




Article

Unsupervised Outlier Detection: A Meta-Learning Algorithm Based on Feature Selection

Vasilis Papastefanopoulos ^{*,†} , Pantelis Linardatos [†]  and Sotiris Kotsiantis [†] 

Department of Mathematics, University of Patras, 26504 Patras, Greece; p.linardatos@upnet.gr (P.L.); sotos@math.upatras.gr (S.K.)

* Correspondence: vasileios.papastefanopoulos@upatras.gr

† These authors contributed equally to this work.

Abstract: Outlier detection refers to the problem of the identification and, where appropriate, the elimination of anomalous observations from data. Such anomalous observations can emerge due to a variety of reasons, including human or mechanical errors, fraudulent behaviour as well as environmental or systematic changes, occurring either naturally or purposefully. The accurate and timely detection of deviant observations allows for the early identification of potentially extensive problems, such as fraud or system failures, before they escalate. Several unsupervised outlier detection methods have been developed; however, there is no single best algorithm or family of algorithms, as typically each relies on a measure of ‘outlierness’ such as density or distance, ignoring other measures. To add to that, in an unsupervised setting, the absence of ground-truth labels makes finding a single best algorithm an impossible feat even for a single given dataset. In this study, a new meta-learning algorithm for unsupervised outlier detection is introduced in order to mitigate this problem. The proposed algorithm, in a fully unsupervised manner, attempts not only to combine the best of many worlds from the existing techniques through ensemble voting but also mitigate any undesired shortcomings by employing an unsupervised feature selection strategy in order to identify the most informative algorithms for a given dataset. The proposed methodology was evaluated extensively through experimentation, where it was benchmarked and compared against a wide range of commonly-used techniques for outlier detection. Results obtained using a variety of widely accepted datasets demonstrated its usefulness and its state-of-the-art results as it topped the Friedman ranking test for both the area under receiver operating characteristic (ROC) curve and precision metrics when averaged over five independent trials.

Keywords: machine learning; data science; unsupervised outlier; detection; meta-learning; feature selection; ensemble-learning



check for updates

Citation: Papastefanopoulos, V.; Linardatos, P.; Kotsiantis, S. Unsupervised Outlier Detection: A Meta-Learning Algorithm Based on Feature Selection. *Electronics* **2021**, *10*, 2236. <https://doi.org/10.3390/electronics10182236>

Academic Editors: Unsang Park and Jihoon Yang

Received: 30 August 2021

Accepted: 10 September 2021

Published: 12 September 2021

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Outlier detection refers to the problem of the identification and, where appropriate, the removal of anomalous observations from data. There is no official definition of what constitutes an outlier [1]; they can be broadly seen as observations that deviate enough from the majority of observations in a dataset to be considered the product of a different generative process. Hence, given a dataset, the percentage of outlier observations is usually small, typically lower than 5% [2].

Numerous real-world applications rely on sophisticated data analyses to filter out outliers and maintain system reliability [3,4]. This is especially true in safety critical environments, where the presence of outliers may imply abnormal activity, such as fraud, or may indicate irregular running conditions in a system, which may hinder its performance significantly and ultimately result in system failure [5,6].

A significant part of the literature focuses on the undesired properties of outliers; they can nevertheless reveal valuable information about previously unknown characteristics of

the systems and entities that generated them. Therefore, shedding light on such characteristics and properties can provide interesting insights and potentially lead to important discoveries [7].

Existing outlier detection methods have been proven to be efficient for a diverse pool of applications, including brain tumour diagnosis, telecommunications fraud identification, credit card fraud detection, component failure prediction, network intrusions [1] and many others that require the processing of high-dimensional data [8] or huge amounts of data streams [9]. The removal of outlier observations from data is also to the benefit of machine learning and statistical modelling, as an outlier-free dataset enables such algorithms to capture the emerging trends more accurately. This makes outlier detection an essential challenge to be addressed during the integration of big data [10] and an invaluable step in any data cleaning process [11].

There are three main machine learning frameworks to approach the problem of outlier detection [1,12]. The first approach, unsupervised outlier detection, assumes little prior knowledge of the data. Under this approach, unlabeled data are split into clusters and any observations separated from the main clusters are flagged as potential outliers. The second approach, supervised outlier detection, tries to explicitly model and learn what constitutes an outlier and what separates an outlier from normal observations. As with any supervised learning setting, a good amount of data needs to have already been labelled explicitly either as outliers or normal observations for a classifier to be trained. The last approach is considered related to semi-supervised classification and is mainly used in situations where labelled anomalous observations are hard to obtain. In this case, a classifier is trained using only labelled examples of normal data, through which a definition of some ‘normality boundary’ is learnt. Subsequently, any new observations that fall outside this boundary are considered outliers.

In this work, a new meta-learning algorithm for unsupervised outlier detection is proposed. While plenty of unsupervised outlier methods are present in the existing literature, they usually put too much weight on a single a measure of ‘outlierness’ such as density or distance, ignoring other measures. The suggested methodology integrates existing outlier-detection techniques using a voting ensemble, but at the same time employs an unsupervised feature selection process in order to overcome the individual shortcomings and finally produce a single confidence score. The effectiveness of the presented method was showcased through extensive empirical evaluation against a wide range of commonly-used techniques for outlier detection, using a variety of datasets and metrics.

The rest of this paper is organised as follows: in Section 2, the most well-known techniques for unsupervised outlier detection are presented. Subsequently, in Section 3, commonly used unsupervised features selection methodologies are analysed. Then, in Section 4, the proposed methodology is introduced, while in Section 5, the conducted experiments are outlined and their results are discussed. Finally, the concluding remarks of this study are provided in Section 6.

2. Unsupervised Outlier Detection Methods

In this section, the most well-known and commonly used techniques for unsupervised outlier detection are briefly presented, namely k-nearest neighbours, local outlier factor, cluster-based local outlier factor, histogram-based outlier score, one-class support vector machines, minimum covariance determinant, principal component analysis, angle-based outlier detection, feature bagging, isolation forest as well as various ensemble voting methodologies. These algorithms also form the benchmark presented in [13], and it should be noted that there are many other well-established methods including but not limited to self-organising maps [14], subspace outlier detection [15], rotation-based outlier detection [16], copula-based outlier detection [17], variational autoencoders [18] and beta-variational autoencoders [19] as well as ensemble methodologies, such as the locally selective combination of parallel outlier ensembles [20].

In Figure 1, these different algorithms and their learnt decision boundaries as well as successes and failures for the dataset are displayed. Through this illustration, it becomes evident that, based on different measures of outlierness, the emerging decision boundaries can considerably vary and therefore so can the obtained predictions for certain datapoints.

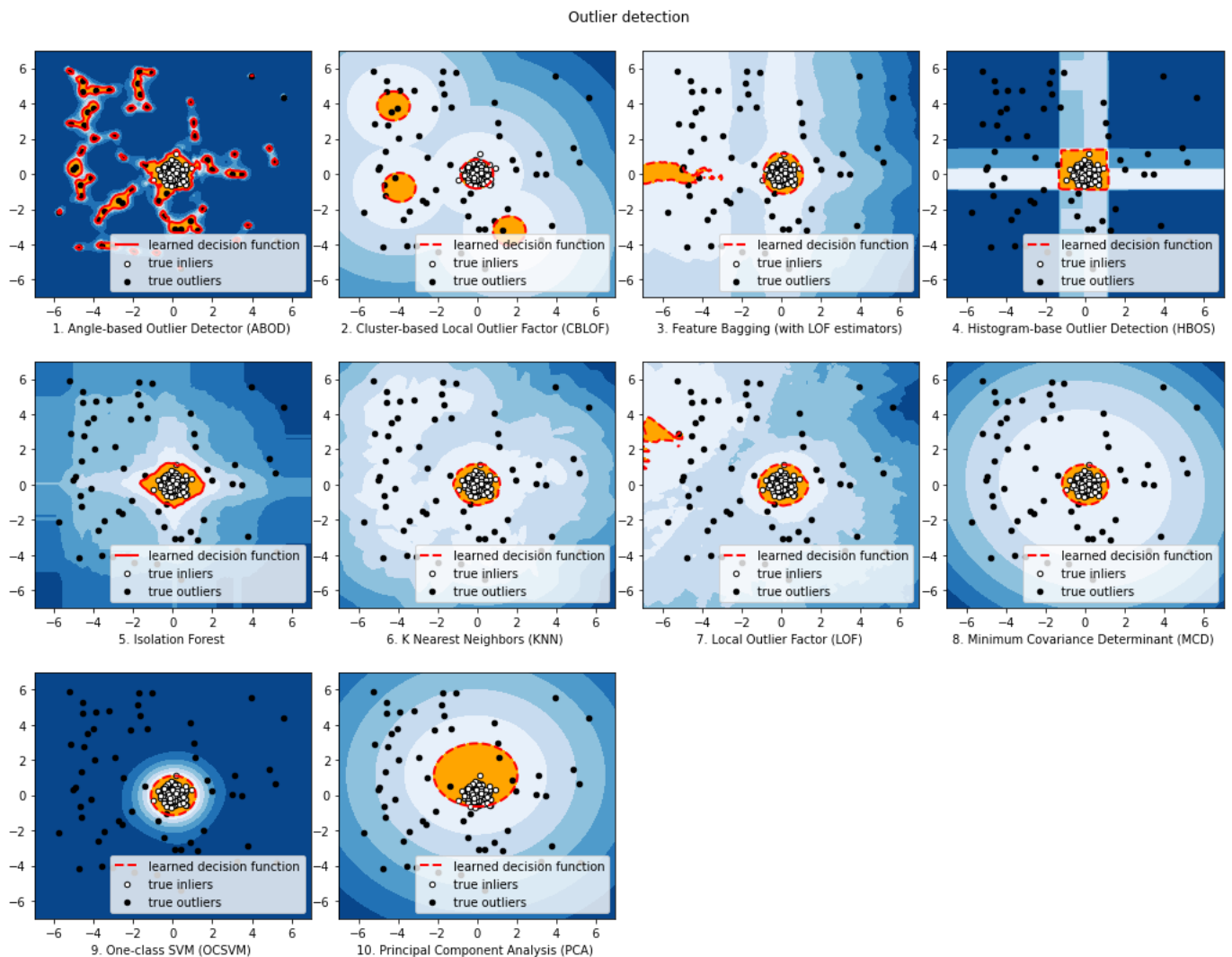


Figure 1. Decision boundaries learnt by different outlier detection methods using the same dataset.

2.1. *k*-Nearest Neighbours (*k*-NN)

The *k*-NN algorithm [21,22] measures the distance between a datapoint and its *k* nearest neighbours and uses it as a proxy to quantify the data density in that area. After the *k* nearest neighbours of a datapoints have been determined, then an outlier score is produced by combining these distances, usually by applying some aggregation function: the maximum, the mean or the median of the *k* calculated distances, one for each of its neighbours, can all be considered as a measure of outlierness for a given observation.

2.2. Local Outlier Factor (LOF)

The LOF algorithm [23] is based on a notion of the local density of a datapoint, according to which a datapoint's locality is determined by its surrounding neighbourhood, while its density is approximated through its distance from its neighbours. By comparing the local density of a datapoint to the local densities of its neighbouring ones, outliers can be considered as points that have a significantly lower density than their neighbours. LOF performs best when the density of the data is not uniform throughout the dataset.

2.3. Cluster-Based Local Outlier Factor (CBLOF)

Proposed in [24], the CBLOF methodology utilises clustering to determine the high-density areas in the dataset. First, k-means is employed to split the data into clusters and subsequently a heuristic approach is applied to further categorise the resulting clusters into two classes: small and large. The outlier confidence score for an observation is calculated by combining the distance of each observation to the centroid of its cluster, with the number of observations belonging to its cluster. In the case where the observation belongs to a small cluster, then the distance to the nearest large cluster is used instead. One of the disadvantages of CBLOF is that by relying on k-means for clustering, its performance is highly dependant on the initial selection of the hyperparameter k.

2.4. Histogram-Based Outlier Score (HBOS)

Assuming feature independence, the HBOS algorithm [25] computes the degree of outlierness by modelling univariate feature densities through histograms with of dynamic or fixed bin width. Subsequently, the produced histograms are combined to calculate an outlier confidence score for each observation, taking into account all the features. Its strong assumptions regarding feature independence, make HBOS computationally cheap compared to multivariate methods but inferior in terms of predictive power.

2.5. One-Class Support Vector Machines (OCSVM)

OCSVM [26] are a special case of the normal SVM algorithm [27] in that they are trained using only the normal, non-outlier observations. As a result, OCSVM learn the boundaries of the distribution of such datapoints and are therefore able to classify any points that fall outside these decision boundaries as outliers. As with any SVM, the choice of the kernel hyperparameter plays a crucial role, with the RBF kernel being generally the most popular one [27].

2.6. Minimum Covariance Determinant (MCD)

MCD is a method for estimating the mean and covariance matrix in a way that tries to minimize the influence of outliers. The idea is to estimate these parameters from a subset of the data that has been carefully chosen to not contain outlier observations, the subset of the data that is most tightly distributed. It can be computed efficiently with the FAST-MCD algorithm of Rousseeuw and Van Driessen [28]. Hardin and Rocke [29] developed a new method for identifying outliers in a one-cluster setting using the MCD, subsequently extending it to the multiple cluster case resulting in a robust outlier detection method [30]. MCD works best on Gaussian-distributed data but could still be relevant on data drawn from a unimodal, symmetric distribution; however, it is not meant to be used with multi-modal data.

2.7. Principal Component Analysis (PCA)

PCA is a linear dimensionality reduction technique that finds the directions of highest variance in the data by decomposing the data covariance matrix in order to find orthogonal eigenvectors [31]. The higher the eigenvalue corresponding to a particular eigenvector, the higher the variance in the direction defined by that eigenvector. Considering the k eigenvectors with the highest eigenvalues, often called the principal eigenvectors, a lower dimensional hyperplane can be defined, capturing most of the variance in the data. For each datapoint, an outlier score can be produced by computing its reconstruction error with respect to the principal eigenvectors as the sum of its projected distances from the principal eigenvectors [32].

2.8. Angle-Based Outlier Detection (ABOD)

Unlike distance-based approaches, the ABOD algorithm [33] determines outlier observations using angles. More specifically, for each observation, the variance in the angles between the difference vectors of an observation to other points is calculated. Subsequently,

a datapoint is considered an outlier if the majority of other datapoints are located in similar directions, meaning lower variance of angles, corresponding to points at the border of a cluster. On the other hand, a lower variance of angles usually translates to many other datapoints are lying in varying directions, corresponding to the inner points of the cluster. The ABOD algorithm, suffering less from the effects of the “curse of dimensionality”, is more suitable and computationally much less expensive for high-dimensional data compared to distance-based approaches.

2.9. Feature Bagging (FB)

A feature bagging outlier detector [34] is a meta-detector consisting of a number of base detectors, such as the kNN, LOF, ABOD outlier detection methods. These base detectors are fit using a variety of random sub-samples of the dataset, while also utilising combination methods, such as averaging, to produce a final robust detector of increased predictive power and less prone to overfitting. The size of the random sub-sample is always the same; however, the size of the sampled features varies for each trained base detector, reducing the correlation among them and increasing their diversity.

2.10. Isolation Forest (iForest)

The Isolation Forest algorithm [35] isolates outliers by taking advantage of two of their main properties: firstly, by definition, they are the minority in a dataset, accounting for far fewer instances and secondly, they are quite different to normal instances, having attributes that deviate from the mainstream distribution. The feature space is partitioned in a recursive fashion, a feature is randomly selected and subsequently a split value between the maximum and minimum values of the selected feature is also randomly selected. Bearing in mind the two aforementioned properties, such a partitioning would result in outlier observations most likely existing closer to the root of the tree, having a shorter average path length, needing fewer splits to be reached. One of the main advantages of isolation forest is its ability to take advantage of sampling techniques, thus being a very fast algorithm with low computational demands [36].

2.11. Ensemble Voting Methods

Due to its unsupervised nature, outlier detection methods often suffer from instability [37]. A solution is to combine the outputs of various detectors in order to produce a meta-detector of increased diversity, robustness and ideally predictive power [37,38]. Common ensemble voting methods include, but are not limited to:

- Average: The outlier scores of all the detectors are averaged to produce a final score;
- Maximisation: The maximum score across all detectors is considered the final score;
- Average of Maximum (AOM): Sub-detectors are divided into subgroups and the maximum score for each subgroup is computed. The final score is the average of all subgroup maximum scores
- Maximum of Average (MOA): Sub-detectors are divided into subgroups and the average score for each subgroup is computed. The final score is the maximum of all subgroup average scores

3. Unsupervised Feature Selection

In supervised feature selection, a subset of features is picked out based on its potency of discriminating samples that belong to different classes. In such a setting, a feature is considered to be relevant when predicting a class if a strong correlation exists between them. On the other hand, in the case of unsupervised learning, defining relevancy is not as easy, due to the lack of labels that would point towards a search direction for relevant features. In this case, the concept of the ‘target’ or ‘label’ is usually related to the intrinsic structure and properties of the data. That said, in both cases, the goal of any feature selection process is to pick out subsets of features capable of categorising instances into separable subsets according to different definitions of the separability [39].

It has been shown [40] that selecting subsets of features, according to some similarity or correlation criteria, can boost unsupervised learning algorithms analogously to how supervised learning algorithms are improved. Several unsupervised learning feature selection methods have been proposed for different kinds of data. In this section, the most important methods algorithms, able to handle any type of data, are presented and briefly analysed.

3.1. Spectral Feature Selection (SPEC)

SPEC [39] is a unified framework, based on spectral graph theory, that enables the joint study of both supervised and unsupervised feature selection. The main idea of SPEC is to represent datapoints as vertices of a graph and assign weights to edges of the graph corresponding to the distance or measure of similarity between points—a commonly used similarity measure being the RBF kernel function. The SPEC framework selects features, by studying the degrees of similarities among samples. Under this framework, features consistent with the graph structure are assigned similar values to datapoints that are close to each other in the graph. Such features would be of increased relevance since they behave similarly in each similar group of samples (i.e., clusters).

Normalised Cut: The Normalised Cut algorithm attempts to find minimal cut in the graph [41]. Subsequently, the top k features separated by this cut are selected, since they are considered to best explain the dataset [39].

Arbitrary Clustering (Generic SPEC): The arbitrary clustering algorithm finds the top k features, resulting in the best graph separability. In this case, this is achieved by calculating the trivial eigenvector of the Laplacian matrix of the graph and using it to normalise scores. This type of normalisation has been found to produce better feature selection results [39].

Fixed Clustering: The fixed clustering algorithm, under the assumption that the data can be split into a user-predefined number of clusters, determines the features that provide the best separability. This is achieved by computing the eigenvectors corresponding to the smallest eigenvalues of the Laplacian matrix of the graph (other than the trivial one), and subsequently calculating the respective cosine similarities to detect the most expressive features [39].

3.2. Unsupervised Lasso

Unsupervised Lasso feature selection [42], based on L1-norm regularisation, performs clustering using an adaptively chosen subset of the features and simultaneously calculates feature importances with respect to it. At the beginning, all features are considered to be of the same importance or weight. Then, at each iteration of the algorithm, the importance or weight of each feature is adjusted according to the clustering objective and penalised according to the L1-penalty. After a given number of iterations, the top k features, in terms of their weights can be selected.

3.3. Weighted K-Means

Weighted k-means [43] is a variation of the k-means clustering algorithm that on top of grouping data into clusters, measures the importance of each feature. More specifically, the addition of an extra step to the k-means algorithmic procedure is proposed, such that the feature weights are iteratively updated, based on their importance in terms of clustering the data. After a set number of iterations is performed, only the k most important features are retained.

4. Proposed Methodology

The aim of this study is to make a contribution to the field of unsupervised outlier detection by proposing a novel meta-learning algorithm for outlier identification. More specifically, drawing inspiration from principles of ensemble learning and unsupervised feature selection, we propose a methodology consisting of three main distinct steps.

The motivation behind the proposed methodology was to obtain the best of many worlds and increase diversity by combining the predictive power of the most powerful outlier detection methods. However, as unsupervised detection algorithms are known to learn quite different decision boundaries depending on the assumptions they use to define ‘outlierness’, the use of a feature selection algorithm is promoted as an intermediate step to ensure that the poorly performing ones do not heavily interfere with the quality of the produced, combined predictions. Lastly, the fully unsupervised nature of the proposed methodology should be highlighted, as in most real-world settings, the number and the pattern of outliers are not known in advance.

4.1. Data Preparation and Training

In terms of data, seventeen (17) datasets of different meta-data, such as number of rows, columns and outliers, coming from a diverse pool of domains were used, as can be seen from Tables 1 and 2.

As a first step, all datasets were split, using 60% for training and the remaining 40% for evaluation purposes. Subsequently, since the scale of the different features/columns in the datasets varied considerably, a standardisation procedure followed: For each dataset, for all the values of each feature, the mean of the feature is subtracted, and the result is divided by the standard deviation of the feature.

The purpose of this standardisation process is for any outlier detection algorithm trained (and later evaluated) to focus on the qualitative measures that the features within the same dataset have to offer rather than its absolute values. That said, it should be noted that while some methods, such as k-NN [21], that rely on distances between datapoints to recognise outliers can heavily suffer from non-standardised features, some others, such as iForest [35], are scale-invariant. In the current training and testing procedures, standardised datasets were supplied to all algorithms.

Table 1. Friedman comparison for ROC.

Rank	Algorithm
4.8823	Proposed methodology
5.0588	Average of maximum
5.1470	Cluster-based local outlier factor
5.3529	Isolation forest
6.0882	K Nearest neighbours (KNN)
6.1176	Histogram-base outlier detection (HBOS)
6.1764	Minimum covariance determinant (MCD)
6.4705	One-class SVM (OCSVM)
6.9411	Principal component analysis (PCA)
8.4705	Feature bagging
8.5882	Angle-based outlier detector (ABOD)
8.7058	Local outlier factor (LOF)

Table 2. Friedman comparison for Precision.

Rank	Algorithm
5.44117	Proposed methodology
5.58823	Cluster-based local outlier factor
5.61764	Histogram-base outlier detection (HBOS)
5.73529	Average of maximum
5.82352	One-class SVM (OCSVM)
5.94117	Isolation forest
6.29411	Principal component analysis (PCA)
6.58823	K nearest neighbours (KNN)
7.17647	Feature bagging
7.82352	Local outlier factor (LOF)
7.97058	Angle-based outlier detector (ABOD)
8	Minimum covariance determinant (MCD)

After the datasets were standardised, numerous broadly-used unsupervised outlier detection estimators were trained using each dataset. In this case, these estimators include the k-NN [21], LOF [23], CBLOF [24], HBOS [25], OCSVM [26], MCD [28], PCA [32], ABOD [33] and iForest estimators [35] as well as an ensemble [34] of LOF estimators.

4.2. Unsupervised Feature Selection and Scoring

Each outlier detection algorithm has its shortcomings, which mainly arise from the assumptions it is based on. Different families of algorithms such as distance-based, angle-based, density-based and more, given the same dataset to learn from, can produce wildly different decision boundaries and vary considerably in terms of predictions for certain datapoints and therefore in terms of performance.

In a supervised scenario, one could, in theory at least, exhaustively or in some other, smarter way find out which algorithms or families perform better for their needs. In an unsupervised setting, however, it is impossible to know which algorithms or families of algorithms work best for a given dataset due to the lack of ground truth; however, still one or more algorithms have to be picked. Moreover, in the case of building an ensemble, if it includes poorly performing algorithms—for a dataset—it is very likely that the overall performance will be driven down by such algorithms.

The proposed methodology aims to firstly identify the most relevant outlier detection estimators for any given dataset—through unsupervised feature selection—and subsequently combine them using ensemble voting. More specifically, given a dataset and the outlier confidence scores produced for each datapoint in the dataset by the trained estimators detailed in Section 4.1 (a total of 10 estimators), the unsupervised spectral feature method ‘Fixed SPEC’ [39] is applied to rank the estimators. This process essentially treats the outlier scores of the datapoints in a dataset as features of a certain relevance with respect to a clustering. In order to rank features in an unsupervised manner, Fixed SPEC exploits the Laplacian Score (LS) [44] of a feature, which reflects the feature’s locality preserving power. Based on the assumption that two datapoints are likely related if they are close to each other and that ‘good’ features (in our case ‘good’ outlier detection algorithms) are the ones that respect this assumption, LS puts more emphasis on the local structure of the data space rather than the global structure. As a result, the importance of a feature can be measured as the degree to which it preserves this assumption.

The hypothesis made in this study is that an unsupervised feature selection process will, for a given dataset, rank its most relevant features (in this case, unsupervised outlier detection algorithms) higher and down-rank the less informative ones. More specifically, the assumption that datapoints (vectors of scores) can be separated into a predefined number of two clusters (informative vs. non-informative) was made.

Implementation-wise, the algorithm is given an $N \times S$ table of outlier scores per datapoint per algorithm, where N corresponds to the number of datapoints in the given

dataset and S to the number of outlier detection methods to choose from. A graph is constructed from this table; its vertices corresponding to the datapoints, while its edges to weights calculated as RBF-distances between datapoints. Subsequently, the eigenvectors corresponding to the K smallest eigenvalues of the Laplacian matrix of the graph (except the trivial one) are calculated, and the cosine similarity between them and the feature vectors are computed. Based on this score, the unsupervised selection algorithm selects the K features as the most informative ones. In our case, K is equal to 6 as the proposed methodology only keeps the 60% of the estimators (a total of 6 scores) that are weighted the highest according to the described procedure.

Finally, after the most relevant estimators have been decided, a voting ensemble is employed to combine them in order to produce a final outlier confidence estimation for a new datapoint. More specifically, the 'average of maximum' voting ensemble is employed, splitting the estimators into subgroups. Each estimator makes a prediction about a datapoint and subsequently the maximum score for each subgroup is computed. The average of all subgroup maximum estimations is considered to be the final score. This voting method is more sophisticated compared to a simple averaging or maximisation technique. It also assumes that as long as mostly powerful classifiers for a given dataset are left after feature selection, taking the mean of the most confident classifiers (maximum scores) in each subgroup, will result in more accurate results.

A step-by-step procedure of the described methodology is outlined in Algorithm 1.

Algorithm 1 Proposed Methodology Steps.

1. For each dataset, its data are split into two sub-sets: 60% for training and 40% for testing.
 2. The training and testing datasets are standardised independently.
 3. Ten base outlier detectors are trained using the standardised training dataset and scores are produced for each datapoint.
 4. Spectral unsupervised feature selection ('Fixed SPEC' with 2 clusters) is applied to the confidence scores, keeping the most relevant base outlier detectors (60%).
 5. For each datapoint in the testing dataset, confidence scores are generated (one per detector), using the most relevant base outlier detectors for this dataset, as calculated in step 4.
 6. The obtained scores are then combined using the 'average of maximum' ensemble voting to produce a final prediction for that datapoint.
-

5. Experiments are Results

The performance of the proposed methodology was evaluated empirically: Three experiments, each designed to evaluate a different part of the proposed algorithm and provide insights into alternatives, were conducted to compare the proposed method against a number of existing approaches in terms of performance, from three different perspectives.

More specifically, the proposed method was benchmarked and compared against ten (10) broadly-used techniques for outlier detection plus a 'naive' ensemble voting classifier (not using unsupervised feature selection).

Following on from the experimentation methodology proposed in [13], seventeen (17) datasets of different metadata, such as number of rows, columns and outliers, coming from a diverse pool of domains were used. For each dataset used, 60% of the dataset was used for training, while the remaining 40% for performance evaluation purposes.

In terms of performance evaluation measurement, two widely accepted metrics were employed [13]: the area under the receiver operating characteristic curve (ROC) and precision [4,45]. On top of these metrics, the non-parametric Friedman test was applied, in order for rankings to be assigned to each algorithm and therefore for more solid comparisons to be made.

To make sure consistent performance results are obtained, each experiment was independently repeated five times. The mean result of these five runs is regarded as the final result for that particular experiment. To conduct these experiments, the well-known Python toolkit for outlier, PYOD, was used [13].

5.1. Experiment 1: Measuring the Performance of the Proposed Methodology

The first experiment is designed to compare the overall performance of the proposed method against a number of existing approaches, ultimately proving its state-of-the-art performance. The performance results for each method using each of the datasets are presented in Tables A1 and A2 for the ROC and precision metrics, respectively.

Furthermore, by applying Friedman’s non-parametric statistical test on these results, the rankings for each method, seen in Table 1 (ROC) and Table 2 (precision), were obtained. The proposed methodology was ranked highest, achieving a better overall performance compared to all other methods.

It is worth noting that the simple/naive ensemble voting method, utilising the ‘average of maximum’ ensemble voting technique—the same averaging technique of the proposed methodology, bar the feature selection procedure—achieved a worse overall score. This highlights the importance of unsupervised feature selection and further proves and justifies its adoption.

Finally, the percentages that each outlier detection method was chosen with, by the unsupervised feature selection procedure, are presented in Table 3.

Table 3. Percentage that each outlier detection method was selected with by the unsupervised feature selection algorithm.

Percentage	Algorithm
0.80	Cluster-based local outlier factor
0.76	Isolation forest
0.73	K Nearest neighbours (KNN)
0.71	Histogram-base outlier detection (HBOS)
0.65	Minimum covariance determinant (MCD)
0.59	One-class SVM (OCSVM)
0.53	Principal component analysis (PCA)
0.45	Feature bagging
0.40	Angle-based outlier detector (ABOD)
0.39	Local outlier factor (LOF)

5.2. Experiment 2: Measuring the Impact of the Proposed Feature Selection Method

A fundamental component of the proposed method is the use of the spectral unsupervised, ‘Fixed SPEC’ with two clusters, feature selection process. That said, there are a number of alternative unsupervised feature selection algorithms that could be employed instead.

To this end, a second experiment was designed to compare the proposed feature selection process against alternatives. Keeping the rest of the proposed methodology fixed, a different feature selection was applied each time. More specifically, using the raw performance results in Table A1 (ROC) and Table A2 (precision), the following alternative feature selection techniques were considered: fixed clustering, arbitrary clustering, normalised cut and weighted k-mean.

Applying Friedman’s non-parametric statistical test, the rankings for each feature selection technique, depicted on Table 4 (ROC) and Table 5 (precision), were produced. According to the results, the proposed methodology works best, both in terms of ROC and precision, when employing the proposed spectral unsupervised feature selection technique.

Table 4. Friedman results of the proposed methodology using different feature selection techniques for ROC.

Rank	Algorithm
3.05882	Fixed clustering
3.32352	Normalised cut
3.5	Lasso
3.88235	Weighted k-means
4.11764	Arbitrary clustering

Table 5. Friedman results of the proposed methodology using different feature selection techniques for precision.

Rank	Algorithm
3.0294	Fixed clustering
3.1764	Normalised cut
3.6764	Lasso
3.8235	Weighted k-means
4	Arbitrary clustering

5.3. Experiment 3: Measuring the Impact of the Proposed Ensemble Voting Method

The third and final component of the proposed methodology is the use of the ‘average of maximum’ ensemble voting technique to aggregate the different scores, produced by the best—according to the feature selection process—methods, into a single prediction. In this experiment, alternative aggregation techniques, namely the simple average, the maximum and the maximum of average, were considered and compared against the proposed one. The results of this experiment with respect to both metrics, ROC and precision, are shown in Tables 6 and 7, respectively, illustrating the superiority of the proposed aggregation technique.

Table 6. Friedman results of the proposed methodology using different aggregation techniques for ROC.

Rank	Algorithm
1.82352	Average of maximum
2.52941	Maximisation
2.64705	Maximum of average
3	Average

Table 7. Friedman results of the proposed methodology using different aggregation techniques for precision.

Rank	Algorithm
2.17647	Average of maximum
2.35294	Maximum of average
2.41176	Maximisation
3.05882	Average

6. Conclusions

In this study, a new meta-learning algorithm for unsupervised outlier detection was presented and empirically validated.

The goal of the contributed methodology is not only to harness the benefits of the existing most powerful outlier detection methods by combining them but also, and most importantly, to mitigate or completely avoid their shortcomings. The former is achieved through ensemble voting, a well-established, reliable and robust solution to several problems of machine learning; the latter through the adoption of the fully unsupervised spectral feature selection algorithm. For a given problem or dataset, firstly, the most informative outlier detection methods are identified and the least explanatory ones are filtered out. Subsequently, they are combined to form a powerful state-of-the-art estimator.

The proposed approach was empirically evaluated through three experiments: in each of these experiments, a diverse pool of seventeen datasets of varying size and number of outliers were used. Results demonstrated the overall superiority of the proposed method against ten broadly-used techniques for outlier detection, as it topped the non-parametric Friedman test rankings, in terms of both the area under the receiver operating characteristic (ROC) curve and precision. A future direction would be to experiment with the proposed methodology in very high dimensional datasets as well as in time series data, assess its accuracy, speed and scalability and make modifications where necessary.

Author Contributions: Conceptualization, V.P.; methodology, S.K. and P.L.; validation, V.P. and P.L.; formal analysis, V.P. and P.L.; investigation, V.P. and P.L.; resources, P.L.; writing—original draft preparation, V.P.; writing—review and editing, S.K.; visualization, V.P. and P.L.; supervision, S.K.; project administration, S.K.; funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

Appendix A

All datasets, results, and the necessary code used in this study to parse the data, develop, train and evaluate the models can be found at the following public GitHub repository: <https://github.com/ML-Upatras/unsupervised-outlier-detection> (accessed on 9 September 2021).

Table A1. Comparison of ROC results (average of 5 trials).

Data	#Rows	#Dims	Outlier Perc	ABOD	CBLOF	FB	HBOS	IForest	KNN	LOF	MCD	OCSVM	PCA	AOM	PROPOSED
arrhythmia	452	274	14.6018	0.75478	0.76042	0.76624	0.83304	0.83536	0.76834	0.767	0.786	0.777	0.7779	0.79932	0.80636
cardio	1831	21	9.6122	0.55894	0.82858	0.51898	0.84824	0.94268	0.70402	0.50432	0.87412	0.94658	0.96004	0.9136	0.91544
glass	214	9	4.2056	0.74278	0.86444	0.7806	0.6897	0.7364	0.81004	0.82102	0.73902	0.62024	0.6295	0.74618	0.71386
ionosphere	351	33	35.8974	0.91228	0.902	0.89394	0.54206	0.84296	0.92442	0.89956	0.94716	0.85314	0.80264	0.8687	0.86776
letter	1600	32	6.25	0.86142	0.75734	0.86804	0.58572	0.59716	0.85534	0.85218	0.77182	0.5809	0.49494	0.79738	0.78382
lympho	148	18	4.0541	0.9043	0.96702	0.96314	0.99784	0.9913	0.96264	0.96092	0.9045	0.96264	0.98046	0.97364	0.98664
mnist	7603	100	9.2069	0.78148	0.84176	0.7292	0.56444	0.78834	0.84392	0.72004	0.84854	0.83976	0.83872	0.8291	0.81276
musk	3062	166	3.1679	0.10668	1	0.53836	0.99994	0.99994	0.7665	0.54098	0.99974	1	0.99998	0.99972	0.99522
optdigits	5216	64	2.8758	0.4515	0.76966	0.4691	0.87552	0.61316	0.37698	0.46556	0.3584	0.49338	0.5019	0.68606	0.70976
pendigits	6870	16	2.2707	0.69892	0.96238	0.48352	0.92942	0.94228	0.75506	0.49278	0.82788	0.92926	0.93426	0.92598	0.9094
pima	768	8	34.8958	0.67502	0.6728	0.63302	0.70656	0.65712	0.7175	0.64198	0.6979	0.63014	0.6485	0.68686	0.69372
satellite	6435	36	31.6395	0.56742	0.72168	0.55492	0.74798	0.69672	0.67996	0.55398	0.79804	0.64634	0.58926	0.72258	0.7279
satimage-2	5803	36	1.2235	0.855	0.99816	0.49052	0.9723	0.9904	0.95222	0.48928	0.99458	0.99548	0.97168	0.99648	0.9914
shuttle	49097	9	7.1511	0.62174	0.62754	0.5201	0.98438	0.99692	0.64868	0.53478	0.98996	0.99154	0.98958	0.99576	0.99396
vertebral	240	6	12.5	0.33178	0.38306	0.35728	0.31012	0.41318	0.35792	0.36848	0.36968	0.45958	0.3993	0.34466	0.3712
vowels	1456	12	3.4341	0.95528	0.91872	0.93292	0.69868	0.72978	0.96316	0.9298	0.69494	0.77312	0.60994	0.9016	0.93778
wbc	378	30	5.5556	0.91486	0.92016	0.94198	0.95988	0.94378	0.94226	0.93542	0.90236	0.93878	0.93096	0.94576	0.94966

Table A2. Comparison of precision results (average of 5 trials).

Data	#Rows	#Dims	Outlier Perc	ABOD	CBLOF	FB	HBOS	IForest	KNN	LOF	MCD	OCSVM	PCA	AOM	PROPOSED
arrhythmia	452	274	14.6018	0.34726	0.42538	0.4468	0.53914	0.53226	0.4468	0.43942	0.3835	0.4537	0.4468	0.48252	0.51108
cardio	1831	21	9.6122	0.19314	0.49558	0.1307	0.46098	0.51398	0.30234	0.1367	0.42454	0.55406	0.65912	0.41988	0.44812
glass	214	9	4.2056	0.23	0.19	0.27	0.04	0.19	0.19	0.27	0	0.19	0.19	0.19	0.19
ionosphere	351	33	35.8974	0.82946	0.8031	0.7551	0.34232	0.66184	0.86012	0.7551	0.86674	0.72184	0.59522	0.72576	0.75152
letter	1600	32	6.25	0.29908	0.19576	0.36988	0.1053	0.07544	0.30538	0.32488	0.11414	0.11834	0.05386	0.22008	0.21428
lympho	148	18	4.0541	0.34	0.56	0.56	0.88	0.76	0.56	0.56	0.56	0.56	0.68	0.56	0.74
mnist	7603	100	9.2069	0.35008	0.40298	0.36356	0.1167	0.28742	0.41504	0.34344	0.35448	0.3772	0.3724	0.37578	0.35494
musk	3062	166	3.1679	0.04908	1	0.19502	0.97548	0.92588	0.2618	0.16998	0.95502	1	0.99	0.96214	0.79334
optdigits	5216	64	2.8758	0.0236	0	0.03268	0.23424	0.00966	0	0.02904	0	0	0	0.00322	0.01612
pendigits	6870	16	2.2707	0.0596	0.32224	0.06194	0.2796	0.31412	0.07138	0.0585	0.06952	0.30956	0.33176	0.26516	0.2249
pima	768	8	34.8958	0.50892	0.47066	0.435	0.5085	0.48504	0.51154	0.45544	0.4928	0.46252	0.50084	0.48514	0.51868
satellite	6435	36	31.6395	0.39006	0.55224	0.40148	0.55862	0.58046	0.49674	0.40076	0.6786	0.5278	0.4679	0.59938	0.59226
satimage-2	5803	36	1.2235	0.24624	0.88134	0.06712	0.6289	0.85068	0.36758	0.07246	0.5631	0.90354	0.82048	0.73912	0.52934
shuttle	49097	9	7.1511	0.19844	0.23528	0.11194	0.97682	0.9357	0.22658	0.15128	0.74876	0.95396	0.94942	0.94168	0.91272
vertebral	240	6	12.5	0.01538	0.01538	0.03076	0.01538	0.03076	0	0.03076	0	0.01538	0	0.01538	0.01538
vowels	1456	12	3.4341	0.4888	0.35528	0.3205	0.14918	0.19304	0.52408	0.35584	0.01112	0.3174	0.13548	0.29576	0.39978
wbc	378	30	5.5556	0.255	0.535	0.5	0.695	0.475	0.475	0.415	0.39	0.475	0.535	0.535	0.495

References

1. Hodge, V.; Austin, J. A survey of outlier detection methodologies. *Artif. Intell. Rev.* **2004**, *22*, 85–126. [[CrossRef](#)]
2. Domingues, R.; Filippone, M.; Michiardi, P.; Zouaoui, J. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognit.* **2018**, *74*, 406–421. [[CrossRef](#)]
3. Boukerche, A.; Zheng, L.; Alfandi, O. Outlier Detection: Methods, Models, and Classification. *ACM Comput. Surv. CSUR* **2020**, *53*, 1–37. [[CrossRef](#)]
4. Zimek, A.; Filzmoser, P. There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1280. [[CrossRef](#)]
5. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv. CSUR* **2009**, *41*, 1–58. [[CrossRef](#)]
6. Wang, H.; Bah, M.J.; Hammad, M. Progress in outlier detection techniques: A survey. *IEEE Access* **2019**, *7*, 107964–108000. [[CrossRef](#)]
7. Aggarwal, C.C. An introduction to outlier analysis. In *Outlier Analysis*; Springer: Berlin, Germany, 2017; pp. 1–34.
8. Aggarwal, C.C.; Philip, S.Y. An effective and efficient algorithm for high-dimensional outlier detection. *VLDB J.* **2005**, *14*, 211–221. [[CrossRef](#)]
9. Alghushairy, O.; Alsini, R.; Soule, T.; Ma, X. A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams. *Big Data Cogn. Comput.* **2021**, *5*, 1.
10. Dong, X.L.; Srivastava, D. Big data integration. In Proceedings of the 2013 IEEE 29th International Conference on Data Engineering (ICDE), Brisbane, Australia, 8–12 April 2013; pp. 1245–1248.
11. Liu, H.; Shah, S.; Jiang, W. On-line outlier detection and data cleaning. *Comput. Chem. Eng.* **2004**, *28*, 1635–1647. [[CrossRef](#)]
12. Meng, F.; Yuan, G.; Lv, S.; Wang, Z.; Xia, S. An overview on trajectory outlier detection. *Artif. Intell. Rev.* **2019**, *52*, 2437–2456. [[CrossRef](#)]
13. Zhao, Y.; Nasrullah, Z.; Li, Z. PyOD: A Python Toolbox for Scalable Outlier Detection. *J. Mach. Learn. Res.* **2019**, *20*, 1–7.
14. Munoz, A.; Muruzábal, J. Self-organizing maps for outlier detection. *Neurocomputing* **1998**, *18*, 33–60. [[CrossRef](#)]
15. Kriegel, H.P.; Kröger, P.; Schubert, E.; Zimek, A. Outlier detection in axis-parallel subspaces of high dimensional data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin, Germany, 2009; pp. 831–838.
16. Almardeny, Y.; Boujnah, N.; Cleary, F. A Novel Outlier Detection Method for Multivariate Data. *IEEE Trans. Knowl. Data Eng.* **2020**. [[CrossRef](#)]
17. Li, Z.; Zhao, Y.; Botta, N.; Ionescu, C.; Hu, X. COPOD: Copula-based outlier detection. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; pp. 1118–1123.
18. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. In Proceedings of the 2nd International Conference on Learning Representations, ICLR, Banff, AB, Canada, 14–16 April 2014.
19. Burgess, C.P.; Higgins, I.; Pal, A.; Matthey, L.; Watters, N.; Desjardins, G.; Lerchner, A. Understanding disentangling in β -VAE. *arXiv* **2018**, arXiv:1804.03599.
20. Zhao, Y.; Nasrullah, Z.; Hryniewicki, M.K.; Li, Z. LSCP: Locally selective combination in parallel outlier ensembles. In Proceedings of the 2019 SIAM International Conference on Data Mining, Calgary, AB, Canada, 2–4 May 2019; pp. 585–593.
21. Angiulli, F.; Pizzuti, C. Fast outlier detection in high dimensional spaces. In *European Conference on Principles of Data Mining and Knowledge Discovery*; Springer: Berlin, Germany, 2002; pp. 15–27.
22. Ramaswamy, S.; Rastogi, R.; Shim, K. Efficient algorithms for mining outliers from large data sets. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 427–438.
23. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
24. He, Z.; Xu, X.; Deng, S. Discovering cluster-based local outliers. *Pattern Recognit. Lett.* **2003**, *24*, 1641–1650. [[CrossRef](#)]
25. Goldstein, M.; Dengel, A. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. In Proceedings of the KI-2012 Poster and Demo Track German Conference on Artificial Intelligence (Künstliche Intelligenz), Saarbrücken, Germany, 24–27 September 2012; pp. 59–63.
26. Schölkopf, B.; Platt, J.C.; Shawe-Taylor, J.; Smola, A.J.; Williamson, R.C. Estimating the support of a high-dimensional distribution. *Neural Comput.* **2001**, *13*, 1443–1471. [[CrossRef](#)]
27. Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*; Cambridge University Press: Cambridge, MA, USA, 2000.
28. Rousseeuw, P.J.; Driessen, K.V. A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **1999**, *41*, 212–223. [[CrossRef](#)]
29. Hardin, J.; Rocke, D.M. The distribution of robust distances. *J. Comput. Graph. Stat.* **2005**, *14*, 928–946. [[CrossRef](#)]
30. Hardin, J.; Rocke, D.M. Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator. *Comput. Stat. Data Anal.* **2004**, *44*, 625–638. [[CrossRef](#)]
31. Aggarwal, C.C. Outlier analysis. In *Data Mining*; Springer: Berlin, Germany, 2015; pp. 237–263.
32. Shyu, M.L.; Chen, S.C.; Sarinnapakorn, K.; Chang, L. A novel anomaly detection scheme based on principal component classifier. In Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in Conjunction with the Third IEEE International Conference On Data Mining (ICDM'03), Melbourne, FL, USA, 19–22 November 2003; pp. 172–179.

33. Kriegel, H.P.; Schubert, M.; Zimek, A. Angle-based outlier detection in high-dimensional data. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 444–452.
34. Lazarevic, A.; Kumar, V. Feature bagging for outlier detection. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Chicago, IL, USA, 21–24 August 2005; pp. 157–166.
35. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422.
36. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data TKDD* **2012**, *6*, 1–39. [[CrossRef](#)]
37. Zimek, A.; Campello, R.J.; Sander, J. Ensembles for unsupervised outlier detection: challenges and research questions a position paper. *ACM Sigkdd Explor. Newsl.* **2014**, *15*, 11–22. [[CrossRef](#)]
38. Dietterich, T.G. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*; Springer: Berlin, Germany, 2000; pp. 1–15.
39. Zhao, Z.; Liu, H. Spectral feature selection for supervised and unsupervised learning. In Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA, 20–24 June 2007; pp. 1151–1157.
40. Alelyani, S.; Tang, J.; Liu, H. Feature selection for clustering: A review. *Data Clust. Algorithms Appl.* **2013**, *29*, 144.
41. Shi, J.; Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
42. Witten, D.M.; Tibshirani, R. A framework for feature selection in clustering. *J. Am. Stat. Assoc.* **2010**, *105*, 713–726. [[CrossRef](#)] [[PubMed](#)]
43. Huang, J.Z.; Ng, M.K.; Rong, H.; Li, Z. Automated variable weighting in k-means type clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 657–668. [[CrossRef](#)]
44. He, X.; Cai, D.; Niyogi, P. Laplacian score for feature selection. *Adv. Neural Inf. Process. Syst.* **2005**, *18*, 507–514
45. Marques, H.O.; Campello, R.J.; Sander, J.; Zimek, A. Internal evaluation of unsupervised outlier detection. *ACM Trans. Knowl. Discov. Data TKDD* **2020**, *14*, 1–42. [[CrossRef](#)]