Benchmarking conventional outlier detection methods*

Elena Tiukhova[1][0000−0002−5050−9417], Manon Reusens[1][0000−0002−0275−0679], Bart Baesens[1,2,3][0000−0002−5831−5668], and Monique Snoeck[1][0000−0002−3824−3214]

[1] Research Center for Information Systems Engineering (LIRIS), KU Leuven, Naamsestraat 69, Leuven 3000, Belgium
monique.snoeck@kuleuven.be
https://feb.kuleuven.be/research/decision-sciences-and-information-management/liris/liris
[2] Department of Decision Analytics and Risk, University of Southampton, United Kingdom
[3] Southampton Business School, University of Southampton, Highfield, Southampton SO17 1BJ, UK

Abstract. Nowadays, businesses in many industries face an increasing flow of data and information. Data are at the core of the decision-making process, hence it is vital to ensure that the data are of high quality and no noise is present. Outlier detection methods are aimed to find unusual patterns in data and find their applications in many practical domains. These methods employ different techniques, ranging from pure statistical tools to deep learning models that have gained popularity in recent years. Moreover, one of the most popular outlier detection techniques are machine learning models. They have several characteristics which affect the potential of their usefulness in real-life scenarios. The goal of this paper is to add to the existing body of research on outlier detection by comparing the isolation forest, DBSCAN and LOF techniques. Thus, we investigate the research question: which ones of these outlier detection models perform best in practical business applications. To this end, three models are built on 12 datasets and compared using 5 performance metrics. The final comparison of the models is based on the McNemar's test, as well as on ranks per performance measure and on average. Three main conclusions can be made from the benchmarking study. First, the models considered in this research disagree differently, i.e. their type I and type II errors are not similar. Second, considering the time, AUPRC and sensitivity metrics, the iForest model is ranked the highest. Hence, the iForest model is the best in the cases when time performance is a key consideration as well as when the opportunity costs of not detecting an outlier are high. Third, the DBSCAN model obtains the highest ranking along the F1 score and precision dimensions. That allows us to conclude that if raising many false alarms is not an important concern, the DBSCAN model is the best to employ.

* Supported by the ING Group

Keywords: Outlier detection · iForest · Local Outlier Factor · DBSCAN

1 Introduction

It is difficult to imagine the modern world without data. Especially since the outbreak of COVID-19, the role of digital technologies has become even more crucial, and the amount of data created every minute is enormous. In 2021, the total amount of data consumed globally was 79 zettabytes, and this number is projected to grow to 180 zettabytes by 2025 [9]. Businesses try to make use of these data by building up machine learning models that facilitate decision-making processes.

For machine learning models to be unbiased and accurate, the data they utilize should be of high quality. One of the possible issues with data quality is the presence of outliers in these data. Outliers are the observations that deviate from what is considered normal and expected [15]. When data contains outliers, this may bias decisions. Outliers might signify some kind of errors, for example, errors in data capture or data processing. In such case of erroneous data, the consistency and accuracy dimensions of data quality are violated, i.e., data is incompatible with previous data and their format and does not correspond to real-world values [26]. When outliers represent correct data, they can nevertheless be of high interest in particular domains, e.g. in fraud detection, medical diagnosis, malware analysis, network intrusion detection, manufacturing quality control, etc.

Artificial intelligence and machine learning have revolutionized the way businesses operate. Similarly, the breakthrough has been made in the outlier detection domain, where thanks to machine learning, models are capable of predicting anomalous observations automatically by learning from data. However, machine learning algorithms differ in detection performance and time efficiency; hence, their business value varies significantly from one domain to another. Many benchmarking studies of outlier detection techniques exist, focusing on the technical aspects of their prediction performance (see Section 2.4). However, they lack the interpretation of these performance results from the business point of view. Moreover, benchmark studies rarely report on the detailed approach of the hyperparameter selection; thus, this gap is addressed in this paper as well.

The main purpose of this paper is to add to the existing body of research on outlier detection by comparing traditional (conventional) outlier detection methods, namely, the isolation forest, the density-based algorithm for discovering clusters and the local outlier factor models. Hence, we investigate the research question: which ones of these outlier detection models perform best in practical business applications. By making such benchmarking, the following contributions will be made to the outlier detection domain. First, the techniques of hyperparameters setting for the outlier detection methods are discussed. Second, the traditional machine learning methods for outlier detection are compared on benchmark outlier detection datasets. Third, the performance metrics suitable for outlier detection are utilized in order to compare the models by means of the McNemar's test and average ranking.

The remainder of this paper is structured as follows. Sections 2 gives a short overview of the traditional machine learning techniques for outlier detection and the existing benchmarking studies on these techniques. The methodology is described in Section 3. Sections 4 and 5 present the results of the research and discuss them. Section 6 presents the limitations and validity threats of the study. Section 7 concludes on the research.

2 Overview of traditional outlier detection methods

Originated in the 1980s in the field of intrusion detection [8], outlier detection was initially performed mainly using statistical techniques such as statistical process control tools, z-scores, box-plots, etc.

The outlier detection domain has its own assumptions that influence the way how research can be performed. Outliers are the data points that do not conform to a defined notion of a normal behavior [6]. However, the outlier detection process is not that straightforward when it comes to defining what is considered normal. The profile of normal can be defined on different levels, e.g., local, global, contextual, collective, etc. Thus, anomaly detection methods differ in a way the normal profile is defined. Also, the notion of being an outlier differs from one domain to another, making a definition of "outlierness" less straightforward. Moreover, the labelled data is not easily available in the outlier detection domain, being a major issue for the researchers [6]. That brings us to the assumption of solving the outlier detection problem using the unsupervised learning techniques. Another characteristic of the outlier detection domain is the unbalanced nature of the data: anomalous classes have fewer observations than normal classes do.

The emergence of machine learning has introduced models which are capable to learn from data and has automated the outlier detection process. The traditional machine learning models for outlier detection combine the best of two worlds: they have statistical underpinnings and at the same time learn from the data they have been trained on. In this paper, we look at the three traditional outlier detection models, namely, the isolation forest (iForest) model, the density-based algorithm for discovering clusters (DBSCAN) model and the local outlier factor (LOF) model. The former uses the notion of isolation to detect outliers, while the latter two models use the notion of density.

2.1 Isolation Forest

The isolation forest model exploits the notion of outliers being few and different from what is considered normal. Thus, the model makes two important assumptions: anomalous observations represent a minority in data and have the values that are different from normal data. In most of the cases these assumptions are realistic as they are aligned with the outlier detection domain characteristics such as the outlier definition and the nature of the data prevalent in the domain (See Section 2). However, these assumptions might not be realistic in the cases when anomalies constitute a substantial amount of the data.

Contrary to most other outlier detection techniques, the iForest model is optimized to detect anomalies [18]. Instead of building the profile of normal data, it uses the isolation approach that is aimed directly to find the unusual patterns in data. The iForest model builds the ensemble of binary isolation trees (iTrees) that isolate instances. Outliers are susceptible to isolation; hence, they are isolated closer to the root of the iTree. The path lengths of iTrees are averaged in the ensemble, and outliers have lower averaged path lengths than normal observations.

The isolation forest model requires specification of two training hyperparameters: the number of trees in the ensemble and the subsampling size [18]. The usage of subsamples to build an iTree allows mitigating the effects of masking and swamping, thus making isolation of outliers easier. The isolation forest model has a linear time complexity. Moreover, it has a major computational advantage over other models, as it does not calculate density or distance measures that are usually time and resource consuming. The iForest model is capable of handling large, high-dimensional datasets, so that it scales well for the big data problems.

2.2 Density-Based Algorithm for Discovering Clusters

The density-based algorithm for discovering clusters model is capable of discovering clusters of arbitrary shape, as it utilizes the density-based notion of clusters [11]. The DBSCAN model requires minimal domain knowledge because it provides a modeler with an intuitive approach of setting the model hyperparameters.

The DBSCAN model makes an assumption that clusters are dense region, so that it separates the low density regions from the high density regions. It uses the notions of core points, border points and outlier points [11]. The neighborhood of a given radius, *Eps*, must contain at least *MinPts* observations for a point to be a core point of the cluster. Border points have less than *MinPts* observation in their neighborhoods, whereas they should be reachable from the core point. The outliers are observations that do not belong to any of the clusters, i.e. they are not core points, and they cannot be reached from the core point. The assumption of clusters being dense regions allows the model to discover the clusters of any shape; however, the DBSCAN model fails to identify the clusters when clusters are of varying density.

2.3 Local Outlier Factor

The local outlier factor model utilizes the notion of density of a local neighborhood of an observation [5]. Contrary to the iForest and DBSCAN models that consider the "outlierness" as a binary property, the LOF model assigns to each observation a degree of it being an outlier. This score is called the Local Outlier Factor. The approach employing the LOF score allows to detect local anomalies that might be difficult to spot with a global approach.

The LOF score measures the observation's outlierness relative to its local neighborhood. The local neighborhood is defined by a hyperparameter *MinPts*

that defines the number of nearest neighbors of an observation that form this neighborhood. The LOF score measures the density of nearest neighbors of an observation relative to the density of an observation itself. If these densities are close to each other, then the LOF score value is around 1, and the observation can be considered as an inlier. If the density of the neighbors is much higher than the density of an observation, then the LOF score is much higher than 1, and the observation is an outlier. However, there is no clear rule for deciding when the observation can be considered an outlier, so that the application of the LOF model in practice requires an intervention from the domain expert's side.

2.4 Related work

Original papers on the isolation forest algorithm and the density-based algorithm for discovering clusters provide a benchmarking of the introduced model with other outlier detection techniques. [18] provides the empirical comparison of the iForest, ORCA, SVM, LOF and Random Forests models. This benchmarking uses the AUC metric and the run time to evaluate the performance of the models. According to both AUC and run time metrics, the iForest model obtains the best performance [18]. The DBSCAN model is compared with the CLARANS model [11]: the DBSCAN model is more efficient according to the run time and is capable of both clustering and discovering noise.

A standalone benchmarking study is presented in the paper about a meta-analysis of the anomaly detection problem [10]. The comparison of the density-based (Robust Kernel Density Estimation, Ensemble Gaussian Mixture Model), model-based (One-Class SVM, Support Vector Data Description), nearest neighbors based (Local Outlier Factor, KNN Angle-based Outlier Detection) and the projection-based (Isolation Forest, Lightweight Online Detector Of Anomalies) approaches is performed using the AUC and average-precision metrics and the statistical hypothesis testing. The study reports that the isolation forest model performs best on average, while the support-vector data description and One-Class SVM models obtain a poor performance [10].

The discussed benchmarking studies lack two important things. First, they lack the contextual interpretation of the result from the business point of view, as these studies focus too much only on the technical characteristics of the performance. Second, most of them employ the AUC metric for performance evaluation that can be misleading when the data are imbalanced [13].

3 Methodology

The benchmarking experiment is designed to contrast the detection performance of outlier detection models. To that end, three traditional outlier detection models, namely the iForest, DBSCAN and LOF models, are selected. The models are trained on the twelve benchmarking datasets that are described in Section 3.1. Before the data can be used for training the models, they are preprocessed in a way described in Section 3.2 and the models are trained on these data in the way

described in Section 3.2. For the training, these models require the hyperparameters to be specified, and the way this specification is performed is described in Section 3.3. Once the models are trained, we can evaluate their prediction performance and compare these models. We use five performance metrics, namely recall, precision, F1 score, AUPRC and time, along with the McNemar's test and the average ranking to contrast the models'performance. The performance metrics and comparison techniques are described in detail in Section 3.4. Detailed implementation of the benchmarking study is available at

`https://github.com/tiu-elena/benchmark-conventional-OD-methods`.

3.1 Datasets

In this paper, models are trained, and the outliers are predicted using 12 benchmark outlier detection datasets. Ten of them are taken from the Outlier Detection Datasets Library [23]. The credit card dataset source is the Credit Card Fraud Detection competition from Kaggle (see the dataset's references in Table 4). The bank dataset is taken from the ADRepository. Detailed information on the dataset sources is displayed in Table 4. The datasets used in this research are diverse in the number of observations, dimensionality and in the percentage of outliers present in the data. The detailed dataset characteristics are displayed in Figure 1.

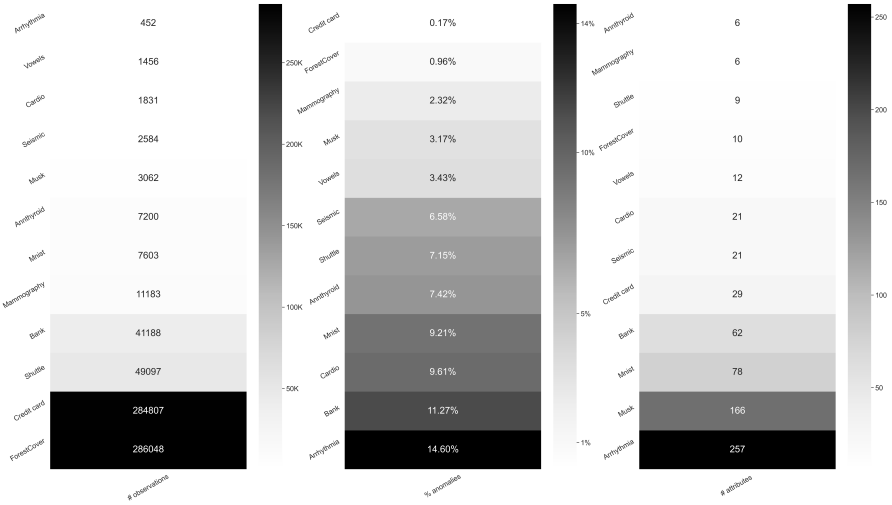

Fig. 1. Datasets characteristics

3.2 Data preprocessing and model implementation

All the models are built using the *scikit-learn* library in Python [21]. The iForest model is constructed using the sklearn.ensemble.IsolationForest class. The

LOF model is built using the sklearn.neighbors.LocalOutlierFactor class. The DBSCAN model is created using the sklearn.cluster.dbscan class. The contamination parameter in the LOF model scikit-learn implementation is set to 10%. To train the models, labels are removed from the data. During the evaluation step, labels along with the models' predictions are used to calculate performance metrics.

As the DBSCAN and LOF models use the notion of density, they are sensitive to the different scales of the features. For that reason, the data is first scaled using min-max normalization. Additionally, the variables that contain only zeros are dropped. Besides, we create dummy variables to represent categorical variables.

3.3 Hyperparameters setting

The hyperparameters of the models in this paper are set based either on the approach described in the original papers on the methods or on the research studies on the hyperparameters setting for these methods. In cases where the described approach is not feasible, the hyperparameters are set by the researchers themselves. These cases are described below in this section.

The hyperparameters of the iForest model are specified in the same way as in the original paper [18]. The subsampling size (train data size) is set to 256 as it is proved to be enough for outlier detection across a wide range of data. Another parameter is the ensemble size (the number of trees) that is set to 100 because the path lengths converge well on the ensemble of this size.

The hyperparameters of the DBSCAN model are set based on the distances to the kth nearest neighbor. The parameter k is specified in the Equation 1 [25].

$$k = 2 \cdot \#features - 1 \tag{1}$$

Equation 2 shows the specification of the *MinPts* parameter [25].

$$MinPts = k + 1 = 2 \cdot \#features \tag{2}$$

To specify the *Eps* parameter, the distances to kth nearest neighbor for each observation are sorted and plotted. The respective distance of the point located in the first "valley" of the graph is set as an *Eps* parameter in the model (Figure 2). On large datasets, we take a subset of the data (10% randomly) for the k-distances plot construction in order to make the distances' calculation feasible. For the Arrhythmia dataset, the approach from the Equation 2 is not feasible: if the approach is applied, the value of k is higher than the number of observations, and the neighborhood specification is not meaningful. For the Arrhythmia dataset, the *k* parameter is set to $\#features \cdot 0.1$ and the *MinPts* is set to $\#observation \cdot 0.1$. The *Eps* value is found using the approach illustrated in Figure 2.

The LOF model hyperparameter, *MinPts*, specification is based on the lower-bound value of 10 from the original paper on the LOF method [5] and the upper bound value of $\#observations \cdot 0.01$ that is set similarly as in the paper [27]. Equation 3 illustrates the *MinPts* parameter specification. The paper [27]

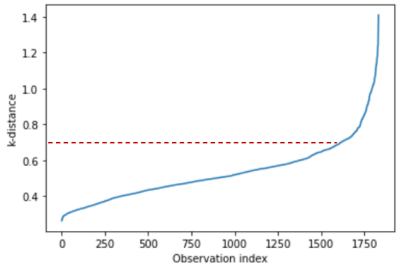

Fig. 2. *Eps* hyperparameter specification

proposes the upper bound value of $\#observations \cdot 0.03$, but we lowered the fraction to 0.01 to make the approach work for the large datasets.

$$MinPts = max(10; \#observations \cdot 0.01) \tag{3}$$

3.4 Model comparison

Once the models for outlier detection are built, the quality of their predictions is evaluated using performance metrics. The class labels that are not used during the training step are used at the model evaluation step and allow using the metrics for classification models. The outlier detection domain is characterized by imbalanced data, i.e., the majority of the observations are non-anomalous. That puts some constraints on the performance metrics which can be used at the performance evaluation step. In this paper, we use the metrics that are focused on the quality of the positive class prediction, i.e., recall, precision, F1 score and AUPRC. Additionally, the total time performance of the model (time to train + time to predict) is measured and used for the models' performance evaluation. The overview of the performance metrics is displayed in Table 1.

The AUPRC metric's thresholds are calculated using the anomaly scores obtained from the models. For the isolation forest model, the score is the anomaly score defined in the original paper [18]. For the LOF model, the score is the local outlier factor defined in the original paper [5]. For the DBSCAN model, the score is the distance to the cluster center for nonanomalous observations and the distance to the center of the closest nonanomalous cluster for outliers.

Once the model predictions are made and the performance metrics are calculated, we need to investigate whether the models perform differently and which model performs the best under what circumstances. In order to answer the first question, we use McNemar's test [12]. The McNemar's test is especially useful in the cases when the model is evaluated on a dataset only once (which is the case in this research, as the model is evaluated on the same dataset it was trained on but with the labels added). The McNemar's test constructs the contingency table and tests the homogeneity of this table. In the case of classification models, this test checks whether two models disagree in the same way or not. The McNemar's test cannot comment on whether the performance of one model is better than the performance of another model.

Table 1. Performance metrics

Metric	Description	Formula
Recall	Shows how many of the actual outliers are captured by the model. Should be high to avoid missing the outliers.	$\frac{TP}{TP+FN}$
Precision	Illustrates the ability of the model to precisely identify outliers. Defines how many of those predicted as outliers are actual outliers. Should be high to avoid false alarms.	$\frac{TP}{TP+FP}$
F1 score	Illustrates the quality of the model to balance precision and recall. Works well for imbalanced datasets.	$\frac{2 \cdot (Recall \cdot Precision)}{Recall + Precision}$
AUPRC	Shows the ability of the model to balance precision and recall on imbalanced datasets.	Area under precision-recall curve across different decision thresholds
Time	Illustrates the computational costs required by the model as well as the speed of the outlier detection. Measured using the process time.	Total time = Time to train + Time to predict

The null hypothesis is that both models make errors in the same proportions, i.e., they have the same error rate [12]. If the test fails to reject the null hypothesis, the models have a similar proportion of errors, and we can conclude that their performances are similar. If the null hypothesis is rejected, then the proportions of errors are different and models perform differently.

4 Results

The performance metrics of the constructed models are displayed in Tables 5, 6, 7, 8, 9. The iForest model has the highest average AUPRC and recall values and the lowest average total time. It also has the lowest median total time. The DBSCAN model obtains the highest values of the F1 score and precision metrics. The average time performance of the iForest model is remarkably better than the performance of the DBSCAN model and the LOF model. The average running time of the iForest model is almost 100 times lower than the LOF performance and around 46 times lower than the DBSCAN performance.

In this paper, the McNemar's test is performed for each dataset and for each pair of the models. The p-values of the test are displayed in Table 2.

As can be seen from Table 2, for most of the datasets the null hypothesis can be rejected with more than 99% confidence. The rejection of the null hypothesis shows that models make errors differently, thus not behaving in a similar way. Also, this confirms the findings of the DBSCAN model having the highest average precision and the iForest model obtaining the highest average recall. The iForest and DBSCAN models do make mistakes, but differently: iForest tend to detect as many outliers out of the true outliers as possible, but it comes at the cost of precision and results in false alarms. On the contrary, the DBSCAN model's predictions are precise and most of the predicted outliers are actual outliers.

Table 2. McNemar's test results (p-values)

Dataset	iForest vs DBSCAN	iForest vs LOF	DBSCAN vs LOF
Arrhythmia	0.54	0.003	$1.4 \cdot 10^{-5}$
Cardiocotography	0.423	$5.4 \cdot 10^{-13}$	$1.1 \cdot 10^{-17}$
ForestCover	0	0	0
Annthyroid	$8.02 \cdot 10^{-22}$	0.0002	$2.5 \cdot 10^{-34}$
Credit card	0	0	0
Mammography	$2.83 \cdot 10^{-158}$	$7.38 \cdot 10^{-13}$	$1.92 \cdot 10^{-96}$
Shuttle	0	0	$4.92 \cdot 10^{-225}$
Mnist	$2.58 \cdot 10^{-108}$	$1.25 \cdot 10^{-49}$	$9.73 \cdot 10^{-23}$
Vowels	$1.05 \cdot 10^{-23}$	0.0001	$9.41 \cdot 10^{-10}$
Seismic	0.01	$5.98 \cdot 10^{-6}$	$4.4 \cdot 10^{-9}$
Bank	0	$4.15 \cdot 10^{-103}$	0
Musk	$2.39 \cdot 10^{-36}$	$8.16 \cdot 10^{-21}$	$2.25 \cdot 10^{-86}$

This difference in the nature of predictions results in the McNemar's test null hypothesis being rejected, as the models make errors in different proportions.

To investigate whether the performance of one model is better than the performance of another model, we calculate the average ranking of the models per performance metric. Additionally, we calculate the overall rank averaged over all the performance metrics. The results are displayed in Table 3.

Table 3. Models ranking

Model	AUPRC	F1 score	Time	Recall	Precision	Average
iForest	**1.75**	1.917	**1.83**	**1.75**	2.042	1.858
DBSCAN	2	**1.75**	1.917	2.083	**1.458**	**1.842**
LOF	2.25	2.333	2.25	2.17	2.5	2.3

As can be seen from Table 3, the overall averaged ranks of the iForest and DBSCAN models are almost the same (the DBSCAN model has slightly higher rank, but the difference is rather small). The analysis of the ranks per performance metric confirms the findings from the averaged values of the performance metrics. The iForest model is ranked the highest on the AUPRC, time and recall metrics. The DBSCAN model is ranked top 1 along the F1 score and precision dimensions. The LOF model is ranked last on all the performance metrics considered.

5 Discussion

The McNemar's test results clearly show that models make mistakes differently. However, this test does not indicate the difference in the overall error rate between the models, as it reports on the difference in proportion of error between the models. In the outlier detection domain, model errors can be of two types: type 1 errors (false positives) and type 2 errors (false negatives). The McNemar's

test results show that models balance these two types differently. This fact is also illustrated by the average ranking of the models that is calculated per performance metric. This ranking shows the dominance of the iForest model for the recall metric; hence, it can better deal with false negatives and is less inclined to report that the "outlierness" is missing when it is in fact not. The DBSCAN model prevails along the precision metric. Thus, this model can better handle the false positives and raises false alarms less often.

According to the ranking, the LOF model never outperforms the iForest and DBSCAN models. It has both lower detection and time performance. However, the LOF model is the best according to the absolute value of the recall metric on large datasets with the lowest percentage of outliers (the Credit card and ForestCover datasets - see the datasets' description in Figure 1) outperforming the iForest model. Yet, it comes at a price of having low precision values and an enormous computational time.

The iForest model is able to not miss out on the outliers at a cost of having false alarms. It obtains high recall values on large datasets (the Credit card, ForestCover and Shuttle datasets), whereas its precision, F1 score and AUPRC values on most of the large datasets (the Credit card and ForestCover datasets) are the lowest. The recall values are mostly decreasing with an increasing percentage of outliers in a dataset; thus, the model becomes less sensitive to outliers once the assumption of outliers being "few" is getting weaker. In contrast, the precision of the iForest model grows for a growing percentage of anomalies, so that the model learns to be more precise when given more anomalous data to learn from. Moreover, the precision of the iForest model mostly grows when the number of features is increasing. Thus, the model has an improved precision once it is given more features to learn from (e.g. Arrhythmia, Musk, Mnist and Bank datasets). However, it is not the case for the Cardiocotography and Shuttle datasets that obtain relatively high precision values with few features. Nevertheless, the iForest model is on average the fastest among all the models considered; hence, it requires less computational power and, therefore, it is less expensive. Also, the iForest model's high time performance allows faster detection of outliers, which is critical in industries like manufacturing where the cost of the unspotted defects is growing with time.

On the contrary, the DBSCAN model is capable of being precise. It is crucial in industries where the cost of dealing with outliers is high; thus, detecting irrelevant observations and checking them afterwards would lead to losses for the businesses. The DBSCAN model obtains the highest values for the precision metric mostly on the datasets of smaller sizes (up to 10 000 observations). Its precision is mostly decreasing for growing data size and increasing for larger numbers of features; however, some datasets do not conform to these trends, making it difficult to generalize about such dependencies.

As a result, more research is needed to investigate the dependencies of the recall and precision values on the dataset's characteristics, as in most of the cases the relationship is not straightforward.

The results show that the performance of conventional outlier detection models vary greatly depending on the datasets characteristics. The practical applicability of outlier detection models in business setting depends on the nature of the data available (its size, dimensionality and contamination level of outliers), the computing resources and their cost, the costs of a false negative and a false positive prediction and the nature of the domain itself. The DBSCAN model shows the biggest potential for the industries with high false positive costs, low computing resources costs, datasets of smaller sizes and larger number of features. It is typical for such domains as insurance fraud detection, where the investigation of fraud is time- and money-consuming. The iForest model can be advantageous in the industries where the computational resources are scarce and the costs of detecting false negatives are high and the datasets sizes are large. These characteristics are crucial in the domains like medical diagnosis, where it is vital to find out any deviations from the normal profile of a patient and the amount of the data is large.

6 Limitations and validity threats

This benchmark study is subject to both internal and external threats to validity. The *internal validity threats* concern the structure of the study itself. The first possible internal validity threat is the choice of the traditional outlier detection methods for this benchmarking study. The ranking of the models depends on how many models we use to obtain the performance metrics and consequently calculate this ranking. However, the three models used in this paper, i.e. the isolation forest, DBSCAN and LOF models, are the most widely used outlier detection methods in the literature nowadays [22] [19] [14]. Nevertheless, this threat opens up an opportunity of future research considering more outlier detection techniques for a benchmarking study. The second possible internal validity threat is how we set the hyperparameters for the models in this research. All the models require hyperparameter specification and are sensitive to these hyperparameters, thus producing different results depending on the setup. However, the hyperparameters setting in this study is based on both the original paper recommendations or the research findings that investigate the optimal ways of setting the hyperparameters for the models. Thus, the research has a solid basis of the way the hyperparameters are set up. The last possible internal validity threat is the choice of the performance metrics and tests to compare the models. The ranking averaged over the performance metrics can change depending on which performance metrics we use. However, in this research, we use a diverse set of performance metrics that measure different aspects of models' performance. These metrics are also appropriate for the outlier detection domain, thus making the research findings sound.

The *external validity threats* relate to generalizability of the study to other settings and domains. The first possible external validity threat is the choice of the datasets. The extent to which the results of this benchmarking study are generalizable is directly influenced by the diversity of the datasets that are used to build the models. This study employs twelve datasets that differ greatly in

three parameters, i.e., in the number of observations and features and in the percentage of outliers. Moreover, these datasets come from different application domains. Such a variety allows for better generalization of the results, making the findings applicable to a broader context. Besides, the detailed datasets' characteristics provided in Figure 1 allow for the transferability of the results and making it possible to apply the findings to other datasets with similar configurations. Another possible external validity threat is the choice of the business context when interpreting the performance of the models. However, the performance metrics used in this research are universal, and their interpretation can be expanded to other domains. Moreover, the datasets used in this benchmarking study are diverse and come from different domains ranging from banking (e.g., Credit Card and Bank datasets) to medicine (e.g., Annthyroid and Cardiocotography datasets).

7 Conclusion

The outlier detection is attracting more and more attention in both research and practical business worlds, as the amount of data generated on a daily basis is constantly growing. Due to the artificial intelligence and machine learning in particular, it is possible to automate the process of detecting outliers and make models learn from the data they have seen. Nevertheless, not all the models are equally efficient in doing so; hence, it is crucial to understand the strengths and weaknesses of each model in order to deploy these models in practice. Thus, it is necessary to investigate which machine learning methods perform the best when applied practically. To this end, three traditional machine learning techniques, namely, the isolation forest model, the density-based algorithm for discovering clusters model and the local outlier factor model, were compared on 12 benchmark datasets. Five performance metrics, namely, AUPRC, precision, recall, F1 score and total running time, were used to measure the performance of the models. The McNemar's test along with the average ranking were applied in order to compare the models.

The first conclusion of this paper concerns the ability of the models to detect outliers. We found that models disagree differently, i.e., their type I and type II errors are not similar. It influences the detection performance of the models: the iForest model has the highest average recall, while the DBSCAN model has the highest average precision.

The second conclusion concerns the iForest model. It obtains the best time performance and is ranked first for the recall metric. That allows us to conclude that when the cost of "not acting" is high and when time and computing resources are limited, the isolation forest model is the best choice.

The third conclusion concerns the DBSCAN model. This model is the most precise one. We conclude that the DBSCAN model is better to utilize in the cases when the cost of detecting outliers is high and false alarms are not encouraged.

Further research can be performed by exploring more outlier detection techniques, e.g., deep learning models that also allows extending the research to other data types, e.g. unstructured data. Furthermore, the comparison of the

models can be extended with other techniques, e.g. the null-hypothesis testing framework.

8 Acknowledgements

The research was sponsored by the ING Chair on Applying Deep Learning on Metadata as a Competitive Accelerator.

References

1. Credit card fraud detection dataset (2016), https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
2. Abe, N., Zadrozny, B., Langford, J.: Outlier detection by active learning. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 504–509 (2006)
3. Aggarwal, C.C., Sathe, S.: Theoretical foundations and algorithms for outlier ensembles. Acm sigkdd explorations newsletter **17**(1), 24–47 (2015)
4. Bandaragoda, T.R., Ting, K.M., Albrecht, D., Liu, F.T., Wells, J.R.: Efficient anomaly detection by isolation using nearest neighbour ensemble. In: 2014 IEEE International Conference on Data Mining Workshop. pp. 698–705. IEEE (2014)
5. 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. pp. 93–104 (2000)
6. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 1–58 (2009)
7. Dal Pozzolo, A., Caelen, O., Johnson, R.A., Bontempi, G.: Calibrating probability with undersampling for unbalanced classification. In: 2015 IEEE Symposium Series on Computational Intelligence. pp. 159–166. IEEE (2015)
8. Denning, D.E.: An intrusion-detection model. IEEE Transactions on software engineering (2), 222–232 (1987)
9. Domo: Data never sleeps 9.0, https://www.domo.com/learn/infographic/data-never-sleeps-9
10. Emmott, A., Das, S., Dietterich, T., Fern, A., Wong, W.K.: A meta-analysis of the anomaly detection problem. arXiv preprint arXiv:1503.01158 (2015)
11. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of KDD. vol. 96, pp. 226–231 (1996)
12. Everitt, B.S.: The analysis of contingency tables. Chapman and Hall/CRC (2019)
13. Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B., Herrera, F.: Learning from imbalanced data sets, vol. 10. Springer (2018)
14. Hossain, F., Uddin, M.N., Halder, R.K.: Analysis of optimized machine learning and deep learning techniques for spam detection. In: 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS). pp. 1–7. IEEE (2021)
15. Johnson, R.A., Wichern, D.W., et al.: Applied multivariate statistical analysis, vol. 6. Pearson London, UK: (2014)
16. Keller, F., Muller, E., Bohm, K.: Hics: High contrast subspaces for density-based outlier ranking. In: 2012 IEEE 28th international conference on data engineering. pp. 1037–1048. IEEE (2012)

17. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 eighth ieee international conference on data mining. pp. 413–422. IEEE (2008)
18. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD) **6**(1), 1–39 (March 2012). https://doi.org/10.1145/2133360.2133363
19. Nofal, S., Alfarrarjeh, A., Abu Jabal, A.: A use case of anomaly detection for identifying unusual water consumption in jordan. WATER SUPPLY **22**(1), 1131–1140 (JAN 2022). https://doi.org/10.2166/ws.2021.210
20. Pang, G., Shen, C., Cao, L., Hengel, A.V.D.: Deep learning for anomaly detection: A review. ACM Computing Surveys (CSUR) **54**(2), 1–38 (2021)
21. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
22. Raj, C., Khular, L., Raj, G.: Clustering based incident handling for anomaly detection in cloud infrastructures. In: 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence). pp. 611–616. IEEE (2020)
23. Rayana, S.: ODDS library (2016), `http://odds.cs.stonybrook.edu`
24. Sathe, S., Aggarwal, C.: Lodes: Local density meets spectral outlier detection. In: Proceedings of the 2016 SIAM international conference on data mining. pp. 171–179 (2016)
25. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Dbscan revisited, revisited: Why and how you should (still) use dbscan. ACM Transactions on Database Systems (TODS) **42**(3) (Jul 2017). https://doi.org/10.1145/3068335
26. Sidi, F., Shariat Panahy, P.H., Affendey, L.S., Jabar, M.A., Ibrahim, H., Mustapha, A.: Data quality: A survey of data quality dimensions. In: 2012 International Conference on Information Retrieval Knowledge Management. pp. 300–304 (2012). https://doi.org/10.1109/InfRKM.2012.6204995
27. Soenen, J., Van Wolputte, E., Perini, L., Vercruyssen, V., Meert, W., Davis, J., Blockeel, H.: The effect of hyperparameter tuning on the comparative evaluation of unsupervised anomaly detection methods. In: Proceedings of the KDD'21 Workshop on Outlier Detection and Description. pp. 1–9 (2021)
28. Tan, S.C., Ting, K.M., Liu, T.F.: Fast anomaly detection for streaming data. In: Twenty-Second International Joint Conference on Artificial Intelligence (2011)
29. Ting, K.M., Zhou, G.T., Liu, F.T., Tan, J.S.C.: Mass estimation and its applications. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 989–998. Association for Computing Machinery, New York, NY, USA (2010), `https://doi.org/10.1145/1835804.1835929`
30. Ting, K., Tan, S., Liu, F.: Mass: A new ranking measure for anomaly detection. Gippsland School of Information Technology, Monash University (2009)

9 Appendices

Table 4. Dataset sources

Dataset	Sources
Arrhythmia	[17], [30], [16]
Cardio	[3], [24]
ForestCover	[17], [30], [29], [28]
Annthyroid	[2], [17], [30]
Credit card	[7], [1]
Mammography	[2], [17], [30]
Shuttle	[2], [17], [30], [29], [28]
Mnist	[4]
Vowels	[3], [24]
Seismic	[24]
Bank	[20]
Musk	[3]

Table 5. AUPRC

Dataset	iForest	LOF	DBSCAN
Arrhythmia	0.440	0.338	0.456
Cardiocorography	0.580	0.186	0.642
ForestCover	0.062	0.248	0.025
Annthyroid	0.320	0.202	0.115
Credit card	0.104	0.453	0.390
Mammography	0.217	0.126	0.083
Shuttle	0.982	0.137	0.162
Mnist	0.269	0.313	0.334
Vowels	0.152	0.289	0.047
Seismic	0.117	0.081	0.116
Bank	0.282	0.190	0.269
Musk	0.996	0.026	1.000
Average	**0.377**	0.216	0.303

Table 6. F1 score

Dataset	iForest	LOF	DBSCAN
Arrhythmia	0.179	0.250	0.412
Cardiocotography	0.529	0.228	0.353
ForestCover	0.091	0.168	0.120
Annthyroid	0.325	0.297	0.179
Credit card	0.066	0.030	0.112
Mammography	0.196	0.199	0.218
Shuttle	0.776	0.158	0.204
Mnist	0.348	0.357	0.293
Vowels	0.179	0.418	0.554
Seismic	0.197	0.098	0.194
Bank	0.314	0.216	0.269
Musk	0.515	0.015	0.957
Average	0.310	0.203	**0.322**

Table 7. Precision

Dataset	iForest	LOF	DBSCAN
Arrhythmia	0.583	0.304	0.583
Cardio	0.535	0.224	0.677
ForestCover	0.049	0.092	0.073
Annthyroid	0.294	0.258	0.339
Credit card	0.034	0.020	0.060
Mammography	0.116	0.122	0.200
Shuttle	0.639	0.136	0.141
Mnist	0.251	0.343	0.538
Vowels	0.119	0.281	0.500
Seismic	0.138	0.081	0.132
Bank	0.363	0.229	0.189
Musk	0.346	0.010	1.000
Average	0.289	0.175	**0.369**

Table 8. Recall

Dataset	iForest	LOF	DBSCAN
Arrhythmia	0.106	0.212	0.318
Cardiocotography	0.523	0.233	0.239
ForestCover	0.689	0.956	0.332
Annthyroid	0.363	0.348	0.122
Credit card	0.833	0.910	0.841
Mammography	0.635	0.527	0.238
Shuttle	0.987	0.190	0.369
Mnist	0.566	0.373	0.201
Vowels	0.360	0.820	0.620
Seismic	0.347	0.124	0.365
Bank	0.277	0.203	0.464
Musk	1.000	0.031	0.918
Average	**0.557**	0.411	0.419

Table 9. Time

Dataset	iForest	LOF	DBSCAN
Arrhythmia	0.411	0.031	0.021
Cardiocotography	0.307	0.099	0.125
ForestCover	10.13	1153.56	119.54
Annthyroid	0.490	0.714	0.844
Credit card	17.9	2636.8	1692
Mammography	0.813	1.151	1.625
Shuttle	2.453	13.859	6.417
Mnist	1.323	2.245	1.729
Vowels	0.297	0.047	0.109
Seismic	0.427	0.255	0.198
Bank	4.927	63.714	41.938
Musk	1.047	0.448	0.438
Average	**3.377**	322.74	155.414
Median	**0.93**	0.932	1.234