Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis

Mattia Carletti[1], Chiara Masiero[2], Alessandro Beghi[1] and Gian Antonio Susto[1*]

Abstract— **In the past recent years, Machine Learning methodologies have been applied in countless application areas. In particular, they play a key role in enabling Industry 4.0. However, one of the main obstacles to the diffusion of Machine Learning-based applications is related to the lack of interpretability of most of these methods. In this work, we propose an approach for defining a 'feature importance' in Anomaly Detection problems. Anomaly Detection is an important Machine Learning task that has an enormous applicability in industrial scenarios. Indeed, it is extremely relevant for the purpose of quality monitoring. Moreover, it is often the first step towards the design of a Machine Learning-based smart monitoring solution because Anomaly Detection can be implemented without the need of labelled data. The proposed feature importance evaluation approach is designed for Isolation Forest, one of the most commonly used algorithm for Anomaly Detection. The efficacy of the proposed method is tested on synthetic and real industrial datasets.**

- Keywords: Anomaly Detection, Industry 4.0, Interpretability, Isolation Forest, Machine Learning

I. INTRODUCTION

In the past recent years, Machine Learning (ML) approaches have been successfully applied to many scientific and technological areas [1]. Nevertheless, there are still limitations that slow down the widespread diffusion of ML-based technologies in many environments, like industry and manufacturing. The present work is inspired by two important obstacles that are still in the way of ML adoption in real environments:

- the lack of reliable tagged data in many scenarios;
- the need for interpretability in many applications.

Motivated by the first item, unsupervised approaches are gaining an increased attention. In particular, ML-based Anomaly Detection (AD) methodologies [2], [3] have seen a widespread diffusion in many industrial [4], [5] and non-industrial [6], [7] applications. AD aims at identifying behaviors that differ significantly from the most common ones in the historical data regarding the system under investigation; such goal is extremely relevant for quality purposes in industrial environment and to enable smart monitoring policies on production/process/machine/sub-system level. Moreover, AD methods naturally allow to deal with multivariate data with non-gaussian, multimodal distributions, overcoming the limitations of classical Statistical Process Control approaches like control charts.

AD methods typically define a so-called *Anomaly Score* (AS), a quantitative index associated with the degree of 'outlierness' of the observation under exam; the AS can be exploited in industrial environment as a sort of health factor [8] of the monitored system and monitoring policies based on thresholds on the AS can be in place: when the AS overcomes a certain threshold, inspections or other monitoring actions can be triggered. However, the main issue in relying on such approach is that Root Cause Analysis is non-trivial. AD methodologies do not provide a simple way to indicate which are the causes of variations in AS, leaving the human operator without guidance in how to react after a threshold crossing have happened. This is a typical case where the need of interpretability (listed above as the second obstacle to ML diffusion) is of paramount importance to make decisions based on a ML module.

Isolation Forest (IF) [9] is a popular AD algorithm that exploits a so-called isolation procedure: an iterative method that aims at defining a set of rules that partition the feature space in a subspace where only an observation amongst the historical data is present; the underlying idea of IF is that outliers should correspond to points in the feature space that can be isolated easier than inliers (i.e. 'normal' observations). The isolation procedure defines a tree-like model of decisions, called *Isolation Tree*. Each node in the tree is linked with a variable, and its children, if present, are determined based on a splitting value. Many Isolation Trees are computed in an IF selecting variables and values for splitting at random, making the IF an ensemble method. AS associated to an observation is computed in IF by evaluating the mean path length of such observation on the various Isolation Trees. In order to obtain the predicted binary labels ('0' for predicted inliers, '1' for predicted outliers) a thresholding operation is performed on the AS associated to all observations. The choice of the threshold value is based on the expected fraction of outliers in the dataset (usually referred to as *contamination*). This approach exhibits state-of-the-art detection performance [4], [5], [7], while being very simple to train and requiring little or even no preprocessing. Given these advantages and the widespread adoption of IF in AD tasks, we have decided to focus on such algorithm. Unfortunately, as typical of AD methods, IF provides no simple results interpretation. In this work, we propose a model-based approach, called *Depth-based*

* Corresponding author: sustogia@dei.unipd.it

[1] M. Carletti, A. Beghi, G.A. Susto are with the Department of Information Engineering at University of Padova, Padova, Italy.

[2] C. Masiero is with Statwolf Data Science SRL, Padova, Italy.

This work has been supported by the Regione Veneto project PreMANI (MANIFATTURA PREDITTIVA: progettazione, sviluppo e implementazione di soluzioni di Digital Manufacturing per la previsione della Qualità e la Manutenzione Intelligente) and the Regione Veneto project ADAPT4.0 (Svilupppo di un sistema adattativo per il riconoscimento di guasti nell'Industria 4.0). We would also like to thank Nvidia Corporation for supporting our research with a Nvidia Titan V GPU.

Isolation Forest Feature Importance (DIFFI), to evaluate feature importance for IF and therefore enabling simple Root Cause Analysis. While a broad dissertation on IF is outside the scope of this paper (we refer the interested readers to [9]), we underline that the inherent structure of IF is governed by random choices, making IF one of the most difficult AD method to be interpreted.

The rest of the paper is organized as follows: Section II is dedicated to review the approaches to define a feature importance in AD algorithms, while in Section III DIFFI is presented. Section IV and V are respectively dedicated to illustrate the datasets (synthetic and industrial) and the experimental results that show the effectiveness of DIFFI. Finally, conclusive remarks and future research directions are outlined in Section VI.

II. LITERATURE REVIEW

The lack of model-specific methods for feature importance computation for IF, and, in general, for many AD algorithms, leads to the exploitation of alternative solutions for determining the contribution of each feature to the detection of anomalies. We identify two major paradigms: i) solutions exploiting feature selection techniques; ii) model-agnostic interpretation methods.

As regards *feature selection techniques*, it is worth highlighting that these are typically used for reducing the dimensionality of data before model training, by keeping only features that are relevant for the task at hand. Feature selection techniques include, among others, wrapper methods [10], which consider alternative subsets of features and rely on the model performance as a metric to evaluate their relevance, and filter methods [10], which filter out irrelevant features as a preprocessing step before training the model. Wrapper methods in general provide the best subset of features for the considered model (and task), at the price of a high computational cost. Filter methods instead are faster to compute but they are independent of the employed algorithm and may thus select features that are not useful for the task at hand; this is generally problematic for AD, since it has been shown that features that are considered informative for supervised tasks are typically not relevant for AD [4], [11].

Model-agnostic interpretation methods extract post-hoc explanations by treating the model under examination as a black-box. Despite their flexibility, such methods are designed for supervised tasks and their adaptation to AD is not always straightforward. Moreover, because of their model-agnostic nature they are not informative about the intrinsic functioning of a specific AD algorithm like IF. When dealing with feature importance evaluation, we distinguish between: (i) *local methods*, which consider one instance at a time and provide an explanation on how each feature affects the associated predicted output; (ii) *global methods*, which consider all instances and give a statement of the global behaviour of the model. Examples of local methods are Individual Conditional Expectation plots [12], Local Interpretable Model-agnostic Explanations (LIME) [13] and SHapley Additive exPlanations (SHAP) [14]. Global methods include Partial Dependence Plots (PDP) [15], Accumulated Local Effects (ALE) plots [16] and Permutation-based Importance (PIMP) [17] (based on the measure of variable importance introduced in [18] for Random Forests). PIMP method is based on a simple procedure composed of two steps, which are repeated for each feature representing the observations. Given a design matrix, where each row is an observation and each column represents a feature, we first permute the column associated with the selected feature and then assess the degradation of prediction performance. Features exhibiting the highest performance degradation due to permutation will likely be the most relevant. Due to its simplicity and easy interpretation, we decided to consider PIMP as benchmark method in this work.

III. PROPOSED APPROACH

We propose a model-based approach for feature importance evaluation that allows users to get deeper insights into the outcome provided by IF. To the best of our knowledge, this is the first contribution in the field of interpretability applied to Isolation Forest algorithm. We remark that (i) the proposed approach is completely unsupervised; (ii) it does not require any modification in the original IF procedure, but the access to the structure of resulting Isolation Trees instead. An unsupervised feature importance evaluation method is pivotal in the context of AD, in particular in the scenario of Industry 4.0. As said, the development of a reliable AD system is often the very first step towards the design of a complete smart monitoring solution, and typically no (or very few) labels are available in the early stages of the process. Indeed, results of the AD stage may be useful also to steer the design of a supervised dataset that can be used in the subsequent steps of the development of a smart monitoring solution, in an active-learning fashion [19].
Before introducing DIFFI, we provide some notations and we recall the basic functioning of IF. Let consider a dataset $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ of N samples, where each sample is represented by a d-dimensional feature vector i.e.

$$\mathbf{x}_i = [f_{i,1}, \dots, f_{i,d}]\,.$$

We perform AD with IF and we let $\hat{y}_i$ be the predicted label for the i-th sample $\mathbf{x}_i$. If the IF predicts $\mathbf{x}_i$ as an inlier, then $\hat{y}_i = 0$, while if $\mathbf{x}_i$ is predicted as an outlier, then $\hat{y}_i = 1$. We split the dataset $\mathcal{D}$ according to the label predicted by the IF, obtaining two subsets:

- predicted inliers $\mathcal{D}_I = \{\mathbf{x}_i \in \mathcal{D} \,|\, \hat{y}_i = 0\}$;
- predicted outliers $\mathcal{D}_O = \{\mathbf{x}_i \in \mathcal{D} \,|\, \hat{y}_i = 1\}$.

Once the IF has been trained, we represent it as a collection of Isolation Trees $\{t_1, \dots, t_T\}$. We denote with $\mathcal{V}_t$ the set of nodes in tree t and we recall that, at each non-leaf node of the

tree, a split of the samples is performed based on the value assumed by a randomly selected feature. For example, at the root node we randomly select a feature, we randomly select a split threshold between the minimum and the maximum value assumed by that feature and we perform a test: the samples for which the selected feature value is less than the split threshold are associated to the left child of the root node, all other samples are associated with the right child. The same procedure is then repeated for each non-leaf node.
For the computation of the feature importance, for each non-leaf node of the tree we introduce a quantity that represents the ability of the node to isolate samples. We will refer to such quantity as *induced imbalance coefficient* and we will denote it with λ. Nodes providing a high imbalance between the number of samples in the left child and those in the right child will be assigned a high coefficient, while nodes providing a low imbalance will receive a low coefficient. Nodes whose associated test gives no improvement to the isolation of samples (i.e. all the samples are assigned either to the left or right child) will receive the lowest coefficient. On the contrary, nodes whose associated test isolates a sample will receive the highest coefficient.
For a generic (non-leaf) node v of the tree t, we indicate with $n(v)$ the number of samples in v, with $n_l(v)$ the number of samples in its left child and with $n_r(v)$ the number of samples in its right child. The corresponding induced imbalance coefficient $\lambda(v)$ is obtained as follows

$$\lambda(v) = \begin{cases} \epsilon, & \text{if } n_l(v) = 0 \text{ or } n_r(v) = 0 \\ 1, & \text{if } n_l(v) = 1 \text{ or } n_r(v) = 1 \\ \tilde{\lambda}(v), & \text{otherwise} \end{cases} \tag{1}$$

where

$$\tilde{\lambda}(v) = \frac{\max(n_l(v), n_r(v))}{n(v)} \tag{2}$$

and $\epsilon \ll 1$ is used in place of 0 for numerical stability. For each sample $\mathbf{x}_i \in \mathcal{D}$, we denote with:

- $d_{\text{leaf}}(i,t)$ the depth of the leaf node associated with sample $\mathbf{x}_i$ in tree t.
- Path$(i,t) \subset \mathcal{V}_t$ the path from root to leaf associated with sample $\mathbf{x}_i$ in tree t.

To conclude, we indicate with Feat(v) the feature associated to the generic node v.
Now we briefly describe the rationale behind this method with reference to a single tree t, a single sample marked as outlier $\mathbf{x}^O \in \mathcal{D}_O$ and a single sample marked as inlier $\mathbf{x}^I \in \mathcal{D}_I$. In general, we would define a feature as 'important' according to its ability to isolate samples, meaning that when it is in the path Path(i,t) associated to $\mathbf{x}_i$, the depth of the corresponding leaf node (where $\mathbf{x}_i$ ends up) $d_{leaf}(i,t)$ is small (i.e. the leaf node is as close as possible to the root node). Since we are interested in solving the AD task, we look for features that are able to isolate only outliers and unable to isolate inliers. In other words, we mark a feature as 'important' for AD if it induces isolation of outliers at small depths and, at the same time, does not contribute to the isolation of inliers.

Algorithm 1 DIFFI algorithm.

function INDUCEDIMBALANCECOEFF($t, \mathcal{S}$)
  for node v in t **do**
    Compute $\lambda(v)$ as in Eq. (1), using samples in $\mathcal{S}$
    $\Lambda[v] \leftarrow \lambda(v)$
  end for
  return Λ
end function

Initialize counter and cumulative importance variables:
for $j = 1, \dots, d$ **do**
  $C_I(f_j), I_I(f_j), C_O(f_j), I_O(f_j) \leftarrow 0$
end for
Update counter and cumulative importance variables:
for Isolation Tree t in IF **do**
  $\Lambda_I \leftarrow$ INDUCEDIMBALANCECOEFF$(t, \mathcal{D}_I)$
  $\Lambda_O \leftarrow$ INDUCEDIMBALANCECOEFF$(t, \mathcal{D}_O)$
  for $\mathbf{x}_i \in \mathcal{D}$ **do**
    for node v in Path(i,t) **do**
      $f =$ Feat(v)
      if $\mathbf{x}_i \in \mathcal{D}_I$ **then**
        $C_I(f) \leftarrow C_I(f) + 1$
        $I_I(f) \leftarrow I_I(f) + \frac{1}{d_{leaf}(i,t)} \cdot \lambda_I(v)$
      else if $\mathbf{x}_i \in \mathcal{D}_O$ **then**
        $C_O(f) \leftarrow C_O(f) + 1$
        $I_O(f) \leftarrow I_O(f) + \frac{1}{d_{leaf}(i,t)} \cdot \lambda_O(v)$
      end if
    end for
  end for
end for
Compute FI for each feature:
for $j = 1, \dots, d$ **do**
  $\text{FI}(f_j) = \frac{I_O(f_j)}{C_O(f_j)} / \frac{I_I(f_j)}{C_I(f_j)}$
end for

Since we are interested in defining Feature Importance (FI) for the IF, we need to extend this idea to an ensemble of Isolation Trees. Moreover, since in general $\mathcal{D}_O$ and $\mathcal{D}_I$ contain more than one element, we need to repeat the procedure for all samples. We introduce the following additional quantities:

- $C_O(f)$ is the number of times the generic feature f appears in the paths from root to leaf, considering for the count all trees in the forest and all samples in $\mathcal{D}_O$ ($C_I(f)$ represents the same quantity when considering for the count all trees in the forest and all samples in $\mathcal{D}_I$).
- $\lambda_O(v)$ is the induced imbalance coefficient associated with the generic node v, computed by using only samples in $\mathcal{D}_O$ ($\lambda_I(v)$ represents the same quantity when only samples in $\mathcal{D}_I$ are used).
- $I_O(f)$ is the cumulative importance for outliers of the

generic feature f and it is computed in an additive fashion. We iterate over all trees in the IF and over all samples marked as outliers. For the generic sample $\mathbf{x}_i \in \mathcal{D}_O$ and the generic tree t we add

$$\frac{1}{d_{leaf}(i,t)} \cdot \lambda_O(v) \tag{3}$$

each time feature f appears in Path(i,t), i.e. for all $v \in$ Path(i,t) such that Feat$(v) = f$. In Eq. (3), the right-hand side multiplicative factor represents only the 'local' effect of the split performed by feature f since it only evaluates the induced imbalance at a specific node location. The left-hand side multiplicative factor, instead, takes into account the contribution of the local split to the overall process of isolation since it considers the depth of the leaf node where the sample will end up. It is reasonable to include also the global effect of a local split as it may happen that a split that seems bad locally (i.e. low induced imbalance) actually facilitates splits performed subsequently by other nodes. Similarly, $I_I(f)$ represents the cumulative importance for inliers and it is computed iterating over all trees in the IF and over all samples marked as inliers.

Once counter and cumulative importance variables have been computed for all features f_j, $j = 1, \dots, d$, the FI for the generic feature f is defined by

$$\text{FI}(f) = \frac{I_O(f)}{C_O(f)} \Big/ \frac{I_I(f)}{C_I(f)}. \tag{4}$$

Algorithm 1 describes how to compute FI according to DIFFI method. The function defined in line 1 computes the *induced imbalance coefficients* for all non-leaf nodes in tree t, considering the samples contained in $\mathcal{S} \subset \mathcal{D}$, and saves them in data structure Λ. Therefore, for a certain tree t, Λ_I will contain coefficients $\lambda_I(v)$ for all nodes $v \in \mathcal{V}_t$, while Λ_O will contain coefficients $\lambda_O(v)$ for the same nodes $v \in \mathcal{V}_t$. Notice that, in order to reduce the amount of computations in Algorithm 1, one can consider only the k predicted inliers with the lowest AS, defining a subset $\mathcal{D}'_I \subset \mathcal{D}_I$ such that $|\mathcal{D}'_I| = k$.

IV. Datasets

In order to test the effectiveness of the proposed method, we consider a synthetic dataset and a real one provided by an industrial partner. It is worth pointing out that, since we are interested in assessing the performance of the proposed feature importance evaluation method, rather than the performance of the AD algorithm itself, true labels are not enough. Indeed, we also need a-priori knowledge about which are the most relevant features.

A. Synthetic dataset

We first consider a synthetic dataset and we design it in a way that only some of the available features are relevant for AD. In particular, the dataset is built on top of a set of 2D points whose Cartesian coordinates are defined as follows

$$\begin{cases} x = \rho\cos(\theta) \\ y = \rho\sin(\theta) \end{cases} \tag{5}$$

The bi-dimensional dataset consists in 900 inliers and 50 outliers, therefore the value of contamination is equal to 5.26%. Inlier points are obtained by setting

$$\theta \sim \mathcal{U}(0, 2\pi), \qquad \rho \sim \mathcal{U}(0, 1),$$

while outlier points are obtained by setting

$$\theta \sim \mathcal{U}(0, 2\pi), \qquad \rho \sim \mathcal{U}(1, 30).$$

Then, $d-2$ uninformative features are added. In particular, we describe each sample $\mathbf{x}_i$ in the dataset with a d-dimensional feature vector

$$\mathbf{x}_i = [f_{i,1}, f_{i,2}, \dots, f_{i,d}]$$

where $f_{i,1} = \rho_i\cos(\theta_i)$ and $f_{i,2} = \rho_i\sin(\theta_i)$ are defined as in Eq. (5) and represent the 'true' features, while $f_{i,j} \sim \mathcal{N}(0,1)$ with $j = 3, \dots, d$ are white noise samples.
Taking into account the characteristics of the dataset, the ideal FI method would mark as important only the features $f_{i,1}$ and $f_{i,2}$, while assigning zero importance to all other features $f_{i,j}$, $j = 3, \dots, d$. For our experiments we set $d = 6$.

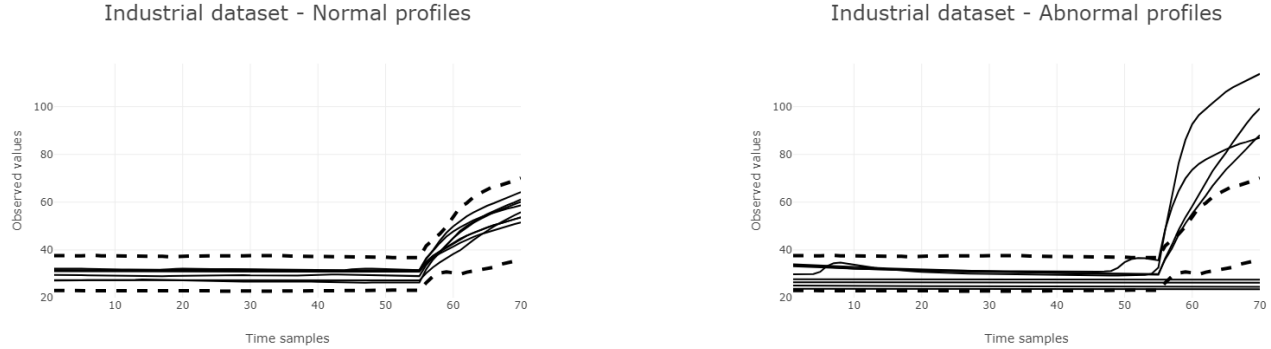

Fig. 1: Industrial dataset: normal profiles (left) and abnormal profiles (right). Dashed lines indicate the upper and lower limits of a control chart based on MAD statistic (evaluated on the whole dataset) with cutoff equal to 2.5.

B. Industrial dataset

The second dataset was provided by an industrial partner and consists in thousands of pressure profiles collected during the vacuum creation process in refrigerators manufacturing. Pressure is expected to decrease in the first stage of the process, due to the action of a pump. Then, a valve is opened to detect the presence of leaking by monitoring how fast pressure grows in a fixed amount of time. Since no labels were available, in order to compare the proposed approach with PIMP we consider a very simple classifier based on a MAD control chart [20]. Examples of normal and abnormal profiles according to MAD control charts are shown in Fig. 1. Notice that such behaviors agree with domain knowledge. On the one hand, when leaking is present, pressure is expected to grow faster than normal. On the other hand, when anomaly is due to non-opening valve, pressure is expected to remain constant also in the final stage. For the AD task, each pressure profile (70 samples) was partitioned into seven non-overlapping windows (of size 10) and for each window three features (i.e. average, minimum and maximum value) were computed. As a result, each pressure profile is represented by a 21-dimensional feature vector. According to domain knowledge, abnormal profiles should differ from the normal ones only in the last 20 samples, while the first part of the time-series should be quite similar. Therefore, we would expect the ideal FI method to mark as important only the features associated with the last two windows (i.e. from Feature 16 to Feature 21).

V. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed approach, we compare it with PIMP, detailed in Section II. We remark that PIMP is a model-agnostic algorithm, i.e. it can deal with each estimator, while the approach we propose, DIFFI, is model-based and only works with Isolation Forest. On the other hand, while PIMP needs true labels, DIFFI is completely unsupervised. The results reported in the following are based on 50 iterations of IF algorithm. For the computation of the induced imbalance coefficients, in Eq. (1) we set $\epsilon = 0.01$. As mentioned in Section III, we limited the number of predicted inliers exploited by DIFFI algorithm; in particular, if we denote with $|\mathcal{D}_O|$ the cardinality of $\mathcal{D}_O$, we only considered the $|\mathcal{D}_O|$ predicted inliers with the lowest AS.
As regards the synthetic dataset, IF has an F1 score (averaged over the 50 iterations) equal to 0.78. As it can be seen in Fig. 2, both PIMP and DIFFI methods manage to identify the two 'informative' features (i.e. Features 1 and 2).
With respect to the industrial dataset, IF has an F1 score (averaged over the 50 iterations) equal to 0.80. As it can be seen in Fig. 2, PIMP method leads to uninformative results, probably due to the high correlation between features (as

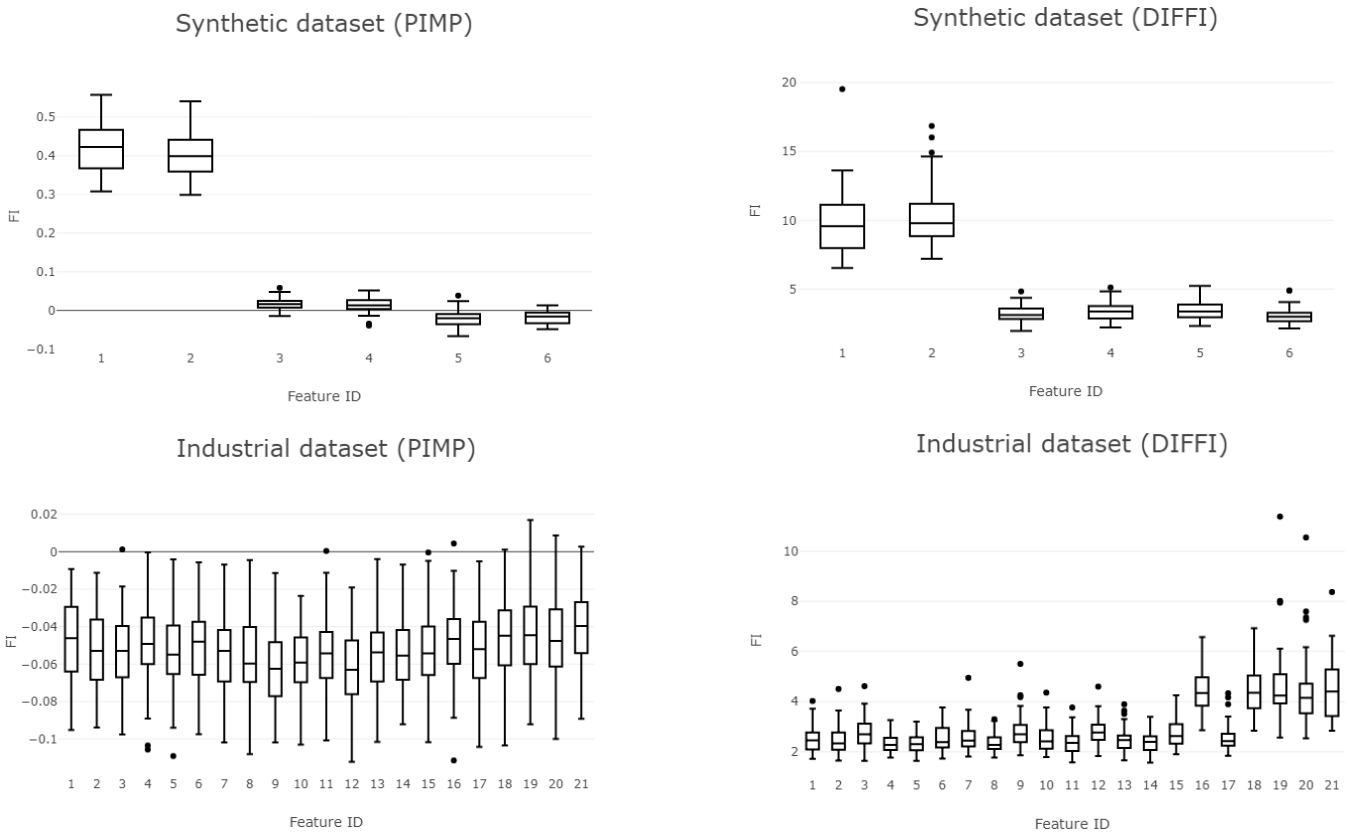

Fig. 2: Top: feature importance for the synthetic dataset computed by PIMP method (left) and DIFFI method (right). Bottom: feature importance for the industrial dataset computed by PIMP method (left) and DIFFI method (right).

shown in Fig. 3). On the contrary, DIFFI highlights Features 16, 18, 19, 20 and 21 in accordance to the portion of the pressure curves relevant for AD indicated by domain experts.

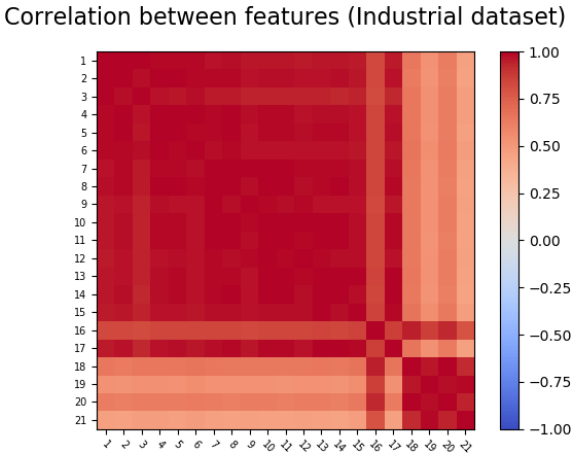

Fig. 3: Correlation between features for the industrial dataset.

VI. CONCLUSIONS AND NEXT STEPS

In this work, we addressed two relevant issues that stand in the way of ML adoption in real world environments such as those typical of Industry 4.0: (i) the lack of supervised datasets, that makes the design of smart monitoring solutions very challenging; (ii) the need for interpretability to foster trust and enable Root Cause Analysis. In particular, we focused on the interpretability of a popular Machine Learning-based Anomaly Detection algorithm, Isolation Forest; Isolation Forest is a state-of-the-art AD technique that is widely used in practice for its performance and computational burden: in this work we designed a model-based approach that is completely unsupervised and provides insight on feature importance at global level. Thus, once an Isolation Forest-powered AD system is deployed, it is possible to perform Root Cause Analysis based on features that are marked as the most relevant by the proposed approach. We plan to explore the following research directions in the future:

- evaluating whether the proposed approach is suitable to provide insights on feature importance also at local level, and adapting it if necessary;
- analysing the impact of dataset unbalancing and feature dimensionality on the performance of the proposed method;
- elaborating on the effect of Isolation Forest setup parameters, such as maximum tree depth and number of trees in the forest.

REFERENCES

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[2] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[3] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. ki Cho, and H. Chen, "Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection," in *ICLR*, 2018.

[4] L. Puggini and S. McLoone, "An enhanced variable selection and Isolation Forest based methodology for anomaly detection with OES data," *Engineering Applications of Artificial Intelligence*, vol. 67, pp. 126–135, 2018.

[5] G. A. Susto, M. Terzi, and A. Beghi, "Anomaly detection approaches for semiconductor manufacturing," *Procedia Manufacturing*, vol. 11, pp. 2018–2024, 2017.

[6] Y. He, X. Zhu, G. Wang, H. Sun, and Y. Wang, "Predicting bugs in software code changes using isolation forest," in *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 2017, pp. 296–305.

[7] L. Meneghetti, M. Terzi, S. Del Favero, G. A. Susto, and C. Cobelli, "Data-Driven Anomaly Recognition for Unsupervised Model-Free Fault Detection in Artificial Pancreas," *IEEE Transactions on Control Systems Technology*, 2018.

[8] G. A. Susto, A. Schirru, S. Pampuri, A. Beghi, and G. De Nicolao, "A hidden-Gamma model-based filtering and prediction approach for monotonic health factors in manufacturing," *Control Engineering Practice*, vol. 74, pp. 84–94, 2018.

[9] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, p. 3, 2012.

[10] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[11] A. Beghi, L. Cecchinato, C. Corazzol, M. Rampazzo, F. Simmini, and G. A. Susto, "A one-class svm based tool for machine learning novelty detection in hvac chiller systems," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 1953–1958, 2014.

[12] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin, "Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation," *Journal of Computational and Graphical Statistics*, vol. 24, no. 1, pp. 44–65, 2015.

[13] M. T. Ribeiro, S. Singh, and C. 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*. ACM, 2016, pp. 1135–1144.

[14] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, 2017, pp. 4765–4774.

[15] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[16] D. W. Apley, "Visualizing the effects of predictor variables in black box supervised learning models," *arXiv preprint arXiv:1612.08468*, 2016.

[17] A. Fisher, C. Rudin, and F. Dominici, "Model Class Reliance: Variable importance measures for any machine learning model class, from the Rashomon perspective," *arXiv preprint arXiv:1801.01489*, 2018.

[18] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[19] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[20] M. O. Abu-Shawiesh, "A simple robust control chart based on MAD," *Journal of Mathematics and Statistics*, vol. 4, no. 2, p. 102, 2008.