Internal Evaluation of Unsupervised Outlier Detection

HENRIQUE O. MARQUES, University of São Paulo
RICARDO J. G. B. CAMPELLO, University of Newcastle
JÖRG SANDER, University of Alberta
ARTHUR ZIMEK, University of Southern Denmark

Although there is a large and growing literature that tackles the unsupervised outlier detection problem, the unsupervised *evaluation* of outlier detection results is still virtually untouched in the literature. The so-called internal evaluation, based solely on the data and the assessed solutions themselves, is required if one wants to statistically validate (in absolute terms) or just compare (in relative terms) the solutions provided by different algorithms or by different parameterizations of a given algorithm in the absence of labeled data. However, in contrast to unsupervised cluster analysis, where indexes for internal evaluation and validation of clustering solutions have been conceived and shown to be very useful, in the outlier detection domain, this problem has been notably overlooked. Here we discuss this problem and provide a solution for the internal evaluation of outlier detection results. Specifically, we describe an index called Internal, Relative Evaluation of Outlier Solutions (IREOS) that can evaluate and compare different candidate outlier detection solutions. Initially, the index is designed to evaluate binary solutions only, referred to as *top-n* outlier detection results. We then extend IREOS to the general case of non-binary solutions, consisting of outlier detection scorings. We also statistically adjust IREOS for chance and extensively evaluate it in several experiments involving different collections of synthetic and real datasets.

CCS Concepts: • **Information systems** → **Data mining**; • **Computing methodologies** → *Anomaly detection*;

Additional Key Words and Phrases: Outlier detection, unsupervised evaluation, validation

ACM Reference format:
Henrique O. Marques, Ricardo J. G. B. Campello, Jörg Sander, and Arthur Zimek. 2020. Internal Evaluation of Unsupervised Outlier Detection. *ACM Trans. Knowl. Discov. Data* 14, 4, Article 47 (June 2020), 42 pages.
https://doi.org/10.1145/3394053

This project was partially funded by FAPESP (Brazil—Grants #2015/06019-0 and #2017/04161-0), CNPq (Brazil), and NSERC (Canada).

Authors' addresses: H. O. Marques, Institute of Mathematics and Computer Sciences (ICMC), University of São Paulo, CEP 13566-590, São Carlos, SP, Brazil; email: hom@icmc.usp.br; R. J. G. B. Campello, School of Mathematical and Physical Sciences (MAPS), University of Newcastle, University Drive, Callaghan, NSW 2308, Australia; email: ricardo.campello@newcastle.edu.au; J. Sander, Department of Computing Science, University of Alberta, T6G 2E8, Edmonton, AB, Canada; email: jsander@ualberta.ca; A. Zimek, Department of Mathematics and Computer Science (IMADA), University of Southern Denmark, Campusvej 55, DK-5230, Odense, Denmark; email: zimek@imada.sdu.dk.

1556-4681/2020/06-ART47 $15.00
https://doi.org/10.1145/3394053

1 INTRODUCTION

One of the central tasks of data mining is *outlier* or *anomaly detection*, the problem of discovering patterns that are exceptional in some sense. Detecting such patterns is relevant for two main reasons: (i) in some applications, such patterns represent spurious data (e.g., sensor failures or noise) that should be removed in a preprocessing step for further data analysis; or, more importantly; (ii) in many applications, such patterns represent extraordinary behaviors that deserve some special attention, such as genes associated with certain diseases, fraud in financial systems, employees with unusual productivity profiles, or customers with uncommon purchasing patterns.

Outlier detection techniques can be categorized in different ways. For instance, a common distinction is that between the methods that assign binary labels ("outlier" *vs.* "inlier" for those observations deemed anomalous *vs.* normal) and methods that assign a score representing a degree to which an observation is considered to be outlier. Another distinction is that between supervised, semisupervised, and unsupervised outlier detection techniques [22]. Supervised techniques assume that a set of observed instances labeled as inliers and outliers are available to train a classifier. In the semisupervised scenario, labeled outliers are not available, and only previously known inliers can be used in order to obtain a (one class) classification model. When no labeled data are available at all, it is necessary to use unsupervised techniques, which do not assume any prior knowledge about which observations are outliers and which are inliers.

In this work, we focus on unsupervised outlier detection scenarios. In general, an outlier in this context can be described as "*an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data*" [6]. In this context, there is no generally applicable definition of "appearance of inconsistency"; its formalization rather depends on the application scenario and the detection method to be used. A common scenario is to apply some outlier detection method to a database with N observations, labeling a certain subset of n such observations as the n most likely outliers, while labeling the remaining $N - n$ observations as inliers. This is referred to as the *top-n* outlier detection problem [2, 4, 7, 17, 29, 34, 37, 45, 51, 63], which is far from trivial, especially when dealing with multivariate data following complex, unknown distributions. Without labeled examples, the main complicating factor in this problem is that the notion of "outlierness" is not precisely and generally defined.

The lack of a single, formal, and agreed-upon definition of unsupervised outlier detection is one of the main reasons why a rich variety of detection methods has been developed, from classic parametric statistical methods [6, 25] to more recent database-oriented approaches conceived to deal with multivariate, possibly large databases. Considering the latter category, a plethora of detection algorithms has emerged in the past 20 years or so. Examples are DB-Outlier [32, 33], k Nearest Neighbors (kNN) Outlier [4, 51], Local Outlier Factor (LOF) [8] and its many variants [30, 35, 36, 48, 58, 65], and Angle-Based Outlier Detection (ABOD) [37], just to mention a few (see, e.g., the works of Schubert et al. [56] and Campos et al. [11] for discussions of these and many more variants). Each of these algorithms, however, uses its own criterion to judge quantitatively the level of adherence of each observation with the concept of outlier, from a particular perspective. This complicates not only the selection of a particular algorithm and/or the choice of an appropriate configuration of parameters for this algorithm in a practical application, but also the assessment of the *quality* of the solutions obtained, especially in light of the problem of defining a measure of quality that is not tied to the criteria used by the algorithms themselves. These issues are interrelated and refer to the problems of model selection and assessment (evaluation or validation) of results in unsupervised learning. These problems have been investigated for decades in the area of unsupervised data clustering [27], but are rarely mentioned and are virtually untouched in the area of outlier detection [67].

In the data clustering domain, the related problems of evaluation and model selection are tackled by using some kind of quantitative index, called validation criterion [27]. In practice, when labels are not available, *internal* validation indexes can be used. These indexes are called internal as they do not make use of any external information (such as class labels) in the evaluation of a solution. Instead, internal indexes measure the quality of an obtained clustering solution based only on the solution and the data objects. Most such indexes are also *relative* in the sense that they can be employed to compare different clustering solutions, pointing out which one is better in relative terms. Therefore they can also be used for model selection. Internal, relative indexes have been shown to be effective and useful tools for the unsupervised clustering evaluation and model selection tasks—e.g., see [21, 43, 61, 62] and references therein.

The areas of clustering and outlier detection are related to each other, and from a certain perspective, they can even be seen as two sides of the same coin. In fact, when referring to an outlier as "*an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism*", as in Hawkins' [25] definition, it is implicitly assumed that there are one or more mechanisms responsible for generating the normal, "unsuspicious" observations. Clusters are possible candidates to model such mechanisms. Surprisingly, although the internal evaluation problem has been extensively studied in data clustering, it has been completely neglected in outlier detection. In this article, we step towards bridging this gap by proposing an internal, relative evaluation measure for unsupervised outlier detection. We start from a definition of "outlierness" that is not tied to any particular criterion used by outlier detection algorithms. Rather, it follows the same, common intuition as a multitude of these algorithms and criteria: an outlier is an observation that is to some extent farther away and can, therefore, be more easily separated from other observations than an inlier. We formulate separability for outlier detection in an objective and principled way, leading to a natural definition of the proposed index.

In summary, we make the following initial contributions in this article:

- We describe the first internal, relative validation measure for evaluation of outlier detection results, Internal, Relative Evaluation of Outlier Solutions (IREOS).
- We describe an improved version of IREOS that is adjusted for chance by removing from the index the theoretical offset that is expected to be observed when evaluating random solutions. In addition to the adjusted index, we also devise means to return p-values with respect to the null hypothesis of a random solution.
- Since the exact procedure to adjust the index for chance can be computationally demanding for large datasets, we also provide a faster version of the proposed procedure, based on Monte Carlo experiments.

This article is an extension of our preliminary publication [41], where IREOS has been specifically conceived for, and experimentally evaluated in, the particular scenario of *top-n* (i.e., binary) outlier detection only. In this extended article, we make the following additional contributions:

- We extend IREOS to the more general scenario of non-binary outlier solutions, which involves the evaluation of outlier detection *scorings*, which is the type of result produced by most widely used database-oriented algorithms in the literature [11]. The extended index reduces to the original one when evaluating binary solutions, i.e., the original, *top-n* index is a particular case of the generalized version proposed here.
- In contrast to the original IREOS, which can be computed efficiently by taking advantage of the imbalanced nature of the binary outlier detection problem, where the number of outliers, n, is given as input and is typically much smaller than the dataset size, N, the proposed extended version cannot rely on this assumption when evaluating non-binary solutions,

which makes it computationally more challenging. In order to mitigate the computational burden, we introduce a number of speedup techniques that allow close approximations of the exact index to be computed in a small fraction of the runtime.
- We extensively evaluate the extended IREOS index using different collections of synthetic (30) and real (27) datasets, both in controlled experiments as well as in practical experiments of model selection involving a number (13) of different outlier detection algorithms.
- As an additional contribution, we show how IREOS can be used as an auxiliary tool to help characterize the suitability of datasets for outlier detection benchmarking, by assessing the corresponding ground truth labels of these datasets as candidate outlier solutions.

The remainder of this article is organized as follows. In Section 2, we discuss related work, the typical approaches for *external* evaluation, and the different requirements and use cases for *internal* evaluation. In Section 3, we introduce IREOS, discussing requirements and solutions, adjustment for chance, properties, and speedup techniques. We experimentally evaluate the proposed index in Section 4 and conclude the article in Section 5.

2 RELATED WORK

In the previous literature, the evaluation of unsupervised outlier detection results has been mostly restricted to making use of labeled datasets to evaluate how algorithms compare to each other, by assessing, in a supervised way, observations previously known to be inliers or outliers according to a particular intuition or semantic (e.g., normal patients vs. patients with an uncommon pathology). Note that in this scenario, referred to as *external* evaluation or validation, the labels are only used to assess the results of otherwise unsupervised outlier detection methods [11]. For the external evaluation of a top-n outlier detection solution, one is given a dataset with n known outliers (ground truth) as well as the observations ranked top-n by the given solution. *Precision-at-n* (prec@n for short) measures the fraction of the true outliers (i.e., labeled in the ground truth as outlier) among the top-n objects of the given solution [12]. If an outlier ranking is to be evaluated beyond the top-n ranks, one could also decide to measure prec@2n, prec@3n, or precision at some other point in the ranking, but the typical choice is to use the number of true outliers as the cutoff value for measuring precision [11]. A common alternative is the Receiver Operating Characteristic (ROC), which compares the candidate ranking against the binary ground truth by plotting the true positive rate against the false positive rate. Variants of these measures and more in-depth considerations about the external evaluation of unsupervised outlier detection results have been discussed, e.g., by Schubert et al. [54] and Campos et al. [11].

For internal evaluation, which is the focus of the current article, the *top-n* IREOS index, preliminarily introduced in our conference article [41], is, to our knowledge, the first internal validation index for unsupervised outlier detection. The absence of any index prior to IREOS had been noted as a gap in the literature that could hinder the development of advanced ensemble selection methods [67], but the potential applications of internal measures are far more diverse. Most fundamental is the practical application of outlier detection methods where users would benefit from unsupervised estimates of the quality of a solution provided by some method. After all, the availability of labeled data required by external evaluation measures is not consistent with the premises of unsupervised learning, and the commonly practiced external evaluation of unsupervised outlier detection algorithms makes sense only when comparing performances of algorithms in controlled experiments.

After our preliminary conference paper [41], a few papers in the literature have acknowledged the importance of internal indexes for the evaluation of outlier detection solutions in specific application scenarios. For instance, Pang et al. [47] use a wrapper approach to optimize an evaluation

measure to select the most relevant features in a dataset (feature selection problem). The proposed measure is based on the average deviation of the top-n scores (outliers) from the median of the remaining scores (inliers). Although this measure might be useful for internal evaluation, it would be limited to the evaluation of top-n solutions only, and only in relative (not absolute) terms. Weng et al. [64] propose to use an ensemble of outlier detectors to label a dataset, taking the resulting labeling as a target ground truth and measuring the quality of individual candidate solutions using external measures (comparing the level of agreement between these individual solutions and the ground truth). However, notice that the target reference solution provided by the ensemble and taken as a ground truth could be readily used as the preferred/selected solution itself. In other words, instead of selecting an individual outlier detector that best approximates the ensemble, the ensemble itself could be adopted. Of course, from a pragmatic point of view, the ensemble is just yet another candidate method rather than a ground truth, and to select between these methods a true measure of unsupervised outlier detection evaluation, such as IREOS, would be required.

3 INTERNAL EVALUATION OF OUTLIER DETECTION

3.1 Problem Statement

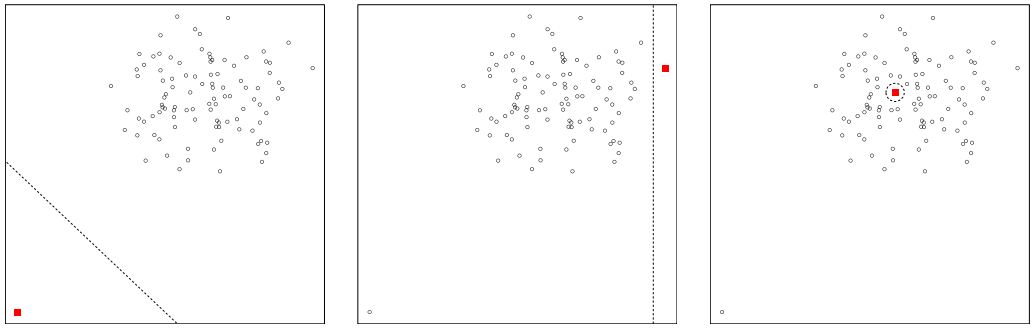
Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be an unlabeled dataset containing N d-dimensional feature vectors, $\mathbf{x}_i$, and assume that one or more unsupervised outlier detection algorithms will produce, for this dataset, a collection Ω of n_Ω candidate outlier detection solutions, $\Omega = \{\omega_1, \ldots, \omega_{n_\Omega}\}$, which one wants to evaluate in the absence of labels. Solutions $\omega_{(\cdot)}$ produced by unsupervised outlier detection algorithms can be given in different formats. The most common format is a scoring $\mathbf{y} = \{y_1, \ldots, y_N\}$, $y_i \in \mathbb{R}^+$, where y_i represents the outlier score associated with the data object/observation $\mathbf{x}_i$, which reflects the degree of outlierness of $\mathbf{x}_i$. This type of solution allows objects $\mathbf{x}_i$ to be sorted and ranked according to their degree of outlierness y_i. When the number of outliers n is known, one can establish a threshold in the ranking in order to select a subset $\mathbf{S} \subset \mathbf{X}$, $|\mathbf{S}| = n$, containing the top-n objects that are labeled as outliers. When represented in this format, we refer to $\omega_{(\cdot)}$ as binary, top-n solutions.

Given a collection Ω of candidate solutions ω_i, whether they are scoring solutions or top-n solutions, we want to independently and quantitatively measure the quality of each individual candidate solution, e.g., in order (i) to assess their statistical significance when compared to the null hypothesis of a random solution; or (ii) to compare them in relative terms so that the best candidates, corresponding to more suitable models (algorithms and parameters), can be selected.

3.2 Preliminary Attempt: A Baseline Index

We start from the intuition that an outlier is an observation that is to some extent farther off and can, therefore, be more easily separated (discriminated) from other observations than an inlier. Initially, we will consider the top-n outlier detection problem, where a binary labeling of n out of N data objects as outliers, corresponding to a better (worse) unsupervised outlier detection solution $\mathbf{S}$, is expected to be more (less) in conformity with this intuition. The basic problem is then to quantify how easy or difficult it is to separate each object $\mathbf{x}_i \in \mathbf{S}$ from other objects. In a good solution $\mathbf{S}$, consisting mostly of genuine outliers correctly detected by some method, the average degree of separability is expected to be high, whereas in a poor solution containing many false positives this average degree of separability should be lower.

We propose to assess the separability of individual data objects using a classifier. We advocate the use of a maximum margin classifier [23, 53, 66], as this type of classifier is able to quantify how distant each object is from the decision boundary while trying to maximize the margin of separability between this boundary and the instances of different classes. This idea is illustrated in



(a) Global outlier (b) Local outlier (c) Inlier

Fig. 1. Illustrative dataset: three different objects labeled as outliers.

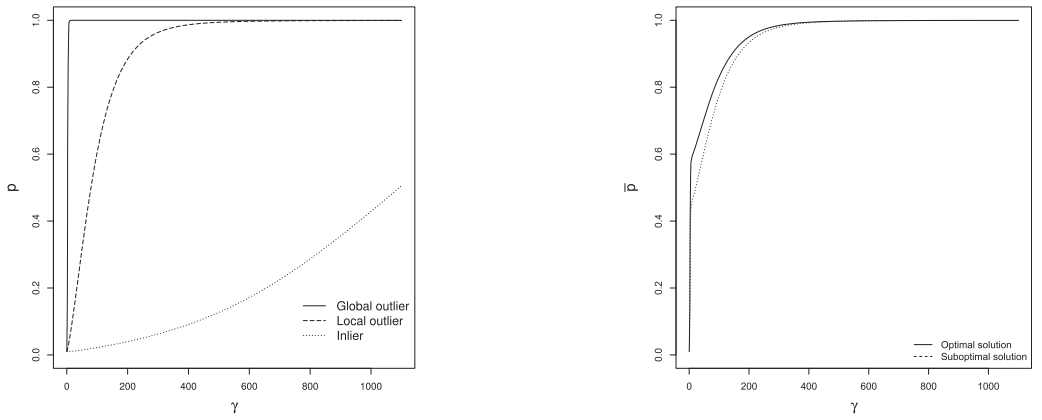
Figure 1. Figure 1(a), 1(b), and 1(c) highlights different objects labeled as an outlier (red square) in different hypothetical outlier detection solutions. In Figure 1(a), the highlighted object, a genuine global outlier, is far away from a maximum margin classification boundary (dashed line) that discriminates it from the other objects. In Figure 1(b), the highlighted object is arguably a local outlier (w.r.t. the neighboring cluster), and the margin is narrower but still wider than that in Figure 1(c). In the case of Figure 1(c), the highlighted object is undoubtedly an inlier and not only the margin is very narrow, but also the decision boundary needs to be nonlinear (i.e., more complex).

The fact that the decision boundary needs to be nonlinear to separate certain objects (as in the example in Figure 1(c)) implies that a nonlinear maximum margin classifier is required for our purpose, such as Nonlinear Support Vector Machines (SVMs) or Kernel Logistic Regression (KLR) [23, 53, 66]. These classifiers use a kernel function to transform the original (possibly non-linearly separable) problem into a linearly separable one. One of the most effective and popular kernel functions is the radial basis kernel, given by:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}. \tag{1}$$

The term γ, which is inversely proportional to the width of such a Gaussian-shaped kernel, is positively related to the flexibility (degree of nonlinearity) of the decision boundary of the corresponding classifiers. In other words, the discrimination capacity of a kernel-based classifier is positively dependent on γ. As a special case, it approaches a linear classifier as γ approaches zero. The effect of γ is similar to that of the order of a polynomial kernel function, starting from linear for first order and getting more and more non-linear as the order increases.

In practical classification tasks, γ can be used to control the compromise between the performance of the classifier on the training data vs. on test data. Here, however, we are not interested at all in the classifier itself or its performance on new, unseen data. We use a classifier merely to measure the degree of difficulty when trying to discriminate between one individual data object and the other data objects. The key observation to achieve this without having to specify a particular value for γ as a parameter is that our original premise tends to hold true, to a lesser or greater extent, no matter the value of γ. In other words, the fundamental assumption "*the more outlier-ish an object is, the easier is it to discriminate from others*" is expected to be observed for different values of γ, although the contrast between easier and more difficult cases may change. This is illustrated in Figure 2(a). We vary the value of γ from zero up to a maximum value γ_{max} (for which all the objects labeled as outliers—Figures 1(a), 1(b), 1(c)—can be individually discriminated from all the others by using a kernel-based classifier). The values along the curves (vertical axis) stand for a measure $p(\mathbf{x}_j, \gamma)$ that quantifies in a normalized interval how far each object $\mathbf{x}_j$ is from the



(a) Separability (b) Weighted Average Separability

Fig. 2. (2(a)) Curves of separability for a maximum margin classifier for each of the objects labeled as outliers in Figure 1; (2(b)) weighted average separability curves of two top-n outlier detection solutions composed of the $n = 2$ outliers (local and global) of Figure 1, differing only in the outlier scores given to them in each of the solutions.

decision boundary. For all values of γ, the two outliers are distinctly farther away from the decision boundary than the inlier.

Thus, we do not need to choose a particular value of γ. Instead, we can measure the overall separability of an object $\mathbf{x}_j$ by computing the *area under the curve* (AUC) over the interval of γ values, i.e.,

$$\int_{\gamma=0}^{\gamma_{max}} p(\mathbf{x}_j, \gamma). \tag{2}$$

Our final goal is, though, to evaluate the separability across the *collection* of data objects labeled as outliers in a given solution S. We, therefore, take the average curve of separability for those objects in S, i.e.,:

$$\bar{p}(\gamma) = \frac{1}{n} \sum_{\mathbf{x}_j \in \mathrm{S}} p(\mathbf{x}_j, \gamma), \tag{3}$$

and then compute the area under this curve to get a single number, i.e.,:

$$\int_{\gamma=0}^{\gamma_{max}} \bar{p}(\gamma). \tag{4}$$

This value can be trivially normalized in $[0, 1]$ by dividing it by its maximum possible value, γ_{max}, thus giving rise to a first, preliminary index,

$$I(\mathrm{S}) = \frac{1}{\gamma_{max}} \int_{\gamma=0}^{\gamma_{max}} \bar{p}(\gamma). \tag{5}$$

As in practice classifiers need to be trained to compute $\bar{p}(\gamma)$ for each γ, we discretize the interval $[0, \gamma_{max}]$ into a finite number of values for γ, from $\gamma_1 = 0$ to $\gamma_{n_\gamma} = \gamma_{max}$. A baseline index can thus be computed (within $[0, 1]$) as:

$$I(\mathrm{S}) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{1}{n} \sum_{\mathbf{x}_j \in \mathrm{S}} p(\mathbf{x}_j, \gamma_l) \right). \tag{6}$$

3.2.1 Non-Binary Top-n Solutions. Typically, algorithms that produce top-n outlier solutions also provide a ranking of the objects according to their score of outlierness as estimated by the algorithm. Even though it is clear that the best solution must rank all outliers before all inliers, while the worst solution would rank all inliers before all outliers, there are $n!$ different best top-n solutions that would be equally well rated by the index in Equation (6). These solutions correspond to rank permutations of the same top-n outliers. The reason why our preliminary index in Equation (6) does not distinguish between them is that it does not use the information of the outlier scores or ranks of the top-n data objects, but rather only the data objects themselves (the subset $\mathbf{S}$). However, the optimal outlier solution should rank more obvious or clear outliers before less obvious outliers, before those borderline cases that could be outliers or inliers, and so on. This issue has been addressed [54] in the context of the *external* evaluation of outlier detection, where the traditional external measures such as ROC AUC and *prec@n* cannot make such a distinction either, even though they have access to a ground truth.

In order to address this problem in our context of *internal* or *relative* evaluation of outlier detection, we propose to introduce weights in Equation (3), which defines the average curve of separability. This way, Equation (3) becomes a weighted average curve of separability, where a weight w_j is given to the separability values $p(\mathbf{x}_j, \gamma)$ of each object $\mathbf{x}_j$, and thus different rankings or scorings of the same set of outliers can be distinguished:[1]

$$\bar{p}(\gamma) = \frac{\sum_{j:\,\mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma) w_j}{\sum_{j:\,\mathbf{x}_j \in \mathbf{S}} w_j}, \tag{7}$$

The use of weights for *external* outlier evaluation has been discussed by Schubert et al. [54], who advocate the usefulness of algorithms that provide outlier scores as a result, pointing out, however, that the direct comparison of such results from different algorithms is far from trivial because different algorithms usually produce scores in completely different scales. For this reason, some kind of normalization is unavoidable when comparing solutions provided by different algorithms. Following the recommendation of Schubert et al. [54], we use the outlier scoring normalization framework proposed by Kriegel et al. [36], as this framework comprises statistically sound methods based on distribution fitting that can accommodate a large variety of outlier detection approaches. By using this framework, the original outlier scores ($\mathbf{y}$) produced by a given algorithm can be transformed into the interval $[0, 1]$ in a way that they can be interpreted as outlier probabilities. The resulting normalized outlier scores, $w_{(\cdot)}$, can then be used as the weights associated to objects $\mathbf{x}_j$, as shown in Equation (7), irrespective of the algorithm that produced the solution under evaluation. The interpretation of the weights as probabilities will also be particularly useful when enhancing our basic notion of separability discussed above towards a more sophisticated model for outlier evaluation, to be discussed later, in Section 3.3.

The intuition behind the use of weights in Equation (7) as described above is the following: when evaluating an ideal or optimal solution, both the outlier scores and the separabilities of the more obvious outliers are expected to be higher than those of the other objects, therefore their product will be higher. On the other hand, in a suboptimal solution, where the top-n objects may be the same as the optimal solution, but those with the highest separability are not the ones with the highest scores (weights) and vice-versa, the product will be penalized. In summary, we expect that solutions assigning higher (lower) outlier scores to objects with higher (lower) separabilities

[1]Notice that the use of a weighted average rather than a weighted sum prevents outlier solutions from artificially achieving higher weighted separability values by indiscriminately increasing the weights.

will result in larger AUC values as computed by the weighted version of Equation (6), given by:

$$I(\mathbf{S}, \mathbf{y}) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{\sum_{j:\, \mathbf{x}_j \in \mathbf{S}} p(\mathbf{x}_j, \gamma) w_j}{\sum_{j:\, \mathbf{x}_j \in \mathbf{S}} w_j} \right). \tag{8}$$

Notice that the index in Equation (8), in addition to the subset of top-n objects, $\mathbf{S}$, also requires as argument the outlier scores, $\mathbf{y}$, which are used to compute the weights $w_{(\cdot)}$ following the method by Kriegel et al. [36]. Hereafter, we refer to this type of solution as *non-binary top-n solutions*.

An illustrative example corresponding to two hypothetical non-binary top-n outlier detection solutions is provided in Figure 2(b). Both hypothetical solutions are composed of the $n = 2$ outliers (global and local) highlighted in Figure 1, differing only in the outlier scores assigned to them in each of the solutions. The optimal solution ranks the global outlier (with an outlier score of 1) before the local outlier (with an outlier score of 0.75), as opposed to the suboptimal solution that ranks the local outlier first (with an outlier score of 1) and the global outlier next (with an outlier score of 0.75). These two different solutions would be equally well rated by the original baseline index in Equation (6). However, using the proposed weighted average curve of separability, the index can properly distinguish these slightly different solutions, as it can be seen from the corresponding weighted average curves of separability in Figure 2(b). As expected, the optimal solution has a larger AUC than the suboptimal solution, which corresponds to a larger value of the new baseline index, $I(\mathbf{S}, \mathbf{y})$, in Equation (8).

3.2.2 Evaluation of Full Outlier Scorings. In many practical applications of unsupervised outlier detection, the number of outliers in a dataset, n, is unknown, and all that is provided by an algorithm is a scoring $\mathbf{y} = \{y_1, \ldots, y_N\}$, $y_i \in \mathbb{R}^+$, containing the outlier scores y_i associated with every data object $\mathbf{x}_i \in \mathbf{X}$ in the dataset. An interesting aspect of our baseline index in Equation (8) is that it can be straightforwardly generalized to this *full scoring* evaluation scenario, by just setting $n = N$, which implies $\mathbf{S} = \mathbf{X}$. From this observation, we can write our baseline index in a more general form that encompasses the top-n scenarios previously discussed as particular cases. Specifically, given an outlier detection solution to be evaluated, ω, our general baseline index is as follows:

$$I(\omega) = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{\sum_{j=1}^{N} p(\mathbf{x}_j, \gamma) w_j}{\sum_{j=1}^{N} w_j} \right). \tag{9}$$

In the general case of *full scoring* evaluation, the outlier solution ω corresponds to a scoring $\mathbf{y} = \{y_1, \ldots, y_N\}$ for all objects in the dataset, the corresponding probability weights $w_{(\cdot)}$ can be computed from these scores, and these weights can be directly applied into Equation (9). It is worth noticing that, in this scenario, the use of weights $w_{(\cdot)}$ normalized as probabilities following the approach by Kriegel et al. [36] is particularly useful because the majority of the data objects, as clear inliers, will tend to have a weight of zero, thus having no undesired (over) influence on the results (which could otherwise occur if the raw outlier scores $y_{(\cdot)}$ were used instead, as these typically have a very long tail of spurious small values for inliers, which, when aggregated, could dominate the results).

In the case previously referred to as *non-binary top-n solutions*, in addition to a scoring $\mathbf{y}$, the outlier solution ω also contains the subset $\mathbf{S}$ of top-n outliers, i.e., $\omega = (\mathbf{S}, \mathbf{y})$. In this case, the weights $w_{(\cdot)}$ corresponding to those objects outside $\mathbf{S}$ (i.e., not in the top-n) must be set to zero before the index is computed with Equation (9). By enforcing $w_j = 0 \; \forall j : \mathbf{x}_j \notin \mathbf{S}$, Equation (9) becomes

equivalent to Equation (8). In the even simpler case of a (binary) top-n solution that does not consider scores associated to the top-n outliers, i.e., $\omega = \mathbf{S}$, if, in addition to setting to zero the weights of objects labeled as inliers, the weights of the top-n outliers are set to 1 ($w_j = 1\ \forall j : \mathbf{x}_j \in \mathbf{S}$), then Equation (9) reduces to Equation (6).

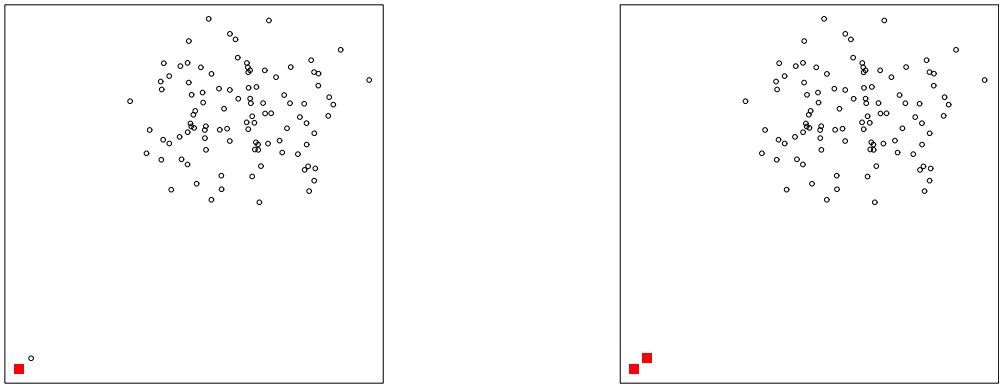
3.3 IREOS Index

3.3.1 Intuitions Missing in the Baseline Index. Our preliminary baseline index introduced in Section 3.2 may work satisfactorily in various application scenarios. Conceptually, however, it does not capture two basic intuitions that we judge important in the realm of outlier detection. Both are related to the possible presence of *clumps* of data objects in the dataset. Clumps, or particles, are subsets of objects lying in the same region of the data space, relatively closer to each other than they are from other objects, but too small to be deemed a cluster. They may exist for different reasons, mainly: (i) just by chance, e.g., in datasets with background noise following a Poison process; or (ii) as a result of anomalies whose instances are relatively rare but tend to be similar to each other, e.g., some genetic mutations or certain types of frauds. Although the semantics behind the possible interpretation of such clumps as outliers would be different, namely, noise in the first case and micro-clusters in the second, in both cases, the analyst may not want to miss the corresponding objects as potential outliers for further investigations.

Unfortunately, what is a clump can depend on both the application domain and a user's expectation. If we do not want our measure to be tied to a specific evaluation perspective that prescribes a clump size, we thus need a mechanism for a user to express in different application scenarios what they judge to be a set of similar objects that is "too small" to be interpreted as a cluster. Therefore, in contrast to the common practice of avoiding parameters in evaluation indexes for unsupervised learning, we advocate here that for outlier detection, it is nonetheless important to provide the users with an optional control mechanism to adjust their expectations about clump sizes. Given a certain expectation about what a maximum clump size should be, we should model an evaluation index so that it is able to differentiate between objects inside clusters and objects in isolated clumps. This is the first intuition that is missing in our baseline index as it was defined in Section 3.2. In order to capture this intuition, we define a maximum clump size, m_{cl}, as an optional control parameter for exploratory data analysis, to be incorporated in our index.

The second intuition that is not captured by the preliminary index is also related to clumps. While it is clear that the evaluation of each object labeled as an outlier and, accordingly, the whole index, should be negatively affected by the presence of other objects nearby (e.g., in a clump), it is intuitive that such a negative impact should be more severe if the nearby objects are assigned a different label (i.e., they are actually deemed inliers). Consider the example in Figure 3, which corresponds essentially to the same dataset as Figure 1 except for an additional object placed near the global outlier at the left bottom corner. On the left, this object's label (inlier) appears to be inconsistent with the label of the original object close by (outlier), and the index should be negatively affected. On the right, even though the original object is less of an outlier now in the presence of the additional object, their common labeling as outliers is more consistent as both objects can be seen as a clump, so the negative impact of the presence of the new object should be smaller.

3.3.2 Incorporating the Missing Intuitions. In order to capture both desired intuitions, we propose the use of classifiers with *soft margins* to compute the values of separability $p(\mathbf{x}_j, \gamma)$ required by our index. The use of soft margin models is common practice both in the literature as well as in real-world applications when using maximum margin classifiers, such as SVMs and KLR [23, 53, 66]. By making use of a soft margin, these classifiers allow the misclassification of objects at the



(a) (b)

Fig. 3. Dataset of Figure 1 with an additional object at the lower left corner: (3(a)) labeled inlier; (3(b)) labeled outlier.

price of a penalty term P_t that is incorporated into the original objective of margin maximization. Such a term is typically in the form:

$$P_t = C \sum_{j=1}^{N} \xi(\mathbf{x}_j), \tag{10}$$

where C is a constant, and $\xi(\mathbf{x}_j)$ stands for the individual penalty component associated with object $\mathbf{x}_j$. The farther an object $\mathbf{x}_j$ is from the margin boundary on the wrong side, the greater the value of $\xi(\mathbf{x}_j)$.[2] The constant C controls the overall cost of penalties. In classification problems, this is a key parameter used to adjust the compromise between under- and overfitting. Here, since we are not interested in the performance of the classifiers for new data, this constant is not critical and should only be big enough so that each object in the dataset can be discriminated from others by these classifiers when following the procedure to compute the proposed index.

Soft margin classifiers allow the use of a generalized penalty component that can assign different costs $C(\mathbf{x}_j)$ to different objects, rather than a single, uniform cost C:

$$P_t = \sum_{j=1}^{N} C(\mathbf{x}_j)\xi(\mathbf{x}_j). \tag{11}$$

A suitable design of terms $C(\mathbf{x}_j)$ makes it possible to simultaneously incorporate both desired intuitions that are missing in our baseline index, related to the modeling of clumps. For the sake of simplicity, let's initially consider the simplest case involving *binary top-n solutions* $\mathbf{S}$, where n out of N objects have been labeled as outliers, and the others have been labeled as inliers. In order to capture the differences between the two distinct scenarios in Figure 3, we can assign full margin violation cost C to the objects labeled as inliers yet only a fraction $\beta \in [0, 1]$ of C to the objects labeled as outliers, i.e., $C(\mathbf{x}_j) = C$ or $C(\mathbf{x}_j) = \beta \cdot C$, depending on whether $\mathbf{x}_j$ has been labeled as an inlier or an outlier in the solution $\mathbf{S}$ under evaluation, respectively. For $\beta = 1$, the method reduces to the ordinary case where objects are treated equally no matter their labels, and thus the separability of the leftmost outlier in Figures 3(a) and 3(b) would be indistinguishable. In the other extreme,

[2]For SVMs, $\xi(\mathbf{x}_j)$ is zero when $\mathbf{x}_j$ lies on the correct side of the margin boundary. For KLR, all objects (rather than only support vectors) can influence the decision boundary and $\xi(\mathbf{x}_j)$ can be non-null but tending to zero as $\mathbf{x}_j$ moves away from the margin boundary on the correct side.

$\beta = 0$, objects labeled as outliers can be misclassified for free. Notice that, when evaluating the separability of a specific object, this is equivalent to removing all other objects labeled as outliers from the dataset. As a consequence, the separability of the leftmost outlier would be much larger in Figure 3(b) (where the neighboring outlier can be ignored by the margin) than in Figure 3(a) (where the neighboring inlier, which is subject to full penalty, forces a much tighter margin). The choice of β would, therefore, tune the influence of other objects depending on their assigned labels and, as such, it addresses our *second desired intuition*.

In order to capture our *first intuition*, the modeling of possible clumps by defining a maximum clump size, m_{cl}, we can set the fraction of the penalty cost C as $\beta = 1/m_{cl}$. This way, we are left with m_{cl} as a single, optional control parameter in our evaluation method. It is optional because, by setting $m_{cl} = 1$, the method reduces to the particular case where clumps are not modeled and the same, full penalty cost is assigned to all objects. As m_{cl} increases, the objects in a clump will individually affect less and less each other's measure of separability, and a larger number of nearby objects will be needed to get a certain negative impact.

Notice that, in the top-n problem, by setting $\beta = 1/m_{cl}$, one needs m_{cl} objects labeled as outliers to have the same effect as a single inlier. Also, notice that it would be contradictory to set $m_{cl} > n$, as no more than n objects can be labeled as outliers in a top-n problem. By considering this conceptual upper bound, one gets $1 \leq m_{cl} \leq n$.

Except when $m_{cl} = 1$, the separability of each object depends on the labels of the other objects and, therefore, seeking a solution that maximizes the proposed index (rather than using it to assess a given solution) would hardly be computationally feasible: in principle, for the top-n problem, it would demand an exhaustive search in a space of size $\binom{N}{n}$, where typically $n \ll N$. In the more general problem of *full outlier scoring evaluation*, discussed next, the size of the space would expand even further due to the infinite number of different possible assignments of outlier scores to each object (rather than a binary inlier/outlier labeling).

We can extend the above formulation for modeling clumps to the more general evaluation problem involving *full outlier scoring solutions*. To achieve this, while keeping the desired properties of the model previously discussed for the top-n case, we can incorporate the normalized outlier scores (weights) $w_{(\cdot)}$ into the cost components, as $C(\mathbf{x}_j) = \beta^{w_j} \cdot C$. For an object $\mathbf{x}_j$ with the maximum probability of being an outlier according to the solution under evaluation, i.e. $w_j = 1$, it follows that $C(\mathbf{x}_j) = \beta \cdot C$. On the other extreme, for an object with the maximum probability of being an inlier, i.e., $w_j = 0$, we have $C(\mathbf{x}_j) = C$. For intermediate values, $0 < w_j < 1$, the higher the outlier score, the higher the penalty cost $C(\mathbf{x}_j)$. This way, we keep the original intuition in this extended, full scoring evaluation model, while having the binary, top-n model clearly as a particular case.

In the general case, therefore, the optional model for clumps to be incorporated into our baseline index, through the use of penalty terms associated with soft margin classifiers as in Equation (11), is therefore:

$$C(\mathbf{x}_j) = \left(\frac{1}{m_{cl}}\right)^{w_j} \cdot C\,, \tag{12}$$

where m_{cl} is the maximum clump size, as a user-defined parameter, and C is a large constant.

3.3.3 Summary and Algorithm. In brief, IREOS is summarized as follows: like the baseline index (Section 3.2), IREOS is computed using Equation (9). However, we make use of classifiers with soft margins in order to compute the terms $p(\mathbf{x}_j, \gamma)$ in that equation, with the cost for margin violations set up as in Equation (12). A high-level pseudo code for computing IREOS for a set Ω of multiple outlier detection solutions (as, e.g., for model selection) is given in Algorithm 1.

As for the classifier to be used in practice, our method is not tied to any specific soft margin classifier. KLR [23, 66], which we have used for all the experiments reported in this article, offers the

ALGORITHM 1: IREOS

Procedure IREOS($\mathbf{X}, \Omega, m_{cl}$)
  foreach ($\omega \in \Omega$) **do**
    switch ω **do**
      case *is binary top-n solution* **do** // $\omega = \mathrm{S}$
        if $\mathbf{x}_j \in \mathrm{S}$ **then**
          $\mathbf{w}[\mathbf{x}_j] = 1$
        else
          $\mathbf{w}[\mathbf{x}_j] = 0$
        end
      end
      case *is non-binary top-n solution* **do** // $\omega = (\mathrm{S}, \mathbf{y})$
        $\mathbf{w}$ = NormalizeScores($\mathbf{y}$)
        if $\mathbf{x}_j \notin \mathrm{S}$ **then**
          $\mathbf{w}[\mathbf{x}_j] = 0$
        end
      end
      case *is scoring solution* **do** // $\omega = \mathbf{y}$
        $\mathbf{w}$ = NormalizeScores($\mathbf{y}$)
      end
    end

    /* γ_{max} = value of γ required by the classifier to separate from the other objects every object labeled as an outlier in all solutions $\omega \in \Omega$; for scoring solutions, objects are considered outliers if their weight w_j is greater than 0.5 (i.e., outlier probability > 50%) */
    $\gamma_{max} = \gamma \mid \mathrm{Classifier}(\mathbf{X}, \mathbf{x}_j, \mathbf{w}, m_{cl}, \gamma) > 0.5, \ \forall \mathbf{x}_j \mid w_j > 0.5, \ \forall \omega \in \Omega$

    setOfGammas = $[0, \gamma_{max}]$ discretized into n_γ values
    foreach ($\gamma \in$ setOfGammas) **do**
      foreach ($\mathbf{x}_j \in \mathbf{X}$) **do**
        prob$[\mathbf{x}_j]$ = Classifier($\mathbf{X}, \mathbf{x}_j, \mathbf{w}, m_{cl}, \gamma$) // Separability $p(\mathbf{x}_j, \gamma)$
      end
      avgProb$[\gamma]$ = WeightedAverage(prob, $\mathbf{w}$) // Inner part of Equation (9)
    end
    ireos$[\omega]$ = NormAUC(avgProb, setOfGammas) // Outer part of Equation (9)
  end
end

following benefits: (i) it automatically provides $p(\mathbf{x}_j, \gamma_l)$ as the probability that object $\mathbf{x}_j$ belongs to the positive (outlier) class; (ii) these terms are not only provided directly as a byproduct of the classifier, but they are naturally normalized (as probabilities) within $[0, 1]$; and (iii) KLR is a classifier known to be robust even in the presence of imbalanced classes and small amounts of training data.

Note, however, that the use of classifiers other than KLR is possible. Other maximum margin classifiers, such as SVM [60], can in principle be readily plugged in provided that the SVM scores are transformed into probabilities [50]. For classifiers other than the maximum margin classifiers, although it is possible to transform the scores into probabilities [44], other questions would have

to be thoroughly investigated. For instance, a general question to be answered would be how to model clumps. Other more specific questions for each classifier would also have to be studied. For example, for Artificial Neural Networks (ANN) [20], one would have to investigate how possible convergence to local minima could affect the separability of the observations and, accordingly, their IREOS scores.

3.4 Adjustment for Chance

The IREOS index, as described above, is ready to be used in practice if one is only interested in comparing in *relative terms* a set of different candidate solutions, e.g., for model selection. However, the interpretation of the index for individual solutions, e.g., for statistical validation, can be misleading. The reason is that IREOS will provide a certain positive value even when evaluating purely random solutions. The situation is worsened by the fact that such a value is data dependent. In fact, note from Figure 2(a) that even inliers will exhibit a non-null value for the AUC of separability. This prevents interpreting and assessing the statistical relevance of a given result in *absolute terms*, which requires the index to be adjusted for chance. Here, we follow the classic statistical framework for chance adjustment, i.e.,:

$$I_{adj}(\omega) = \frac{I(\omega) - E\{I\}}{I_{max} - E\{I\}}, \tag{13}$$

where $I_{adj}(\omega)$ is the resulting (adjusted) index, $I(\omega)$ is the original index in Equation (9), $I_{max} = 1$ is the maximum value that the index can take, and $E\{I\}$ is its expected value assuming that the outlier weights are randomly assigned to the objects (for scoring solutions), or assuming that the n data objects labeled as outliers in a top-n solution are chosen randomly. For random solutions, I_{adj} is expected to take values around zero. The maximum is still 1, but the index now can take negative values to indicate solutions even worse than what one would expect to obtain by chance.

The expected value, and for top-n solutions, also the variance that can be used for statistical validation, can be computed exactly for the choice of $m_{cl} = 1$. For $m_{cl} > 1$ or for statistical validation in scenarios other than the binary top-n case, Monte Carlo simulations can be used to estimate the relevant statistics. In Appendix A, we provide detailed explanations on how to perform adjustment for chance as well as statistical validation in the different scenarios.

3.5 Complexity

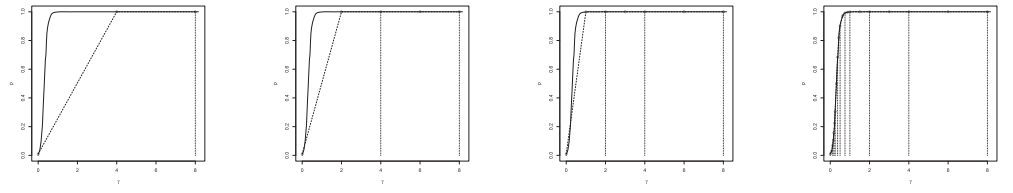
The asymptotic computational complexity of the algorithm depends on the complexity of the classifier, $O(f(N,d))$, as a function of the database size N and dimensionality d. For each candidate solution ω, we need to compute IREOS (Equation (9)), which demands training $N \cdot n_\gamma$ classifiers in the case of *full scoring evaluation*, thus resulting in an overall complexity of $O(N \cdot n_\gamma \cdot f(N,d))$ in this case. When the index is adjusted for chance, we may need to evaluate n_{MC} different random solutions in Monte Carlo simulations in order to estimate the expected index, as discussed in Appendix A.2, leading to a complexity of $O(n_{MC} \cdot N \cdot n_\gamma \cdot f(N,d))$. This is the complexity in the most general case. For *top-n evaluations*, we only need to train $n \cdot n_\gamma$ classifiers, where the usual assumption is that $n \ll N$, which reduces the complexity to $O(n_{MC} \cdot n \cdot n_\gamma \cdot f(N,d))$. If Monte-Carlo simulations are not required (because $m_{cl} = 1$ or because statistical validation is not needed, as in the case of model selection, where all we need is to relatively compare different candidate solutions), the complexity is further reduced to $O(n \cdot n_\gamma \cdot f(N,d))$.

The above complexity can be mitigated in many different ways, starting from the observation that every single classifier can be trained simultaneously and in a completely independent way. In the case where we have $O(n_c)$ computer cores for parallel processing, where n_c is the number of classifiers to be trained, the complexity of IREOS reduces to that of the classifier used. In

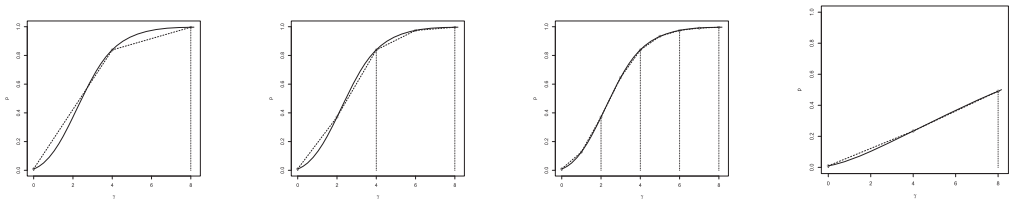
other words, IREOS is highly parallelizable and can be implemented in a straightforward way in distributed (e.g., cloud) environments using parallel computing frameworks such as MapReduce [14]. In order to make the index even more computationally efficient, we provide strategies to avoid large amounts of unnecessary computations in the three major components involved in the complexity of the index: n_γ separability values across different values of γ for a given object, separability values across N different objects for a given value of γ, and the computational burden to train each classifier for a dataset of size N.

3.5.1 Separability Curves. In practice, in order to calculate the separability curve for a given data object, we need to discretize the continuous range of γ values, $[0, \gamma_{max}]$, into n_γ discrete values. In the experiments performed in our preliminary publication [41], we arbitrarily set $n_\gamma = 100$ as a setting that empirically provides a very accurate approximation of the AUC of separability as discretized into n_γ equally spaced values. There are, however, two major limitations with this approach. First, it does not provide any control of the trade-off between the resulting level of approximation (which is not known in advance) and the corresponding computational burden. The only information available is that, by increasing n_γ, the approximation (i.e., discretization) error will decrease at the price of an increase in the computational cost, but the user cannot calibrate the desired approximation error and automatically derive the smallest n_γ that provides that error. It is possible, for instance, that a very similar (and already good enough) approximation can be achieved with much fewer than $n_\gamma = 100$ values, in which case this arbitrary setting will waste a large amount of computing power to unnecessarily and only slightly improve the quality of the approximation. In addition, an approach that discretizes values evenly into equally spaced intervals may not be the most effective and efficient. Instead, a smaller number of γ values can be sampled from less critical regions of the separability curve, particularly when the curve has already flattened and therefore requires much fewer values for a good approximation than required in the leftmost part of the curve.

To overcome these limitations and to significantly speedup the computation of IREOS, instead of using a fixed, arbitrary number n_γ, we will use a variable number n_γ according to a user-specified tolerance for an estimated approximation error. To that end, we make use of the technique of adaptive quadrature (or adaptive numerical integration) [9, 19, 38]. This technique will approximate the separability curve by subdividing it recursively into smaller subintervals. The procedure begins with the approximation of the separability curve with the minimum possible number of points (three). If this approximation does not reach the user-specified precision, the separability curve is subdivided into two parts (bisection), and the same procedure is applied recursively and independently on both (equally spaced) intervals, until the user-specified precision is reached. This procedure is illustrated in Figure 4, where the three different separability curves in Figure 2(b), corresponding to a global outlier, a local outlier, and an inlier, are approximated by this procedure. Initially, in Figure 4(a), the separability curve represented by the solid line is approximated by the dotted line using only three points. Since the approximation is not good enough, the separability curve is divided into two parts, the area of these two parts are calculated independently, and the procedure is applied recursively to both of them. Figure 4(b) illustrates the bisection as a vertical line as well as the approximated curve on both sides (five points in total) as a dotted line. Since the area on the right side has already reached a satisfactory approximation, only the left part will be further subdivided (Figure 4(c)). This procedure is repeated recursively, until each subdivision reaches a sufficiently good approximation (Figure 4(d)). The other two separability curves are approximated using the same procedure, in Figures 4(e)–4(g) and 4(h), respectively. Note that the number of points needed to achieve a good approximation varies in each case.



(a) Approximation using 3 points (global outlier)
(b) Approximation using 5 points (global outlier)
(c) Approximation using 7 points (global outlier)
(d) Approximation using 21 points (global outlier)



(e) Approximation using 3 points (local outlier)
(f) Approximation using 5 points (local outlier)
(g) Approximation using 9 points (local outlier)
(h) Approximation using 3 points (inlier)

Fig. 4. Approximation of the separability curves in Figure 2(b) using the adaptive quadrature technique. The original curves are shown as solid lines, whereas the approximate curves are shown as dotted lines. Bisections are highlighted as vertical lines.

The stopping condition of this procedure is that the approximation error should be less than a user-specified threshold. Since the exact error cannot be determined as it would require the exact computation of the separability curves, which is what the use of an approximation is meant to avoid in the first place, an estimated error bound can be used instead. Since IREOS is defined conceptually as the AUC of the continuous average curve of separability $\bar{p}(\gamma)$, which we want to approximate by a finite number of points within the interval $[0, \gamma_{max}]$, determined using the adaptive quadrature technique, the error bound for this approximation can be calculated analytically through a closed-formula [9]. In particular, there is a class of error bound, known as *a posteriori* error estimate, that gives us a computable estimate based on the computed approximation only.

There are different approaches to calculate this type of error estimate [19]. The most commonly used approach is based on the difference between the approximation of the curve over an entire interval, using n_γ points, and the approximation of the curve after a bisection of the original interval, as the sum of the approximations over the two resulting subintervals, i.e., using $2n_\gamma$ points (although the midpoint is shared by both intervals, it is technically used twice, once in each interval).

Let $I(0, \gamma_{max})$ be the approximate IREOS index computed over the entire interval $[0, \gamma_{max}]$ as in Equation (9), i.e., using n_γ discrete points. In addition, let $I(0, \frac{\gamma_{max}}{2})$ and $I(\frac{\gamma_{max}}{2}, \gamma_{max})$ be the approximate IREOS index computed separately in each bisection subinterval, $[0, \frac{\gamma_{max}}{2}]$ and $[\frac{\gamma_{max}}{2}, \gamma_{max}]$, respectively, each of which uses n_γ points. Since IREOS is, by definition, an AUC, it follows that $I(0, \frac{\gamma_{max}}{2}) + I(\frac{\gamma_{max}}{2}, \gamma_{max})$ corresponds to the approximate IREOS index computed over the entire interval $[0, \gamma_{max}]$ *after* the bisection, i.e., using $2n_\gamma$ points, in contrast to $I(0, \gamma_{max})$, which uses n_γ points only (i.e., *before* the bisection). In addition, let I_{exact} be the exact value of the index as if the exact (continuous) curves of separability were known. For smooth and continuous curves, such as separability curves, we can estimate the error of the approximation using $2n_\gamma$ points as [5, 13, 19]:

$$\begin{aligned} \mathcal{E}_{2n_\gamma}(I) &:= \left| I_{exact} - \left(I(0, \tfrac{\gamma_{max}}{2}) + I(\tfrac{\gamma_{max}}{2}, \gamma_{max}) \right) \right| \\ &\approx \tfrac{1}{3} \left| \left(I(0, \tfrac{\gamma_{max}}{2}) + I(\tfrac{\gamma_{max}}{2}, \gamma_{max}) \right) - I(0, \gamma_{max}) \right| \end{aligned} \tag{14}$$

Equation (14) can be interpreted as follows: the absolute error between the exact curve and its (bisected) approximation using $2n_\gamma$ points, $\mathcal{E}_{2n_\gamma}(I)$ (the first expression above, by definition), is approximately $\frac{1}{3}$ of the absolute difference between the approximation using $2n_\gamma$ points and the (non-bisected) approximation using n_γ points only. Notice that, while the former expression is not computable because we do not know I_{exact}, the components in the latter expression are not only computable but also readily available as we perform the adaptive quadrature procedure. After each new bisection step is performed, we can recursively apply Equation (14) to the bisected subinterval, update the total estimated error (as a sum of the errors in all current subintervals), until this error falls below the user-defined threshold, when the adaptive quadrature can be interrupted.

It is worth noticing that, for the sake of simplicity, the above description of the adaptive quadrature procedure has subsumed the use of the midpoint rule as the method for numerical integration, but there are other rules that can be applied. In case of higher-order methods, the ratio between the error when using $2n_\gamma$ points and the error when using only n_γ points can be much more accentuated than $\frac{1}{3}$. For instance, for Simpson's rule, which is used in our code and experiments, the ratio is $\frac{1}{15}$ [9]. This means that the estimated error decays much more quickly and, therefore, a smaller number of points is required to achieve a given targeted approximation quality.

3.5.2 Separability Values Across the Dataset. Due to the normalization applied to the outlier scores, following the approach of Kriegel et al. [36] to compute weights w_j as outlier probabilities associated with the data objects $\mathbf{x}_j$, many objects in the dataset will be assigned a weight of zero. Since the separability $p(\mathbf{x}_j, \gamma)$ of each object $\mathbf{x}_j$ is multiplied by the corresponding weight w_j in Equation (9), the separabilities of all objects with weight equal to 0 do not need to be computed, which in practice can drastically reduce the number of classifiers to be trained.

The normalization proposed by Kriegel et al. [36] presumes the regularization of the scores in such a way that the expected score for an object considered to be an inlier by the outlier detection method that produced the scores will have an associated probability (weight) around zero. A number of outlier detection algorithms in the literature [8, 10, 48, 65] readily provide an expected score as a mathematical property, so that regularization can be done just by subtracting this value from the outlier scores—e.g., the expected score for an inlier is around 1 for LOF [8] and around 0 for GLOSH [10]. In case of algorithms for which the expected score of inliers cannot be trivially determined, the regularization must be performed based on assumptions about the distribution of the outlier scores; e.g., if a Normal distribution is assumed, the regularization can be performed by subtracting the mean or median from the scorings, "squeezing" the scores of objects at or below the mean/median to zero. Notice that, in practice, for algorithms whose expected score for inliers is known, e.g., LOF, a large amount of the data objects, as inliers, will tend to have null weight, and therefore it is not unrealistic to assume that, even in the *full scoring evaluation* scenario, the separability values and the corresponding classifiers may only need to be computed for a fraction n of the data objects, where $n \ll N$.

In practice, one of the most important applications of IREOS could be *model selection*, where the analyst faces a collection of candidate outlier detection solutions produced by different algorithms, or different parameters of the same algorithm, and wants to relatively compare and rank these solutions, in a completely unsupervised way, in order to pick the best one(s) for further investigation. In this scenario, it is possible to further speed-up computations by taking advantage of a *pruning strategy*, explained below.

Since larger weights $w_{(\cdot)}$ have a greater influence on the composition of the index in Equation (9), computing the separability values only for a subset of the data objects with the largest weights may suffice to determine that one solution is necessarily better or worse than others. By sorting the weights in each candidate solution and then computing IREOS incrementally by processing

Table 1. Illustrative Example of the Pruning Strategy for Model Selection Involving Three Hypothetical Candidate Outlier Detection Solutions

Solution 1				Solution 2				Solution 3			
$p(\mathbf{x}_i, \gamma)$	w_i	$I_{partial}$	$\hat{I}_{max}$	$p(\mathbf{x}_j, \gamma)$	w_j	$I_{partial}$	$\hat{I}_{max}$	$p(\mathbf{x}_l, \gamma)$	w_l	$I_{partial}$	$\hat{I}_{max}$
1	0.3	0.3	1	0.2	0.3	0.06	0.76	0.6	0.35	0.21	0.86
0.87	0.2	0.474	0.974	0.6	0.15	0.15	0.7	0.2	0.25	0.26	0.66
1	0.135	0.609	0.974	0.9	0.14	0.276	0.686	0.5	0.15	0.335	0.585
1	0.13	**0.739**	0.974	1	0.1	0.376	**0.686**	0.4	0.07	0.363	**0.543**
0.3	0.12	0.775	0.89	0.95	0.09	0.4615	0.6815	0.3	0.05	0.378	0.508
0.6	0.1	0.835	0.85	0.4	0.08	0.4935	0.6335	0.9	0.045	0.4185	0.5035
0.2	0.015	0.838	0.838	0.2	0.08	**0.5095**	0.5695	0.92	0.035	0.4507	**0.5007**
0.3	0	0.838	0.838	1	0.06	0.5695	0.5695	1	0.03	0.4807	0.5007
0.5	0	0.838	0.838	0.1	0	0.5695	0.5695	1	0.02	0.5007	0.5007

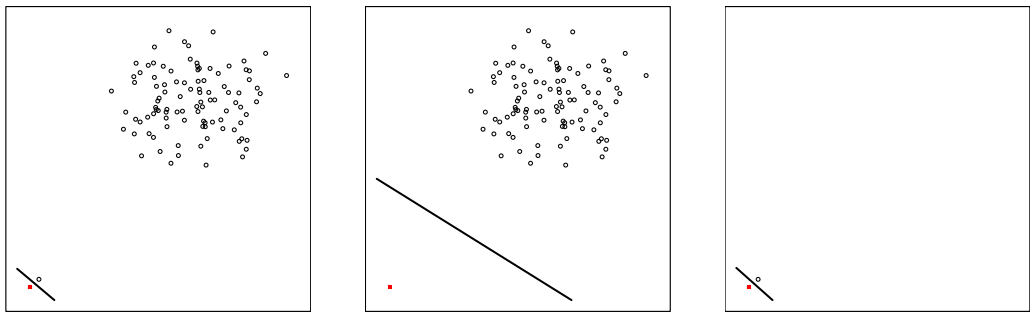
Processing of Solution 1 can be stopped earlier, at the 4th iteration, since it cannot be surpassed by Solutions 2 or 3. Processing of Solution 3 can be stopped at the 7th iteration, as it cannot surpass Solutions 1 or 2.

the normalized products $(p(\mathbf{x}_j, \gamma) \cdot w_j)/(\sum_i w_i)$ in decreasing order of weights w_j, simultaneously across all candidate solutions, we can monitor the current partial value of the index ($I_{partial}$) for each solution as well as the maximum value that the index can reach for that solution ($\hat{I}_{max}$) assuming that all the data objects that have not yet been processed would have maximum separability of 1. Since both the separability as well as the weight values cannot be negative, i.e., $p(\mathbf{x}_j, \gamma) \geq 0$ and $w_j \geq 0$, $I_{partial}$ can only increase as we process more objects. Contrarily, $\hat{I}_{max}$ can only decrease. Hence, by keeping track and comparing the values of $I_{partial}$ of each solution against the values of $\hat{I}_{max}$ for the other solutions, we can stop the calculation of the index of a solution earlier, when it cannot surpass or be surpassed by any other solution.

An example of this pruning strategy is illustrated in Table 1, where three outlier detection solutions are evaluated. The outlier weights are sorted in decreasing order, and the separability of the corresponding data objects are computed incrementally (from top to bottom). In each iteration, one data object is processed at each candidate solution, and the values of $I_{partial}$ and $\hat{I}_{max}$ for that solution are updated and compared with those of the other solutions. Once $I_{partial}$ of a solution is already larger than the $\hat{I}_{max}$ values of all the other solutions that are still active (i.e., being processed), the process of this solution can be interrupted. In the example in Table 1, this occurs in the 4th iteration, when Solution 1 no longer needs to be processed as its partial index ($I_{partial} = 0.739$) cannot be reached by the indexes in Solutions 2 ($\hat{I}_{max} = 0.686$) and 3 ($\hat{I}_{max} = 0.543$). We can also halt processing a solution once its $\hat{I}_{max}$ value is already smaller than $I_{partial}$ of all the other solutions that are still active. In our example, this occurs in the 7th iteration, when Solution 3 no longer needs to be processed as its maximum possible index value ($\hat{I}_{max} = 0.5007$) cannot reach the index in Solution 2 ($I_{partial} = 0.5095$).[3]

3.5.3 Classifiers and Computational Complexity Revisited. The computational cost of IREOS is dominated by the cost of training multiple classifiers. While maximum margin classifiers are known for their high efficacy, they are also known for being computationally demanding. Fortunately, different techniques are available in the literature to lessen the computational burden of

[3]There is also a 3rd scenario in which we can stop processing a solution: let Ω be the set of candidate solutions. If $\hat{I}_{max}$ of a solution ω_i is already lower than the $\hat{I}_{partial}$ values of a subset of solutions $\Omega_\mathbf{A} \subset \Omega$, while $\hat{I}_{partial}$ of ω_i is already higher than the $\hat{I}_{max}$ values of the remaining solutions, i.e., those in the subset $\Omega \backslash \{\Omega_\mathbf{A}, \omega_i\}$, then ω_i cannot surpass or be surpassed by any other solution and can be stopped.



(a) Original Dataset (b) Closest object downsampled (c) Whole cluster downsampled

Fig. 5. Illustrative example of subsampling and the influence of close and distant objects on an object's separability.

these classifiers. These techniques can be categorized into two main approaches: *algorithmic* and *data selection*. The first approach consists of faster algorithms for the Quadratic Programming (QP) solver required by maximum margin classifiers [31, 49], while the latter approach focuses on the selection of training data in order to reduce the number of examples (N) used to train the classifiers [40, 46]. Our approach to speeding up the computation of our index combines the best of both worlds: while taking advantage of a fast QP solver [31], we also elaborate on a suitable approach for the selection of the training data.

The most straightforward approach to reduce N is to randomly subsample the dataset available for training [40]. However, when evaluating the individual separability of a data object, an informed subsampling is required instead, because separability can be highly affected depending on the objects filtered out by the sampling procedure. In Figure 5, we can see two different subsamples taken from the dataset illustrated in Figure 5(a). Notice that, when evaluating the separability of the object marked as a red square, while the subsample in Figure 5(b) completely modified the decision boundary by filtering out a single object, the boundary in Figure 5(c) has not been affected even though an entire cluster has disappeared. While the use of soft margin classifiers (as opposed to the strict margin illustrated in Figure 5) significantly reduces the influence of single data objects on the computation of separability, the message learned from the extreme case illustrated in Figure 5 is still valid: closer objects have more influence on the decision boundary and, accordingly, on the separability of a given object.

While closer neighbors noticeably affect the decision boundary for the separability of a given object, other more distant objects are likely to be irrelevant. In fact, for certain classifiers, such as SVMs, only the support vectors matter. For KLR, while all objects influence the decision boundary to some extent, the influence tends to zero very quickly as an object moves away from the margin boundary on the correct side. Thus, in order to select the most effective training subset for our classifiers, we follow previous work from the literature [42, 46] and propose to use only the kNN of each object (rather than the whole dataset) to evaluate its separability.

When trained from the N objects of a dataset to compute the separability of one particular object, the KLR classifier, which is used in all our experiments, runs in $O(d \cdot N^3)$ time, where d is the data dimensionality. By using a subsample containing only the kNN of the object in question, this complexity is drastically reduced to $O(d \cdot k^3)$, plus $O(N)$ to determine the kNN using an ordinary linear search, i.e., $O(d \cdot k^3 + N)$. Then, in the worst case scenario where, despite the different pruning strategies previously described, the separability (and the kNN) of all the N objects in the dataset would be required to compute IREOS, for n_γ different values of γ, the total complexity of the index would be $O(N \cdot n_\gamma \cdot d \cdot k^3 + N^2)$.

If appropriate index structures are in place, such as a Kd-Tree, the kNN for all N observations can be determined in $O(N \log N)$ time. Moreover, notice that the exact kNN are not fundamentally important, because, by using a subsample, we are already computing an approximate value of separability in the first place, so an approximate-NN approach [18, 57] should suffice. By using approximate rather than exact nearest neighbors, the k (approximate) nearest neighbors for all N observations may be determined in $O(N)$ time, typically with the aid of hashing structures. In this case, the complexity of IREOS in the worst case reduces to $O(N \cdot n_\gamma \cdot d \cdot k^3 + N)$, i.e., $O(N \cdot n_\gamma \cdot d \cdot k^3)$. For top-$n$ solutions, the complexity is $O(n \cdot n_\gamma \cdot d \cdot k^3)$.

The value of k is a compromise between the quality of approximation and runtime. It is important to notice that most existing unsupervised outlier detection algorithms require the computation of the kNN for all observations in the dataset, where k is a user-defined parameter of these algorithms (typically a small constant such that $k \ll N$). If the same value of k given as input to the algorithm or algorithms that produced the outlier detection solution(s) to be evaluated is also used to compute IREOS, the kNN computed by the algorithms can be reused with no extra cost to IREOS. This makes the worst-case complexity of IREOS to be $O(N \cdot n_\gamma \cdot d \cdot k^3)$ for scoring solutions or $O(n \cdot n_\gamma \cdot d \cdot k^3)$ for top-n solutions even when using exact (rather than approximate) nearest neighbors. Finally, notice that these results subsume centralized processing, but one can significantly speed up computations using parallelization schemes (as we do in our code).

3.5.4 Final Remarks. Before demonstrating that the index developed in this chapter works well in practice, we want to point out that we are aware of a vulnerability that could potentially allow "adversarial" outlier detection methods to fool the index to some extent. Recall that the basic definition of the index for a full scoring (Equations (8) and (9)) is a weighted average, where the average of separability values, weighted by an outlier detection method's scores, is taken over the sum of the scores assigned by the method. It is easy to see that a method could get the maximum possible IREOS score, if it would somehow be able to detect the strongest outlier (in terms of separability), and would then assign this outlier a weight of 1 and all other data objects a weight of 0. While this property is not desirable, it is practically not very relevant, since outlier detection methods are typically not designed to fool an evaluation index. An alternative evaluation index that does not have this property can be based on a weighted Pearson correlation coefficient (with separability values as weights). However, the drawback of the weighted correlation is that not all of the speed-up strategies developed in this section easily apply, making it computationally much more demanding. We have implemented and experimented with an evaluation index based on weighted Pearson Correlation, and its performance in terms of quality is indeed comparable to the index proposed in this chapter, but not systematically better, and much worse in terms of runtime. Therefore, we recommend to use the index as proposed in this section, and evaluate its performance in detail in the next section.

4 EVALUATION

4.1 Datasets

We combine different strategies to annotate datasets used for evaluation, following statistical considerations, following the semantical notion of unusual classes, or following common procedures and examples from the literature.

4.1.1 Synthetic Datasets. For experiments on synthetic data, we adopt previously published benchmarking datasets that have been designed and used to evaluate outlier detection methods [68, 69]. We use a collection referred to as "*batch1*" [68, 69]. The 30 datasets in this collection

vary in the dimensionality $d \in [20, \ldots, 40]$, in the number of clusters $c \in [2, \ldots, 10]$, and for each cluster independently in the number of points $n_{c_i} \in [600, \ldots, 1000]$. For each cluster, the points are generated following a Gaussian model with randomly selected parameters that are attribute-wise independent. The sampled cluster is randomly rotated and the covariance matrix is rotated accordingly. Based on the covariance matrix, the Mahalanobis distance between the mean of a cluster and each cluster point is computed. The distribution of the Mahalanobis distances follows a χ^2 distribution with d degrees of freedom. Points with distances on the right the tail of this distribution can be seen as outliers.

4.1.2 Real-world Datasets and Preliminary Benchmarking Experiments. The first collection of real-world datasets is composed of 23 real datasets from a publicly available outlier detection repository that contains approximately 1,000 variants of 23 main datasets [11]. As these datasets are originally from clustering and classification tasks and, therefore, there is no guarantee that the ground truth labels of objects as inliers and outliers in these datasets are in conformity with the spatial distribution of the data (i.e., in principle it is unknown if the semantic represented by the labels is properly captured in the feature space where the objects have been described), Campos et al. [11] proposed two indexes to characterize the suitability of each candidate dataset for outlier detection benchmarking. One of the indexes is called *difficulty*, which measures the disagreement between the results of a collection of representative outlier detection methods and the ground truth labels, such that a high value of the index indicates that the methods have difficulty in detecting the objects labeled as outliers in the ground truth. Therefore, the ground truth labels of a dataset with a high difficulty value are not consistent with the notion of outliers as captured by a variety of algorithms, which may suggest that the labels are not aligned with the spatial distribution of the data. The second index proposed by Campos et al. [11] is called *diversity*, which measures the disagreement of a given collection of outlier methods with respect to the outlier scores that they assign to data objects labeled as outliers in the ground truth. A low diversity score combined with a low difficulty score suggests that the outliers in the ground truth can be detected by most algorithms (an easy dataset). A high diversity score combined with a high difficulty score can occur, e.g., if most of the algorithms are unable to detect most of the outliers, but a few outliers, which vary from algorithm to algorithm, can be detected by each algorithm.

A limitation of the measures of difficulty and diversity mentioned above is that they rely on a collection of representative outlier detection algorithms. Therefore, if these measures suggest that a dataset is very hard, we do not know for sure if there is a problem with the existing algorithms only (in which case the dataset is suitable for benchmarking in studies involving future algorithms) or if there is a problem with the dataset (the labels are fully or partially inconsistent with the data distribution and, therefore, the dataset may not be suitable for benchmarking). For instance, a low diversity score combined with a high difficulty score could occur, e.g., if the outliers in the ground truth are assigned low outlier scores (as inliers) by most algorithms, either because the algorithms are not good enough or because the objects labeled as outliers cannot be detected as such (or both). In order to help disambiguate these scenarios, we propose the use of IREOS as an additional, complementary measure of *hardness* for a dataset, which has the advantage of being algorithm independent, i.e., unlike the measures of difficulty and diversity, it depends only on the data. To that end, we take the ground truth labels of a dataset as a candidate (binary, top-n) solution and compute IREOS, adjusted for chance, for this "solution". Higher values (close to 1) would indicate that the ground truth labels are highly correlated with the separability of the objects labeled as outliers, while low values (around zero) would suggest the opposite.

In Figure 6, we reproduce an experiment from Campos et al. [11], where a scatter-plot of the diversity and difficulty scores for several downsampled variants of a variety of base datasets is

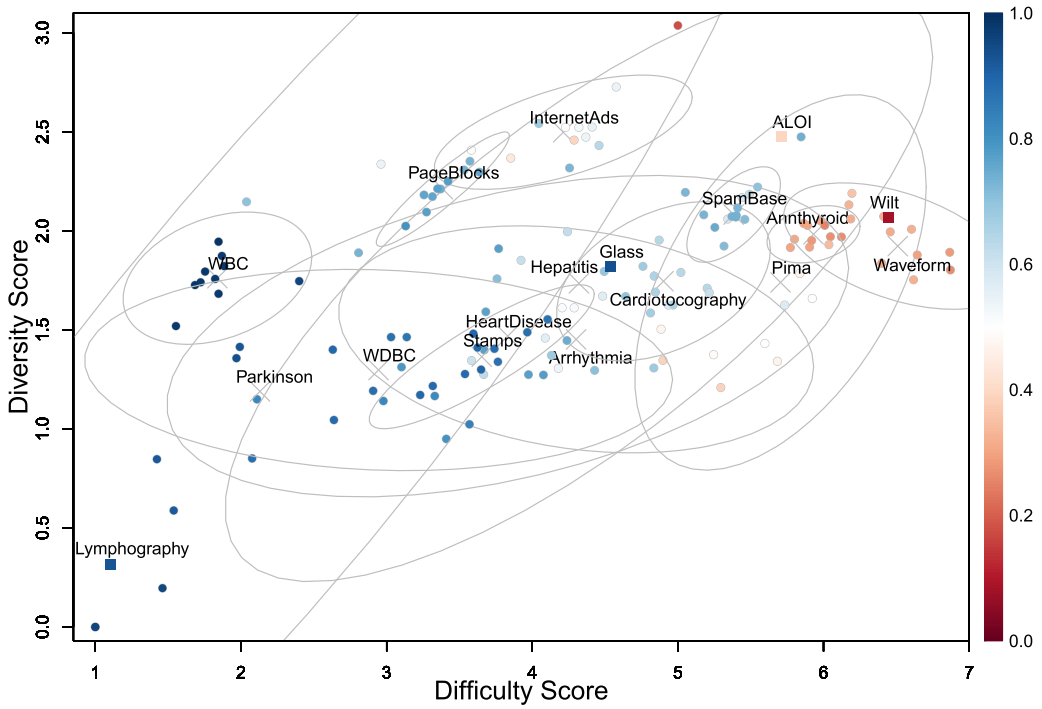


Fig. 6. Scatter-plot of *difficulty* vs. *diversity* for several downsampled variants of a variety of base datasets, following [11], but coloring points according to the IREOS assessment of the ground truth labels (adjusted for chance, with $m_{cl} = n$, where n is the number of outliers in the ground truth). Hotter colors represent *lower* IREOS values, i.e., the ground truth only agrees to a smaller degree with the notion of separability adopted by IREOS. Note that the IREOS score appears to be compatible with the difficulty score. The result obtained with $m_{cl} = 1$ is very similar.

displayed. For each dataset with different downsampled variants, the 95% confidence ellipse is shown as well as the individual points for each variant. The means of the ellipses are indicated by an "x", labeled with the corresponding base dataset. The datasets with only one variant are shown as a filled square. Unlike the original plot [11], our plot in Figure 6 maps the value of the adjusted IREOS for each dataset into a cold/hot color. Notice that datasets with high difficulty and diversity scores do tend to exhibit lower values of IREOS. Since these two strategies to assess the hardness of a dataset are based on completely different evaluation mechanisms, their compatible results suggest that both can be useful and that the intuition of separability adopted by IREOS seems to capture well the (not always explicit) intuition of the various outlier detection methods used in the study of Campos et al. [11].

Supported by these results, and to increase the chances that the solutions of the unsupervised outlier detection methods are not completely meaningless with respect to the ground truth labels, we chose for each of the 23 datasets from Campos et al. [11], the variant with the lowest difficulty score, whenever there are multiple downsampled variants of the dataset available for the chosen percentage of outliers. We chose from the variants with 5% of outliers, normalized, and without duplicates.

The collection of 23 real-world datasets [11], some of which are displayed in Figure 6, does not include 4 of the 11 datasets used in the experiments in our preliminary publication [41], namely,

Isolet, *Multiple Features*, *Optical Digits*, and *Vowel*. For the sake of completeness, here we will also include these four datasets, resulting in 27 datasets in total.

4.2 Methods and Measures

In our experiments, we evaluate results by contrasting the unsupervised assessment of candidate outlier solutions performed by IREOS against the supervised assessment using the ground truth, i.e., using the labels as provided in the datasets.[4] We then study the relationship between the quality assessments of the solutions with respect to the ground truth and the quality assessments of the solutions computed by IREOS. To assess the quality of a given scoring solution with respect to the ground truth, i.e., in a supervised way, we compute the Area Under the ROC Curve (ROC AUC) of the outlier scores in the solution under evaluation against the labels in the ground truth. For the real-world datasets, the ground truth labels are binary (i.e., outlier/inlier, according to a certain classification perspective), but this is not necessarily the case for the synthetic datasets, where the true probability distributions are known, and p-values (or rather their complement) can be used as reference outlier probabilities when computing the ROC AUC.

In addition to evaluating the IREOS assessment against the assessment using ground truth, we also contrast the assessment made by IREOS against a baseline. For that purpose, we use as the baseline a method originally designed for feature selection.[5] It is worth noting that the baseline index used here has not been originally published or by any means used for internal evaluation of outlier detection. Instead, the index was originally proposed as a filter method for feature selection. In order to use a feature selection method for internal evaluation of outlier detection solutions, the candidate solutions can be seen as a set of features (each candidate outlier detection solution corresponds to a feature). We take advantage of the fact that certain feature selection methods rank the features according to their importance, so, by treating candidate outlier solutions as features, these candidates will then be ranked.

In order to produce a diverse collection of candidate outlier solutions to be assessed, we collected 10 different solutions for each dataset. For the *real-world datasets*, these solutions were produced by the same outlier detection algorithms involved in the benchmarking study of Campos et al. [11], namely: COF [58], FastABOD [37], INFLO [30], KDEOS [55], KNN [51], KNNW [3, 4], LDOF [65], LDF [39], LOF [8], LoOP [35], ODIN [24], and SimplifiedLOF [56]. In addition to these algorithms, we also included a recent outlier detection method called GLOSH [10], for a total of 13 algorithms. All these algorithms share a parameterization in terms of a local neighborhood size, hereafter referred to as k, which is the main parameter in all of these algorithms (and the only parameter in many of them). For those algorithms that have additional parameters, following Campos et al. [11], we used the default values suggested by the original authors. For each real dataset, the algorithms were run by varying their neighborhood size k between 1 and 100.[6] In order to select a diverse subset of 10 candidate solutions for each dataset, we ensured that the best and the worst solutions according to their ROC AUC were selected, and the remaining solutions were selected trying to keep an interval as equally spaced as possible between solutions in terms of their ROC AUC values.

For the *synthetic datasets*, since the actual distributions used to generate the data are known, algorithms are not needed to produce suitable candidate solutions. Instead, the candidate solutions were produced artificially, in a fully controlled way. We start from the perfect solution given by

[4]Notice that these labels are not used by our index in any way, except if the index is specifically applied to assess the ground truth itself as a candidate solution, as in Figure 6.

[5]We would like to thank the anonymous reviewer for suggesting this method as the baseline for IREOS.

[6]Notice that, except for GLOSH, all these results have been precomputed and are available in the repository associated with the benchmarking study of Campos et al. [11].

the ground truth (ROC AUC = 1), which is as follows: based on the known covariance matrix of the multivariate normal distribution that generated each data object, the Mahalanobis distance between each object and the mean of the distribution is computed. The distribution of the Mahalanobis distances follows a χ^2 distribution with d degrees of freedom, from which an outlier probability for each data object can be readily obtained as the complement of the p-value. For each synthetic dataset, the corresponding p-value complements are taken as a non-binary ground truth, i.e., a perfect scoring solution. In order to produce a diverse collection of 10 candidate outlier solutions to be assessed, for each dataset, we start from the ideal solution as described and iteratively produce new solutions by swapping the outlier score of one outlier (p-value < 0.025) with the score of a random inlier. This way, at each new iteration, the ROC AUC deteriorates as we are worsening the ideal solution by flipping the scores of outliers and inliers (as per a 97.5% confidence level). Once again, in order to select a diverse subset of 10 candidate solutions for each dataset, we ensured that the best and the worst solutions according to their ROC AUC were selected, and the remaining solutions were selected trying to keep an interval as equally spaced as possible between solutions in terms of their ROC AUC values.

It is worth noticing that the outlier scores in the candidate outlier solutions obtained for the synthetic datasets already can be interpreted as (complementary) outlier probabilities (p-values), so the normalization procedure used by IREOS to produce regularized weights from the raw outlier scores is not necessary. As for the real datasets, whose candidate solutions have been obtained from a diverse collection of algorithms producing outlier scores with different characteristics, the normalization is required. In these cases, we applied the Gaussian scaling setting of the normalization framework by Kriegel et al. [36], which has been recommended by the authors based on the overall results presented in their article.

4.2.1 Categories of Experiments. We perform two main categories of experiments. The *first category* of experiments is designed to assess the compromise between computational efficiency and efficacy of the strategies proposed in Section 3.5 to speed up the computation of IREOS. First, each proposed strategy is evaluated separately in terms of computational gain and loss of approximation quality in relation to the original index. Then, all the proposed strategies are combined and compared against the original index.

The *second category* is designed to assess the effectiveness of IREOS, which is tested in two different experimental scenarios. In each of these scenarios, we assess the effectiveness of both original and approximate IREOS with all speedup strategies combined, as well as compare them to a baseline index. In the first scenario, we measure the goodness of fit between the rankings obtained by assessing the 10 candidate solutions of each dataset in a supervised way (using ROC AUC) and in an unsupervised way (using IREOS, approximate IREOS, and the baseline index). The goodness of fit is measured by computing the Spearman correlation between these two rankings of the same 10 candidate solutions, for each dataset. The second scenario involves model selection: using the quality assessments of the solutions computed by the three different indexes, we selected the best solution of each dataset according to their respective scores, these solutions are then compared in terms of ROC AUC against the best, worst, and expected (average) ROC AUC values across the entire collection of candidate solutions.

4.2.2 Parameters. We evaluate IREOS with $m_{cl} = 1$ (no modeling of clumps) and $m_{cl} = n$, where n is the number of outliers according to the ground truth (for synthetic datasets, n is the number of data objects with p-value < 0.025, i.e., around 2.5% of the dataset size). Since n is unknown in practical application scenarios, we also experiment with another (empirical and generic, not domain-specific or application-oriented) setting for the case where the optional mechanism for modeling clumps is used: $m_{cl} = \sqrt{5\% \cdot N}$.

For the evaluation of feature importance/relevance in an unsupervised setting by the baseline index, we use the Laplacian Score (LS) [26], which makes its evaluation based on the structure of a nearest neighbor graph. For the construction of such graph, we consider three neighborhood sizes k, namely: 1 and 100, as these were the minimum and maximum values used by the outlier detection algorithms themselves, and $\sqrt{N}$, which is common practice in the literature [16].

Following our preliminary publication [41], the number of discrete γ values for the computation of the index when the speedup strategies are not in place (including adaptive quadrature for numerical integration) is $n_\gamma = 100$. As previously discussed, the constant penalty cost for soft margin violations, C, only needs to be large enough to perform its intended role, and our experiments show that the results obtained with values of C varying several orders of magnitude are very similar and do not change the conclusions. For instance, we showed earlier [41] that the differences in the results for C varying from 100 to 800,000 are negligible. The experiments reported in this article have been run with $C = 100$.

4.3 Results

4.3.1 First Category of Experiments (Efficiency of Approximations). In this first category of experiments, we evaluate the fast, approximate strategies to compute IREOS discussed in Section 3.5. We start with a comparison between the computational cost of the index when using a fixed number $n_\gamma = 100$ of points to compute the separability curves and the cost when using the adaptive quadrature approach. We experiment with the estimated approximation error set to 0.01, 0.005, and 0.001. The results are shown in Table 2, where columns with header "Classifiers" show the average number of classifiers trained to evaluate a solution, whereas columns with header "Diff" show the average absolute difference between the index as computed with $n_\gamma = 100$ and as computed with a variable n_γ determined by the adaptive quadrature technique. All the averages were taken over the assessment of 30 candidate solutions, 10 for $m_{cl} = 1$, 10 for $m_{cl} = \sqrt{5\% \cdot N}$, and 10 for $m_{cl} = n$. For the estimated error of 0.01, which was the largest one used, the number of classifiers needed to evaluate a solution with adaptive quadrature falls as low as 6% of the amount required with n_γ fixed, while the observed approximation errors (Diff) are in the second or third decimal digit. For an estimated error of 0.001, all datasets required less than a third of the classifiers needed with n_γ fixed, and most Diff values were in the third or fourth decimal digit.

Our next experiment assesses the reduction of the computational cost when avoiding the unnecessary computation of separability curves for data objects with zero outlier probability (weight $w_{(\cdot)} = 0$) as well as by using the pruning strategy that stops early when ranking candidate solutions for model selection. The results are shown in Table 3. Notice that, in this comparison scenario, Diff is irrelevant, because: (a) Pruning by $w_{(\cdot)} = 0$ produces an exact result (Diff = 0), not an approximation; and (b) in model selection, the exact value of the index is not important, only the ranking of the candidate solutions matters, and the stop early pruning strategy produces the same ranking of candidate solutions as the one that would be obtained by sorting the solutions according to the index computed without any pruning. Notice from Table 3 that each of these pruning techniques is able to reduce computations to less than 50% of the original cost in most datasets.

Our third experiment assesses the trade-off between average runtime and quality of approximation of IREOS with the separability of each object computed using the object's kNN only, as opposed to using all the other objects in the dataset. We varied the number of the nearest neighbors, k, as: 10, 50, 100, and 250. The results are shown in Table 4, for a representative sub-collection of the datasets, all of which are computationally demanding. As can be seen, this approach provides by far the largest computational gains. In fact, for these datasets, each classifier required by the approximate IREOS has been computed on average with at most around 2% of the original runtime, in the case of the largest neighborhood, $k = 250$. Except for dataset Wilt with $k = 10$ (the

Table 2. Comparison Between IREOS with Fixed n_γ and Variable n_γ as Determined by the Adaptive Quadrature Approximation

Dataset	Fixed n_γ	Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.01$		Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.005$		Ad. Quad.: $\mathcal{E}_{2n_\gamma}(I) = 0.001$	
	Classifiers	Classifiers	Diff	Classifiers	Diff	Classifiers	Diff
Annthyroid	694,200	41,704.2 (6.01%)	0.035	62,081.2 (8.94%)	0.00155	76,952.8 (11.09%)	0.00036
Arrhythmia	25,600	1,536 (6%)	0.04594	5,684.2 (22.2%)	0.00347	8,143.2 (31.81%)	0.00075
Cardiotocography	173,400	10,614.8 (6.12%)	0.01484	16,917.4 (9.76%)	0.01709	27,562.2 (15.9%)	0.01543
Glass	21,400	1,288.8 (6.02%)	0.05718	3,492.6 (16.32%)	0.00173	4,996.8 (23.35%)	0.00031
HeartDisease	15,700	942 (6%)	0.03257	2,162.2 (13.77%)	0.00466	3,368 (21.45%)	0.00098
Hepatitis	7,000	420 (6%)	0.04733	1,354.4 (19.35%)	0.0033	1,889.8 (27%)	0.00057
InternetAds	168,200	10,341 (6.15%)	0.02633	28,822.2 (17.14%)	0.00705	44,124.4 (26.23%)	0.00521
Ionosphere	35,100	2,137.8 (6.09%)	0.04363	6,925.8 (19.73%)	0.00922	9,435.4 (26.88%)	0.00835
Isolet	91,000	5,460 (6%)	0.02557	18,553 (20.39%)	0.0052	26,693.2 (29.33%)	0.00089
Lymphography	14,800	888 (6%)	0.03175	2,108.4 (14.25%)	0.00453	3,246.2 (21.93%)	0.00105
MultipleFeature	40,800	2,448 (6%)	0.03655	8,070.8 (19.78%)	0.00529	12,005.6 (29.43%)	0.00107
OpticalDigits	114,400	6,889.8 (6.02%)	0.01573	17,985.8 (15.72%)	0.00786	28,545 (24.95%)	0.00146
PageBlocks	513,900	30,919.4 (6.02%)	0.01231	39,310.6 (7.65%)	0.00287	48,580.8 (9.45%)	0.00067
Parkinson	5,000	300 (6%)	0.04883	888 (17.76%)	0.00222	1,303.6 (26.07%)	0.00048
Pima	52,600	3,225.4 (6.13%)	0.02824	7,307.2 (13.89%)	0.00768	11,381.2 (21.64%)	0.0062
Shuttle	101,300	6,078 (6%)	0.01641	7,087.2 (7%)	0.0155	8,810.2 (8.7%)	0.01529
SpamBase	266,100	16,701.4 (6.28%)	0.02191	43,310.2 (16.28%)	0.00502	62,923 (23.65%)	0.001
Stamps	32,500	1,960.6 (6.03%)	0.02733	3,916.2 (12.05%)	0.00939	6,134.4 (18.88%)	0.00823
Vowel	10,000	600 (6%)	0.0545	980 (9.8%)	0.05453	1,576.6 (15.77%)	0.05357
Waveform	344,300	21,198.4 (6.16%)	0.00765	55,365.6 (16.08%)	0.00754	86,595.8 (25.15%)	0.00192
WBC	22,300	1,352.4 (6.06%)	0.05597	4,047.4 (18.15%)	0.00382	5,801 (26.01%)	0.00317
WDBC	36,700	2,231.4 (6.08%)	0.04083	6,520.2 (17.77%)	0.00432	9,673.8 (26.36%)	0.00209
Wilt	481,900	28,979.4 (6.01%)	0.01374	37,752.4 (7.83%)	0.01388	48,340.8 (10.03%)	0.01992
WPBC	19,800	1,188 (6%)	0.05295	3,778.2 (19.08%)	0.00299	5,555 (28.06%)	0.00048

Columns "Classifiers" show the average number of classifiers trained to evaluate a solution. Columns "Diff" show the average absolute difference between the index as computed with $n_\gamma = 100$ and as computed with a variable n_γ determined by the adaptive quadrature technique.

smallest neighborhood), the observed absolute difference (i.e., the approximation error) Diff was in the second or third decimal digit.

The results of an experiment that combines all the speedup strategies simultaneously are summarized in Table 5. In this case, for each dataset, we report the average time to compute the entire index for one candidate solution, if a single processing core was used.[7] The estimated error for adaptive quadrature and the number of nearest neighbors were set to 0.005 and 250, respectively.[8] As can be seen, with exception of the InternetAds dataset, IREOS can be computed in a few minutes without any major loss of accuracy, even considering a sequential (rather than parallelized) implementation.

4.3.2 Second Category of Experiments (Effectiveness). The results of the second category of experiments, for the first scenario involving the correlation between the rankings of candidate

[7]Although our experiments have been run with our parallelized code in a multi-core server, we report the sum of the runtimes required to train classifiers across multiple processing nodes, as a worst-case scenario assuming a non-parallelized implementation.

[8]This is also the setting used in the second category of experiments (effectiveness).

Table 3. Comparison between IREOS with and without Pruning Strategies

Dataset	No pruning	Pruning: $w_{(\cdot)} = 0$	Pruning: Stop Early
	Classifiers	Classifiers	Classifiers
Annthyroid	694,200	232,710 (33.52%)	193,816.7 (27.92%)
Arrhythmia	25,600	13,820 (53.98%)	13,753.33 (53.72%)
Cardiotocography	173,400	69,490 (40.07%)	66,096.67 (38.12%)
Glass	21,400	7,480 (34.95%)	7,010 (32.76%)
HeartDisease	15,700	7,250 (46.18%)	6,866.667 (43.74%)
Hepatitis	7,000	3,320 (47.43%)	3,236.667 (46.24%)
InternetAds	168,200	79,830 (47.46%)	78,270 (46.53%)
Ionosphere	35,100	21,890 (62.36%)	20,700 (58.97%)
Isolet	91,000	45,400 (49.89%)	44,393.33 (48.78%)
Lymphography	14,800	6,480 (43.78%)	6,210 (41.96%)
MultipleFeature	40,800	19,430 (47.62%)	19,080 (46.76%)
OpticalDigits	114,400	57,420 (50.19%)	53,643.33 (46.89%)
PageBlocks	513,900	204,170 (39.73%)	184,410 (35.88%)
Parkinson	5,000	2,230 (44.6%)	2,053.333 (41.07%)
Pima	52,600	20,930 (39.79%)	19,926.67 (37.88%)
Shuttle	101,300	47,120 (46.52%)	42,490 (41.94%)
SpamBase	266,100	172,240 (64.73%)	168,766.7 (63.42%)
Stamps	32,500	12,820 (39.45%)	12,406.67 (38.17%)
Vowel	10,000	3,310 (33.1%)	3,060 (30.6%)
Waveform	344,300	163,570 (47.51%)	158,636.7 (46.08%)
WBC	22,300	8,720 (39.1%)	8,166.667 (36.62%)
WDBC	36,700	15,660 (42.67%)	14,833.33 (40.42%)
Wilt	481,900	171,230 (35.53%)	148,256.7 (30.77%)
WPBC	19,800	8,200 (41.41%)	7,960 (40.2%)

Table 4. Comparison Between IREOS with and Without the use of kNN Approximation to Compute Separability

Dataset	Full Dataset	$k = 10$		$k = 50$		$k = 100$		$k = 250$	
	Runt. Class.	Diff	Runtime Class.	Diff	Runtime Class.	Diff	Runtime Class.	Diff	Runtime Class.
Cardiotocogr.	382ms	0.0565	0.07ms (0.02%)	0.0195	0.57ms (0.15%)	0.0172	1.69ms (0.44%)	0.0161	8.49ms (2.22%)
InternetAds	22s	0.0382	0.65ms (≈0%)	0.0121	7.01ms (0.03%)	0.0102	30.18ms (0.13%)	0.0087	493.93ms (2.17%)
PageBlocks	3s	0.0908	0.04ms (≈0%)	0.0418	0.38ms (0.01%)	0.0383	1.26ms (0.04%)	0.0294	6.89ms (0.2%)
SpamBase	1s	0.0369	0.07ms (≈0%)	0.0292	0.69ms (0.04%)	0.0336	2.54ms (0.15%)	0.0319	11.12ms (0.66%)
Waveform	1s	0.0308	0.04ms (≈0%)	0.0130	0.47ms (0.03%)	0.0078	1.58ms (0.08%)	0.0038	8.65ms (0.46%)
Wilt	2s	0.1105	0.04ms (≈0%)	0.0510	0.34ms (0.01%)	0.0410	1.13ms (0.04%)	0.0266	6.20ms (0.23%)

The reported runtimes are the average times to compute each classifier. The number of classifiers does not change in this strategy.

solutions as determined by unsupervised outlier evaluation indexes and by the ground truth, are summarized in Table 6.[9] For the real datasets, which are individual datasets, the entries denote the value of the Spearman correlation between the indexes scores and ROC AUC values for the

[9]Since the original IREOS does not take advantage of the speedup strategies, we have not computed the index for the datasets ALOI, KDDCup99, and PenDigits, as these large datasets are computationally very demanding.

Table 5. Comparison Between IREOS with and without Fast, Approximate Strategies

Dataset	No Approximation	Combined Approximate Strategies	
	Runtime Index	Runtime Index	Diff
Annthyroid	2348:54:58	0:4:5 (≈0%)	0.00265
Arrhythmia	0:13:56	0:1:45 (12.64%)	0.00356
Cardiotocography	18:26:45	0:1:24 (0.13%)	0.01707
Glass	0:2:9	0:0:18 (14%)	0.00182
HeartDisease	0:1:0	0:0:14 (23.53%)	0.00467
Hepatitis	0:0:7	0:0:1 (19.07%)	0.0033
InternetAds	1061:10:47	1:52:59 (0.18%)	0.00989
Ionosphere	0:13:47	0:1:4 (7.83%)	0.00924
Isolet	93:21:28	0:10:9 (0.18%)	0.0078
Lymphography	0:0:55	0:0:9 (16.92%)	0.00453
MultipleFeature	8:24:43	0:4:38 (0.92%)	0.00558
OpticalDigits	8:51:46	0:1:57 (0.37%)	0.00785
PageBlocks	491:47:43	0:2:24 (0.01%)	0.00172
Parkinson	0:0:3	0:0:0 (18.89%)	0.00222
Pima	0:30:43	0:0:45 (2.47%)	0.0079
Shuttle	3:18:20	0:0:39 (0.33%)	0.0161
SpamBase	124:9:0	0:5:45 (0.08%)	0.01186
Stamps	0:7:3	0:0:23 (5.53%)	0.00965
Vowel	0:0:40	0:0:1 (3.06%)	0.05453
Waveform	179:39:0	0:3:59 (0.04%)	0.00972
WBC	0:2:32	0:0:21 (13.88%)	0.00382
WDBC	0:13:19	0:0:42 (5.37%)	0.004
Wilt	359:23:27	0:2:8 (0.01%)	0.00333
WPBC	0:4:4	0:0:20 (8.57%)	0.00299

The reported runtimes are the average times needed to compute the entire index assuming a sequential (non-parallelized) implementation.

corresponding set of 10 candidate solutions. For the collection of synthetic data, the entries denote the average and the standard deviation of the Spearman correlation between the indexes scores and ROC AUC values over the multiple datasets of the collection.

Notice that, for the synthetic data, the Spearman correlation between IREOS (both original and approximate) and ROC AUC is maximum across the entire collection of 30 datasets. It shows that the ordering of the candidate solutions based on the notion of outlierness as captured by IREOS is perfectly aligned with the ordering of candidate solutions based on the notion of outlierness according to the underlying statistical mechanism that generated the data, irrespective of m_{cl}. Note that the way the solutions for synthetic datasets are generated makes the solutions numerically very similar, which causes LS to produce very similar scores for the set of solutions (with small variations), leading to a Spearman correlation close to zero (random).

For the real-world datasets, which represent a more intricate evaluation scenario where the ground truth labels may be fully or partially inconsistent with the spatial distribution of the data, as discussed in Section 4.1, there is a clear correlation most of the time. However, low or even negative values of correlation can be observed for datasets that appear to have inconsistent labelings according to the measures of diversity and difficulty proposed by Campos et al. [11], e.g., Annthyroid and Wilt (see Figure 6). The Wilt database, in particular, showed a strong negative

Table 6. Spearman Correlation Between Unsupervised Outlier Evaluation Indexes and ROC AUC (Based on ground truth labels)

Spearman correlation	IREOS $m_{cl} = 1$	IREOS $m_{cl} = \sqrt{5\% \cdot N}$	IREOS $m_{cl} = n$	ApproxIREOS $m_{cl} = 1$	ApproxIREOS $m_{cl} = \sqrt{5\% \cdot N}$	ApproxIREOS $m_{cl} = n$	LS $k = 1$	LS $k = \sqrt{N}$	LS $k = 100$
ALOI	—	—	—	−0.115	−0.115	**0.103**	−0.188	−0.139	−0.139
Annthyroid	**0.248**	−0.285	−0.394	**0.248**	−0.285	−0.37	−0.382	−0.248	−0.248
Arrhythmia	0.539	0.673	0.721	0.539	0.673	0.721	0.576	**0.83**	0.515
Cardiotocography	0.758	0.915	**0.976**	0.758	0.915	**0.976**	−0.188	0.345	0.636
Glass	0.467	**0.479**	0.382	0.467	**0.479**	0.394	0.273	0.321	0.285
HeartDisease	**0.927**	0.903	0.903	**0.927**	0.903	0.903	0.273	0.758	0.77
Hepatitis	**0.939**	0.903	0.903	**0.939**	0.903	0.903	0.188	0.879	0.042
InternetAds	0.636	**0.915**	0.782	0.709	0.891	0.721	−0.794	−0.794	−0.721
Ionosphere	0.648	0.721	0.709	0.648	0.721	0.709	**0.879**	0.794	0.624
Isolet	0.273	0.6	0.6	0.2	0.588	0.588	**0.806**	0.697	0.394
KDDCup99	—	—	—	**0.855**	0.782	0.733	0.37	0.309	0.527
Lymphography	0.661	**0.758**	**0.758**	0.661	0.721	**0.758**	−0.358	−0.661	−0.006
MultipleFeature	0.152	**0.515**	**0.515**	0.212	0.503	0.503	−0.406	0.103	−0.079
OpticalDigits	0.733	**0.782**	**0.782**	0.648	**0.782**	**0.782**	0.552	0.6	0.418
PageBlocks	0.927	**0.964**	0.952	0.927	**0.964**	0.952	0.758	0.794	0.855
Parkinson	**0.515**	**0.515**	−0.37	**0.515**	**0.515**	**0.515**	0.042	0.115	0.309
PenDigits	—	—	—	0.6	**0.661**	**0.661**	0.455	0.467	0.467
Pima	0.758	0.83	**0.879**	0.758	**0.879**	**0.879**	0.479	0.515	0.539
Shuttle	**0.867**	**0.867**	**0.867**	**0.867**	**0.867**	**0.867**	0.576	0.818	0.394
SpamBase	0.794	0.842	0.842	0.794	0.842	0.842	**0.867**	0.77	0.709
Stamps	0.588	0.636	**0.685**	0.588	0.636	**0.685**	−0.224	−0.467	−0.067
Vowel	**0.927**	0.709	0.661	**0.927**	0.709	0.648	0.079	−0.733	−0.527
Waveform	0.818	0.855	**0.891**	0.794	0.855	**0.891**	0.879	0.818	0.758
WBC	0.042	0.042	0.042	−0.018	0.042	0.042	**0.079**	−0.079	−0.103
WDBC	0.915	**0.952**	**0.952**	0.915	0.915	0.915	0.758	0.673	0.127
Wilt	−0.794	−0.903	−0.939	−0.794	−0.903	−0.915	−0.709	−0.576	**-0.552**
WPBC	0.479	0.648	0.806	0.479	0.648	0.806	**0.879**	0.855	0.685
Synthetic	**1 ± 0**	**1 ± 0**	**1 ± 0**	**1 ± 0**	**1 ± 0**	**1 ± 0**	−0.152 ± 0.329	−0.036 ± 0.119	−0.028 ± 0.109

IREOS (columns 2–4), approximate IREOS (columns 5–7), and Baseline LS (columns 8–10).

correlation across all the measures. The Principal Component Analysis (PCA) visualization of the five-dimensional Wilt database, using the first two principal components, shown in Figure 7, can help explain this result. As can be seen, most of the outliers, plotted in red, lie deep in the denser region of the cluster, which can explain the negative correlation with IREOS, since most objects labeled as outliers in the ground truth seem to be clear inliers from the viewpoint of the spatial distribution of the data.

When comparing the results for different values of clump size, the correlations in Table 6 are similar across different m_{cl} values for most datasets, which suggests that this optional parameter is not critical. As previously discussed, whether clumps should be modeled or not, and, if so, what the expectations about the maximum size of a clump (as opposed to a cluster) should be, are problem-specific decisions. With our three general settings for m_{cl} across all the datasets, we notice that, while for most datasets the results are similar, for certain datasets, such as Isolet, Multiple Features, and WPBC, modeling clumps ($m_{cl} > 1$) provides better results, whereas the opposite occurs for a few other datasets, most noticeably Parkinson and Vowel. Overall, our generic (i.e., blind, not

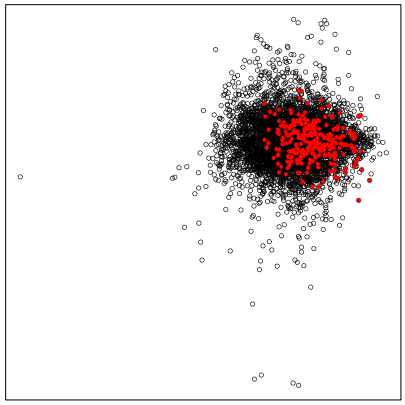
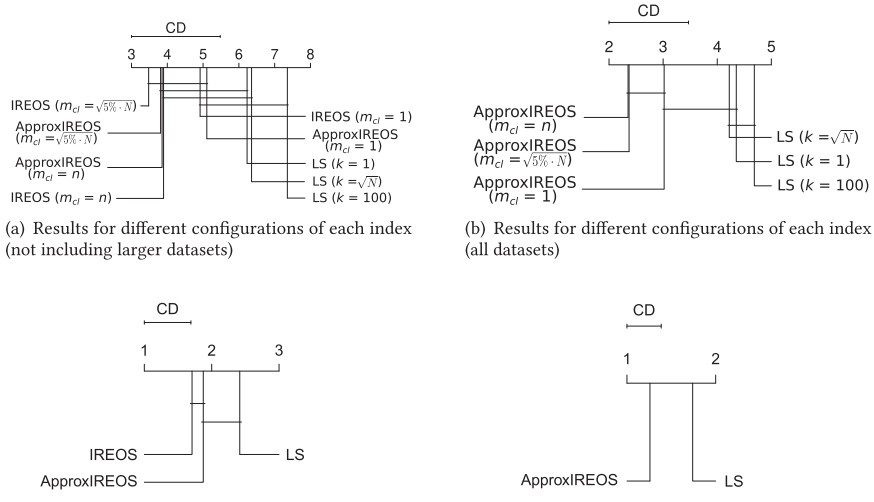


Fig. 7. PCA visualization of the Wilt dataset using the first two principal components. Outliers according to the ground truth are displayed in red.



(a) Results for different configurations of each index (not including larger datasets)

(b) Results for different configurations of each index (all datasets)

(c) Best result from each index across all configurations (not including larger datasets)

(d) Best result from each index across all configurations (all datasets)

Fig. 8. Average ranking of the indexes over the Spearman correlation experiment.

application-specific) heuristic $m_{cl} = \sqrt{5\% \cdot N}$ seems to provide a good compromise, if not the best result, in almost all cases. Comparisons of performances of IREOS for different values of m_{cl} must be taken with a grain of salt, though, because the ground truth (i.e., labels based on some sort of semantic) can be seen as only one of the different possible perspectives of what outliers should be in each dataset.

The results of the first scenario presented in Table 6 can also be summarized in Figure 8.[10] Figure 8(a) shows the average ranking of Spearman correlation for the different configurations of the indexes. The length of the upper bar (CD) indicates the critical difference of the well-known

[10]Note that when the comparison involves the three indexes, we do not include the datasets ALOI, KDDCup99, and PenDigits, as we did not compute IREOS for them.

Friedman/Nemenyi statistical test [15] at significance level $\alpha = 0.05$. When we compare the average ranking of original IREOS, where the index is computed without any approximation strategy, to the approximate IREOS, we can see that both indexes exhibit similar performance for the different configurations of clump size. In fact, when we compare the correlation results from Table 6 between the original and approximate IREOS, we can see that, apart from a few localized differences, the overall results remain stable, and the conclusions do not change. As can also be noticed and already discussed earlier, when we use modeling of clumps ($m_{cl} = \sqrt{5\% \cdot N}$ and $m_{cl} = n$), IREOS achieves superior results. For example, when compared only the approximate IREOS against the baseline in Figure 8(b) (comparison over a higher number of datasets), the use of modeling of clumps achieves superior results with statistical difference when compared to any configuration of LS. Notice, however, that even for $m_{cl} = 1$ (no modeling of clumps), IREOS is, on average, better than any parametrization of LS, but using this configuration, there is a statistical difference only for LS with $k = 100$. In Figure 8(c) and 8(d), instead of showing the results for each evaluation index and configuration separately, we take the best result from each index (highest correlation for each dataset, irrespective of index configuration). From Figure 8(c), we can see that while overall results remain stable when comparing the original to the approximate IREOS, the original IREOS tends to provide better results, albeit without statistical difference. In this setup, we can also see even more clearly the superiority of IREOS over LS. Even in the scenario with a smaller number of datasets (Figure 8(c)), IREOS presents superior results with statistical difference when compared to LS, while approximate IREOS presents statistical difference only in the scenario with a higher number of datasets (Figure 8(d)).

The results for the experimental scenario involving model selection are summarized in Table 7, which shows the ROC AUC values corresponding to the worst, the expected (average), and the best candidate outlier solution for each dataset, along with the ROC AUC value of the solution selected as best according to indexes (i.e., in an unsupervised way). For each selected solution, the table also indicates which algorithm produced this solution. To get a better sense of where the selected solutions are located within the distribution of the ROC AUC values for all candidate solutions, we also show box plots of the distributions for each dataset in Figure 9. The position of the solutions selected by the indexes are indicated by special symbols in the plots.

By using the original IREOS with $m_{cl} = n$ for the model selection, one would select the most accurate solution according to the ground truth in 10 out of the 24 datasets. IREOS with $m_{cl} = 1$ makes the best choice for 9 out of the 24. IREOS with $m_{cl} = \sqrt{5\% \cdot N}$ makes the best choice for 11 out of the 24. The approximate IREOS has similar performance, however, in a higher number of datasets. Irrespective of the settings of clump size, approximate IREOS makes the best choice for 10 out of the 27 datasets. In cases where the best solution according to IREOS (both original and approximate) is not the one with highest ROC AUC value, the choice is often much better than the expected (average) and worst values, and often close to the quality of the best possible selection, for at least two of the three clump size settings used. Exceptions are the datasets ALOI, Annthyroid, Wilt, and Glass. The ALOI, Annthyroid, and Wilt datasets were already expected to exhibit poor results due to the label inconsistency problem discussed in the previous experimental scenario and in Section 4.1. The low value of ROC AUC in the best solution chosen by IREOS for the Glass dataset can be explained due to the large number of ties in the scorings produced by the solution. This solution produced only 8 non-null values of scorings for the 214 objects in the dataset. When the ROC curve is computed, past these 8 objects, the curve will walk diagonally, resembling a random solution, as shown in Figure 10(a), which explains the low ROC AUC value. The PCA visualization of the Glass dataset for the first two principal components (Figure 10(b)) suggests that the eight objects detected as outliers in the solution selected by IREOS, plotted in red, are apparently not in clear disagreement with the spatial distribution of the data.

Table 7. Summarization of the ROC AUC Values for all Candidate Solutions Used for Model Selection

ROC AUC	Min	Avg	Max	IREOS $m_{cl} = 1$		IREOS $m_{cl} = \sqrt{5\%N}$		IREOS $m_{cl} = n$		ApproxIREOS $m_{cl} = 1$		ApproxIREOS $m_{cl} = \sqrt{5\%N}$		ApproxIREOS $m_{cl} = n$		LS $k = 1$		LS $k = \sqrt{N}$		LS $k = 100$	
ALOI	0.544	0.675	0.805	–		–		–		**0.632**	KNN	**0.632**	KNN	**0.632**	KNN	0.602	KNNW	0.602	KNNW	0.602	KNNW
Annthyroid	0.528	0.63	0.732	**0.642**	KNNW	0.574	KNN	0.551	KNN	**0.642**	KNNW	0.574	KNN	0.596	LOF	0.551	KNN	0.619	INFLO	0.619	INFLO
Arrhythmia	0.5	0.737	0.916	0.879	GLOSH	0.879	GLOSH	0.879	GLOSH	0.879	GLOSH	0.879	GLOSH	0.879	GLOSH	**0.916**	LDF	**0.916**	LDF	0.77	LoOP
Cardiotocog	0.536	0.689	0.839	0.74	LOF	0.74	LOF	**0.839**	LOF	0.74	LOF	0.74	LOF	**0.839**	LOF	0.536	LDOF	0.74	LOF	0.74	LOF
Glass	0.492	0.699	0.904	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH	**0.539**	GLOSH
Heart D.	0.387	0.687	0.97	0.847	KNNW	0.847	KNNW	0.847	KNNW	0.847	KNNW	0.847	KNNW	0.847	KNNW	0.387	LDOF	**0.909**	KNNW	0.847	KNNW
Hepatitis	0.075	0.522	0.96	**0.96**	COF	**0.96**	COF	**0.96**	COF	**0.96**	COF	**0.96**	COF	**0.96**	COF	0.075	LOF	**0.96**	COF	0.473	SimLOF
InternetAds	0.413	0.656	0.895	**0.843**	FABOD	**0.843**	FABOD	**0.843**	FABOD	**0.843**	FABOD	**0.843**	FABOD	**0.843**	FABOD	0.413	COF	0.467	COF	0.413	COF
Ionosphere	0.514	0.745	0.96	0.868	LoOP	0.868	LoOP	0.868	LoOP	0.868	LoOP	0.868	LoOP	0.868	LoOP	**0.914**	LDF	0.868	LoOP	0.868	LoOP
Isolet	0.116	0.569	1	0.116	GLOSH	**1**	COF	**1**	COF	0.116	GLOSH	0.91	ODIN	0.91	ODIN	0.819	ODIN	0.819	ODIN	0.819	ODIN
KDDCup99	0.364	0.68	0.99	–		–		–		**0.923**	KNNW	**0.923**	KNNW	**0.923**	KNNW	0.645	INFLO	0.645	INFLO	0.645	INFLO
Lymphogr.	0.43	0.748	1	**0.95**	LDOF	**0.95**	LDOF	**0.95**	LDOF	**0.95**	LDOF	**0.95**	LDOF	**0.95**	LDOF	0.606	LDOF	0.606	LDOF	0.68	SimLOF
Multiple F.	0.233	0.638	0.994	0.918	LOF	**0.994**	COF	**0.994**	COF	0.918	LOF	0.918	LOF	0.918	LOF	0.233	GLOSH	0.766	KDEOS	0.766	KDEOS
Optical D.	0.164	0.596	0.999	**0.999**	KNN	0.737	LDF	0.737	LDF	**0.999**	KNN	0.737	LDF	0.737	LDF	0.737	LDF	0.737	LDF	0.556	KDEOS
PageBlocks	0.506	0.728	0.943	0.896	LOF	**0.943**	LDF	**0.943**	LDF	0.896	LOF	**0.943**	LDF	**0.943**	LDF	0.896	LOF	**0.943**	LDF	**0.943**	LDF
Parkinson	0.448	0.765	1	**1**	FABOD	**1**	FABOD	0.51	LDF	**1**	FABOD	**1**	FABOD	**1**	FABOD	0.875	COF	0.875	COF	0.917	COF
PenDigits	0.431	0.715	0.993	–		–		–		**0.993**	GLOSH	**0.993**	GLOSH	**0.993**	GLOSH	0.623	LOF	0.623	LOF	0.623	LOF
Pima	0.499	0.67	0.834	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	**0.762**	LOF	0.725	LOF
Shuttle	0.415	0.708	0.992	**0.992**	LDF	0.87	KNNW	0.87	KNNW	**0.992**	LDF	0.87	KNNW	0.87	KNNW	**0.992**	LDF	**0.992**	LDF	0.683	ODIN
SpamBase	0.463	0.625	0.779	**0.779**	KNNW	**0.779**	KNNW	**0.779**	KNNW	**0.779**	KNNW	**0.779**	KNNW	**0.779**	KNNW	0.712	INFLO	0.712	INFLO	0.712	INFLO
Stamps	0.38	0.655	0.921	**0.921**	KNN	**0.921**	KNN	**0.921**	KNN	**0.921**	KNN	**0.921**	KNN	**0.921**	KNN	0.687	INFLO	0.687	INFLO	0.687	INFLO
Vowel	0.223	0.614	1	**1**	FABOD	**1**	FABOD	0.916	COF	**1**	FABOD	**1**	FABOD	0.916	COF	0.916	COF	0.487	INFLO	0.4	GLOSH
Waveform	0.5	0.66	0.796	0.708	INFLO	**0.796**	LDF	**0.796**	LDF	0.708	INFLO	**0.796**	LDF	**0.796**	LDF	0.708	INFLO	0.737	SimLOF	0.737	SimLOF
WBC	0.438	0.721	0.997	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH	**0.997**	GLOSH
WDBC	0.497	0.745	0.988	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH	**0.988**	GLOSH
Wilt	0.34	0.527	0.713	**0.424**	KNNW	**0.424**	KNNW	0.382	KNNW	**0.424**	KNNW	**0.424**	KNNW	0.382	KNNW	0.382	KNNW	0.382	KNNW	0.382	KNNW
WPBC	0.401	0.493	0.583	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF	**0.543**	LDF

Nine last columns indicate the solutions selected by IREOS (columns 5–7), approximate IREOS with all speedup strategies combined (columns 8–10), and by the baseline LS (columns 11–13) with different configurations.

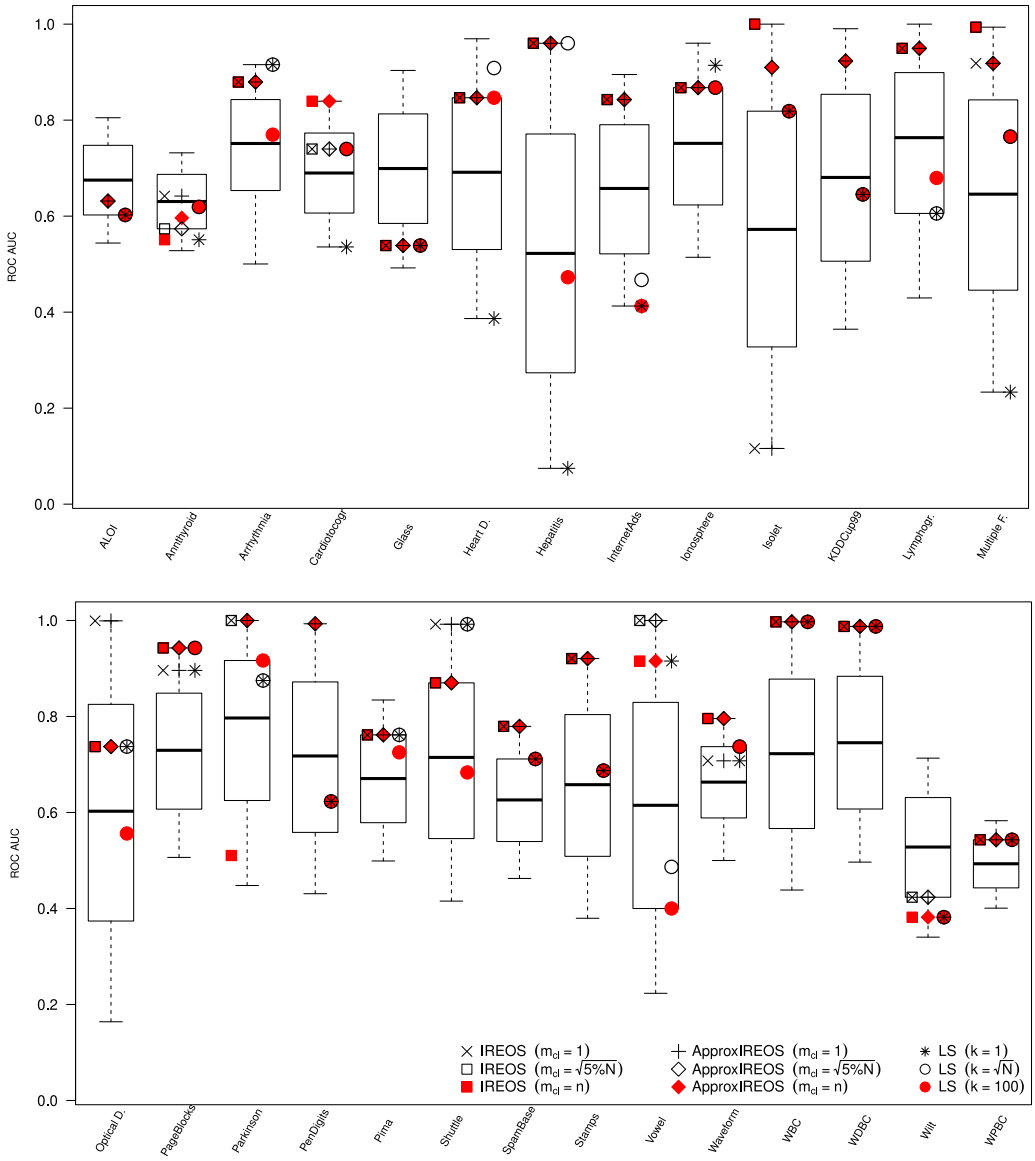
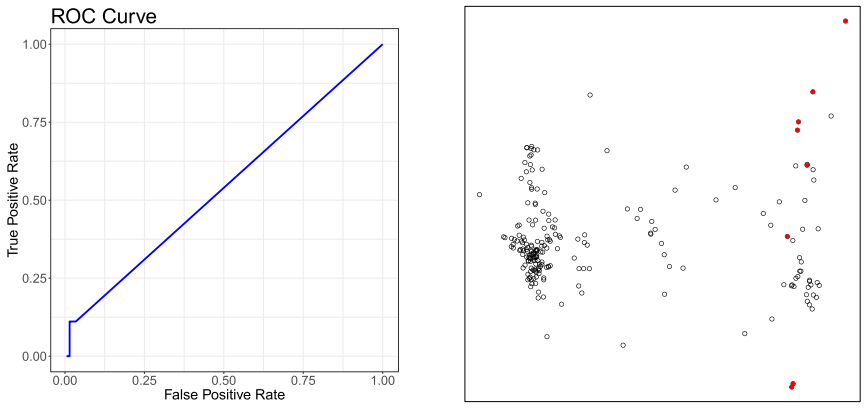


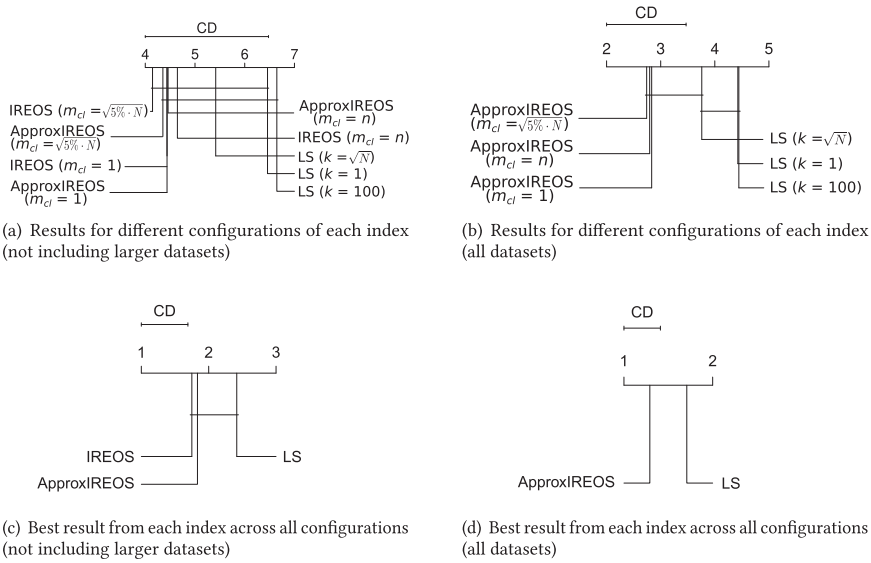
Fig. 9. Distribution of the ROC AUC values for all candidate solutions used in the model selection experiments. The position of the solutions selected by the indexes as best are indicated by their respective special symbols (left column: IREOS variants, center column: Approximate IREOS variants, right column: LS variants).

In Figure 11, we summarize the model selection experiment. In Figure 11(a), we can see again IREOS outperforming LS in all configurations. In fact, we can observe that in the model selection experiment (Table 7 and Figure 9), LS using $k = \sqrt{N}$ makes the best choice only for 6 out of the 27 datasets. For the other configurations, the performance is even worse. LS with $k = 1$ makes the best choice for 4 out of the 27, and LS with $k = 100$ makes the best choice for 3 out of the 27. When comparing this scenario of model selection (Figure 11(b)) to the previous scenario of



(a) ROC Curve (b) PCA visualization

Fig. 10. Best solution selected by IREOS variants (exact and approximate) for the Glass dataset. Outliers highlighted in red.



(a) Results for different configurations of each index (not including larger datasets)

(b) Results for different configurations of each index (all datasets)

(c) Best result from each index across all configurations (not including larger datasets)

(d) Best result from each index across all configurations (all datasets)

Fig. 11. Average ranking of the indexes over the model selection experiment.

the correlation (Figure 8(b)), the difference between modeling clumps or not modeling clumps has reduced, the average ranking for $m_{cl} = \sqrt{5\% \cdot N}$ and $m_{cl} = n$ has increased, and there is no longer statistical difference with respect to LS with $k = \sqrt{N}$. For the other two configurations of LS ($k = 1$ and $k = 100$), approximate IREOS is better with statistical difference for any configuration of m_{cl}. In Figure 11(c) and 11(d), we repeat the average ranking, but now considering the best solution selected by each index irrespective of their configurations. In this setup, we can see again that IREOS (both original and approximate) has a clear advantage over LS.

4.3.3 Final Remarks. We performed two different categories of experiments. In the first category of experiments, we evaluated the fast, approximate strategies to compute IREOS. We performed three different experiments. In the first experiment, we evaluated the approximation of

the separability curves via the adaptive quadrature. Computing the index using adaptive quadrature can be seen as an improvement rather than an approximation of the index as it represents both more efficient and more effective computation. By using the adaptive quadrature, there is no need to set an arbitrary value for n_γ. Instead, the user has to specify a tolerance error, which is common practice in many machine learning algorithms that involve iterative optimization, such as ANNs, SVMs, KLR, k-means, among others. In the second experiment, we assessed the reduction of the computational cost when avoiding unnecessary computation via pruning. Pruning by $w_{(\cdot)}$ produces the exact index, and therefore, it should be performed whenever it is possible. In the third experiment, we evaluated the approximation of the index with the separability of each object computed using the object's kNN only. This approximation provides by far the largest computational gains, reducing the classifier complexity from $O(N^3)$ to $O(k^3)$. Therefore, it should always be considered for very large datasets.

In the second category, we evaluated the effectiveness of IREOS in an experimental scenario involving the correlation between the rankings of candidate solutions as determined by IREOS and by the ground truth, as well as in another scenario involving practical experiments of model selection. In these two different experimental scenarios, we evaluated IREOS using the optional parameter m_{cl}, which allows the domain expert to adjust the notion of clump size vs. cluster size, in three different settings. We set $m_{cl} = 1$ to represent cases where clumps are not modeled, $m_{cl} = n$ to represent cases where the clumps are modeled with domain information, and $m_{cl} = \sqrt{5\% \cdot N}$ to represent cases where the clumps are modeled in the absence of domain information. When comparing the results for different values of clump size, the results are similar across different m_{cl} values for most datasets, which suggests that this optional parameter is not critical. However, we notice that, while for most datasets the results are similar, modeling clumps ($m_{cl} > 1$) tended to provide better results overall (see Figures 8 and 11). Therefore, in absence of domain information to decide whether clumps should be modeled or not, or what should be the maximum size of a clump, we recommend our generic (i.e., not application specific/informed) heuristic $m_{cl} = \sqrt{5\% \cdot N}$, which has systematically provided a good compromise, if not the best result, in almost all cases.

In the second category of experiments, we also compared IREOS against a baseline method. On average, all the different configurations of IREOS provided better results than any of the different configurations of the baseline, with statistical significance for some of the configurations. When only the best result of each index is considered, irrespective of their configurations, we observed statistically significant difference between IREOS (superior) and the baseline. Notice that the baseline used in our experiments has not been proposed specifically for this task, and we do not recommend its use in practical applications for different reasons. First, in contrast to IREOS, where the optional parameter for modeling of clumps has a clear interpretation, allowing domain experts to adjust the notion of clump size *vs.* cluster size, the neighborhood size k of the baseline (LS) is difficult to interpret, particularly as computed in the outlier scorings space. Second, the importance/relevance of each candidate outlier solution, as seen as candidate features for a feature selection method, depends on the other solutions/features, such that a given solution may be deemed more (or less) important depending on the remaining candidate solutions under assessment.

5 CONCLUSIONS

We addressed in this article the long-term open problem [67] of internal and relative evaluation of outlier detection results, that is, the assessment of the quality of results of unsupervised outlier detection methods without referring to external information (such as class labels). In the typical application scenario of outlier detection, such external information about what the outliers are is not available in advance, and results need to be assessed by domain experts.

IREOS is the first measure to allow such quality assessment of solutions automatically and, as a consequence, to select better solutions (models, parametrizations) for a given problem. We discussed the properties of IREOS, including derived statistics, p-values, and adjustment for chance. Experiments with synthetic and real data, using both artificially generated candidate solutions as well as real candidate solutions from outlier detection algorithms, show the usefulness of IREOS for unsupervised quality assessment of outlier detection results.

It is important to acknowledge that no single index can capture all possible facets of the unsupervised outlier detection problem. In fact, in the related area of data clustering, relying on multiple indexes is considered good practice and a variety of indexes have been proposed over the past few decades [61]. Some works in the clustering literature have taken advantage of such diversity to build ensembles of validation indexes [28, 62]. In the realm of unsupervised outlier detection, for which evaluation is still in its infancy, the development of new evaluation indexes is an important topic for future work. For instance, the use of different classifiers, other than those used in our article, could give rise to new variants of IREOS. Different types of classifiers could lead to indexes with different biases, which nevertheless would still follow the same fundamental intuition behind IREOS. We hope, though, that our work will also stimulate the development of indexes supported on different grounds.

A ADJUSTMENT FOR CHANCE AND STATISTICAL VALIDATION

A.1 Exact Computation

A.1.1 Scoring Solutions. Initially, we consider the evaluation of *full scorings*, where a random solution according to the adopted null model assumes that the outlier weights $w_{(\cdot)}$ are randomly and independently assigned to objects following a uniform distribution within $[0, 1]$. In order to adjust IREOS for chance, all we need is to derive term $E\{I\}$ in Equation (13) for this null model. Notice that this term can be written from Equation (9) as:

$$E\{I\} = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} E\{\bar{p}(\gamma_l)\}, \tag{15}$$

where

$$\bar{p}(\gamma_l) = \frac{\sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot w_j}{\sum_{j=1}^{N} w_j} = \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot \tilde{w}_j, \tag{16}$$

$$\tilde{w}_j = \frac{w_j}{\sum_{j=1}^{N} w_j}. \tag{17}$$

Term $E\{\bar{p}(\gamma_l)\}$ in Equation (15) can be written from Equation (16) as:

$$E\{\bar{p}(\gamma_l)\} = E\left\{ \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot \tilde{w}_j \right\}. \tag{18}$$

The random scoring solutions $w_{(\cdot)}$ are independent from the data objects, but the objects' separabilities $p(\cdot, \cdot)$ will depend on $w_{(\cdot)}$ when the optional modeling of clumps is in place, i.e., when $m_{cl} > 1$ in Equation (12). In this case, it is hard to derive $E\left\{\sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot \tilde{w}_j\right\}$ analytically, due to the complex relation between $p(\cdot, \cdot)$ and $w_{(\cdot)}$, which depends on the kernel-based soft margin classifier formulation and optimization solution. For this reason, we initially assume the scenario where the optional model of clumps is not in place, i.e., $m_{cl} = 1$.

Recall that, when $m_{cl} = 1$, the classifiers just try to discriminate between each data object $\mathbf{x}_j$ and the other objects, no matter their outlier labeling or scorings. In this case, $p(\cdot, \cdot)$ does not depend

on $w_{(\cdot)}$ and, thus, Equation (18) can be written as:[11]

$$E\{\bar{p}(\gamma_l)\} = \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot E\left\{\tilde{w}_j\right\}. \tag{19}$$

Term $E\{\tilde{w}_j\}$ can be computed analytically as follows. From Equation (17), it is clear that $\sum_{j=1}^{N} \tilde{w}_j = 1$, thereby:

$$E\left\{\sum_{j=1}^{N} \tilde{w}_j\right\} = \sum_{j=1}^{N} E\left\{\tilde{w}_j\right\} = E\{1\} = 1, \tag{20}$$

and since the weights w_j are i.i.d.:

$$E\{\tilde{w}_1\} = E\{\tilde{w}_2\} = \cdots = E\{\tilde{w}_N\}, \tag{21}$$

which in Equation (20) yields:

$$N \cdot E\{\tilde{w}_j\} = 1, \tag{22}$$

and, accordingly:

$$E\{\tilde{w}_j\} = \frac{1}{N}. \tag{23}$$

Substituting Equation (23) into Equation (19), we have the following result:

$$E\{\bar{p}(\gamma_l)\} = \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \cdot E\{\tilde{w}_j\} = \frac{1}{N} \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l), \tag{24}$$

which in Equation (15) yields:

$$E\{I\} = \frac{1}{n_\gamma} \sum_{l=1}^{n_\gamma} \left(\frac{1}{N} \sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l) \right). \tag{25}$$

A.1.2 Top-n Solutions. For binary top-n solutions, the previous null model is no longer applicable as there are not any scores available. The adopted null model in this case assumes that a random solution consists of a subset of n objects to be labeled as outliers, which are drawn from the dataset $\mathbf{X}$ with uniform probability. We can therefore interpret such a random solution as a sample $\mathbf{S}_R \subset \mathbf{X}$ of size $|\mathbf{S}_R| = n$ taken without replacement from a population $\mathbf{X}$ with size $|\mathbf{X}| = N$.

Recall that, in binary top-n solutions, the average separability term $\bar{p}(\gamma_l)$ is not weighted, and it is computed over the separabilities of the top-n objects only, i.e.,:

$$\bar{p}(\gamma_l) = \frac{1}{n} \sum_{\mathbf{x}_j \in \mathbf{S}_R} p(\mathbf{x}_j, \gamma_l), \tag{26}$$

from which we can take the expectation as:

$$E\{\bar{p}(\gamma_l)\} = \frac{1}{n} \sum_{x_j \in \mathbf{S}_R} E\{p(\mathbf{x}_j, \gamma_l)\} = E\{p(\mathbf{x}_j, \gamma_l)\}. \tag{27}$$

This is an instance of the well-known result that the expected value for the mean of an i.i.d. sample of size n is the mean of the population. Here, for a given γ_l, our (finite) population consists of the N precomputed values $p(\mathbf{x}_j, \gamma_l)$ for all data objects $\mathbf{x}_j$ in the database $\mathbf{X}$. In fact, for $m_{cl} = 1$, the separabilities $p(\cdot, \cdot)$ depend only on the data, not on any particular realization $\mathbf{S}_R$ of random

[11]When $m_{cl} = 1$, the separabilities can be pre-computed for each data object, irrespective of the outlier solution under evaluation and, accordingly, they become constant terms in the adopted null model.

candidate outliers, and therefore they can be independently precomputed. Taking their average gives the exact value for $E\{\bar{p}(\gamma_l)\}$, i.e.,:

$$E\{\bar{p}(\gamma_l)\} = E\{p(\mathbf{x}_j, \gamma_l)\} = \frac{1}{N}\sum_{j=1}^{N} p(\mathbf{x}_j, \gamma_l). \tag{28}$$

Substituting this equation into Equation (15) yields once again the very same result in Equation (25). In other words, even though the null models are different, the expected value of the index is the same, given by Equation (25), for both scoring or binary top-n solutions.

A.1.3 Statistical Validation—Top-n Solutions. The variance is not needed for the adjustment for chance in Equation (13), but it can be useful for statistical validation when this type of validation is required. Since our index is given by a sum of random variables $\frac{1}{n_\gamma}\bar{p}(\gamma_l)$ over γ_l, we can compute the variance of the index as:

$$\mathrm{Var}\{I\} = \mathrm{Var}\left\{\frac{1}{n_\gamma}\sum_{l=1}^{n_\gamma}\bar{p}(\gamma_l)\right\} = \frac{1}{n_\gamma^2}\sum_{l_1,l_2=1}^{n_\gamma}\mathrm{Cov}\left(\bar{p}(\gamma_{l1}), \bar{p}(\gamma_{l2})\right), \tag{29}$$

which can also be rewritten equivalently as:

$$\mathrm{Var}\{I\} = \frac{1}{n_\gamma^2}\sum_{l=1}^{n_\gamma}\mathrm{Var}\{\bar{p}(\gamma_l)\} + \frac{2}{n_\gamma^2}\sum_{l_1=1}^{l_2-1}\sum_{l_2=2}^{n_\gamma}\mathrm{Cov}\left(\bar{p}(\gamma_{l1}), \bar{p}(\gamma_{l2})\right). \tag{30}$$

This equivalent form emphasizes the possible lack of independence of $\bar{p}(\gamma_l)$ over γ_l, specifically when the second term is not null. For the first term, it follows from Equation (26) that:[12]

$$\mathrm{Var}\{\bar{p}(\gamma_l)\} = \frac{1}{n^2}\sum_{\mathbf{x}_j\in\mathbf{S}_R}\mathrm{Var}\{p(\mathbf{x}_j, \gamma_l)\} = \frac{1}{n}\mathrm{Var}\{p(\mathbf{x}_j, \gamma_l)\}, \tag{31}$$

i.e., the variance of the sample mean is the variance of the population over the sample size. Analogously, for the covariance one has

$$\mathrm{Cov}\left(\bar{p}(\gamma_{l1}), \bar{p}(\gamma_{l2})\right) = \frac{1}{n}\mathrm{Cov}\left(p(\mathbf{x}_j, \gamma_{l1}), p(\mathbf{x}_j, \gamma_{l2})\right), \tag{32}$$

Equations (31) and (32), which are required by Equation (30), can both be exactly computed once we have precomputed the whole population $p(\cdot,\cdot)$, which is possible when $m_{cl} = 1$. In this case, provided that the sample size is not critically small, the Central Limit Theorem (CLT) ensures that, for each γ_l, the sample mean $\bar{p}(\gamma_l)$ follows at least approximately a Normal distribution, i.e.,

$$\bar{p}(\gamma) \sim \mathcal{N}(E\{\bar{p}(\gamma)\}, \mathrm{Var}\{\bar{p}(\gamma)\}). \tag{33}$$

This means that, for random binary solutions $\mathbf{S}_R$ and $m_{cl} = 1$, our index in Equation (6) is given by a sum of normally distributed variables $\frac{1}{n_\gamma}\bar{p}(\gamma_l)$ over γ_l. The sum of normally distributed random variables

$$\mathbf{X} \sim \mathcal{N}\left(\mu_\mathbf{X}, \sigma_\mathbf{X}^2\right) \tag{34}$$

and

$$\mathbf{Y} \sim \mathcal{N}\left(\mu_\mathbf{Y}, \sigma_\mathbf{Y}^2\right) \tag{35}$$

[12] Since the population is of a finite size (N), though, when considering sampling without replacement and sample sizes n significantly large w.r.t. N (e.g., more than 5%), a finite population correction factor $(N-n)/(N-1)$ can be used to adjust the computed variance [59].

is also normally distributed, i.e.,:

$$\mathbf{X} + \mathbf{Y} \sim \mathcal{N}\left(\mu_{\mathrm{X}} + \mu_{\mathrm{Y}}, \sigma^2_{\mathrm{X+Y}}\right) \tag{36}$$

where [52]

$$\sigma^2_{\mathrm{X+Y}} = \sigma^2_{\mathrm{X}} + \sigma^2_{\mathrm{Y}} + 2\,\mathrm{Cov}(\sigma_{\mathrm{X}}, \sigma_{\mathrm{Y}}) \tag{37}$$

This leads to the interesting result that, in this particular setting (binary top-n solutions and $m_{cl} = 1$), our index IREOS, as a sum of sample means, will follow at least approximately a Normal distribution according to the CLT, i.e., $I \sim \mathcal{N}(E\{I\}, \mathrm{Var}\{I\})$, with mean $E\{I\}$ and variance $\mathrm{Var}\{I\}$ computed in an exact way as described above.

Since we know such a mean and variance for the population, we thus can go beyond the ordinary adjustment for chance (Equation (13)) and perform statistical validation as well. Particularly, if we are given a certain outlier detection solution, $\mathbf{S}$, and the corresponding value for our adjusted index, $I_{adj}(\mathbf{S})$, we can assess the statistical significance of $I_{adj}(\mathbf{S})$ by means of a z-test. In this case, a p-value can be trivially computed based on the Normal assumption by contrasting $I_{adj}(\mathbf{S})$ against the null hypothesis of a random solution [59].

A.2 Approximate Computation Via Monte Carlo

The exact computations described above presume $m_{cl} = 1$. For different evaluation setups, $p(\mathbf{x}_j, \gamma_l)$ can no longer be independently precomputed for each object $\mathbf{x}_j \in \mathbf{X}$, as the separability of a given object as assessed by the classifiers now depends also on the binary labels or the degrees of outlierness assigned to the other objects of the dataset, for each possible random solution. This means that, for a given γ_l, the size of our finite population expands from N to at least $\binom{N}{n}$ (for the binary top-n problem) and, as such, it can easily become intractable for exhaustive computations. Even when $m_{cl} = 1$, precomputing N terms $p(\mathbf{x}_j, \gamma_l)$ for each γ_l (i.e., $N \cdot n_\gamma$ in total) may be computationally prohibitive for large databases as well, as each term demands to train an independent classifier.

To make adjustment for chance feasible when $m_{cl} > 1$ or N is large, and also to make statistical validation possible in evaluation scenarios other than the binary top-n case, we can use Monte Carlo simulations in order to estimate the relevant statistics rather than trying to compute them in an exact and exhaustive way. The idea is to sample a number n_{MC} of random outlier detection solutions whereby the desired statistical moments can be estimated. In particular, the expected value in Equation (13) can be directly estimated from the sample.

When statistical validation is needed, we also need to estimate the baseline distribution under the null hypothesis. There are different alternatives. For binary top-n solutions, if the normality assumption is evoked from the CLT, a parametric approach is possible based on a t-student distribution with the sample estimates for the mean and variance (i.e., a t-test, which is known to be robust even when normality is not fully satisfied [1]). Alternatively, p-values can be directly derived from observed histograms in a non-parametric way [27] for each candidate outlier solution ω_i.

The sample size, n_{MC}, clearly represents a trade-off between computational burden and accuracy. Larger (smaller) samples lead to more (less) accurate estimations yet from a larger (smaller) number of trained classifiers. Rather than setting the value for n_{MC} arbitrarily, one can also determine n_{MC} automatically, by specifying (i) a certain significance level as the probability that the sample mean will fall within, and (ii) a prespecified confidence interval around the population mean [59].

REFERENCES

[1] C. Alan Boneau. 1960. The effects of violations of assumptions underlying the t test. *Psychological Bulletin* 57, 1 (1960), 49–64. DOI: https://doi.org/10.1037/h0041412

[2] Fabrizio Angiulli and Fabio Fassetti. 2009. DOLPHIN: An efficient algorithm for mining distance-based outliers in very large datasets. *ACM Transactions on Knowledge Discovery from Data* 3, 1 (2009), 4:1–4:57. DOI: https://doi.org/10.1145/1497577.1497581

[3] Fabrizio Angiulli and Clara Pizzuti. 2002. Fast outlier detection in high dimensional spaces. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'02)*. DOI: https://doi.org/10.1007/3-540-45681-3_2

[4] Fabrizio Angiulli and Clara Pizzuti. 2005. Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering* 17, 2 (2005), 203–215. DOI: https://doi.org/10.1109/TKDE.2005.31

[5] Uri M. Ascher and Chen Greif. 2011. *A First Course in Numerical Methods*. Society for Industrial and Applied Mathematics.

[6] Vic Barnett and Toby Lewis. 1994. *Outliers in Statistical Data* (3rd ed.). John Wiley & Sons.

[7] Stephen D. Bay and Mark Schwabacher. 2003. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*. 29–38. DOI: https://doi.org/10.1145/956750.956758

[8] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)*. 93–104. DOI: https://doi.org/10.1145/342009.335388

[9] Richard L. Burden, J. Douglas Faires, and Annette M. Burden. 2016. *Numerical Analysis*. Cengage learning.

[10] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. 2015. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data* 10, 1 (2015), 5:1–5:51. DOI: https://doi.org/10.1145/2733381

[11] Guilherme O. Campos, Arthur Zimek, Jörg Sander, Ricardo J. G. B. Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E. Houle. 2016. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery* 30, 4 (2016), 891–927. DOI: https://doi.org/10.1007/s10618-015-0444-8

[12] Nick Craswell. 2009. *Precision at n*. Springer, 2127–2128. DOI: https://doi.org/10.1007/978-0-387-39940-9_484

[13] Philip J. Davis and Philip Rabinowitz. 1984. *Methods of Numerical Integration* (2nd ed.). Academic Press.

[14] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: Simplified data processing on large clusters. *Communications of the ACM* 51, 1 (2008), 107–113. DOI: https://doi.org/10.1145/1327452.1327492

[15] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7 (2006), 1–30.

[16] Richard O. Duda, Peter E. Hart, and David G. Stork. 2000. *Pattern Classification* (2nd ed.). Wiley-Interscience.

[17] Amol Ghoting, Srinivasan Parthasarathy, and Matthew Eric Otey. 2008. Fast mining of distance-based outliers in high-dimensional datasets. *Data Mining and Knowledge Discovery* 16, 3 (2008), 349–364. DOI: https://doi.org/10.1007/s10618-008-0093-2

[18] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB'99)*. 518–529.

[19] Pedro Gonnet. 2012. A review of error estimation in adaptive quadrature. *ACM Computing Surveys* 44, 4 (2012), 22:1–22:36. DOI: https://doi.org/10.1145/2333112.2333117

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. The MIT Press.

[21] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. 2001. On clustering validation techniques. *Journal of Intelligent Information Systems* 17, 2–3 (2001), 107–145. DOI: https://doi.org/10.1023/A:1012801612483

[22] Jiawei Han, Micheline Kamber, and Jian Pei. 2011. *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

[23] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2013. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd ed.). Springer-Verlag, New York.

[24] Ville Hautamäki, Ismo Kärkkäinen, and Pasi Fränti. 2004. Outlier detection using k-nearest neighbour graph. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)*. 430–433. DOI: https://doi.org/10.1109/ICPR.2004.671

[25] Douglas M. Hawkins. 1980. *Identification of Outliers*. Chapman and Hall.

[26] Xiaofei He, Deng Cai, and Partha Niyogi. 2005. Laplacian score for feature selection. In *Proceedings of the 18th International Conference on Neural Information Processing Systems (NIPS'05)*. 507–514.

[27] Anil K. Jain and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice-Hall.

[28] Pablo A. Jaskowiak, Davoud Moulavi, Antonio C. S. Furtado, Ricardo J. G. B. Campello, Arthur Zimek, and Jörg Sander. 2016. On strategies for building effective ensembles of relative clustering validity criteria. *Knowledge and Information Systems* 47, 2 (2016), 329–354. DOI: https://doi.org/10.1007/s10115-015-0851-6

[29] Wen Jin, Anthony K. H. Tung, and Jiawei Han. 2001. Mining top-n local outliers in large databases. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*. 293–298. DOI: https://doi.org/10.1145/502512.502554

[30] Wen Jin, Anthony K.H. Tung, Jiawei Han, and Wei Wang. 2006. Ranking outliers using symmetric neighborhood relationship. In *Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'06)*. 577–593. DOI: https://doi.org/10.1007/11731139_68

[31] S. S. Keerthi, K. B. Duan, S. K. Shevade, and A. N. Poo. 2005. A fast dual algorithm for kernel logistic regression. *Machine Learning* 61, 1–3 (2005), 151–165. DOI: https://doi.org/10.1007/s10994-005-0768-5

[32] Edwin M. Knorr and Raymond T. Ng. 1998. Algorithms for mining distance-based outliers in large datasets. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)*. 392–403.

[33] Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov. 2000. Distance-based outliers: Algorithms and applications. *The VLDB Journal* 8, 3–4 (2000), 237–253. DOI: https://doi.org/10.1007/s007780050006

[34] George Kollios, Dimitrios Gunopulos, Nick Koudas, and Stefan Berchtold. 2003. Efficient biased sampling for approximate clustering and outlier detection in large data sets. *IEEE Transactions on Knowledge & Data Engineering* 15, 5 (2003), 1170–1187. DOI: https://doi.org/10.1109/TKDE.2003.1232271

[35] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. 2009. LoOP: Local outlier probabilities. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'09)*. 1649–1652. DOI: https://doi.org/10.1145/1645953.1646195

[36] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. 2011. Interpreting and unifying outlier scores. In *Proceedings of the 2011 SIAM International Conference on Data Mining (SDM'11)*.13–24. DOI: https://doi.org/10.1137/1.9781611972818.2

[37] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. 2008. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*. 444–452. DOI: https://doi.org/10.1145/1401890.1401946

[38] Guy F. Kuncir. 1962. Algorithm 103: Simpson's rule integrator. *Communications of the ACM* 5, 6 (1962), 347. DOI: https://doi.org/10.1145/367766.368179

[39] Longin Jan Latecki, Aleksandar Lazarevic, and Dragoljub Pokrajac. 2007. Outlier detection with kernel density functions. In *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM'07)*. 61–75. DOI: https://doi.org/10.1007/978-3-540-73499-4_6

[40] Yuh-Jye Lee and Olvi L. Mangasarian. 2001. RSVM: Reduced support vector machines. In *Proceedings of the 2001 SIAM International Conference on Data Mining (SDM'01)*. 1–17. DOI: https://doi.org/10.1137/1.9781611972719.13

[41] Henrique O. Marques, Ricardo J.G.B. Campello, Arthur Zimek, and Jörg Sander. 2015. On the internal evaluation of unsupervised outlier detection. In *Proceedings of the 27th International Conference on Scientific and Statistical Database Management (SSDBM'15)*. 7:1–7:12. DOI: https://doi.org/10.1145/2791347.2791352

[42] B. Micenková, R. T. Ng, X. H. Dang, and I. Assent. 2013. Explaining outliers by subspace separability. In *Proceedings of the 13th International Conference on Data Mining (ICDM'13)*. 518–527. DOI: https://doi.org/10.1109/ICDM.2013.132

[43] Glenn W. Milligan and Martha C. Cooper. 1985. An examination of procedures for determining the number of clusters in a data set. *Psychometrika* 50, 2 (1985), 159–179. DOI: https://doi.org/10.1007/BF02294245

[44] Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*.625–632. DOI: https://doi.org/10.1145/1102351.1102430

[45] Gustavo H. Orair, Carlos H. C. Teixeira, Wagner Meira, Jr., Ye Wang, and Srinivasan Parthasarathy. 2010. Distance-based outlier detection: Consolidation and renewed bearing. *Proceedings of the VLDB Endowment* 3, 1–2 (2010), 1469–1480. DOI: https://doi.org/10.14778/1920841.1921021

[46] Navneet Panda, Edward Y. Chang, and Gang Wu. 2006. Concept boundary detection for speeding up SVMs. In *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*. 681–688. DOI: https://doi.org/10.1145/1143844.1143930

[47] Guansong Pang, Longbing Cao, Ling Chen, and Huan Liu. 2017. Learning homophily couplings from non-IID data for joint feature selection and noise-resilient outlier detection. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. 2585–2591. DOI: https://doi.org/10.24963/ijcai.2017/360

[48] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. 2003. LOCI: fast outlier detection using the local correlation integral. In *Proceedings of the 19th International Conference on Data Engineering (ICDE'03)*. 315–326. DOI: https://doi.org/10.1109/ICDE.2003.1260802

[49] John Platt. 1998. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. Technical Report.

[50] John Platt. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans (Eds.). MIT Press, 61–74.

[51] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD'00)*. 427–438. DOI: https://doi.org/10.1145/342009.335437

[52] Sheldon M. Ross. 2009. *Introduction to Probability Models* (10th ed.). Elsevier.
[53] Bernhard Schölkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press.
[54] Erich Schubert, Remigius Wojdanowski, Arthur Zimek, and Hans-Peter Kriegel. 2012. On evaluation of outlier rankings and outlier scores. In *Proceedings of the 2012 SIAM International Conference on Data Mining (SDM'12)*. 1047–1058. DOI: https://doi.org/10.1137/1.9781611972825.90
[55] Erich Schubert, Arthur Zimek, and Hans-Peter Kriegel. 2014. Generalized outlier detection with flexible kernel density estimates. In *Proceedings of the 2014 SIAM International Conference on Data Mining (SDM'14)*. 542–550. DOI: https://doi.org/10.1137/1.9781611973440.63
[56] Erich Schubert, Arthur Zimek, and Hans-Peter Kriegel. 2014. Local outlier detection reconsidered: A generalized view on locality with applications to spatial, video, and network outlier detection. *Data Mining and Knowledge Discovery* 28, 1 (2014), 190–237. DOI: https://doi.org/10.1007/s10618-012-0300-z
[57] Erich Schubert, Arthur Zimek, and Hans-Peter Kriegel. 2015. Fast and scalable outlier detection with approximate nearest neighbor ensembles. In *Proceedings of the 20th International Conference on Database Systems for Advanced Applications (DASFAA'15)*. 19–36. DOI: https://doi.org/10.1007/978-3-319-18123-3_2
[58] Jian Tang, Zhixiang Chen, Ada Wai-chee Fu, and David W. Cheung. 2002. Enhancing effectiveness of outlier detections for low density patterns. In *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)*. 535–548. DOI: https://doi.org/10.1007/3-540-47887-6_53
[59] Mario F. Triola. 2007. *Elementary Statistics* (10th ed.). Pearson.
[60] Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.
[61] Lucas Vendramin, Ricardo J. G. B. Campello, and Eduardo R. Hruschka. 2010. Relative clustering validity criteria: A comparative overview. *Statistical Analysis and Data Mining* 3, 4 (2010), 209–235. DOI: https://doi.org/10.1002/sam.v3:4
[62] Lucas Vendramin, Pablo A. Jaskowiak, and Ricardo J. G. B. Campello. 2013. On the combination of relative clustering validity criteria. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management (SSDBM'13)*. 4:1–4:12. DOI: https://doi.org/10.1145/2484838.2484844
[63] Nguyen Hoang Vu and Vivekanand Gopalkrishnan. 2009. Efficient pruning schemes for distance-based outlier detection. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECMLP-KDD'09)*. 160–175. DOI: https://doi.org/10.1007/978-3-642-04174-7_11
[64] Y. Weng, N. Zhang, and C. Xia. 2018. Multi-agent-based unsupervised detection of energy consumption anomalies on smart campus. *IEEE Access* 7 (2018), 2169–2178. DOI: https://doi.org/10.1109/ACCESS.2018.2886583
[65] Ke Zhang, Marcus Hutter, and Huidong Jin. 2009. A new local distance-based outlier detection approach for scattered real-world data. In *Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'09)*. 813–822. DOI: https://doi.org/10.1007/978-3-642-01307-2_84
[66] Ji Zhu and Trevor Hastie. 2005. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics* 14, 1 (2005), 185–205. DOI: https://doi.org/10.1198/106186005X25619
[67] Arthur Zimek, Ricardo J. G. B. Campello, and Jörg Sander. 2013. Ensembles for unsupervised outlier detection: Challenges and research questions a position paper. *SIGKDD Explorations Newsletter* 15, 1 (2013), 11–22. DOI: https://doi.org/10.1145/2594473.2594476
[68] Arthur Zimek, Ricardo J. G. B. Campello, and Jörg Sander. 2014. Data perturbation for outlier detection ensembles. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management (SSDBM'14)*. 13:1–13:12. DOI: https://doi.org/10.1145/2618243.2618257
[69] Arthur Zimek, Matthew Gaudet, Ricardo J. G. B. Campello, and Jörg Sander. 2013. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. 428–436. DOI: https://doi.org/10.1145/2487575.2487676

Received October 2019; revised February 2020; accepted April 2020