). Third, the training data can contain anomalies, this setting is called *with anomalies* (A), or fourth, it can consist of solely normal points, this setting is called *no anomalies* (No-A). The last two settings evaluate how the contribution of $X1$ changes when the decision boundary of the IFOREST also changes.

Figure 3 clearly illustrates that our method produces explanations that reflect the real contribution of each attribute, while local-DIFFI tends to produce explanations that favor an even contribution of both attributes. For example, looking at the plots in the first row, the points located at the right edge of the heatmap are mostly anomalous with respect to $X1$. Our method outputs a contribution close to 1 for $X1$, while local-DIFFI outputs values close to 0.6, indicating that $X1$ is only slightly more important than $X2$ to explain the anomaly score. We argue that local-DIFFI is able to correctly rank attributes by their importance, but our method is also able to assign correct values to the attributes' contributions.

To compare the impact of training with or without anomalies, we can compare the first and third columns of Figure 3 (IB) or the second and fourth columns (OOB). When using anomalies in the training data, our method accurately identifies the contribution of each attribute over the whole data space. When no anomalies are in the training data, the explanation is correct mostly where the clusters are located. This is expected as without anomalies, the IFOREST does not describe the space outside of the clusters, so it provides no information about the attributes' contributions.

Finally, when the IFOREST is not retrained on the example to explain (OOB), our method outputs contributions that are closer to an even split between both attributes. This is expected as the IFOREST has less information about which attributes are relevant to isolate this particular point. This effect is more visible when the training data contain no anomalies (third columns against forth column in Figure 3) than when the training data contains anomalies (first column against second column of Figure 3). Training on anomalies provides some information about what attribute is relevant to isolate examples in some parts of the data space.

## 6 Conclusion and Future Work

This paper presents a method to explain the predictions of the IFOREST. It employs a weighting scheme that evaluates how much each attribute of an example contributed to its isolation, forming the basis of the explanation vector for that example. Our method performs on par with the state-of-the-art SHAP method, but is an order of magnitude faster. For future work, we plan to extend our method to global model interpretability, and provide more precise explanations by considering attribute values that make a point anomalous.

## References


Alvarez-Melis, D.; and Jaakkola, T. S. 2018. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049* .

Bolton, R. J.; Hand, D. J.; and Others. 2001. Unsupervised profiling methods for fraud detection. *Credit scoring and credit control VII* 235–255.

Campos, G. O.; Zimek, A.; Sander, J.; Campello, R. J.; Micenková, B.; Schubert, E.; Assent, I.; and Houle, M. E. 2016. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data mining and knowledge discovery* 30(4): 891–927.

Carletti, M.; Terzi, M.; and Susto, G. A. 2020. Interpretable Anomaly Detection with DIFFI: Depth-based Feature Importance for the Isolation Forest .

Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; and Stein, C. 2009. *Introduction to algorithms*. MIT press.

Domingues, R.; Filippone, M.; Michiardi, P.; and Zouaoui, J. 2018. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition* 74: 406–421.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository. URL http://archive.ics.uci.edu/ml.

García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; and Vázquez, E. 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* 28(1): 18–28.

Lin, J.; Keogh, E.; Ada Fu; and Van Herle, H. 2005. Approximations to magic: finding unusual medical time series. In *18th IEEE Symposium on Computer-Based Medical Systems (CBMS'05)*, 329–334.

Liu, F. T.; Ting, K. M.; and Zhou, Z. 2008. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*, 413–422.

Lundberg, S. M.; and Lee, S.-I. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*, 4765–4774.

Miller, T. 2019. Explanation in artificial intelligence: Insights from the social sciences.

Molnar, C. 2019. *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Model-Agnostic Interpretability of Machine Learning .

Thiprungsri, S.; and Vasarhelyi, M. A. 2011. Cluster Analysis for Anomaly Detection in Accounting Data: An Audit Approach. *International Journal of Digital Accounting Research* 11.