electronics

MDPI

Article

Insider Threat Detection Model Enhancement Using Hybrid Algorithms between Unsupervised and Supervised Learning

Junkai Yi and Yongbo Tian *

School of Automation, Beijing Information Science & Technology University, Beijing 100192, China; yijk@bistu.edu.cn

* Correspondence: 2021020439@bistu.edu.cn

Abstract: Insider threats are one of the most costly and difficult types of attacks to detect due to the fact that insiders have the right to access an organization's network systems and understand its structure and security procedures, making it difficult to detect this type of behavior through traditional behavioral auditing. This paper proposes a method to leverage unsupervised outlier scores to enhance supervised insider threat detection by integrating the advantages of supervised and unsupervised learning methods and using multiple unsupervised outlier mining algorithms to extract from the underlying data useful representations, thereby enhancing the predictive power of supervised classifiers on the enhanced feature space. This novel approach provides superior performance, and our method provides better predictive power compared to other excellent abnormal detection methods. Using only 20% of the computing budget, our method achieved an accuracy of 86.12%. Compared with other anomaly detection methods, the accuracy increased by up to 12.5% under the same computing budget.

Keywords: representation learning; unsupervised ensemble; outlier scoring; principal component analysis; extreme gradient boosting

check for updates

Citation: Yi, J.; Tian, Y. Insider Threat Detection Model Enhancement Using Hybrid Algorithms between Unsupervised and Supervised Learning. *Electronics* **2024**, *13*, 973. https://doi.org/10.3390/electronics13050973

Academic Editor: Zbigniew Kotulski

Received: 19 January 2024
Revised: 22 February 2024
Accepted: 1 March 2024
Published: 3 March 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Recognizing the significant threat posed by insiders, organizations and cybersecurity firms are intensifying efforts to detect and mitigate these risks. Insiders' authorized access to networks, familiarity with structures, and security procedures complicate detection [1]. Traditional auditing is ineffective against such threats due to insiders' knowledge of sensitive data locations. Acknowledging this, authoritative bodies like the CERT Insider Threat Center, in conjunction with the National Cybersecurity and Communications Integration Center, offer strategic guidelines for identifying and handling insider threats in corporate environments [2].

Traditionally, detection systems have predominantly relied on rule-based pattern matching. To identify anomalous behaviors among users, experts were required to define all conceivable patterns of normal behavior to distinguish them from malevolent actions. However, with the advent of advanced technologies and the rapid evolution of user behaviors, defining all such patterns has become unfeasible.

Presently, most of the insider threat detection methods mainly revolve around the identification of behavioral traces of internal users, and the construction of corresponding models of internal user behavior to ascertain potential threats [3]. With the advancement of machine learning and deep learning, security professionals are increasingly exploring the use of these technologies to accurately capture malicious threats within massive and complex heterogeneous logs of normal activities. The primary research strands for deep learning-based insider threat detection include few-shot learning, self-supervised learning, deep survival analysis, multi-model learning, etc. [4]. These studies seek to address the challenges of applying deep learning to insider threat detection to enhance detection performance.

Few-shot learning is particularly aligned with the characteristics of insider threats, marked by the scarcity of threatening insiders; it is dedicated to classifying samples from unknown classes with only a few labeled examples [5]. Yuan et al. [6] have combined self-supervised pretraining with metric-based few-shot learning to detect insider threats, outperforming models such as OC-SVM, RNN, etc. The advantages of few-shot learning lie in its ability to perform insider threat detection with a limited number of samples, leveraging prior knowledge. Its limitations stem from the current assumption of a fixed task distribution in few-shot learning. This implies that if new types of attacks, starkly different from observed ones, emerge, the few-shot learning model may fail to detect such anomalies. In the domain of self-supervised learning, the objective is to train models using labels that are easily derived from the input data itself, circumventing the need for human annotation. Zhang et al. [7] developed a method that harmonizes ensemble and self-supervised learning for supervised insider threat detection, achieving up to a 99.2% AUC value on the CERT 4.2 dataset. The advantage of employing self-supervised learning for insider threat detection lies in its potential ability to identify malicious insiders without the use of any labels. However, the creation of self-supervised tasks typically necessitates handcrafted rules tailored for each dataset.

Deep survival analysis is intended to model datasets where the outcome of interest is the time until an event occurs. Alhajjar et al. [8] have devised a novel insider threat detection methodology based on survival analysis techniques, utilizing the Cox proportional hazards model to predict insider threat incidents with greater accuracy, leveraging data such as activity logs, login information, and psychological evaluations. The aim is to predict both the occurrence of insider threat events and their approximate timing. The deployment of deep survival analysis models to capture temporal information of user activities presents great potential for the early detection of insider threats. One of the challenges in applying deep survival analysis models is that they often require extensive event data for training, which can be difficult to amass in significant volumes.

Multi-modal learning amalgamates multiple models to enhance the detection capabilities for insider threats. Liu et al. [9] proposed MUEBA, a multi-modal UEBA system for spatiotemporal analysis, which incorporates individual historical analysis and group behavior analysis to detect insider threats. Utilizing attention-based LSTM to increase the model's sensitivity to abnormal activities, it extends the iForest algorithm in terms of attribute selection and iTree construction. Experimental evaluations on the publicly available CERT-4.2 dataset indicate that our system surpasses any single model in terms of stability and precision. By integrating data across multiple modalities, user patterns can be captured from diverse perspectives, thereby improving detection accuracy. However, within the scope of insider threat detection, the acquisition of multi-modal data, such as psychological data of users, presents challenges due to privacy concerns.

There are also some other approaches for insider threat detection; for example, Moriano P et al. [10] proposed a generic framework for time-series graph analysis for detecting the anomalous behavior of insider users. Happa et al. [11] proposed an insider threat detection method based on a hybrid Gaussian model, which combines the judgment of subject matter experts to improve the accuracy of the detection. Soh C et al. [12] described a method for building an employee sentiment profile using a deep learning approach. The method identifies potential risks of insider threats early by detecting abnormal changes in employee emotions and predicting abnormal behaviors and potential attacks. Zhang et al. [13] used unsupervised learning techniques using deep belief networks (DBNs) to study insider behaviors in order to detect insider threats. Their model is divided into four phases: log collection, log preprocessing (converting data into standard digital format), deep learning of insider behavioral characteristics, and log classification.

However, the aforementioned methods that employ deep learning or other technologies to enhance the capability of insider threat detection face challenges such as high computational resource consumption, the necessity for extensive data labeling, and the rarity of insider threat samples. These challenges have resulted in the application of deep

learning in the field of insider threat detection not yielding as significant performance improvements as its application in other domains.

Le et al. [14] proposed a method that combines supervised and unsupervised learning, aimed at enhancing the capability of logistic regression functions for anomaly detection through the application of various unsupervised outlier detection functions. This hybrid approach does not require large amounts of training data as well as computational consumption of deep learning models, while offering significant performance advantages over traditional insider threat detection methods. However, this approach lacks in-depth consideration when selecting the unsupervised outlier detection functions, and it does not fully take into account the impact of data granularity on the outcomes. We believe that this hybrid approach is very promising, provided that it is improved and combined with a suitable methodology. Therefore, this paper proposes an insider threat detection model enhancement using hybrid algorithms between unsupervised and supervised learning. The model uses an unsupervised algorithm to characterize user behavior, and the output score of the unsupervised outlier scoring function algorithm is regarded as a nonlinear transformation of the original feature space, which provides a richer representation of data features for the original data and enhances the classification effect of the supervised algorithm. The model scores the outliers of each user's daily behavior at different time granularities for user behavior log data in multiple servers. The scoring matrix is constructed, and the scoring matrix is downscaled and fused with the original feature space to form a new feature space to enhance the prediction ability of the supervised classifier on the new feature space, and to realize the detection of abnormal user behaviors under low computational budget. The contributions of this paper are as follows:

1. The behavioral characteristics of insider threat agents are analyzed and the impact of different temporal granularity on the algorithmic model is considered;
2. Unsupervised algorithms are utilized for outlier scoring, where the transformed outlier scores generated by the unsupervised outlier detection function are regarded as a nonlinear transformation of the original feature space, constituting a scoring feature space, which is augmented by downscaling and fusing it with the original feature space;
3. The evaluation of publicly available datasets demonstrates the ability of the proposed method to detect malicious insiders under very low investigation budgets.

2. Related Works

Upon the triumphant integration of machine learning within the realms of intrusion detection and abnormal detection tasks [15,16], there has been a burgeoning application of these methods to the detection of insider threats. This transfer involves adapting abnormal detection methodologies from assorted domains, enabling the extraction of insightful patterns from considerable data pools to pinpoint atypical behavior among insiders. Previous scholarly endeavors have significantly advanced the field by introducing a spectrum of machine learning techniques, including unsupervised, supervised, and semi-supervised approaches [17–19]. Prominently, outlier detection methodologies are extensively harnessed to distinguish aberrations within datasets [20–22], underpinning the critical role of data-driven strategies in the proactive identification of potential insider threats.

However, there are limitations to using supervised outlier detection because outliers in the data usually represent only a small portion of the dataset they are contained in. In addition, unlike traditional classification methods, true values are usually not available in outlier detection. For supervised algorithms, this highly unbalanced dataset and underlabeled data result in limited generalization capabilities for these methods. A number of unsupervised algorithms have been developed for outlier detection. Le et al. [18] proposed an unsupervised learning based abnormal detection method for insider threat detection by using four unsupervised learning methods with different workings and exploring various representations of data with temporal information. Li et al. [23] converted audit logs into grayscale images, and identified anomalies by applying geometric transformations to the

grayscale image. Aldairi et al. [24] used the abnormal scores generated by the unsupervised algorithm from the previous cycle as the trust scores for each user, which were fed into the next cycle of the model, and showed their importance and impact in detecting insiders. Tian et al. [25] characterized users' daily activities from multiple perspectives, using a variety of deep learning algorithms to compute deviations between realistic actions and daily behavioral norms. These methods specialize in exploring information related to outliers, such as local density, global correlation, and hierarchical relationships of unlabeled data. Integrated learning with appropriate combinations of multiple algorithms has been shown to improve outlier detection performance [26].

Integrated learning typically combines multiple base classifiers to create a more powerful algorithm than its counterpart alone. Over the past decades, many integration frameworks have been proposed, such as bagging [27] and stacking [28], which are still used in recent research. While integration methods have been explored in both supervised and unsupervised applications, outlier integration techniques have rarely been investigated. Since outlier detection algorithms are usually unsupervised and lack true labeling, they are not simple to construct. Most existing outlier integration methods are unsupervised and use enhancement methods such as feature bagging. However, the predictive power of supervised methods is usually too dependent on the proportion of labeled data that may be present in the dataset. Therefore, stacking-based outlier integration can be used to utilize label-related information from supervised learning as well as complex data representations from unsupervised outlier methods, and methods that combine unsupervised and supervised algorithms have been fruitful, such as the models of Carcillo [29] and Zhao [30]. And Zhao's model has good insider threat detection performance, so it is used as the main comparison method in this paper.

Based on existing research, this paper extends and improves the work of Carcillo [24] et al. and Micenková [31], and proposes an insider threat detection model enhancement using hybrid algorithms between unsupervised and supervised learning, which enhances the original feature space by stacking various unsupervised outlier detection functions. The unique feature is that instead of relying on the computationally expensive Easy Ensemble [32] method, the approach uses XGBoost 2.0.3 [33] for unbalanced data processing. In addition, this paper evaluates and compares various unsupervised algorithms for efficient computation and considers the computation of outlier scores at different data granularity levels (from daily to weekly time granularity) under different computational budgets. To reduce the risk of added features and model overfitting, the framework also obtains the maximum amount of information in the outlier score matrix through principal component analysis to reduce the risk of model overfitting. Overall, the approach is efficient, easy to implement, and empirically proven effective for insider threat detection.

3. Methodology

In this section, the overall architecture of the enhanced supervised insider threat detection algorithm based on unsupervised outlier scores is presented and its components, unsupervised learning representation, feature space transformation, and supervised detection classification are described in turn.

3.1. General Model Architecture

The overall framework of the model is shown in Figure 1.

The model uses an unsupervised algorithm to score the input data for outliers in the absence of labeled data, discovers patterns and features in the data by learning the intrinsic regularities and structure of the data themselves, and calculates the abnormal score for each data point based on the distribution of the input data, which can reflect the extent to which the data points are outliers in the original feature space, i.e., the extent to which they deviate from the normal data points. The method treats the multiple abnormal scores output by the unsupervised outlier scoring function algorithm as a nonlinear transformation of the original feature space, thus providing new features to the original data to better distinguish

between clustered and normal points. In order to avoid overfitting the model due to too many new features, the system will downsize the new feature space, which can effectively reduce the complexity of the data and extract key features to reduce the complexity of the model. The new feature space after dimensionality reduction is merged with the original feature space to form a new feature space. In this new feature space, the supervised classifier can better identify outliers and normal points, and more accurately identify internal malicious user abnormal behavior.

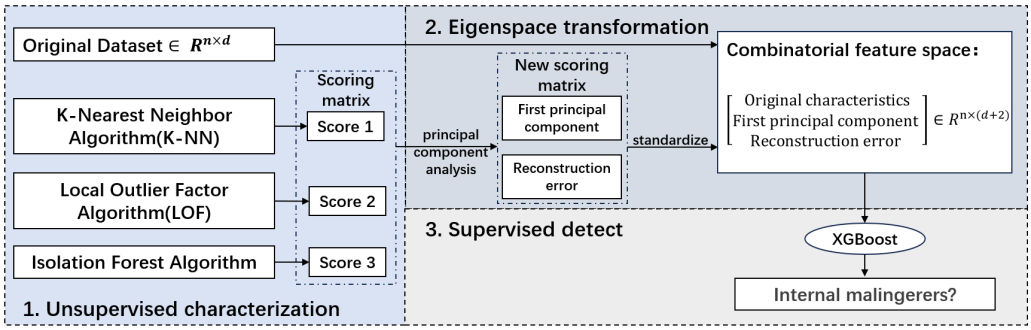


Figure 1. Overall framework diagram of the model.

3.2. Unsupervised Characterization

Outliers identified by unsupervised algorithms may be perceived as learned representations of the original data and can also be understood as a form of unsupervised feature engineering, aimed at enhancing the original feature space. By considering outliers in unsupervised algorithms as a learned representation of the original data, they serve to augment the intrinsic feature space with new dimensions of analysis that may reveal more nuanced insights into the data patterns and behaviors. Let the primitive feature space $X \in R_{n\times d}$ denote the set of n data points with d features. Since outlier detection is a binary classification, the vector $y \in \{0,1\}$ assigns outlier labels, where 1 represents an outlier and 0 represents a normal point. Let L be a set of labeled observations of X as follows:

$$L = \{(x_1, y_1), \ldots, (x_n, y_n)\} \in R_{n\times d} \quad (1)$$

The outlier scoring functions are defined as mapping functions Φ, where each scoring function describes the degree of outlierness of each sample by using the real-valued vector $\Phi i(X) \in R^{n\times 1}$ on the output dataset X as the transformed outlier score. The outlier score function can be used with various unsupervised outlier detection methods. In order to ensure diversity and comprehensiveness, three unsupervised algorithms, k-nearest neighbor (KNN), local outlier factor (LOF), and isolation forest (IF), are selected as the outlier scoring function in this paper. These three algorithms complement each other by evaluating the data from different perspectives.

Specifically, the KNN algorithm evaluates data mainly from the perspective of distance metric [34]. It determines whether a data point is an outlier by calculating the distance between the data point and its neighboring data points. Normal data points are usually close to each other in the data space, while outliers are usually far away from other data points. Therefore, the KNN algorithm is based on the distance metric and can effectively identify outliers. Take binary categorized data as an example, as shown in Figure 2, where c_1 and c_2 are the two categories of the samples respectively, with circles and stars denoting the sample points in the categories, and x is the unknown sample. The samples pointed out by the arrows are the five samples (k = 5) with the closest distance to the unknown sample, and the category of c_1 has the most samples, so the KNN algorithm predicts the category of the unknown sample as c_1.

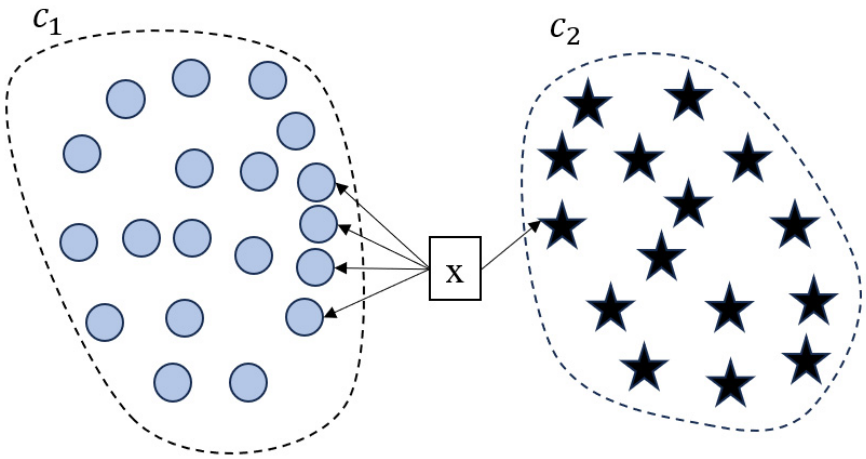


Figure 2. The schematic diagram of the KNN algorithm.

The KNN algorithm flow is as follows: suppose the training set is $S = \{(x_i, y_i), i \in [1, l]\}$, where $x_i \in R^m$, $y_i \in \{c_1, c_2, \ldots, c_n\}$, x_i is the feature vector of the ith sample of the training set, y_i is the label of the ith sample, n is the number of sample categories, l is the number of samples in the training set, and m is the dimension of the feature vector.

1. For an unknown sample x, find the k closest samples to it in the training set S, and define the set of k samples as N_k. The distance is usually calculated using the Euclidean distance, which is given by the following formula:

$$d_{i,j} = \| x_i - x_j \| \tag{2}$$

2. Determine the category y of the unknown sample x according to the principle of majority voting:

$$y = \arg max \sum_{x_i \in N_k} I(y_j, c_i) \tag{3}$$

where $i = 1, \ldots, l, j = 1, \ldots, n$, I is the indicator function:

$$I(x, y) = \begin{cases} 1, x = y \\ 0, x \neq y \end{cases} \tag{4}$$

The LOF algorithm focuses on the evaluation of relative density [26]. It calculates the density of each data point relative to its neighboring data points, which is used to determine the anomalousness of the data points. Normal data points are usually in regions of higher density, while outliers may appear in regions of lower or higher density. The LOF algorithm helps to accurately identify these anomalies through the calculation of relative density. As in Figure 3, set c_3 is a low-density region, using white circles to represent the sample points in it; set c_4 is a high-density region, using black circles to represent the sample points in it. According to the traditional density-based outlier detection algorithm, the distance between point p and the neighboring points in c_4 is less than the distance between any data point in c_3 and its neighboring points, and point p will be regarded as a normal point, whereas locally, point p is a de facto isolated point, and the LOF algorithm can effectively realize outlier detection for this situation. The LOF algorithm can effectively realize outlier detection in this case.

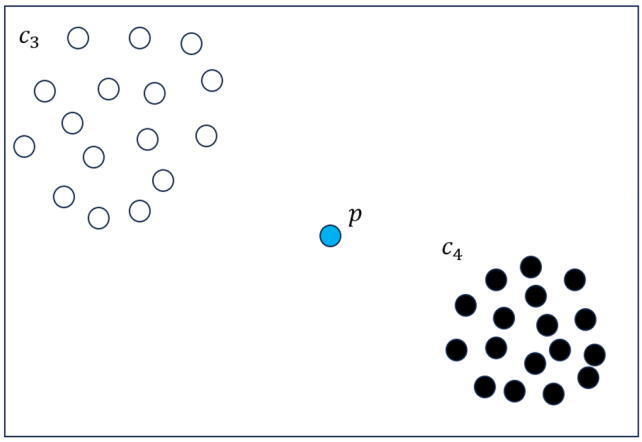


Figure 3. The schematic diagram of the LOF algorithm.

The basic principle of the LOF algorithm is to calculate the local density deviation of a point p from its neighbors in its k-nearest neighbor set as the outlier degree of the point LOF, and the larger the LOF is, the more outlier point p is. The flow of the LOF algorithm is as follows:

1. Calculate the kth reachable distance of each point from each point in the neighborhood:

$$reach_dist_k(o, p) = \max\{d_k(o), d(o, p)\} \tag{5}$$

where $d_k(o)$ is the kth distance from point o in the domain, and $d(o, p)$ is the distance from point o to point p. Point p represents an object that is currently being evaluated as a potential outlier. Point o denotes an object within the dataset, and is typically one of the k-nearest neighbors to p, such as the points in c_3 and c_4. The algorithm harnesses these neighbors to define a local neighborhood and through this neighborhood, it estimates the local density of object p. Should the density of p be significantly lower than its neighbors (these o points), then its local outlier factor (LOF) value would be relatively higher, thereby identifying it as an outlier.

2. Calculate the local kth local reachable density for each point:

$$lrd_k(p) = \frac{1}{\left(\frac{\sum_{o \in N_k(p)} reach_dist_k(o,p)}{|N_k(p)|}\right)} \tag{6}$$

where $N_k(p)$ is the kth distance neighborhood of point p.

3. Calculate the kth local outlier for each point:

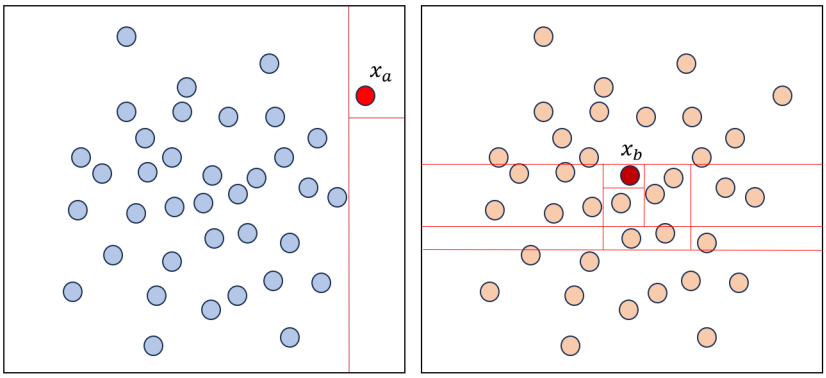
$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} = \frac{\frac{\sum_{o \in N_k(p)} lrd_k(o)}{|N_k(p)|}}{rd_k(p)} \tag{7}$$

where $N_k(p)$ is the kth distance neighborhood of point p.

4. For the data points belonging to the largest n local outliers, output the set of outlier points $O = \{o_1, o_2, \ldots, o_n\}$.

The advantage of the LOF algorithm is that there is no need to know the distribution of the dataset in advance, and it can quantify the degree of outliers by calculating the LOF value to find out the outliers. However, the LOF algorithm also has some limitations: its effect is easily affected by the selection of k value; when the dataset contains clusters with different densities, the points at the boundary of sparse clusters are easily misjudged as outliers by the algorithm; the algorithm involves a large number of data sorting, searching, and distance calculations, which results in a large time cost when facing a large dataset.

The isolation forest algorithm (iForest) is a special abnormal detector because it uses an isolation mechanism to detect anomalies [35]. In isolated forests, anomalies are defined as outliers that are easily foxed, which can be interpreted as points that are sparsely distributed and far away from dense populations, and in the feature space, the regions with right-scattered scores indicate that the solution rate of an event occurring in that region is very low, and thus the data that fall in these regions can be considered anomalous. Figure 4 shows the process of training the subsample with cuts. Point x_a in Figure 4a falls in a region with sparsely distributed edges, and was "isolated" after only two cuts. In contrast, point x_b in Figure 4b falls in a region of higher density, and thus cuts many times before it is partitioned into a separate subspace.



(a) Isolating x_a (b) Isolating x_b

Figure 4. The schematic diagram of the iForest algorithm. (**a**) shows the process of isolating the anomaly sample x_a from the sample points. (**b**) shows the process of separating the anomaly sample x_b from the sample points.

The flow of the iForest algorithm is as follows:

1. Assuming that the data contain a total of N data points, then the maximum height of the isolated trees that make up the isolated forest is $N-1$, at which point the path of the isolated forest cutting the line-loss dataset can be computed according to the following equation:
$$f(x) = \ln(x) + 0.577 \tag{8}$$
where $f(x)$ is the path from leaf node x to the root node of the isolated tree.
2. Then, based on the similarity between the isolated tree and the binary search tree, the length of the cut path of the isolated tree can be calculated with the expression as follows:
$$R(N) = 2f(N-1) - \left(\frac{2(N-1)}{N}\right) \tag{9}$$
where $R(N)$ denotes the average length of the isolated tree cut paths derived from the binary search tree.
3. Finally, based on the above equation, the abnormal score of the data is calculated with the following expression:
$$F(x,N) = 2^{\frac{E(f(x))}{R(N)}} \tag{10}$$
where $F(x,N)$ denotes the abnormal score of the medium data. If the requested $F(x,N)$ tends to 0, it indicates that the data node is a normal node; on the contrary, if the requested $F(x,N)$ tends to 1, it is considered that there is an obvious abnormality in the data node, and it will be judged as abnormal data.

Choosing these three different types of algorithms ensures the assessment of data anomalies from multiple perspectives, improving the robustness and generalization performance of the integration. If abnormal detection algorithms with similar perspectives are selected, they may produce similar outputs with a lack of diversity, and may even

record similar errors while imposing an unnecessary computational burden. Based on these considerations, these three unsupervised algorithms were finally selected to obtain more comprehensive and accurate outlier scores.

3.3. Eigenspace Transformation

The KNN, LOF, and isolation forest from the unsupervised characterization process are combined to generate k outlier scoring functions to construct a transformation function matrix: $\Phi = [\Phi_1, \ldots, \Phi_K]$, which generates an outlier scoring matrix for the original feature space X on the k outlier scoring matrices for each of the basic scoring functions. $\Phi(\cdot)$ is applied to the original dataset X, and the outlier scoring matrix $\Phi(\mathrm{X})$ is as follows:

$$\Phi(\mathrm{X}) = \left[\Phi_1(X)^T, \ldots, \Phi_K(X)^T\right] \in R^{n \times k} \tag{11}$$

The abnormal scoring matrix $\Phi(\mathrm{X})$ is downscaled using principal component analysis. The principle is to normalize and transform the original dataset $\Phi(\mathrm{X})$ into the matrix $\Phi(\mathrm{T})$, $T = XW$, where the ith column of the matrix $\Phi(\mathrm{W})$, W_i is the ith eigenvector of $X^T\mathrm{X}$, and the m linearly uncorrelated variables in $\Phi(\mathrm{T})$ are called principal components. We obtain only its first principal component set $\mathrm{PC}_1 = {\mathrm{W}_1}^T\mathrm{x}$ with the first principal component minimum reconstruction error $\mathrm{RE}_1 = \| \mathrm{x} - \mathrm{W}_1{\mathrm{W}_1}^T \|$. The purpose is to retain the maximum amount of information in the outlier score matrix and minimize the added features to reduce the risk of model overfitting. Then, the newly generated scoring matrix is normalized using Z-Score so that the scoring results produced by different algorithms can be effectively evaluated. Finally, the new feature space constituted by combining the dimensionality reduced outlier score matrix with the original feature set. This novel feature space furnishes additional information, facilitating the supervised classification algorithm to more accurately distinguish between normal and anomalous behavior. Finally, leveraging the newly derived feature space, an XGBoost classifier is employed for supervised learning to predict anomalous activities.

3.4. Supervised Detect

Extreme gradient boosting, often referred to as XGBoost, is a tree-based integration method developed by Chen [33]. It is a scalable, high-performance implementation of gradient boosting trees specifically designed to optimize computational speed and model performance. Nowadays, it is widely used in medical detection [36], geological exploration [37], and other fields. However, XGBoost does not work well with unbalanced datasets such as credit card risk prediction, cyber human intrusion detection, and medical testing, and often needs to be combined with sampling techniques to improve performance [38–40]. Data imbalance occurs when a class (or subset) in a dataset represents a very small percentage of the overall class. Data imbalance leads to the usual degradation of classifier performance.

Insider threat detection is a particularly challenging binary classification task due to the inherent data imbalance; outliers, which represent potential threats, typically constitute a relatively small proportion of the dataset. This scarcity increases the difficulty of accurately identifying such outliers. To address this imbalance, techniques such as bootstrap aggregation and the Easy Ensemble method are commonly employed within the realm of outlier detection. The Easy Ensemble approach creates several balanced subsets by downsampling the majority class, and amalgamates the decisions from the base classifier for each subset.

Nevertheless, the implementation of these methods can be computationally expensive and their performance may vary across different problems. In contrast to other conventional enhancement algorithms, such as those involving gradient boosting, XGBoost incorporates a regularization component that mitigates the risk of overfitting. This results in not only more precise predictions but also expedited model execution. Owing to these advantages, XGBoost was selected as the preferred supervised classifier in this study, offering a sub-

stitute for Easy Ensemble that enhances computational efficiency. The operation of the XGBoost algorithm is characterized as follows.

For a dataset $\mathcal{D} = \{(x_i, y_i)\}(|\mathcal{D}| = n, x_i \in R^m, y_i \in R)$ with n samples and m features, the predicted value of the sample $\phi(x_i)$ is the sum of the results $f_i(x_i)$ across the decision trees. Assuming that K decision trees are generated, the sample predictions are:

$$\phi(x_i) = \sum_{i=1}^{K} f_i(x_i) \tag{12}$$

Then, the decision tree generation process is described. The loss function of the algorithm is:

$$L(\phi) = \sum_i l(y_i, \widehat{y}_i) + \sum_k \Omega(f_k) \tag{13}$$

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \| W \|^2 \tag{14}$$

where $l(y_i, \widehat{y}_i)$ is the sample loss, $\Omega(f_k)$ is the regularization function, the penalty coefficient γ reduces the number of leaf nodes of the tree T, $W = (w_1, w_2, w_3, \cdots, w_T)$ is the vector composed of all the leaf node weights, and the λ coefficient restricts the size of the leaf node weights.

The decision tree f_t generated in the tth iteration reduces the loss after the $t-1$th iteration, as shown in the following equation:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \widehat{y}_i^{(t-1)} + f_t(x_i)\right) + \sum_{i=1}^{t-1} \Omega(f_i) + \Omega(f_t) \tag{15}$$

A second-order Taylor expansion of the loss function is performed as shown in Equations (16) and (17):

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} \left[l\left(y_i, \widehat{y}_i^{(t-1)}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \sum_{i=1}^{t-1} \Omega(f_i) + \Omega(f_t) \tag{16}$$

$$g_i = \partial_{\widehat{y}_i^{(t-1)}} l\left(y_i, \widehat{y}_i^{(t-1)}\right), h_i = \partial^2_{\widehat{y}_i^{(t-1)}} l\left(y_i, \widehat{y}_i^{(t-1)}\right) \tag{17}$$

where leaf $(x_i) = j$ means that the sample x_i is classified to the jth leaf node by the decision tree f_t, and the result of the decision tree $f_t(x_i)$ is the weight w_j of the leaf node. After removing the constants $l\left(y_i, \widehat{y}_i^{(t-1)}\right)$ and $\sum_{i=1}^{t-1} \Omega(f_i)$, the loss of the decision tree generated by the tth iteration to classify the sample to the Tth leaf node is:

$$\widetilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \tag{18}$$

where $I_j = \{i \mid \text{leaf}(x_i) = j\}$ denotes the index of the sample in the jth leaf node. The loss of leaf node j is partialized with respect to w_j to find the weight of leaf node w_j^* that minimizes the loss:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{19}$$

Equation (19) is called the leaf node weighting formula. Substituting w_j^* into the leaf node loss function, the loss of leaf node j is obtained as:

$$\mathcal{L}_{before} = -\frac{1}{2} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma \tag{20}$$

Taking Equation (20) as the structural loss of the node, the samples in the node are divided into two nodes according to the value σ of the ith feature, the left node $S_{\mathrm{L}} = \{x_i \mid \sigma(x_i) \leqslant \theta\}$ and the right node $S_{\mathrm{R}} = \{x_i \mid \sigma(x_i) > \theta\}$, and the value of θ is called the splitting point. The samples are indexed as I_{L} and I_{R}, and the structural losses after splitting are:

$$\mathcal{L}_{\text{after}} = -\frac{1}{2}\left(\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda}\right) + 2\gamma \tag{21}$$

So the structural gain obtained by leaf node j with θ of the ith feature as the splitting point is:

$$\mathcal{L}_{\text{split}} = \mathcal{L}_{\text{before}} - \mathcal{L}_{\text{after}} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda}\right] - \gamma \tag{22}$$

Equation (22) is called split gain formula [28], which is used to measure how good a split node is. The splitting coefficient γ is used to reduce the complexity of the structure and prevent the decision tree from overfitting. The XGBoost algorithm searches for splitting points in each feature according to the splitting gain formula in order to split; it then calculates the weights of the leaf nodes in order to update the sample prediction value after the construction of the decision tree is completed, and then constructs a decision tree with the new sample prediction value in the next iteration; finally, it takes the sum of the prediction results of multiple decision trees as the final prediction value.

Overall, our model utilizes a trio of unsupervised algorithms to concurrently evaluate outlier scores, endowing it with robust anomaly detection capabilities. For attackers who may deliberately attempt to circumvent detection mechanisms, as long as their behavior significantly deviates from normal patterns, our model is capable of identifying such irregularities. These detected actions are then subjected to a supervised XGBoost classifier for further validation and categorization. Consequently, our model exhibits exceptional robustness, equipping it to cope with adversaries employing diverse strategies.

4. Experiment and Result Analysis

4.1. Experimental Platforms and Dataset

The model is deployed in our lab's network traffic security monitoring platform, Lenovo SR660 V2 (made in China), Intel Xeon Silver 4314, 2.4 GHz, 16-core, 32 GB memory, runs on Linux and Ubuntu systems, and is deployed in a distributed manner when heavy traffic passes through, while monitoring the platform's data in the programming language Python 3.9.

Our model has good generalization ability because all three unsupervised algorithms and the supervised XGBoost algorithm have good generalization, and the anomalous features can be extracted by the powerful outlier detection algorithm when facing different datasets. In order to verify the effectiveness of the model in real insider threat detection, the CERT r4.2 [41] insider threat dataset is used in this paper to verify the performance of the model. The CERT r4.2 is a publicly available dataset collected from real-world enterprise environments constructed by Carnegie Mellon University and ExactData to simulate the behavior of malicious insider threat users of an enterprise in terms of intellectual property or security information theft, fraud, and sabotage, for the purpose of researching, developing, and testing insider threat mitigation methods. The dataset records logs of users' operations in multiple domains, as well as users' positional and psychographic information, and sets up different scenarios to simulate the behavior of insider threat users. It contains more than 20 GB of various syslog files, recording all the activities of 1000 users over a 500-day period (including weekends).

The dataset is extremely unbalanced in terms of the number of malicious insiders; it includes only 70 malicious insiders. It is also worth mentioning that there are some missing

data behaviors, such as some users powering off their computers directly after using their devices, and there will be no logout operations after logging in. In this regard, we are interested in extracting the counts of different types of operations performed by users on a weekly or daily basis using the time-based preprocessing method mentioned by Le [42], in order to construct feature vectors representing user activities and profile information at different levels of granularity.

4.2. Performance Comparisons

We evaluate the adaptability of our method on the CERT r4.2 insider threat dataset. To demonstrate the effectiveness of our model (KLI-XG), it is compared with a variety of other excellent anomaly detection methods under different investigative budgets, including (a) KNN, (b) LOF, (c) OneClass-SVM, (d) iForest, and (e) XGBOD.

Within the realm of internal threat detection, the choice of temporal granularity significantly affects the detection algorithm's sensitivity to user behavior patterns, which can subsequently impact the accuracy of the detection results. A granularity that is too coarse may neglect or misinterpret subtle anomalous behaviors; conversely, an overly fine granularity may result in numerous inconsequential alerts, leading to information overload. In monitoring for anomalous activities among insider users, various studies explore differing temporal windows, such as hourly, daily, or weekly, to pinpoint anomalies. The objective of such research is to identify an optimal balance that allows for the timely discovery of anomalies without the inconvenience posed by the sheer volume and complexity of data. For example, certain methodologies are geared toward identifying anomalies within short time frames, like unusual activities that occur within several consecutive hours—potentially indicative of more precise threat patterns. Conversely, other approaches utilize extended time cycles to look for anomalies that may require a longer duration to become evident. Regarding the three unsupervised machine learning algorithms employed in our model, they are more suited for the short-term detection of anomalies within a multitude of datasets. We propose that opting for daily and weekly intervals stands as a balanced choice.

We selected the ROC-AUC value and accuracy rate as the evaluation metrics for various models. The ROC-AUC value is a statistical metric used to measure the performance of classification models. ROC stands for Receiver Operating Characteristic curve, and AUC represents the Area Under the Curve. The ROC curve is a two-dimensional graph that displays the relationship between the True Positive Rate (TPR) and the False Positive Rate (FPR) at varying classification thresholds, reflecting the performance of the classification model. The AUC value, which is the size of the area under the ROC curve, offers a method to quantify the average performance of the model. AUC values range from 0 to 1, with an AUC of 1 indicating a perfect classifier and an AUC of 0.5 signifying no better than random guessing. Typically, the higher the AUC, the better the model's classification performance. The results of ROC-AUC and accuracy are shown in Tables 1 and 2.

Table 1. ROC-AUC results for abnormal detection with different data shares on the CERT dataset.

Time Granularity	Percentage of Data	KNN	LOF	OC-SVM	iForest	XGBOD	KLI-XG
Day	0.1%	0.6313	0.8391	0.9747	0.7273	0.9848	0.9781
	1.0%	0.6596	0.8131	0.9947	0.7414	0.9980	0.9984
	5.0%	0.6240	0.7643	0.9872	0.7543	0.9983	0.9985
	10.0%	0.6004	0.8554	0.9930	0.7448	0.9953	0.9982
	20.0%	0.6111	0.7444	0.9928	0.8194	0.9949	0.9980
Week	0.1%	0.7246	0.8375	0.9625	0.9375	0.9771	0.9882
	1.0%	0.7169	0.8225	0.9878	0.7531	0.9906	0.9981
	5.0%	0.7333	0.7852	0.9837	0.8054	0.9950	0.9961
	10.0%	0.7073	0.8655	0.9827	0.8159	0.9941	0.9948
	20.0%	0.7565	0.8074	0.9871	0.8142	0.9945	0.9980

Table 2. Accuracy results for abnormal detection with different data shares on the CERT dataset.

Time Granularity	Percentage of Data	KNN	LOF	OC-SVM	iForest	XGBOD	KLI-XG
Day	0.1%	0.0000	0.0184	0.3330	0.0058	0.3330	0.4516
	1.0%	0.0432	0.0896	0.6250	0.0344	0.5833	0.7083
	5.0%	0.1054	0.1267	0.6902	0.1597	0.6772	0.7497
	10.0%	0.1223	0.2014	0.7318	0.4048	0.7441	0.8179
	20.0%	0.3145	0.3267	0.7804	0.6654	0.7778	0.8447
Week	0.1%	0.0026	0.0056	0.2095	0.0028	0.4317	0.4437
	1.0%	0.1137	0.0909	0.4545	0.0344	0.6545	0.6978
	5.0%	0.1962	0.1813	0.5577	0.1364	0.7023	0.7603
	10.0%	0.3827	0.2408	0.6340	0.3814	0.7854	0.7922
	20.0%	0.5834	0.3449	0.7463	0.6690	0.8089	0.8612

The data in Table 1 show that our model maintains a high performance in terms of ROC-AUC values, comparable to the XGBOD algorithm, and exhibits some performance advantages at most temporal granularities. The data share in the table refers to the proportion of data extracted from the dataset, ranging from 0.1% to 20%. In particular, at 20% data share, our method achieves a ROC-AUC value of 0.998 at weekly or daily time granularity. Compared to Zhang et al.'s model based on self-supervised learning [7] mentioned in the introduction, our AUC has the same advantage compared to their highest AUC value of 0.992. We observe a more substantial improvement in the score compared to a single method utilizing KNN, LOF, and Iforest, which confirms the idea that a combination of multiple unsupervised learning algorithms can improve performance. In terms of accuracy, the data in Table 2 show that our model outperforms other algorithms such as XGBOD at different temporal granularities, with a maximum lead of 12.5% and an average lead of 6.3% at 1.0% daily granularity. Finally, our method achieves 86.12% accuracy with 20% data percentage, which can be interpreted as using only 20% of the traffic data to get 86.12% detection accuracy, which puts less pressure on the server. When all the data are analyzed by the server, this will put more pressure on the server, so the advantage of our method is that it can use a lower computational budget to achieve a good detection accuracy.

Experimental results show that our method outperforms other methods, achieving high levels of ROC-AUC and accuracy despite low investigation budgets. Specifically, we employ an unsupervised algorithm for outlier scoring, which effectively reveals malicious insider behavior and provides XGboost with auxiliary information to identify insiders. This approach is unique in that it utilizes unsupervised learning algorithms to automatically detect outliers without human intervention, which not only improves efficiency but also reduces human error. These data fully demonstrate the accuracy and effectiveness of our method in identifying insider threats. Compared to other methods, it exhibits higher accuracy and more stable operational performance under low investigation budgets.

4.3. Ablation Experiments

To assess the validity of outlier scores, we designed ablation experiments comparing the following scenarios:

1. Original-XG: no outlier scoring is used, and the XGboost classifier is used for the original features;
2. Score-XG: the XGboost classifier is used after using outlier scoring matrices combined with the original features;
3. KLI-XG: the KLI-XG is used using the algorithm proposed in this paper, i.e., using principal component analysis to process the outlier scoring matrix again after combining it with the original features, and finally using XGboost for classification;
4. All-XG: using all the data, i.e., combining the original features, outlier scoring matrix, and scoring matrix after principal component analysis dimensionality reduction, and finally using XGboost for classification.

The results of the experiments are shown in Tables 3 and 4.

Table 3. ROC-AUC results of ablation experiments.

Time Granularity	Percentage of Data	Orig-XG	Score-XG	All-XG	KLI-XG
Day	0.1%	0.9428	0.8906	0.8872	0.9781
	1.0%	0.9983	0.9539	0.9982	0.9984
	5.0%	0.9884	0.9982	0.9879	0.9985
	10.0%	0.9932	0.9197	0.9980	0.9982
	20.0%	0.9979	0.9481	0.9982	0.9980
Week	0.1%	0.8458	0.8646	0.9012	0.9882
	1.0%	0.9466	0.9095	0.9425	0.9981
	5.0%	0.9837	0.9668	0.9961	0.9961
	10.0%	0.9946	0.9941	0.9917	0.9948
	20.0%	0.9945	0.9963	0.9947	0.9980

Table 4. Accuracy results of ablation experiments.

Time Granularity	Percentage of Data	Orig-XG	Score-XG	All-XG	KLI-XG
Day	0.1%	0.4467	0.4481	0.4379	0.4516
	1.0%	0.7083	0.6874	0.6667	0.7083
	5.0%	0.7135	0.7202	0.7254	0.7497
	10.0%	0.7813	0.7974	0.7488	0.8179
	20.0%	0.8285	0.8103	0.8336	0.8447
Week	0.1%	0.4154	0.4213	0.1276	0.4437
	1.0%	0.6667	0.6050	0.6061	0.6978
	5.0%	0.7218	0.7268	0.7346	0.7603
	10.0%	0.7728	0.7212	0.7631	0.7922
	20.0%	0.8275	0.8169	0.8433	0.8612

Combining the data in Tables 3 and 4, the results of the ablation experiments show that the KLI-XG method using outlier scoring helps to identify insider threats with better performance. Outlier scoring is able to detect anomalous data that are inconsistent with normal behavior, which may indicate a potential insider threat. By identifying and analyzing outliers, this method can effectively reveal the malicious or abnormal behavior of insiders, and detect and analyze abnormal behavior in a more focused manner, thus improving the accuracy and performance of identifying insider threats. It is interesting to note that if we use all the scoring data from the model construction, i.e., the outlier scoring matrix composed by the unsupervised algorithm with the reduced scoring matrix, there is a small decrease in performance, which we believe is due to some of the data being duplicated in the characterization, which is in line with our pre-experiment assumptions.

5. Conclusions

This paper proposes an insider threat detection model enhancement using hybrid algorithms between unsupervised and supervised learning for detecting malicious insiders in insider threat datasets. The method consists of three parts:

1. Augmenting the data representation using an unsupervised outlier detection algorithm;
2. Processing the outlier score matrix to constitute the augmented feature space;
3. Applying an XGBoost classifier to predict the augmented feature space.

Experimental data from the CERT R4.2 dataset demonstrate that our approach achieves significantly improved results with up to 12.5% higher accuracy compared to other insider threat detection methods that combine supervised and unsupervised learning. Specifically, applying various established unsupervised outlier detection algorithms on the raw data generates outlier scores with potentially better representations. Furthermore, combining

these outlier scores with the original feature space after dimensionality reduction can improve overall outlier prediction. This study extends previous research by showing that even using few outlier scores can significantly improve insider malingerer detection rates. In addition, this paper designs and tests a comparison of insider threat detection algorithms under different computational budgets. Compared to other good anomaly detection methods, the method proposed in this paper provides better predictive power, achieving 86.12% accuracy with only 20% of the computational budget and possessing the highest AUC value of 0.998.

As adversarial organizations enhance their operational sophistication, the complexity of detecting their incursions correspondingly increases, necessitating the application of more advanced feature extraction and construction techniques. Paralleling the approach of augmenting complexity within neural network architectures, our model is designed for modular enhancements or the incorporation of novel methodologies at pivotal junctions to effectively counter sophisticated threats. Our research endeavors will next pivot towards addressing the deterioration in classifier performance stemming from data imbalances, with a distinct aim to identify or develop strategies resilient to such imbalances. Further inquiries will delve into the prospect of integrating elevated unsupervised algorithmic approaches within our framework, with a resolute goal to improve the model's precision and reliability.

Author Contributions: Conceptualization, J.Y.; Methodology, J.Y. and Y.T.; Software, Y.T.; Formal analysis, J.Y. and Y.T.; Investigation, J.Y.; Resources, J.Y. and Y.T.; Data curation, Y.T.; Writing—original draft, Y.T.; Writing—review and editing, J.Y. and Y.T.; Funding acquisition, J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is contained within the article.

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

References

1. Al-Mhiqani, M.N.; Ahmad, R.; Abidin, Z.Z.; Yassin, W.; Hassan, A.; Abdulkareem, K.H.; Ali, N.S.; Yunos, Z. A review of insider threat detection: Classification, machine learning techniques, datasets, open challenges, and recommendations. *Appl. Sci.* **2020**, *10*, 5208. [CrossRef]
2. Kwon, D.; Kim, H.; Kim, J.; Suh, S.C.; Kim, I.; Kim, K.J. A survey of deep learning-based network anomaly detection. *Clust. Comput.* **2019**, *22*, 949–961. [CrossRef]
3. Xiong, W.; Lagerström, R. Threat modeling—A systematic literature review. *Comput. Secur.* **2019**, *84*, 53–69. [CrossRef]
4. Yuan, S.; Wu, X. Deep learning for insider threat detection: Review, challenges and opportunities. *Comput. Secur.* **2021**, *104*, 102221. [CrossRef]
5. Wang, Y.; Yao, Q.; Kwok, J.T.; Ni, L.M. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 63. [CrossRef]
6. Yuan, S.; Zheng, P.; Wu, X.; Tong, H. Few-shot insider threat detection. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Virtual Event, 19–23 October 2020; pp. 2289–2292.
7. Zhang, C.; Wang, S.; Zhan, D.; Yu, T.; Wang, T.; Yin, M. Detecting Insider Threat from Behavioral Logs Based on Ensemble and Self-Supervised Learning. *Secur. Commun. Netw.* **2021**, *2021*, 4148441. [CrossRef]
8. Alhajjar, E.; Bradley, T. Survival analysis for insider threat: Detecting insider threat incidents using survival analysis techniques. *Comput. Math. Organ. Theory* **2021**, *28*, 335–351. [CrossRef]
9. Liu, J.; Zhang, J.; Du, C.; Wang, D. MUEBA: A Multi-model System for Insider Threat Detection. In Proceedings of the International Conference on Machine Learning for Cyber Security, Guangzhou, China, 2–4 December 2022; Springer Nature: Cham, Switzerland, 2022; pp. 296–310.
10. Moriano, P.; Pendleton, J.; Rich, S.; Camp, L.J. Insider threat event detection in user-system interactions. In Proceedings of the 2017 International Workshop on Managing Insider Security Threats, Dallas, TX, USA, 30 October 2017; pp. 1–12.
11. Happa, J. Insider-threat detection using gaussian mixture models and sensitivity profiles. *Comput. Secur.* **2018**, *77*, 838–859.
12. Soh, C.; Yu, S.; Narayanan, A.; Duraisamy, S.; Chen, L. Employee profiling via aspect-based sentiment and network for insider threats detection. *Expert Syst. Appl.* **2019**, *135*, 351–361. [CrossRef]

13. Zhang, J.; Chen, Y.; Ju, A. Insider threat detection of adaptive optimization DBN for behavior logs. *Turk. J. Electr. Eng. Comput. Sci.* **2018**, *26*, 792–802. [CrossRef]
14. Le, D.C.; Zincir-Heywood, A.N. Evaluating insider threat detection workflow using supervised and unsupervised learning. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24 May 2018; pp. 270–275.
15. Yu, K.; Tan, L.; Mumtaz, S.; Al-Rubaye, S.; Al-Dulaimi, A.; Bashir, A.K.; Khan, F.A. Securing critical infrastructures: Deep-learning-based threat detection in IIoT. *IEEE Commun. Mag.* **2021**, *59*, 76–82. [CrossRef]
16. Li, B.; Wu, Y.; Song, J.; Lu, R.; Li, T.; Zhao, L. DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 5615–5624. [CrossRef]
17. Liu, L.; Chen, C.; Zhang, J. Unsupervised insider detection through neural feature learning and model optimisation. In *Network and System Security, Proceedings of the 13th International Conference, NSS 2019, Sapporo, Japan, 15–18 December 2019*; Proceedings 13; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 18–36.
18. Le, D.C.; Zincir-Heywood, N. Anomaly detection for insider threats using unsupervised ensembles. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 1152–1164. [CrossRef]
19. Da Costa, K.A.; Papa, J.P.; Lisboa, C.O.; Munoz, R.; de Albuquerque, V.H.C. Internet of Things: A survey on machine learning-based intrusion detection approaches. *Comput. Netw.* **2019**, *151*, 147–157. [CrossRef]
20. Blázquez-García, A.; Conde, A.; Mori, U.; Lozano, J.A. A review on outlier/anomaly detection in time series data. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 56. [CrossRef]
21. Smiti, A. A critical overview of outlier detection methods. *Comput. Sci. Rev.* **2020**, *38*, 100306. [CrossRef]
22. Boukerche, A.; Zheng, L.; Alfandi, O. Outlier detection: Methods, models, and classification. *ACM Comput. Surv. (CSUR)* **2020**, *53*, 55. [CrossRef]
23. Li, D.; Yang, L.; Zhang, H.; Wang, X.; Ma, L.; Xiao, J. Image-based insider threat detection via geometric transformation. *Secur. Commun. Netw.* **2021**, *2021*, 1777536. [CrossRef]
24. Aldairi, M.; Karimi, L.; Joshi, J. A trust aware unsupervised learning approach for insider threat detection. In Proceedings of the 2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI), Los Angeles, CA, USA, 30 July–1 August 2019; pp. 89–98.
25. Tian, B.; Su, Q.; Yin, J. Anomaly detection by leveraging incomplete anomalous knowledge with anomaly-aware bidirectional gans. *arXiv* **2022**, arXiv:2204.13335.
26. Alghushairy, O.; Alsini, R.; Soule, T.; Ma, X. A review of local outlier factor algorithms for outlier detection in big data streams. *Big Data Cogn. Comput.* **2020**, *5*, 1. [CrossRef]
27. González, S.; García, S.; Del Ser, J.; Rokach, L.; Herrera, F. A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Inf. Fusion* **2020**, *64*, 205–237. [CrossRef]
28. Zhang, H.; Li, J.L.; Liu, X.M.; Dong, C. Multi-dimensional feature fusion and stacking ensemble mechanism for network intrusion detection. *Future Gener. Comput. Syst.* **2021**, *122*, 130–143. [CrossRef]
29. Carcillo, F.; Le Borgne, Y.-A.; Caelen, O.; Kessaci, Y.; Oblé, F.; Bontempi, G. Combining unsupervised and supervised learning in credit card fraud detection. *Inf. Sci.* **2021**, *557*, 317–331. [CrossRef]
30. Zhao, Y.; Hryniewicki, M.K. Xgbod: Improving supervised outlier detection with unsupervised representation learning. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
31. Micenková, B.; McWilliams, B.; Assent, I. Learning outlier ensembles: The best of both worlds–supervised and unsupervised. In Proceedings of the ACM SIGKDD 2014 Workshop on Outlier Detection and Description under Data Diversity (ODD2), New York, NY, USA, 24 August 2014.
32. Bendou, Y.; Hu, Y.; Lafargue, R.; Lioi, G.; Pasdeloup, B.; Pateux, S.; Gripon, V. Easy—Ensemble augmented-shot-y-shaped learning: State-of-the-art few-shot classification with simple components. *J. Imaging* **2022**, *8*, 179. [CrossRef]
33. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
34. Abu Alfeilat, H.A.; Hassanat, A.B.; Lasassmeh, O.; Tarawneh, A.S.; Alhasanat, M.B.; Salman, H.S.E.; Prasath, V.S. Effects of distance measure choice on k-nearest neighbor classifier performance: A review. *Big Data* **2019**, *7*, 221–248. [CrossRef]
35. Jiang, J.; Li, T.; Chang, C.; Yang, C.; Liao, L. Fault diagnosis method for lithium-ion batteries in electric vehicles based on isolated forest algorithm. *J. Energy Storage* **2022**, *50*, 104177. [CrossRef]
36. Shi, H.; Wang, H.; Huang, Y.; Zhao, L.; Qin, C.; Liu, C. A hierarchical method based on weighted extreme gradient boosting in ECG heartbeat classification. *Comput. Methods Programs Biomed.* **2019**, *171*, 1–10. [CrossRef]
37. Ding, H.; Liu, K.; Chen, X.; Xiong, L.; Tang, G.; Qiu, F.; Strobl, J. Optimized segmentation based on the weighted aggregation method for loess bank gully mapping. *Remote Sens.* **2020**, *12*, 793. [CrossRef]
38. Thabtah, F.; Hammoud, S.; Kamalov, F.; Gonsalves, A. Data imbalance in classification: Experimental evaluation. *Inf. Sci.* **2020**, *513*, 429–441. [CrossRef]
39. Abad, Z.S.H.; Maslove, D.M.; Lee, J. Predicting discharge destination of critically ill patients using machine learning. *IEEE J. Biomed. Health Inform.* **2020**, *25*, 827–837. [CrossRef] [PubMed]

40. Chang, Y.-C.; Chang, K.-H.; Wu, G.-J. Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. *Appl. Soft Comput.* **2018**, *73*, 914–920. [CrossRef]
41. Glasser, J.; Lindauer, B. Bridging the gap: A pragmatic approach to generating insider threat data. In Proceedings of the 2013 IEEE Security and Privacy Workshops, San Francisco, CA, USA, 23–24 May 2013; pp. 98–104.
42. Le, D.C.; Zincir-Heywood, N.; Heywood, M.I. Analyzing data granularity levels for insider threat detection using machine learning. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 30–44. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.