computers

MDPI

Article

Towards Benchmarking for Evaluating Machine Learning Methods in Detecting Outliers in Process Datasets

Thimo F. Schindler [1,*], Simon Schlicht [2] and Klaus-Dieter Thoben [1,2,*]

[1] BIBA-Bremer Institut für Produktion und Logistik GmbH, Hochschulring 20, D-28359 Bremen, Germany
[2] Department of Integrated Product Development, University of Bremen, Bibliothekstraße 1, D-28359 Bremen, Germany; simsch@uni-bremen.de
* Correspondence: sth@biba.uni-bremen.de (T.F.S.); tho@biba.uni-bremen.de (K.-D.T.)

Abstract: Within the integration and development of data-driven process models, the underlying process is digitally mapped in a model through sensory data acquisition and subsequent modelling. In this process, challenges of different types and degrees of severity arise in each modelling step, according to the Cross-Industry Standard Process for Data Mining (CRISP-DM). Particularly in the context of data acquisition and integration into the process model, it can be assumed with a sufficiently high degree of probability that the acquired data contain anomalies of various kinds. The outliers must be detected in the data preparation and processing phase and dealt with accordingly. If this is sufficiently implemented, it will positively impact the subsequent modelling in terms of accuracy and precision. Therefore, this paper shows how outliers can be identified using the unsupervised machine learning methods autoencoder, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), Isolation Forest (iForest), and One-Class Support Vector Machine (OCSVM). Following implementing these methods, we compared them by applying the Numenta Anomaly Benchmark (NAB) and sufficiently presented the individual strengths and disadvantages. Evaluating the correctness, distinctiveness and robustness criteria described in the paper showed that the One-Class Support Vector Machine was outstanding among the methods considered. This is because the OCSVM achieved acceptable anomaly detections on the available process datasets with comparatively little effort.

Keywords: machine learning; outlier detection; benchmarking anomaly detection; deep learning

Citation: Schindler, T.F.; Schlicht, S.; Thoben, K.-D. Towards Benchmarking for Evaluating Machine Learning Methods in Detecting Outliers in Process Datasets. *Computers* **2023**, *12*, 253. https://doi.org/10.3390/computers12120253

Academic Editor: Dae-Kyoo Kim

Received: 26 October 2023
Revised: 22 November 2023
Accepted: 27 November 2023
Published: 4 December 2023

1. Introduction

In the context of modern factory processes and advancing digitalization in companies, it can be shown that new technologies, such as machine learning (ML) methods, can exploit previously unused energy-saving potential in process parameters to increase energy efficiency and sustainability.

Compared to offline ML-based utilization and evaluation of production processes, the real-time integration of machine learning processes into production is challenging and can cause various problems. Our previous article described the general challenges of integrating and utilizing ML-based methods and outlined possible technical and methodological approaches to mitigate these challenges. Upon its challenges, the machine learning integration concept was built up concerning Cross-Industry Standard Process for Data Mining (CRISP-DM), Define-Understand-Collect-Analyze-Realize (DUCAR), SCRUM and constantly refers to the integration carried out in a production environment based on results from application-oriented research projects [1].

In today's world, the concept of energy efficiency is crucial to industry from the perspectives of sustainability and cost-effectiveness. Energy use can be improved through sensory monitoring, collection of diverse processes, and data application of data models. The research initiative on which this paper is based deals with the digital monitoring of

processes at a quay of an industrial port within the Weser estuary to the North Sea. This project aims to make more efficient and sustainable use of port infrastructure by increasing digitalization and integrating machine learning.

Processes are mapped onto digital models through sensory monitoring in data-driven process modeling. With the help of data analysis techniques, decisions can be made based on a process model that improves a process in terms of factors such as energy efficiency. The sensory recording of real-world processes is likely to result in outliers caused by measurement or processing errors, for example, and negatively influence the validity of these process models. Against this background and given the ever-growing amount of data, this paper aims to investigate four different machine learning methods for outlier detection and evaluate them concerning their suitability in data-driven process modeling. For this purpose, the applied unsupervised learning methods autoencoder, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), Isolation Forest (iForest), and One-Class Support Vector Machine (OCSVM) were evaluated and compared with each other based on their application in the Numenta Anomaly Benchmark (NAB). This work mainly aimed to evaluate four unsupervised machine learning methods using the Numenta Anomaly Benchmark to assess our implemented ML-based approaches to identify outliers in process datasets. These findings are also reflected in the development and automated selection of the DBSCAN parameters to calculate the evaluation value continuously. In comparison, the machine learning methods iForest and One-Class Support Vector Machine are less complex for selecting model parameters and, therefore, easier to implement for outlier detection. The One-Class Support Vector Machine delivered results as a simple technique, while the autoencoder delivered more complicated results than the OCSVM. In the evaluation based on criteria such as accuracy, distinctiveness, and robustness, the One-Class Support Vector Machine was at the top of the methods analyzed, as it achieved satisfactory recognition results with relatively little effort. Our evaluation shows balanced results and highlights the identified strengths and weaknesses of individual methods.

2. Related Work

Anomaly detection is a very comprehensive field of research. Singh, K., et al. [2] show ten different application areas and, based on different characteristics of these, highlight that other techniques are better and worse suited for anomaly detection, referring to further literature. This reveals the research environment of anomaly detection to be multi-layered and complex in terms of the ever-varying circumstances and properties of different application domains. In this paper, we address the application to real-world process model data, which can be classified as sensor networks according to the classification of Singh, K., et al. [2].

Furthermore, prior research has probed the challenges and advantages of integrating machine learning in manufacturing processes [3]. This study identified various challenges of applying machine learning in production environments, especially concerning process modeling, data integration, and AI-driven solutions. Several innovative technologies and methodologies were proposed to address these challenges. Our research also featured practical applications, such as enhancing energy efficiency using machine learning [3]. This earlier work is a precursor to our current research, offering valuable insights into the intricacies of integrating machine learning into manufacturing. It also provides a foundation for our focus on anomaly detection within real-world process model data.

A key component of our research approach lies in the real-time application of outlier detection methods in data from real process models. To validate this approach and provide a basis for comparing different detection methods, we rely on the Numenta Anomaly Benchmark (NAB). The NAB is an established resource in anomaly detection research specifically designed to evaluate the performance of outlier detection algorithms in real-time environments [4]. It provides a range of benchmark datasets, as well as metrics and evaluation criteria that allow us to test and compare the performance of our detection methods in different real-time scenarios and produce both objective and conclusive results. In [5], Freeman et al. proposed a framework for intelligent method selection based on

time series characteristics, offering guidelines to save time and effort. Meanwhile, Maciag et al. introduced OeSNN-UAD, an Online evolving Spiking Neural Network (OeSNN-UAD) adapted for unsupervised anomaly detection in streaming data [6]. The proposed OeSNN-UAD detector was experimentally compared with state-of-the-art unsupervised and semi-supervised detectors of anomalies using data streams from the Numenta Anomaly Benchmark repository, demonstrating its effectiveness in outperforming other solutions in that context.

The need to apply machine learning techniques in our research area of process modeling results largely from the enormous amounts of data generated in this application area. To determine the current state of machine learning research for anomaly detection, Nassif et al. conducted a comprehensive systematic literature review [7]. This review covered a period of two decades, from 2000 to 2020, and aimed to document the development and trends in the field. The results of this literature review illustrate that research in machine learning for anomaly detection has grown continuously and dynamically over the past two decades [7]. The autoencoder, DBSCAN, the Isolation Forest, and the OCSVM are frequently used machine learning techniques for anomaly detection in different applications.

Authors in [8] utilized autoencoders for monitoring data, employing stacked autoencoders for outlier detection. Conversely, Ahmad et al. introduced an autoencoder-based monitoring system specifically designed to detect anomalies in rotating machines [9]. Real-time inconsistencies in autonomous driving systems were addressed by Hussain et al. through the utilization of various autoencoders [10]. Proposing continual anomaly detection systems, Stocco, A., et al. incorporated autoencoders into their approach [11]. In a separate study [12], the authors evaluated autoencoders' performance in anomaly detection, employing custom reconstruction-based evaluation metrics.

Researchers have explored diverse applications of DBSCAN (Density-Based Spatial Clustering of Applications with Noise) for outlier detection and anomaly identification. In anomaly detection in monthly temperature data, DBSCAN, as applied by Celik et al. [13], showcased its advantages over traditional statistical methods. For disease classification, Ijaz et al. proposed a Hybrid Prediction Model (HPM) integrating DBSCAN-based outlier detection [14], achieving superior performance in predicting diabetes and hypertension. Flight safety analysis was extended by Sheridan et al. through the application of DBSCAN to identify anomalous flights during the approach phase [15].

The Isolation Forest has found applications in diverse domains as researchers address fraud detection, power grid monitoring, and cybersecurity challenges. In credit card fraud detection, John, H., et al. applied Isolation Forest, achieving an accuracy of 76% compared to Local Outlier Factor [16]. Regarding power grid operations, Khaledian et al. developed a Synchrophasor Anomaly Detection and Classification (SyADC) tool utilizing Isolation Forest along with KMeans and local outlier probability (LoOP) algorithms [17]. Addressing cybersecurity concerns, Ripan et al. introduced an Isolation Forest Learning-Based Outlier Detection Model [18].

The One-Class Support Vector Machine (OCSVM) emerges as a potent tool for anomaly detection across diverse domains. Applied by Mourão-Miranda et al. to classify patterns of functional magnetic resonance imaging (fMRI) responses in depressed patients [19], OCSVM revealed correlations with depression severity and treatment outcomes. In the realm of Wireless Sensor Networks (WSNs), H. Shia et al. conducted a comparative study on outlier detection techniques, showcasing OCSVM's efficiency in achieving high detection rates for abnormal data with short detection times [20]. Addressing power system anomaly detection, Wang et al. employed OCSVM optimized by an improved Particle Swarm Optimization algorithm [21], demonstrating the effectiveness of the proposed algorithm in ensuring the safe and stable operation of power systems. In another study [22], the authors focused on Internet of Things anomaly detection, extending OCSVM with Nyström and Gaussian Sketching approaches.

The literature review by Nassif et al. identified key focus areas, methodological advances, and application areas that have been addressed in the literature [7]. It became

clear that machine learning is gaining importance as a critical component for detecting outliers in complex data landscapes. These findings will be highly relevant in our research context as they reflect the current state of research and recent developments in machine learning and outlier detection.

3. Materials and Methods

This chapter presents the basics of process modeling in the context of a short introduction, and the anomaly detection methods implemented based on machine learning methods are explained in sufficient detail. This chapter concludes with the description and application of the Numenta Anomaly Benchmark, which is used to evaluate the presented and implemented outlier detectors and is explained in sufficient detail. Furthermore, the term outlier is adequately defined in the context of process modeling, and an overview of the various causes and types of outliers is presented. Moreover, machine learning methods are introduced thematically based on the outlier detectors we implement. Subsequently, the anomaly detectors based on machine learning are presented in detail. Their operating principles are explained, followed by an evaluation based on the results of the NAB benchmark. Table 1 shows the implemented methods for anomaly detection based on machine learning and continues to classify their application (e.g., clustering analysis).

Table 1. Investigated machine learning techniques for outlier detection.

Technique	Categorization in Machine Learning
Autoencoder	Dimensionality reduction
DBSCAN	Clustering
Isolation Forest	Hybrid
One-Class Support Vector Machine	Classification

We implemented the detectors in a consistent programming environment. Therefore, a unified sliding window was used to apply the four machine-learning methods to data streams. This approach makes it possible to apply the methods in real time and to train the models incrementally to optimize the accuracy of outlier detection.

3.1. Data-Driven Process Modelling

Society and industry are undergoing a digital transformation. The fourth industrial revolution stands above all for digitalization and networking of production and work processes and using technologies such as artificial intelligence (AI), robotics, and cloud computing. Innovations of Industry 4.0, such as data-driven process modeling, expand the possibilities of data analysis and thus contribute to process optimization. Data-driven process modeling describes the monitoring and digital modeling of real-world processes. The sensory acquisition of data is the basis for the modeling. Digital process models are used to analyze and optimize processes. At this point, the field of artificial intelligence with machine learning methods offers possibilities for data analysis and further gaining knowledge about processes [23] (p. 271). According to Mockenhaupt et al., four analysis techniques are distinguished in data analysis:

- Descriptive analysis (descriptive information gathering);
- Diagnostic analysis (pattern recognition);
- Predictive analysis (predictive forecasting ability);
- Prescriptive analysis (action-oriented) [23] (p. 272).

Digitization requires dealing with ever-increasing amounts of data. The amount of data the entire human race had generated since the beginning of time until 2003 was generated in just ten minutes in 2013 [23] (p. 124). Automatic data collection through sensor technology is an exemplary cause of this data growth. For this reason, the topic of Big Data is also essential in process modeling, and thus, this work will be briefly introduced below. The term Big Data describes datasets with a complexity in size and structure that exceeds

the complexity of conventional datasets. More concretely, Big Data can be characterized by volume, variety, and velocity [24] (p. 96). Volume refers to the enormous amount of data that can be produced and collected. The dimensionality of datasets is also a relevant topic here. The aspect of variety refers to variable formats and sources that differ, for example, in complexity or structure. The speed at which data is produced is the third aspect of this description. Some systems require real-time analysis and anomaly detection of the data to make critical decisions with as little delay as possible. These aspects define Big Data as mass data whose analysis requires special preprocessing steps due to its complexity in size and structure. For data-driven process modeling, it is evident that the quality of a model is significantly dependent on the data quality. However, modeling real-world processes often negatively affects the model quality due to measurement errors and noise [25] (p. 23). Outliers are also a type of error that can falsify a process model and, thus, analysis results. Consequently, data preprocessing is an essential part of process modeling, in which the detection of outliers is also thematically classified.

3.2. Outliers and Anomalies

According to Mehrotra et al., anomalies or outliers are significant deviations from the norm [26] p. 4. This short and open definition focuses on the core aspect of outliers. They represent a peculiarity in data distribution and can, therefore, strongly impact statistical analyses and predictive models. In the literature, the terms outlier and anomaly are handled differently [26] (p. 4). In this paper, both terms are used interchangeably in analogy to the work of Mehrotra et al. concerning data-driven process modeling. Hawkins further defines outliers: "An outlier is an observation that deviates from the other observations to such an extent that it is suspected to have been generated by some other mechanism" [27]. In addition to the properties in datasets, this definition also addresses the cause of outliers as a relevant criterion. Causes of outliers can be, for example, errors in data collection as well as data processing. If an outlier is caused by such an "other mechanism", it no longer reflects the real-world behavior of the system. This leads to the modeled behavior deviating from the depicted reality. Furthermore, Collett et al. classify outliers into two categories: Data points that look suspicious "in the eyes of the analyst" and those that deviate from the rest of the dataset by an objective statistical measure [28].

This distinction suggests that outliers are not always identifiable by the deviation of statistical ratios but can also deviate from the norms in other contexts. Local outliers are an example of this. These occur in a limited data area as deviations from the surrounding values. In contrast, global outliers occur as deviations in a wide range of the data. Examples of a local and a global outlier are shown in Figure 1. Accurate analysis of datasets for outlier detection requires consideration of anomalies' respective properties and characteristics. We assume that different types of outliers have specific characteristics that must be detected individually. In addition to the distinction between local and global, outliers can be divided into punctual, contextual, and collective outliers [2] (p. 310), [7] (pp. 78658–78659). These types are shown as examples in Figure 2.

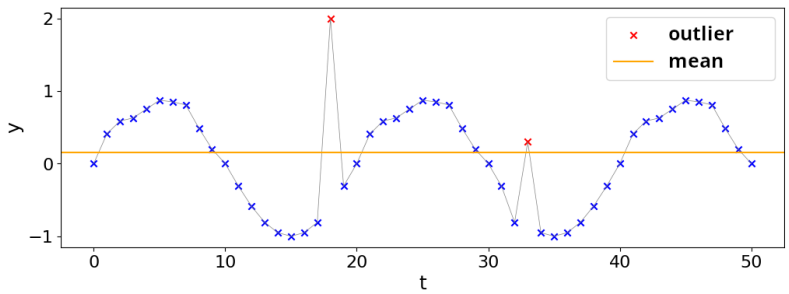

Figure 1. While the global outlier deviates fundamentally from all other data points ($t = 18$), the local outlier is anomalous in the context of the surrounding data points but is close to the mean value of the dataset ($t = 33$).

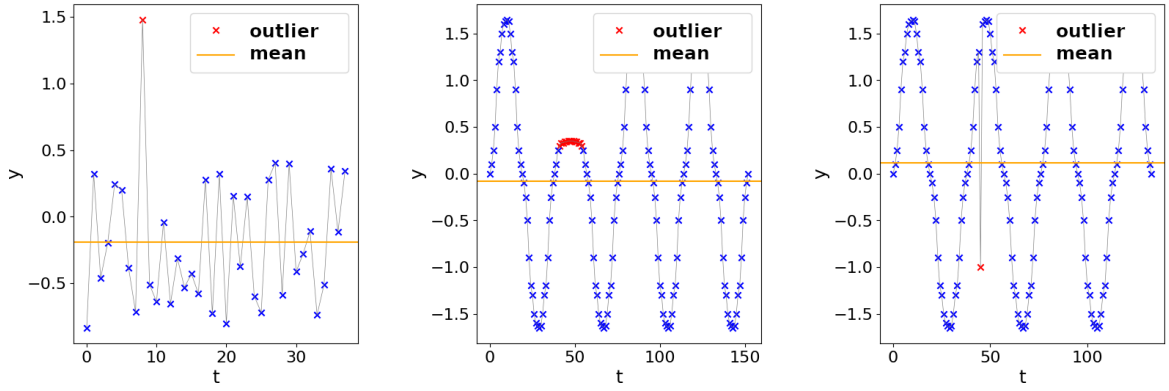

Figure 2. (**a**) Punctual outliers, also called point outliers, are data points that deviate as individual points from the surrounding data points. (**b**) In contrast, collective outliers are related data points that differ from the expected distribution. (**c**) Contextual outliers only appear anomalous in a specific context [2] (pp. 310–311). Local outliers, for example, can be described as contextual because they appear abnormal in the context of their surrounding data points but may represent typical values in other contexts.

To detect outliers, the properties of the different types must be considered. For detecting local outliers, for example, methods that assume the immediate surroundings of the affected data point are suitable. In contrast, global outliers can be detected by considering the entire dataset. The outliers presented here suggest that anomalies need not exclusively represent measurement errors. Collective and contextual outliers can also indicate systematic failures or unusual events that impact a group of data points. Outlier detection is essential in data analysis and process capture used in many application areas. Examples include network intrusion detection, fraud detection, image processing, and quality control [2] (p. 308), [29] (p. 399). It should be noted that anomaly detection is highly dependent on the application domain and its characteristics. The definition of outliers and the appropriate detection methods may vary depending on the application domain. Factors in formulating the problem are the data structure, the types of outliers, the availability of labels, and the detection output [2] (pp. 308–309). For this reason, the anomaly detection technique must be adapted to the circumstances depending on the application. Many anomaly detection techniques operate unsupervised [7] (p. 78664). This means that there are no labels that classify the data into normal and anomalous. Such unsupervised methods work with further assumptions about anomalies fundamental to the detection: outliers occur much less frequently than average data [2] (p. 312) and deviate statistically from it [30] (p. 36). In this paper, the focus is on unsupervised methods.

The result of a machine learning method for detecting outliers can be a binary label or a continuous evaluation. Labeling methods specify a predicted data point's class (e.g., outlier or no outlier). In contrast, scoring methods predict a continuous outlier score that can be used to view the data points sorted by the degree of anomaly [31] (p. 7). In the broadest sense, this distinction describes a common differentiation of machine learning methods into classification and regression. Choosing a threshold value makes it possible to convert the continuous outlier score of evaluative methods into a binary label. In the review of classifications, predicted labels related to a target class are differentiated into the categories shown in Table 2 [25] (pp. 90–91). Concerning the type of anomalous data points, *True Positive* (TP) describes the correct detection of an outlier and *True Negative* (TN) the non-detection of a normal data point. In contrast, a *False Positive* describes a false detection of a normal data point, and a *False Negative* is a non-detection of an actual outlier.

Table 2. Differentiation of classification results.

Detection Result	Outlier (Positive)	No Outlier (Negative)
Outlier detected	True Positive (TP)	False Positive (FP)
Outlier not detected	False Negative (FN)	True Negative (TN)

These concepts can be quantified by the True Positive Rate (TPR) and False Positive Rate (FPR), which provide a more nuanced understanding of model performance, where:

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR, \tag{1}$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - TNR. \tag{2}$$

The TPR and FPR offer valuable insights into a model's ability to identify anomalies while minimizing false alarms correctly.

3.3. Implemented and Tested Anomaly Detection Methods

For the selection of methods for this analysis, this work is limited to unsupervised learning methods since the availability of labeled data for training a model is often expensive and often cannot be assumed in the application domain of data-driven process modeling. However, the primary aim of this study is to thoroughly explore machine learning techniques for detecting anomalies in the context of data integration and its associated dependencies while also addressing the challenges in data integration previously outlined in our preceding article [3]. Thus, in addition to limiting the selection to unsupervised methods, care was taken to consider differently acting methods from different areas of machine learning. The methods covered are the autoencoder (dimensionality reduction), DBSCAN (clustering), the Isolation Forest (hybrid), and the One-Class Support Vector Machine (classification), which are briefly described below.

3.3.1. Autoencoder

Autoencoders (AE) are commonly employed in the field of anomaly detection [7] (p. 78665) and dimensional reduction. These models belong to the category of machine learning methods known as dimensionality reduction techniques, and more specifically, they represent a distinct variant within neural networks. The architectural structure of an autoencoder is visually described in Figure 3.

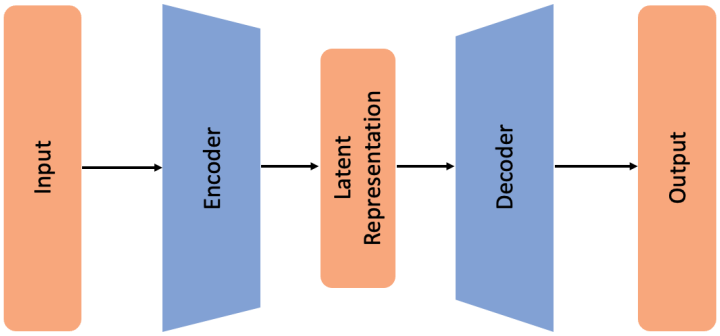

Figure 3. Consisting of two sequentially arranged parts, the first of which is the encoder and the second the decoder, autoencoders are generally used to map input data to a smaller, coded data space and thus learn the essential properties of datasets [32] (p. 1). This information about the input data, compressed by the encoder, is then reconstructed by the decoder to map the input data as losslessly as possible [33] (p. 603). (Own illustration).

For proper functionality, the encoder's input space and the decoder's output space must match the dataset's dimensionality under consideration. The compressed feature space should also consist of a substantially smaller number of nodes. As a result, essential and abstract information and correlations between features are learned and embodied by a smaller representation space. In anomaly detection, these properties of autoencoders are exploited by using the encoded representation of data to reconstruct data points using the decoder. The reconstruction error epsilon is used to measure the outlier score since outliers usually deviate from the data model and consequently cannot be reconstructed as accurately by the learned coding [32] (p. 2), [34] (p. 666). One challenge with the way autoencoders work is that although they are used as an unsupervised method, they are usually trained on uncontaminated datasets [34] (p. 665) to learn the properties of the normal data and be able to detect outliers due to the reconstruction error using this model. Since real-world use cases often do not guarantee the existence of uncontaminated data, this affects the anomaly detection result. Suppose one trains an AE on a contaminated dataset. In that case, not only the representation of the normal data but that of all data is learned, and outliers may be easier to reconstruct. However, since anomalies in datasets are often much rarer than regular data points, it is still possible that the completely unsupervised use of an autoencoder will lead to helpful detection results. Alternatively, Zhou et al. present an approach for robust autoencoders that clean contaminated datasets by applying regularization methods in the training phase [34]. Inserted filter layers sort out data from the dataset that the previously trained model cannot sufficiently represent. In this way, a trained representation of the data that can be reconstructed well and a residual of anomalies that cannot be represented well by the learned model of the autoencoder is created. Nonetheless, this approach is outside the scope of this study and its research context, but it holds promise as a subject for future investigation.

3.3.2. DBSCAN

Clustering is a frequently used method in unsupervised machine learning for outlier detection [7] (p. 78665). In general, clustering methods work according to the principle of forming clusters from data points according to specific decision criteria, e.g., distance or density. The anomaly detection application identifies data points that do not belong to a cluster as outliers. With DBSCAN (Density-Based Spatial Clustering of Applications with Noise), such a clustering method finds its way into our paper. DBSCAN uses the number of data points in the environment as a decision criterion for forming clusters. For outlier detection, those data points that do not belong to a group are subsequently detected as outliers. They are classified as noise [35] (p. 228), [36] (p. 3). DBSCAN uses a special cluster notation, which means that the density estimation does not have to be carried out between points [37] (p. 2). This notation is shown in Figure 4.

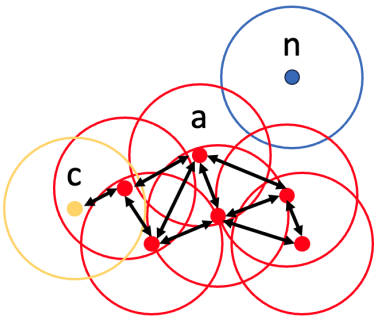

Figure 4. A point belongs to a cluster if it is in the *Eps*-neighbourhood from a core point of the cluster. In addition, if there are *MinPts* data points in its *Eps*-neighborhood, it becomes a core point itself, otherwise it is a border point. Here, point *a* represents a core point, point *c* is a border point, and point *n* is classified as noise and thus an outlier. In this figure, *MinPts* is set to 4 [37] (p. 3). Adapted from Chire (https://commons.wikimedia.org/wiki/File:DBSCAN-Illustration.svg, accessed on 1 December 2023), "DBSCAN-Illustration", CC BY.

3.3.3. One-Class Support Vector Machine (OCSVM)

Support Vector Machines (SVMs) are methods for building a discriminative model of binary datasets. They are used for classification and regression. An SVM is based on the idea of maximum separability of classes of a dataset and realizes this separation by learning a hyperplane that partitions a dataset with maximum distance to the nearest data points [38] (p. 352). A distinction is made here between linear SVMs and non-linear SVMs. If datasets are not linearly separable, the data are mapped into a higher-dimensional feature space using the kernel trick, in which they can then be linearly separated [25] (p. 99), [38] (p. 352). For anomaly detection in the context of this work, the One-Class Support Vector Machine is used. This unsupervised method learns a hyperplane between data and the origin of the coordinate system to separate outliers from the rest of the data [38] (p. 357).

A dataset $X = \{x_1, x_2, ..., x_n\}$ with $x_i \in \mathbb{R}^d$ is separated from the origin of the coordinate system by a hyperplane such that the space between the origin and hyperplane $\rho/||w||$ becomes maximal. In this example, we assume that all vector components of the data from X are positive and can be linearly separated from the origin. Figure 5 shows a soft-margin machine. It is called such because the constructed hyperplane does not necessarily separate all data from the origin but allows some data points within the delimited space using a slack variable ξ. Introducing this variable into the optimization problem of the constructed hyperplane creates a trade-off between the size of the slack variable and the margin. It results in

$$\min \phi(w, \rho) = \min_{w,\rho} \frac{1}{2} w \bullet w - \rho + \frac{1}{\nu n} \sum_{i=1}^{n} \xi[i] \tag{3}$$

under consideration of the constraints

$$w \bullet x_i \geq \rho - \xi[i], \tag{4}$$

$$\xi[i] \geq 0 \tag{5}$$

for $i = 1, ..., n$ [39] (p. 211).

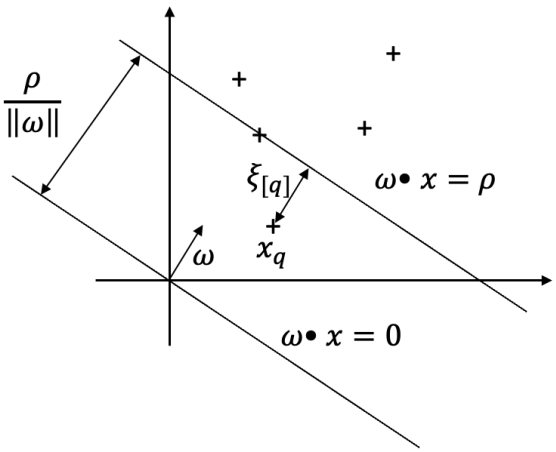

Figure 5. The implementation of the tolerance ξ to some inappropriate data points is used to detect them as outliers. The introduced parameter ν defines the maximum of points which are not separated from the origin by the hyperplane and takes values between $0 < \nu \leq 1$ [38] (p. 358), [39] (pp. 211–212). Reproduced with permission from Wiley Books.

3.3.4. Isolation Forest

The Isolation Forest or iForest differs from the other machine learning outlier detection methods in one fundamental way: This method does not focus on modeling data without anomalies to identify outliers based on deviation from that model. Instead, it assumes that anomalies are rare and sufficiently different compared to the rest of the data. An Isolation

Forest explicitly learns to isolate outliers from the dataset, using the complexity of isolation as a measure of outlier score [40] (p. 416). It is an averaging ensemble of isolation trees that draw linear decision boundaries in the data space equivalent to random decision trees to separate samples. An isolation tree recursively separates a dataset $X = x_1, ..., x_n$ into multiple parts that the path can uniquely describe. Due to the equivalent structure of isolation trees and binary search trees, the anomaly degree of a point x of a dataset with n data points results in

$$s(x, n) = 2 \cdot -\frac{E(h(x))}{c(n)}, \tag{6}$$

where $E(h(x))$ describes the average of the path length of an ensemble of isolation forests and $c(n)$ corresponds to the average path length of an unsuccessful search in a binary search tree [40] (p. 415). An Isolation Forest uses two parameters: The sample size ψ and the number of trees t. For the parameters, Liu et al. propose $\psi = 256$ and $t = 100$ for a wide range of datasets and properties of these [40] (pp. 417–418).

3.4. Numenta-Anomaly-Benchmark (NAB)

The Numenta Anomaly Benchmark (Release 1.1) [4] evaluates the previously presented and implemented anomaly detection methods. NAB is an open-source environment in which anomaly detectors can be applied, tested, and uniformly compared in a real-time environment. In the NAB, detectors are tested on time series data passed to the Detector Under Test (DUT), as shown in Figure 6 in a simulated data stream [4] (p. 38). For this reason, the environment is suitable for comparing detectors concerning the use case in data-driven process modeling. NAB not only provides a unified environment for the application of detectors but also some datasets and a scoring system that evaluates detectors according to the following requirements in real-time application:

- Detection of all anomalies;
- Detection speed;
- No false alarms;
- Detection in real-time (no looking ahead);
- Automation [4] (pp. 39–40).

Anomaly windows are defined for evaluating these factors [4] (p. 40). These windows extend around anomalies in the test datasets. Detected outliers are weighted by the relative position in the anomaly window to evaluate the detection speed. Appropriate thresholding is very crucial in anomaly detection. Therefore, the Numenta Anomaly Benchmark determines the threshold value, which is set to maximize the NAB score.

$$\sigma^a(y) = (a_{TP} - a_{FP})(\frac{1}{1 + e^{5y}}) - 1, \tag{7}$$

where a_{TP} and a_{FP} stand for weights of the application profile a. These weights can be adjusted depending on the application to change the focus of the scoring system. In our assessment, the default score S^{std} was used. For a dataset X with a set of all data points Y_X detected as anomalous and the number of all false-negative detections f_X, the score results in [4] (p. 41)

$$S_X^a = (\sum_{y \in Y_X} \sigma^a(y)) + a_{FN} f_X. \tag{8}$$

If several datasets are checked, the anomaly scores are added and normalized to the range $[0, 100]$ [4] (p. 41). The four machine learning methods described in Section 3.3 were implemented in the NAB as detectors. For this, the benchmark provides a base class from which detectors must inherit and override the NAB function *handleRecord()* in which the detection occurs. This function realizes the data stream, passes the next data point of the time series to the DUT, and expects an anomaly score in the range $[0, 1]$ as a return value,

where a larger value represents a higher probability of an anomaly. The process is shown in Figure 6. The NAB analyses the anomaly scores of a detector in an optimization phase to determine a threshold value for the optimal NAB score. To compare the detectors in more detail, each detector was evaluated individually on each dataset and then averaged.

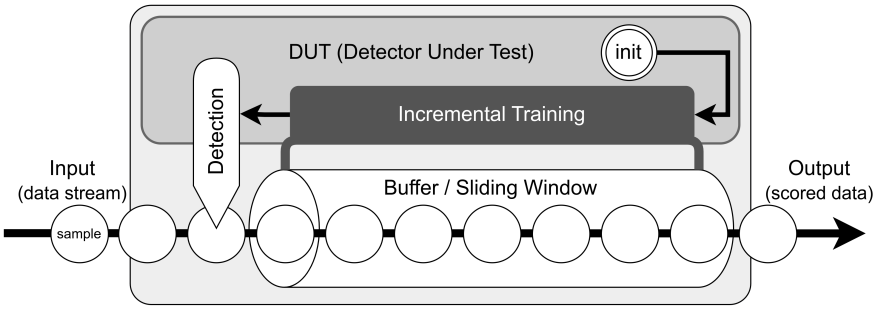

Figure 6. In application, data points of the data stream are buffered, and the models are trained on data of a fixed window size; in this case, a defined time [41] (p. 1148). This training is done periodically on changing windows of a fixed size. (Own illustration).

We employed a sliding window to incrementally train the models with the available data to ensure comparability among the detection methods. The incremental training is shown in Figure 6. For our evaluation, the window size of 96 h and a training interval of one-quarter of the window size, i.e., 24 h, was used for all detectors and datasets. For the evaluation of the anomaly detection methods, the datasets *artificialWithAnomaly* and *realKnownCause* from NAB are used [4].

In the following, we explain the concrete model structures and special features of the implementation of the detectors. Table 3 also shows the parameters used for the different detection methods. Due to the scope of the manipulated variables in implementing the autoencoder, this table only refers to the optimization algorithm of the neural network and the loss function for this method. For the implementation of the autoencoder and the Isolation Forest, the data were z-normalized. The standardization Z of dataset X is carried out using the mean value X and the standard deviation σ as follows [42] (p. 274):

$$Z = \frac{X - \overline{X}}{\sigma}. \tag{9}$$

The remaining detectors were applied to unscaled data. For our evaluation, a sequential autoencoder (AE) was implemented with Keras (2.6.0) [43]. Compared to the other methods, the unique feature is that the AE can process entire time sequences as input. Thus, the AE not only considers all data points of the window but explicitly learns the sequence of the data to detect outliers. The encoder consists of three dense layers, with the number of neurons halving in each layer. After the first layer, a dropout of 0.2 was introduced to counteract the overfitting of the network. Dropout is a regularisation technique in which randomly selected neurons and their connections are temporarily turned off during training. The decoder also uses three dense layers with a dropout of 0.2 after the first layer. ReLu (Rectifier Linear Unit) is used as the activation function in the first five layers and the sigmoid function in the last layer to obtain an output of the range $[0, 1]$. The outlier score is formed by forming the absolute error between the last data point of the input and output and scaled to the value range $[0, 1]$. When implementing DBSCAN, the dataset-dependent parameters $MinPts$ and Eps pose a challenge. As shown in Figure 4, the hyperparameters are essential for the detection result. Due to the application in the sliding window, the characteristics of the dataset under consideration change with each execution. For this reason, the choice of parameters was automated. The approaches of Sander et al. for $MinPts$ and of Akbari et al. for Eps were implemented in an additional function [44] (p. 182), [45]. On the unidimensional datasets and combined with the automated determination of Eps, we got results if $MinPts = 2d$ was chosen. Since the hyperparameters are adjusted in each

training step, the detector re-initialization occurs in the training phase. Furthermore, it should be noted that DBSCAN is a labeling procedure. No continuous evaluations are returned, only the distinction between noise and clusters. Discrete labels would give the training process a decisive disadvantage in the optimization and evaluation phase of the NAB since threshold values have no relevance to the result. For this reason, an extension of the anomaly evaluation was added, which calculates the mean value of the distances to the *MinPts* nearest neighbors as a continuous evaluation from all data points labeled as outliers. In this way, labeled outliers can be distinguished. Data points more distant from their neighbors are more likely to be anomalous. The outlier score is also scaled to the $[0, 1]$ value range. The procedure was implemented with the help of the library scitkit-learn (0.21.1) [46].

Table 3. Parameter of the implemented techniques in application.

Technique	Parameter	Value
Autoencoder [1]	Optimizer	adam
	Loss function	mse
DBSCAN	*MinPts*	$2d$
	Eps	according to [45]
iForest	t	100
	ψ	256
	Contamination	0.1
OCSVM	Kernel Function	rbf
	ν	0.01

[1] Due to the massive scope of the manipulated variables in implementing the autoencoder, this table only refers to the optimisation algorithm of the neural network and the loss function for this method.

Implementing the Isolation Forest was also carried out using scitkit-learn (0.21.1) [46]. The parameters were defined according to the specifications of Liu et al. [40]. The decision function is used to calculate the outlier score and then scaled. The One-Class Support Vector Machine was also implemented using scitkit-learn (0.21.1) [46].

4. Results

For the evaluation of the anomaly detection methods, the datasets *artificialWithAnomaly* and *realKnownCause* are used by the NAB, and the following criteria are calculated and presented as the result of the benchmarking:

4.1. Correctness

For the evaluation of the correctness of the anomaly detection, the application-oriented standard score S^{std} from the NAB is used on the artificially generated datasets. These six datasets contain various anomalies (punctual, contextual, collective) and local and global outliers. As described earlier, the NAB score S^{std} highlights the false positive detections against the detected anomaly windows concerning speed.

4.2. Distinctiveness

The evaluation of distinctiveness is also carried out on these datasets by the evaluation metric Area Under the Curve (AUC). AUC describes the area under the ROC (Receiver Operating Characteristic) curve. This curve shows the relationship between the True Positive and False Positive rates for different thresholds [47] (pp. 901–902). AUC is thus a measure of how well the classifier can distinguish between positive and negative anomalies and is often used to evaluate methods for anomaly detection [7] (p. 78690). However, for the datasets from the Numenta Anomaly Benchmark, it should be noted that the anomaly windows determine the labels for this evaluation metric. Therefore, a positive label does not necessarily describe a true anomaly but only the range of the anomaly window. For

this reason, the values for AUC determined in this evaluation can only be compared with other detectors if the detection was also carried out on datasets labeled in this way.

4.3. Robustness

The robustness criterion assesses the adaptability of the methods to datasets with diverse properties, examining their resilience and limitations. Evaluation is conducted on the realKnownCause datasets from NAB, comprising seven real-world datasets from various application domains. In this context, the NAB score (S^{std}) and the AUC value are jointly employed to measure the detectors' real-time performance. To enhance comparability, the NAB score is scaled by a factor of 0.01, and the resultant scaled score is combined with the AUC value. This combined assessment comprehensively measures the methods' robustness across different datasets. The scaling of the NAB score is crucial to ensure equal weighting in the evaluation, considering that the NAB score ranges from 0 to 100 while the AUC ranges from 0 to 1. This adjustment guarantees a balanced contribution of both scores to the overall assessment.

4.4. Outcome of the Individual Anomaly Detection Methods

Table 4 shows the benchmark results of all detectors on the 13 test datasets from the NAB. Datasets 1–6 represent the artificially generated datasets *artificialWithAnomaly* from the NAB to evaluate correctness and distinctiveness. As explained previously, robustness is assessed using the *realKnownCause* datasets from the NAB. These datasets are labelled 7–13. In the practical application of the detectors, the window size was increased by 10 for dataset 7 and by a factor of 5 for dataset 11 due to the smaller sampling interval. The NAB score focuses on the real-world application of outlier detectors and primarily evaluates the detection of anomaly depth windows rather than accuracy. Using the metric AUC, a deeper examination of the distinctiveness of the detectors is made by examining the discriminability of the classes at different thresholds.

Table 4. Results of applying detectors to test datasets. The numbered datasets are from NAB [4]. Datasets 1 to 6 are from *artificialWithAnomaly*, Datasets 7 to 13 are from *realKnownCause* and its names are: (1) art_daily_flatmiddle, (2) art_daily_jumpsdown, (3) art_daily_jumpsup, (4) art_daily_nojump, (5) art_increase_spike_density, (6) art_load_balancer_spikes, (7) ambient_temperature_system_failure, (8) cpu_utilization_asg_misconfiguration, (9) ec2_request_latency_system_failure, (10) machine_temperature_system_failure, (11) nyc_taxi, (12) rogue_agent_key_hold, (13) rogue_agent_key_updown [1].

	Autoencoder		**DBSCAN**		**iForest**		**OCSVM**	
Dataset	S^{std}	**AUC**	S^{std}	**AUC**	S^{std}	**AUC**	S^{std}	**AUC**
1	40.210	0.474	93.030	0.580	**99.670**	**0.820**	93.030	0.634
2	42.900	**0.766**	**93.030**	0.617	55.710	0.611	**93.030**	0.659
3	**93.030**	**0.773**	**93.030**	0.675	59.640	0.697	**93.030**	0.670
4	**93.030**	**0.837**	0.000	0.578	34.640	0.380	0.000	0.553
5	0.000	0.564	0.000	0.583	0.000	**0.587**	0.000	0.543
6	87.190	0.669	86.440	0.595	92.270	**0.779**	**97.740**	0.484
Mean	59.393	**0.681**	60.922	0.605	56.988	0.646	**62.805**	0.591
7	46.660	**0.682**	**91.450**	0.586	0.000	0.583	47.540	0.581
8	0.000	0.722	98.560	0.533	0.000	**0.746**	**98.720**	0.639
9	**95.310**	0.591	83.930	0.592	67.970	0.578	80.770	**0.621**
10	**48.380**	**0.885**	41.450	0.569	0.000	0.840	0.170	0.638
11	**53.110**	**0.608**	16.440	0.550	0.000	0.591	0.000	0.464
12	0.000	—	18.390	0.597	22.740	0.610	**23.980**	**0.721**
13	39.890	0.699	57.840	0.578	64.640	0.742	**73.540**	**0.791**
Mean	40.479	**0.698**	**58.294**	0.572	22.193	0.670	46.389	0.636

[1] The datasets are provided by the Numenta Anomaly Benchmark. Further documentation and the actual datasets will be taken from the NAB Repository available at github.com (accessed on 1 December 2023)/Numenta/NAB. NAB is licensed under the GNU Affero General Public License v3.0.

When applying the methods, the system memory usage was additionally measured as an example when using the individual methods. It is noticeable that the autoencoder has a much higher memory requirement than the other methods due to the formation of sequences from the data and the complex architecture, which at 129.4 MiB is more than ten times that of the DBSCAN method (11.6 MiB). OCSVM is the least expensive method at 4.9 MiB, with Isolation Forest at 7.2 MiB.

4.5. Autoencoder

The autoencoder performs comparatively well on artificially generated datasets. This is particularly evident concerning accuracy. With an AUC value of 0.681, the AE has the highest distinctiveness among the detectors examined. What is striking here is the detection accuracy in dataset 4, in which all other anomaly detectors performed predominantly worse. Dataset 4 is the dataset *art_daily_nojump* [4], which contains a collective contextual outlier. The oscillation of the values is interrupted by the failure of a peak and remains in the minimum range for the duration of one oscillation. Figure 7 shows the result of the detection of the AE and, as an example, that of the isolation forest.

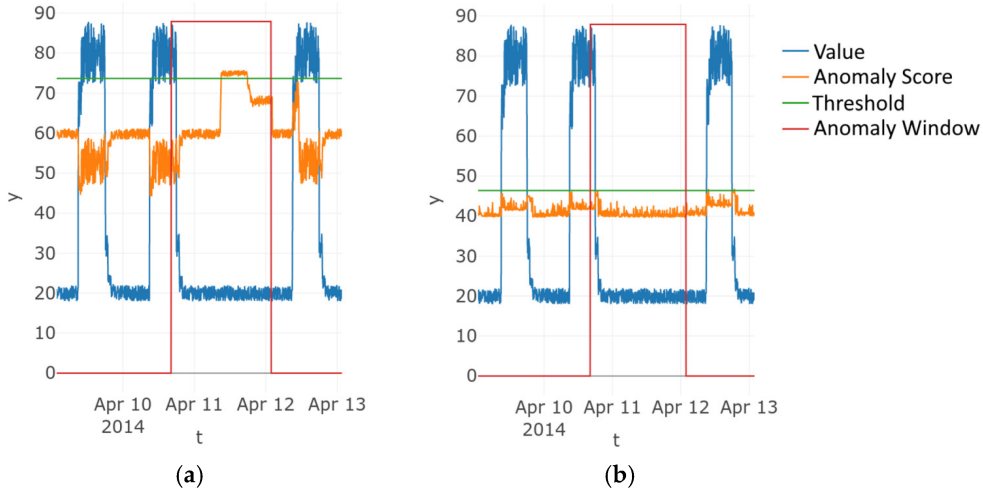

Figure 7. The plot shows the detection result due to the detector's anomaly score and threshold. The blue graph represents the artificially generated dataset art_daily_nojump from Numenta [4] on which the detection was performed. (**a**) The autoencoder can detect complex contextual outliers by respecting the sequence of the data. (**b**) The Isolation Forest does not achieve acceptable detection results in contrast.

This is where the strength of the autoencoder becomes clear. Because it is trained on sequences and the order of the data is taken into account, it can react to contextual outliers and recognize them as such. On the real datasets, however, the AE performs less well. Despite its relatively high AUC value, some false-positive detections lead to a weak NAB score. In general, it is noticeable that the outlier evaluation of the AE shows unstable behavior compared to the other investigated methods. This may be due to insufficient training data and training on contaminated data, among other reasons. It was not applied to dataset 12 due to gaps in the dataset, as some input sequences could not be formed.

4.6. DBSCAN

DBSCAN shows similar results when applied to the artificially generated and real datasets. Due to the automated parameter selection, the method is comparatively well applicable to different datasets. Among all the methods compared, however, the accuracy of DBSCAN is predominantly weak. This is partly because collective outliers are challenging to detect for clustering methods, as they can form their clusters when they occur, which are consequently not detected as outliers. Through the real-time application, the first emerging

data points of a collective outlier can be detected. Still, the subsequently occurring points, in some cases, lead to the formation of their cluster.

4.7. One-Class Support Vector Machine (OCSVM)

The One-Class Support Vector Machine (OCSVM) shows average to good results in the application. On the artificially generated datasets, the difficulty of the OCSVM in recognizing contextual outliers becomes apparent. Despite the comparatively low distinguishability, the OCSVM achieves the best rating for the criterion correctness. Overall, the results are therefore acceptable. The real-world datasets show inconsistent results and lead to a mixed evaluation.

4.8. Isolation Forest

The performance of Isolation Forest on the artificial datasets is very mixed. Despite comparatively good distinctiveness, four of the seven real datasets have a NAB score of 0, which indicates that isolated occurrences of false positives strongly influence the evaluation in the real-time application.

4.9. Summary

In terms of implementation, the autoencoder offers many parameters and variations that must be tuned to the dataset's characteristics under investigation. The large number of possible settings places high demands on the implementation. DBSCAN also requires effort due to parameter automation and the extension of outlier evaluation for a continuous score. In comparison, the iForest and OCSVM methods are less susceptible to parameter selection. They are comparatively straightforward to implement, with the One-Class Support Vector Machine parameter selection proving less sensitive.

5. Conclusions

This paper aimed to investigate machine learning methods for outlier detection, focusing on their suitability for data-driven process modeling. Our previous paper [3] comprehensively outlined the challenges of integrating and using machine learning-based methods in a production environment to steer the process. In particular, we highlighted the importance of data quality in the context of data processing and integration. This paper shows how anomalies and outliers can be detected using ML-based methods and applying NAB benchmarking. Further, we demonstrate how these methods can be tested and compared with each other based on artificial and non-artificial datasets. For this purpose, the primary problem and motivation for evaluating anomaly detection methods have been explained. Furthermore, the theoretical foundations of ML methods and the Numenta Anomaly Benchmark have been presented. The central focus of this paper was to evaluate four unsupervised machine learning methods using the criteria described in Section 4 and apply the Numenta Anomaly Benchmark to the ML-based methods we have implemented. The evaluation of the criteria correctness, distinctiveness, and robustness put the One-Class Support Vector Machine first among the methods considered since acceptable detection results could be achieved with comparatively little effort. The complexity of the implementation concerning the vulnerability of the hyperparameter ν was also decisive for this result. For complex use cases and using more resources, the autoencoder can train a more complex model that can provide better detection results if the training and parameter selection are precisely matched to the data characteristics. Thus, the One-Class Support Vector Machine results in a relatively simple technique, and the autoencoder is more complex than the OCSVM.

The evaluation also helped identify specific merits and limitations concerning different types of outliers, which enabled a more precise method distinction. Therefore, examples include the autoencoder's high effectiveness in detecting complex contextual outliers and the ability of DBSCAN to detect local punctual outliers. The Numenta Anomaly Benchmark is also suitable for comparing methods for outlier detection in real time. In the context

of modeling real-world processes using machine learning, there is also the possibility of integrating a comparison of different anomaly detection methods into real-time monitoring systems within data-driven process modeling. Thus, in the future, the findings from this work can be further deepened and incorporated into developing a real-time-based method for detecting outliers in integrated data-driven process modeling. It is essential to note that the application of autoencoders for outlier detection, for example, requires a comprehensive understanding of the process to approximate the complex representation of the process sufficiently accurately by the model.

Author Contributions: Conceptualization, T.F.S. and S.S.; methodology, T.F.S. and S.S.; software, T.F.S. and S.S.; validation, T.F.S. and S.S.; formal analysis, T.F.S. and S.S.; investigation, S.S.; resources, T.F.S. and S.S.; data curation, T.F.S. and S.S.; writing—original draft preparation, T.F.S.; writing—review and editing, T.F.S., S.S., and K.-D.T.; visualization, T.F.S. and S.S.; supervision, S.S. and K.-D.T.; project administration, T.F.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research Port2Connect-project (19H22008) was funded by the German Federal Ministry for Digital and Transport (BMDV) in the "Innovative Port Technologies" (IHATEC II) program.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: The authors would like to thank their project partners and the anonymous reviewers for their valuable input.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Schäfer, F.; Mayr, A.; Schwulera, E.; Franke, J. Smart Use Case Picking with DUCAR: A Hands-On Approach for a Successful Integration of Machine Learning in Production Processes. *Procedia Manuf.* **2020**, *51*, 1311–1318. [CrossRef]
2. Singh, K.; Upadhyaya, S. Outlier detection: Applications and techniques. *Int. J. Comput. Sci. Issues (IJCSI)* **2012**, *9*, 307–323.
3. Schindler, T.F.; Bode, D.; Thoben, K.D. Towards Challenges and Proposals for Integrating and Using Machine Learning Methods in Production Environments. In Proceedings of the International Conference on System-Integrated Intelligence, Genova, Italy, 7–9 September 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 3–12.
4. Lavin, A.; Ahmad, S. Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark. In Proceedings of the 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 9–11 December 2015. [CrossRef]
5. Freeman, C.; Merriman, J.; Beavers, I.; Mueen, A. Experimental Comparison of Online Anomaly Detection Algorithms. In Proceedings of the Thirty-Second International Flairs Conference, Sarasota, FL, USA, 19–22 May 2019.
6. Maciag, P.S.; Kryszkiewicz, M.; Bembenik, R.; Lobo, J.L.; Del Ser, J. Unsupervised Anomaly Detection in Stream Data with Online Evolving Spiking Neural Networks. *Neural Netw.* **2021**, *139*, 118–139. [CrossRef] [PubMed]
7. Nassif, A.B.; Talib, M.A.; Nasir, Q.; Dakalbab, F.M. Machine learning for anomaly detection: A systematic review. *IEEE Access* **2021**, *9*, 78658–78700. [CrossRef]
8. Wan, F.; Guo, G.; Zhang, C.; Guo, Q.; Liu, J. Outlier Detection for Monitoring Data Using Stacked Autoencoder. *IEEE Access* **2019**, *7*, 173827–173837. [CrossRef]
9. Ahmad, S.; Styp-Rekowski, K.; Nedelkoski, S.; Kao, O. Autoencoder-based Condition Monitoring and Anomaly Detection Method for Rotating Machines. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020. [CrossRef]
10. Hussain, M.; Ali, N.; Hong, J.E. DeepGuard: A framework for safeguarding autonomous driving systems from inconsistent behaviour. *Autom. Softw. Eng.* **2021**, *29*, 1. [CrossRef]
11. Stocco, A.; Tonella, P. Towards Anomaly Detectors that Learn Continuously. In Proceedings of the 2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), Coimbra, Portugal, 12–15 October 2020. [CrossRef]
12. Hussain, M.; Suh, J.W.; Seo, B.S.; Hong, J.E. How Reliable are the Deep Learning-based Anomaly Detectors? A Comprehensive Reliability Analysis of Autoencoder-based Anomaly Detectors. In Proceedings of the 2023 Fourteenth International Conference on Ubiquitous and Future Networks (ICUFN), Paris, France, 4–7 July 2023. [CrossRef]
13. Celik, M.; Dadaser-Celik, F.; Dokuz, A.S. Anomaly detection in temperature data using DBSCAN algorithm. In Proceedings of the 2011 International Symposium on Innovations in Intelligent Systems and Applications, Istanbul, Turkey, 15–18 June 2011. [CrossRef]
14. Ijaz, M.; Alfian, G.; Syafrudin, M.; Rhee, J. Hybrid Prediction Model for Type 2 Diabetes and Hypertension Using DBSCAN-Based Outlier Detection, Synthetic Minority Over Sampling Technique (SMOTE), and Random Forest. *Appl. Sci.* **2018**, *8*, 1325. [CrossRef]

15. Sheridan, K.; Puranik, T.G.; Mangortey, E.; Pinon-Fischer, O.J.; Kirby, M.; Mavris, D.N. An Application of DBSCAN Clustering for Flight Anomaly Detection During the Approach Phase. In Proceedings of the AIAA Scitech 2020 Forum, Orlando, FL, USA, 6–10 January 2020. [CrossRef]
16. John, H.; Naaz, S. Credit Card Fraud Detection using Local Outlier Factor and Isolation Forest. *Int. J. Comput. Sci. Eng.* **2019**, *7*, 1060–1064. [CrossRef]
17. Khaledian, E.; Pandey, S.; Kundu, P.; Srivastava, A.K. Real-Time Synchrophasor Data Anomaly Detection and Classification Using Isolation Forest, KMeans, and LoOP. *IEEE Trans. Smart Grid* **2021**, *12*, 2378–2388. [CrossRef]
18. Ripan, R.C.; Sarker, I.H.; Anwar, M.M.; Furhad, M.H.; Rahat, F.; Hoque, M.M.; Sarfraz, M. An Isolation Forest Learning Based Outlier Detection Approach for Effectively Classifying Cyber Anomalies. In *Advances in Intelligent Systems and Computing*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 270–279. [CrossRef]
19. Mourão-Miranda, J.; Hardoon, D.R.; Hahn, T.; Marquand, A.F.; Williams, S.C.; Shawe-Taylor, J.; Brammer, M. Patient classification as an outlier detection problem: An application of the One-Class Support Vector Machine. *NeuroImage* **2011**, *58*, 793–804. [CrossRef]
20. Shia, H.H.; Ali Tawfeeq, M.; Mahmoud, S.M. High Rate Outlier Detection in Wireless Sensor Networks: A Comparative Study. *Int. J. Mod. Educ. Comput. Sci.* **2019**, *11*, 13–22. [CrossRef]
21. Wang, Z.; Fu, Y.; Song, C.; Zeng, P.; Qiao, L. Power System Anomaly Detection Based on OCSVM Optimized by Improved Particle Swarm Optimization. *IEEE Access* **2019**, *7*, 181580–181588. [CrossRef]
22. Yang, K.; Kpotufe, S.; Feamster, N. An Efficient One-Class SVM for Anomaly Detection in the Internet of Things. *arXiv* **2021**, arXiv:2104.11146. https://doi.org/10.48550/ARXIV.2104.11146.
23. Mockenhaupt, A. *Digitalisierung und Künstliche Intelligenz in der Produktion*; Springer: Wiesbaden, Germany, 2021. [CrossRef]
24. O'Leary, D.E. Artificial intelligence and big data. *IEEE Intell. Syst.* **2013**, *28*, 96–99. [CrossRef]
25. Runkler, T.A. *Data Mining: Modelle und Algorithmen Intelligenter Datenanalyse*, 2nd ed.; Computational Intelligence; Springer: Wiesbaden, Germany, 2015.
26. Mehrotra, K.G.; Mohan, C.K.; Huang, H. *Anomaly Detection Principles and Algorithms*; Springer: Berlin/Heidelberg, Germany, 2017; Volume 1.
27. Hawkins, D.M. *Identification of Outliers*; Springer: Berlin/Heidelberg, Germany, 1980; Volume 11.
28. Collett, D.; Lewis, T. The subjective nature of outlier rejection procedures. *J. R. Stat. Soc. Ser. C Appl. Stat.* **1976**, *25*, 228–237. [CrossRef]
29. Aggarwal, C.C. Applications of Outlier Analysis. In *Outlier Analysis*; Springer International Publishing: Cham, Switzerland, 2017; pp. 399–422. [CrossRef]
30. Omar, S.; Ngadi, A.; Jebur, H.H. Machine learning techniques for anomaly detection: An overview. *Int. J. Comput. Appl.* **2013**, *79*, 33–41. [CrossRef]
31. Zimek, A.; Filzmoser, P. There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1280. [CrossRef]
32. Chen, Z.; Yeo, C.K.; Lee, B.S.; Lau, C.T. Autoencoder-based network anomaly detection. In Proceedings of the 2018 Wireless Telecommunications Symposium (WTS), Phoenix, AZ, USA, 17–20 April 2018; pp. 1–5.
33. Ye, A.; Wang, Z. Autoencoders. In *Modern Deep Learning for Tabular Data: Novel Approaches to Common Modeling Problems*; Apress: Berkeley, CA, USA, 2023; pp. 601–680. [CrossRef]
34. Zhou, C.; Paffenroth, R.C. Anomaly detection with robust deep autoencoders. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 665–674.
35. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, OR, USA, 2–4 August 1996; Volume 96, pp. 226–231.
36. Wibisono, S.; Anwar, M.; Supriyanto, A.; Amin, I. Multivariate weather anomaly detection using DBSCAN clustering algorithm. *Proc. J. Phys. Conf. Ser.* **2021**, *1869*, 012077. [CrossRef]
37. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Trans. Database Syst.* **2017**, *42*, 1–21. [CrossRef]
38. Hejazi, M.; Singh, Y.P. One-class support vector machines approach to anomaly detection. *Appl. Artif. Intell.* **2013**, *27*, 351–366. [CrossRef]
39. Hamel, L.H. *Knowledge Discovery with Support Vector Machines*; John Wiley & Sons: Hoboken, NJ, USA, 2011; pp. 103–113, 209–217.
40. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation Forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422. [CrossRef]
41. Hota, H.; Handa, R.; Shrivas, A.K. Time series data prediction using sliding window based RBF neural network. *Int. J. Comput. Intell. Res.* **2017**, *13*, 1145–1156.
42. Fahrmeir, L.; Heumann, C.; Künstler, R.; Pigeot, I.; Tutz, G. Stetige Zufallsvariablen. In *Statistik: Der Weg zur Datenanalyse*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 251–287. [CrossRef]
43. Keras. 2015. Available online: https://keras.io (accessed on 1 December 2023).
44. Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Min. Knowl. Discov.* **1998**, *2*, 169–194. [CrossRef]

45. Akbari, Z.; Unland, R. Automated determination of the input parameter of DBSCAN based on outlier detection. In Proceedings of the Artificial Intelligence Applications and Innovations: 12th IFIP WG 12.5 International Conference and Workshops, AIAI 2016, Thessaloniki, Greece, 16–18 September 2016; Proceedings 12; Springer: Berlin/Heidelberg, Germany, 2016; pp. 280–291.
46. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
47. Campos, G.O.; Zimek, A.; Sander, J.; Campello, R.J.; Micenková, B.; Schubert, E.; Assent, I.; Houle, M.E. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Min. Knowl. Discov.* **2016**, *30*, 891–927. [CrossRef]