Hunting inside n-quantiles of outliers (Hino)

Jessy Colonval[1 [0000−0001−6792−0264]] and Fabrice Bouquet[1 [0000−0001−9181−1172]]

Université de Franche-Comté, Institut Femto-ST, CNRS,
16 route de Gray, Besançon, France
{jessy.colonval,fabrice.bouquet}@femto-st.fr

Abstract. This paper presents an approach (called **Hino**) to detect outliers present in a data set, also called aberrations or anomalies. These data may reduce the quality of data analysis and lead to erroneous results. In the case of learning algorithms, they can deviate their behavior, i.e. reduce their efficiency. Thus, outlier detection is crucial where it improves performance by providing better data quality and reduces the influence of outliers. Inter Quartiles Range (**IQR**) is a popular statistical detection method, which has the advantage of being simple and fast in calculation time. It is based on the distribution quartiles of a data set and considers the most extreme values as outliers. This means that this method only searches for **point outliers**, which is a restrictive and naive approach. Indeed, nothing prevents an element from having an extreme value while remaining consistent with the rest of the elements. The proposed method is also a statistical detection method based on quantiles, but it looks for **contextual outliers** instead of **point outliers** and consider the context of a point to determine whether it is an outlier or not. The effectiveness of **Hino** is compared with the original **IQR** method and other approaches, including **Isolation Forest**, **SVM**, and **LOF**, using 16 real and 278 synthetic data sets.

Keywords: Outliers detection · Data filtering · Interquartile range · Machine learning.

1 Introduction

An important principle in the use of artificial intelligence algorithms (AIs) is the quality of the learning data. So, the implementation of tools to help filter the data is a critical point [10]. However, data sets may contain outliers, which can be defined as "an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data" [3]. Outliers can reduce the quality of data analysis and lead to erroneous results. And in the case of AIs, they can deviate behavior of AIs and reduce their efficiency with two effects:

1. Weaken the predictive power of the model obtained after the learning phase.
2. Weaken the score obtained from the model during its validation phase.

Therefore, it is essential to identify and eliminate them directly. Thus, the quality of the data is improving and outlier's influence is reducing [9].

Three categories of outliers are identified [8]:

1. **Points outliers** are isolated individuals, they are far from other data.
2. **Contextual outliers** are isolated individuals in a specific context, i.e. according to their relations between their attributes and their behavioral values.
3. **Collective outliers** are a set of individuals consistent with each other but isolated from the rest of the data.

The **IQR** approach considers the distribution of a point to determine if it is isolated which makes it a statistical detection method. And since the context is not taken into consideration, then this method detects **point outliers**. Moreover, this method considers that a point with at least one of these values isolated is sufficient to consider it as an outlier. However, we will see that this method is ineffective for detecting aberrations. The objective is to extend the **IQR** approach so that it is able to detect **contextual outliers** and tolerate that the values of a point can be isolated a minimum number of times before considering the point fully isolated. This new approach is called **Hunting inside n-quantiles of outliers (Hino)**.

Since **Hino** is looking for contextual outliers and the data sets used in the experimentation are spatial (see Section 5), then the terminology defines two types of attributes [1, Chapter 11]:

1. **Behavioral attributes** are attributes of interest measured for each point, commonly called *class*. For example: the type of glass, the presence of a heart abnormality, or the description of an image. A behavioral value is a value of this attribute. For example, a behavioral attribute that describes 7 different glass categories, here **1** would be one of the existing behavioral values.
2. **Contextual attributes** is an attribute expressing the characteristics of a point, commonly called *feature*. For example: the composition of a glass pane, the number of heartbeats per minute, or the color of a pixel.

The Section 2 present the **IQR** method of detection by quantiles, its functioning and the problems it raises. Then, the Section 3 present the **Hino** approach, its functioning and its meta-parameters. Follow-up of the Section 4 which proposes a methodology to generate the synthetic data sets used (see Section 4.1) and equations to calculate the **Hino** meta-parameters (see Section 4.2). Finally, the performance detection of **Hino** is compared with three methods (**IQR**, **Isolation Forest**, **SVM** and **LOF**) on a panel of 278 synthetic data sets (see Section 5.1) and on a panel of 16 real data sets (see Section 5.2) to conclude with a discussion of some biases present in the evaluation of real data sets and on the values used for the meta-parameters (see Section 5.3).

2 Detection method based on the interquartile range

In general, the median divides a data set into two more or less equal parts (depending on if the number of elements is even or odd), so approximately half of the elements are below the median and the rest are above it. On the same

principle, it is possible to make this division into two unequal parts such that a percentage p of the data is less than a certain number and a percentage $1-p$ is greater than that number. This number is called the $100p$ *empirical percentile* or the *pth empirical quantile* and is denoted by $q_n(p)$ [5].

To calculate the interquartile range, the data set is divide into four parts. Using the previous definition we define: the 25th empirical percentile $q_n(0.25)$ is called the *lower quartile* (Q1) and the 75th empirical percentile $q_n(0.75)$ is called the *upper quartile* (Q3). Together with the median, they allow the division of a data set into four parts more or less equal in number of elements. The distance between these two quartiles gives an indication of the skewness of the data set. This distance is called **IQR** (**I**nter **Q**uartile **R**ange), and it equals to:

$$IQR = Q3 - Q1 = q_n(0.75) - q_n(0.25) \tag{1}$$

The interquartile range is used to describe the spread of a distribution. Thus, it is possible to detect outliers by considering the most scattered elements as outliers. For this, it is necessary to define thresholds that will cover the majority of the points so that only the most scattered ones are considered as outliers. The lower fence is equal to $Q1 - 1.5 * IQR$ and the upper fence is equal to $Q3 + 1.5 * IQR$. The constant 1.5 allows controlling the sensitivity of the interval and thus the rule of detection. A larger one would cause outliers to be considered healthy, while a smaller one would cause some points to be detected as outliers. Thus, with a Gaussian distribution, then a constant equal to 1.5 gives fences covering -2.698σ (see equation 2) and 2.698σ or 99.3% of the points.

$$\begin{aligned} & Q1 - 1.5 * IQR = Q1 - 1.5 * (Q3 - Q1) \\ = & -0.6745\sigma - 1.5 * (0.6745\sigma - (-0.6745\sigma)) \\ = & -0.6745\sigma - 1.5 * 1,349\sigma = -2.698\sigma \end{aligned} \tag{2}$$

Figure 1, gives a visualization of quartiles and fences as a box plot and relates them to a Gaussian distribution. The majority of the points are included between these fences and only a small part of the points, the most extreme, will be excluded. Vinutha et al. [11] use the interquartile range as an outlier detection method. For each contextual attribute, this range is calculated and all points within are considered as outliers.

This method have three problems that make hazardous the use of the interquartile range as an outlier detection method. First, it removes the points that have extreme values who

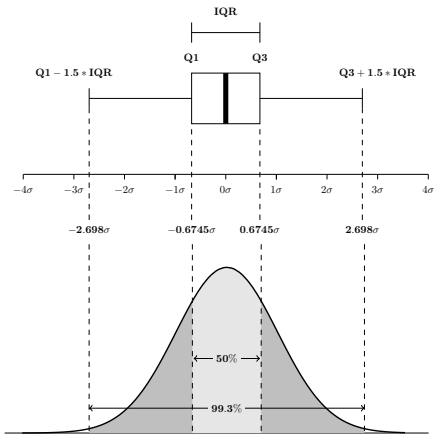


Fig. 1: Box plot and the Gaussian probability density of interquartile range

aren't always outliers. Indeed, nothing prevents an inlier to be extreme. Second, it doesn't consider the individual distribution of a behavioral value. One of them can be essentially present on the extreme values which would be suppressed with this approach. Third, there is no **tolerance limit** to determine if a point is an outlier. With only one extreme values among the contextual attribute, the large data sets are disadvantaged and risk to have a high false positivity rate because a point will have more chance to have an extreme value.

3 Extension of the IQR method

As a reminder, an outlier can be defined as an abrupt change. **IQR** detects only **point outliers** and considers as outliers the points with at least one extreme values. As explained previously, this approach is problematic. Thus, our objective is to extend **IQR** to be able to detect **contextual outliers** and tolerate the isolation of a point for few attributes before being an outlier.

For this purpose, we propose to separate the data set into several quantiles for each contextual attribute, instead of only 4. For each quantile, we check that all the behavioral values that constitute it are also present in adjacent quantiles. Otherwise, all representatives of absent values are considered isolated for this attribute. Except in the case where all the representatives of a value are present in a single quantile. A point isolated for few attributes is not necessarily an outlier. Indeed, depending on the total number of attributes, an isolation for just one of them does not have the same impact. For example, an isolation for 1 attribute out of 10 does not have the same significance for 100. Our approach tolerates a certain level of isolation before identifying a point as an outlier. To summarize, an outlier is a point "too often" isolated.

Our extension is summarized in the Algorithm 1. It takes as parameters the data set (P), the number of quantiles used (n_qtils), the tolerance limit ($limit$) and the maximum percentage of points that can be considered as outliers (p_{max}) to preserve the consistency of data set. And it returns the mapping between the points in P and a Boolean to indicate if they are outliers. Except P, the other arguments strongly influence the detection result. However, it is difficult to determine the best values for each of them to obtain an optimal outlier detection.

- p_{max} is set at 20%, greater, we assume the data set will be distorted.
- n_qtils is calculated according to the number of points they must contain. They must be small enough so that the points they contain share similar characteristics and make it easier to detect abrupt changes.
- $limit$ is calculated according to the number of contextual attributes and behavioral values. When the number of contextual attributes and/or behavioral values are high, there's a greater chance of obtaining many isolated values.

This algorithm has been implemented in Python and its time complexity is equal in the worst case to $O(p(ac + a + 1))$, while in the best case to $O(p(ac + \frac{a}{a+1} + 1))$, where p is the number of points, a is the number of attributes and

c is the number of behavioral values. Plus, its memory complexity is equal to $O(p(a + \frac{a}{a+1}))$.

Algorithm 1: Hino.

input : Let P be a set of n points
Let $n\_qtils \geq 2$ be the number of quantiles
Let $limit > 0$ be the maximum of breaking rule
Let p_{max} be the maximum percentage of outliers
output: Let $result$ be an mapping indicating if a point is an outlier

A_c be the set of contextual attributes labels in P
`/* Counters that track the frequency of isolated points. */`
$n\_cdt \leftarrow \{(p, 0) | p \in P\}$
for $a \in A_c$ **do**
  $qtils \leftarrow$ A list of list that groups the points divided by n_qtils quantiles on the attribute a
  for $i \in [0, card(qtils)[$ **do**
    `/* Gets the behavioral values of the previous, next and current quantile, respectively. */`
    $prev\_cls \leftarrow \emptyset$
    if $i > 0$ **then**
      $prev\_cls \leftarrow$ Behavioral values in $qtils[i-1]$
    $next\_cls \leftarrow \emptyset$
    if $i < card(qtils) - 2$ **then**
      $next\_cls \leftarrow$ Behavioral values in $qtils[i+1]$
    $cur\_cls \leftarrow$ Behavioral values in $qtils[i]$
    `/* Missing behavioral values are those in the current quantile, but absent from the two adjacent ones. */`
    $missing\_cls \leftarrow cur\_cls \setminus (prev\_cls \cup next\_cls)$
    `/* If one is missing, then the counters of their points in the current quantile are increased. */`
    if $card(missing\_cls) > 0$ **then**
      for $pq \in qtils[i]$ **do**
        $pq\_cls \leftarrow$ Behavioral value of pq
        $all\_here \leftarrow$ True if all of pq_cls are in $qtils[i]$
        `/* In that case their counters are not incremented. */`
        if $pq\_cls \in missing\_cls$ ***and not*** all_here **then**
          $n\_cdt[pq] + +$
`/* Repeat with a limit incremented if a behavioral value is fully identified as an outlier or if more than` p_{max} `points are outliers. */`
repeat
  $result \leftarrow \{(p, False) | p \in P\}$
  for $(p, count) \in n\_cdt$ **do**
    $result[p] = count > limit$
  $limit + +$
until *(countTrue(result)* $\leq p_{max}$ ***and not*** *viableCls(result))*

4 Determination of meta-parameters

The first meta-parameters is the number of quantiles and it equals to $\frac{p}{c+1}$ to ensure that each quantile is large enough so that each behavioral value is represented at least once. The second meta-parameter is the maximum percentage of outliers set at 20%, we assume that over this value a data set may become inconsistency. The third meta-parameter is the **tolerance limit**, which is determined by the number of attributes and behavioral values. However, building an equation from these values is difficult, so it's estimated empirically using a regression study based on synthetic data sets (see Section 4.2).

4.1 Generation of synthetics data sets and these outliers

Synthetic data sets are created with the same algorithm designed to generate *Madelon* [6]. They all have only one behavioral attribute whose values are homogeneously distributed. And all contextual attributes are useful for establishing the behavioral value. The original generation does not contain any outliers, but they are added after by using different approaches of *machine learning* algorithms (**ML**) from the Python library *scikit-learn*[1]: *SVC*[2], *K Neighbors Classifier*[2], *Random Forest Classifier*[3], *Extra Trees Classifier*[3], *Gradient Boosting Classifier*[2] and *Logistic Regression*[4].

The goal is to detect the most useful points for predictions in order to change their behavioral value. Thus, the chance of having strong outliers which are harmful to the predictions is increased. While a purely random method wouldn't prevent the selection of points that are not very useful for the prediction. For this purpose, the data sets are randomly divided 10 times into a pair of training **T** (70%) and test **V** (30%) sets. For each pairs, the **ML** are trained with **T** and predict the points in **V**. The number of times each point is wrongly predicted are counted. Thus, each point will have an associated counter with a value between 0 and $60 = 10 * 6$ (**ML** algorithms). Those with the lowest values will be the points that have been most frequently correctly predicted. Finally, the n points that will become outliers are the first n with the lowest scores. Where there are more than 2 behavioral values, the new one is chosen randomly.

4.2 Tolerance limit estimation

The tolerance limit is determined by a regression study based on 346 data sets which contain 5% of outliers. Plus, they have distinct characteristics in order to cover a large set of attribute and behavioral value number (see Table 1). The number of points is small in order to perform a large amount of computation in a reasonable time. Since **Hino** works with quantiles of fixed sizes, so the number of points has no influence on the outlier detection.

[1] https://scikit-learn.org/stable/index.html
[2] All meta-parameters have the default values.
[3] All meta-parameters have the default values, except `n_estimator` set to 200.
[4] All meta-parameters have the default values, except `max_iter` set to 1 000.

This study consists of determining what is the optimal **tolerance limit** for each data set. It is based on the assumption that there is a relationship between this limit, the numbers of contextual attributes and behavioral values. More concretely, all possible **tolerance limits** are used in order to determine which one is the best. A detection is performed for each limit and a **sensitivity** and **specificity** score is calculated. These scores are determined with the numbers of true positive (**TP**), true negative (**TN**), false positive (**FP**) and false negative (**FN**).

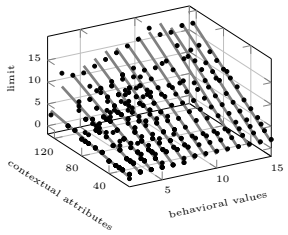
Points (p)	**Attributes** (a)	**Behavioral values** (c)
250	10 . . . 50, 25	2 . . . 5
500	10 . . . 70, 25, 75	2 . . . 5
1 000	10 . . . 100, 25, 75	2 . . . 7
2 500	10 . . . 100, 25, 75	2 . . . 8
5 000	10 . . . 100, 25, 75, 125	2 . . . 11

Table 1: data sets used in regression study with a step of 10 for a and 1 for c

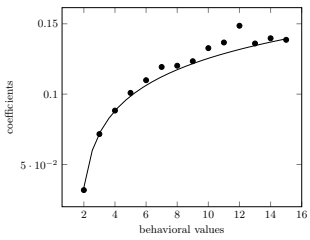
$$sensitivity = \frac{TP}{TP + FN} \quad (3) \qquad specificity = \frac{TN}{TN + FP} \quad (4)$$

The **sensitivity** measures the ability of a test to give a positive result when a hypothesis is verified, and it is calculated using equation 3. It corresponds to the percentage of correctly detected outliers. While the **specificity** measures the ability of a test to give a negative result when the hypothesis is not verified, and it is calculated using equation 4. It corresponds to the percentage of undetected points that are actually healthy. These metrics are interpreted together, a good **tolerance limit** must give these scores as close to 100% as possible.

Plus, a good **tolerance limit** must not remove all points of a behavioral value, and it must not remove too many points, e.g. 20%. The optimal limit is selected from the remaining ones, and is the one with the highest sum of **sensitivity** and **specificity**. Thus, those found for each data set are visualized on a 3D graph according to the number of contextual attributes and behavioral values in Figure 2a. This limit is shown to increase with the number of attributes and behavioral values, which confirms the relationship between them.



(a) Graph of linear regression computed on optimal limits obtained



(b) Logarithmic regression of linear's coefficients

$[c = 2]0.03183a$ $[c = 3]0.07163a$ $[c = 4]0.08832a$ $[c = 5]0.1009a$ $[c = 6]0.1099a$ $[c = 7]0.1193a$ $[c = 8]0.1202a$
$[c = 9]0.1234a$ $[c = 10]0.1327a$ $[c = 11]0.1367a$ $[c = 12]0.1486a$ $[c = 13]0.136a$ $[c = 14]0.1397a$ $[c = 15]0.1386a$

(c) Linear function for each behavioral value

Fig. 2: Regression study

From this relation, a second regression is performed in order to determine the slope of each linear line in Figure 2a according to the number behavioral values. Note that the equations of these lines are shown in Figure 2c. As shown in Figure 2b, these slopes do increase with the characteristics of the data set, but logarithmically. Thus, this regression gives the following equation:

$$0.0205 * log_2(-1.623730 + c) + 0.062579 \tag{5}$$

Finally, it is possible to establish an equation that will approach the optimal **tolerance limit** with the number of attributes and behavioral values:

$$\lfloor (0.0205 * log_2(-1.623730 + c) + 0.062579) * a \rceil \tag{6}$$

The result is only an estimation and is not guaranteed to be the optimal **tolerance limit**. This is due to the choices made on the characteristics of the synthetic data sets and the approximations made during the regressions. However, we can be confident that this estimation gives a viable solution.

5 Experimentation

The efficiency of the **IQR** (see Section 2), **Isolation Forest** [7], **SVM** [2], **LOF** [4] and **Hino** (see Section 3) methods are compared to their ability to correctly detect outliers on two kinds of data sets: synthetics (different from those in Table 1) and real. Computations have been performed on the supercomputer facilities of the *Mésocentre de calcul de Franche-Comté*. The results obtains are present in a GitHub directory[5].

5.1 On synthetic data sets

An additional 278 data sets with their outliers are created with the same way as those used in the Section 4.1, which has the characteristics present in Table 2. **LOF**, **Isolation Forest** and **SVM** come from the Python library *scikit-learn* and performed with the default settings. Meta-parameters used for **Hino** are already established in Section 4.2.

Points (p)	**Attributes** (a)	**Behavioral values** (c)
375	10 . . . 50	2 . . . 5
1 250	10 . . . 100	2 . . . 7
3 000	10 . . . 100	2 . . . 8
6 000	10 . . . 100	2 . . . 9
10 000	10, 50, 75, 100	2, 3, 5, 7, 9, 11
25 000	10, 50, 75, 100	2, 3, 5, 7, 9, 11

Table 2: Data sets used in evaluation with a step of 10 for a and 1 for c

Outliers detected are compared with the real ones in order to determine a **sensitivity** and **specificity**. Results are given in Figure 3 and the best method is the one that has these two scores closest to 100%. The results conclude:

[5] https://github.com/JessyColonval/Hino

- **IQR** have a linear relationship between these scores, it means that the number of false positives is proportional to the number of outliers detected.
- The whole **sensitivity** of **Isolation forest** remains $< 20\%$ (the majority is $< 10\%$ or $= 0\%$) while its **specificity** is between 80% and 100%. It means few or no outliers are removed, but it's always removing inliers.
- **SVM** and **LOF** preserve correctly the inliers with a score of **specificity** $> 96\%$. However, they remove a fewer percent of outlier, i.e. $< 5\%$.
- The **sensitivity** of **Hino** are $< 60\%$ (the majority are between 10 and 40%), while the **specificity** are $> 70\%$ (the majority are $> 80\%$). It means that generally, it remove more outlier than inliers unlike **IQR** but also a larger number than the **Isolation Forest**.

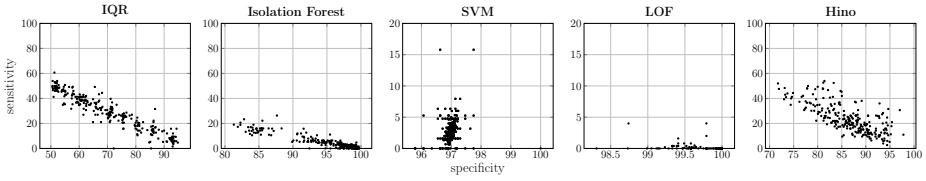


Fig. 3: Specificity and sensitivity on the 5 methods

5.2 On real data sets

In order to confirm the effectiveness of **Hino** compared to state-of-the-art methods, the outlier detection is computed on 16 real data sets from *UCI Machine Learning Repository*[6]. Few are used, due to the difficulty of finding ones with expected characteristics. Their contextual attributes are numerical values and they represents a panel of different characteristics (see Table 3) in order to observe the behavior of these methods according to these different characteristics.

Data sets	Points	Attributes	Behavioral values
annthyroid	7 200	21	3
breastW	683	9	2
cardio	2 126	21	10
glass	214	9	6
ionosphere	351	33	2
isolet	7 797	617	26
letter recognition	20 000	16	26
mammography	11 183	6	2
multiple features	2 000	619	10
musk	6 598	166	2
parkinson	756	752	2
pendigits	10 992	16	10
satimage2	6 435	36	6
shuttle	58 000	9	7
wine	178	13	3
wine-quality	1 599	11	6

Table 3: Characteristics of real data sets.

Two detections are analyzed with **Hino**: one with the meta-parameters calculation described in Section 4.2, and other one with the best solution among all possible **tolerance limits**. The goal is to validate the **tolerance limit** provide

[6] https://archive.ics.uci.edu/ml/index.php

by equation 6 and observe the variation with the optimal one. Once detected, these outliers are removed from the data set, and the relevance of this removal is measured using two metrics by using 6 **ML** algorithms described in Section 4.1:

- **Cross-validation** measures the prediction performance of these algorithms. We assume that the presence of outliers reduces the performance of these algorithms, then it determines which removal was the most beneficial.
- **False positive** measure the performance of these **ML** algorithms to correctly predict the outliers detected. We assume that the outliers removed from the training set cannot be correctly predicted by these algorithms..

Table 4 summarizes and presents these results in 2 separate columns:

- **Cross-validation** cells contain the percentage of cross-validation (1^{st} line); the standard deviation (2^{nd} line) and **tolerance limit** (only for **Hino**) | percentage of outliers | ✓ if all behavioral values are kept, ✗ otherwise (3^{rd} line). **Tolerance limits** annotated with a '*****' are those that have been modified to obtain a percentage of outliers $< 20\%$.
- **False positive** cells contain the percentage of points falsely detected as outliers, and the standard deviation of this percentage.

Cases where no outliers have been detected are indicated by the value **None** with the **cross validation** of the full data set. Cells with **Error** imply that the computations of the metrics could not be done because of an overly distorted data set. For example, when a behavioral value has only one representative.

A method is considered better when: it doesn't suppress any behavioral value; the number of points removed isn't too high; the cross-validation is higher and the false positive is lower. Sometimes the choice is not obvious, and the method that is considered better depends on the weight given to these different metrics. These choices are symbolized by two color: **gray** when one solution is better; **light gray** when multiple solutions are closed and better than the others.

According to Table 4, **Hino** gives equivalent or better results for almost all data sets with the estimation of the meta-parameters proposed in Section 4 (column **Hino**). Except for *wine*, where the best solutions are given by **SVM** and **LOF**. But by manually changing the **tolerance limits**, it is possible to get better results (column **Best of Hino**). Thus, out of 16 data sets, the results are clearly improved for 8, remain equivalent for 5 and do not change for the last 3.

5.3 Discussion

The use of **ML** algorithms to compute **false positives** introduces a bias. Indeed, a point detected as an outlier, but correctly predicted, does not necessarily mean that the detection has failed. In a case where too many inliers have been removed from the training set, then it is possible that the **ML** algorithms no longer have enough information to correctly predict the outliers. Thus, these outliers would be wrongly predicted because the training set is too distorted. This situation is less likely when few points are removed, but it should be read carefully and in addition to the other metrics.

	IQR		Isolate Forest		SVM		LOF		Hino		Best of Hino	
Data sets	**Cross validation**	**False positive**	**Cross validation**	**False positive**	**Cross validation**	**False positive**	**Cross validation**	**False positive**	**Cross validation**	**False positive**	**Cross validation**	**False positive**
annthyroid	Error 53.6%		97.68% ±0.19 5.54% | ✓	94.22% ±0.07	97.12% ±0.22 3.01% | ✓	97.98% ±0.06	98.12% ±0.22 11.22% | ✓	94.72% ±0.12	97.65% ±0.19 2 | 10.4% | ✓	94.43% ±0.06	No change	
breastW	97.13% ±1.12 28.1% | ✓	91.66% ±0.36	98.61% ±0.29 37.63% | ✓	94.22% ±0.90	97.09% ±0.91 7.32% | ✓	99.67% ±0.0	96.73% ±1.19 24.60% | ✓	95.08% ±0.25	98.24% ±0.86 2* | 12.7% | ✓	89.66% ±0.17	97.61% ±1.18 3 | 3.95% | ✓	81.48% ±0.00
cardio	81.4% ±1.94 56.3% | ✗	60.35% ±0.19	81.23% ±1.26 16.04% | ✓	72.71% ±0.35	82.06% ±1.5 3.15% | ✓	71.67% ±0.3	82.45% ±1.69 1.69% | ✓	71.67% ±3.15	83.16% ±1.30 3 | 5.9% | ✓	39.72% ±0.59	84.76% ±1.3 2 | 12.61% | ✓	43.77% ±0.57
glass	73.46% ±4.87 36.4% | ✗	28.12% ±1.09	74.75% ±5.05 16.82% | ✓	48.7% ±1.71	72.28% ±4.66 1.87% | ✓	49.17% ±1.76	73.49% ±5.15 15.89% | ✓	40.98% ±1.25	73.95% ±4.21 2* | 9.35% | ✓	29.25% ±1.29	72.16% ±5.55 4 | 0.93% | ✓	0.00% ±0.00
ionosphere	95.12% ±2.58 48.1% | ✓	59.47% ±0.43	90.81% ±1.97 32.48% | ✓	86.54% ±0.52	91.21% ±1.88 2.85% | ✓	91.67% ±0.00	94.23% ±2.00 34.76% | ✓	55.61% ±0.72	92.46% ±2.19 6* | 16.8% | ✓	78.02% ±0.47	92.57% ±2.26 9 | 4.56% | ✓	63.54% ±0.00
isolet	Error 96.6%		93.1% ±0.49 14.21% | ✓	93.4% ±0.14	93.53% ±0.44 2.98% | ✓	90.55% ±0.25	93.60% ±0.50 0.05% | ✓	75.00% ±0.00	95.08% ±0.40 97 | 13.7% | ✓	76.61% ±0.23	No change	
letter recognition	90.04% ±0.38 47.4% | ✓	68.37% ±0.23	91.48% ±0.27 13.57% | ✓	88.15% ±0.09	91.64% ±0.32 3.01% | ✓	92.75% ±0.12	91.48% ±0.25 0.19% | ✓	94.52% ±0.23	None - 91.72% 3 | 0.0%		91.93% ±0.24 0 | 8.16% | ✓	64.47% ±0.15
mammo-graphy	99.41% ±0.04 37.6% | ✓	94.33% ±0.03	99.26% ±0.03 19.88% | ✓	91.06% ±0.2	98.69% ±0.12 3.08% | ✓	91.23% ±0.16	98.66% ±0.00 1.78% | ✓	90.98% ±0.19	99.85% ±0.03 1* | 10.7% | ✓	83.75% ±0.05	99.24% ±0.15 3 | 1.11% | ✓	27.28% ±0.47
multiple features	Error 97.65% | ✗	35.3% ±0.74	99.0% ±0.45 35.45% | ✓	91.15% ±0.2	98.29% ±0.45 3.00% | ✓	90.14% ±0.54	98.33% ±0.47 0.05% | ✓	3.33% ±10.54	99.43% ±0.25 78 | 17.9% | ✓	85.2% ±0.37	98.7% ±0.34 119 | 0.7% | ✓	25.6% ±2.18
musk	91.38% ±4.61 97.4% | ✓	59.22% ±0.37	96.39% ±0.34 6.68% | ✓	98.78% ±0.06	96.45% ±0.27 2.99% | ✓	99.31% ±0.25	96.54% ±0.35 0.32% | ✓	93.17% ±0.41	98.31% ±0.23 6 | 4.5% | ✓	43.46% ±0.31	97.47% ±0.36 11 | 1.55% | ✓	22.91% ±0.72
parkinson	Error 100%		87.04% ±1.78 2.78% | ✓	86.19% ±1.14	86.45% ±1.9 3.04% | ✓	87.03% ±1.08	86.67% ±1.67 3.04% | ✓	83.26% ±0.79	96.93% ±0.80 95* | 19.8% | ✓	9.48% ±0.29	96.94% ±1.06 96 | 19.71% | ✓	9.47% ±0.29
pendigits	98.58% ±0.20 4.7% | ✓	85.18% ±0.17	98.87% ±0.23 51.91% | ✗	50.45% ±0.17	98.52% ±0.16 2.98% | ✓	94.99% ±0.16	98.67% ±0.20 1.56% | ✓	76.34% ±0.45	98.63% ±0.20 2 | 16.7% | ✓	97.27% ±0.04	98.58% ±0.12 9 | 0.01% | ✓	0.00% ±0.00
satimage2	89.19% ±0.46 8.4% | ✓	96.65% ±0.16	88.48% ±0.75 19.40% | ✓	58.24% ±4.04	89.28% ±0.52 2.97% | ✓	97.56% ±0.06	89.89% ±0.52 1.54% | ✓	76.35% ±0.49	90.04% ±0.48 9* | 18.9% | ✓	88.20% ±0.10	90.10% ±0.48 18 | 0.05% | ✓	93.89% ±1.76
shuttle	99.83% ±0.05 79.6% | ✗	81.91% ±0.0	99.78% ±0.03 15.62% | ✗	60.8% ±0.04	Error 2.79%		Error 19.00%		99.36% ±0.03 6* | 0.02% | ✓	56.97% ±1.28	99.36% ±0.03 7 | 0.01% | ✓	51.67% ±4.56
wine	98.03% ±1.66 9.6% | ✓	90.2% ±0.0	95.96% ±2.81 13.48% | ✓	95.83% ±0.00	98.3% ±1.45 4.49% | ✓	88.12% ±1.01	98.11% ±1.54 2.81% | ✓	70.67% ±1.41	96.94% ±2.37 2* | 8.4% | ✓	95.56% ±0.0	98.14% ±1.35 3 | 1.12% | ✓	91.67% ±0.00
winequality	64.42% ±2.17 25.2% | ✓	51.85% ±0.61	63.99% ±1.95 16.45% | ✓	46.05% ±0.57	63.48% ±1.85 2.88% | ✓	35.72% ±1.16	63.66% ±1.69 2.94% | ✓	41.31% ±1.02	63.08% ±2.15 7* | 1.06% | ✓	0.00% ±0.00	No change	

Table 4: Detection methods comparison on real data sets

The way the **tolerance limit** equation is established in Section 4.2 is strongly related to the maximum percentage of inlier that is agreed to be removed (i.e. 20%). A different hypothesis would change this equation and the resulting detection. Plus, for the preservation of data sets, the algorithm shifts these limit when the maximum number of outliers is reached or when a behavioral value is completely removed. In practical, in Table 4, the limits of 9 detections has been shifted: 7 cause the maximum number of outliers and 2 cause the integrity of behavioral values. Despite this precaution, the **tolerance limit** doesn't always provide the optimal solution. As Table 4 shows, there is a much better solution in 7 over 16 cases. Thus, the **tolerance limit** suit better as a meta-parameter.

6 Conclusion

This paper presented a parameterized method of outlier detection based on the quantile principle and a calculation to determine the parameters to be used. This method was compared with a state-of-the-art method (**IQR**, **Isolation**

Forest, **SVM** and **LOF**) on 278 synthetic data sets and 16 real data sets. It was clearly shown that the **Hino** approach was the more efficient. The computational efficiency of meta-parameters for **Hino** has been discussed and can be studied to adapt it to take into account the context of data sets. Finally, this approach can be easily integrated into standard machine learning library (as scikit-Learn or Tensor Flow) to help preparation of data sets.

References

1. Aggarwal, C.C.: Outlier Analysis. Springer International Publishing, Cham, 2 edn. (2017). https://doi.org/10.1007/978-3-319-47578-3
2. Amer, M., Goldstein, M., Abdennadher, S.: Enhancing one-class support vector machines for unsupervised anomaly detection. In: Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description. pp. 8–15. ODD '13, Association for Computing Machinery, New York, NY, USA (Aug 2013). https://doi.org/10.1145/2500853.2500857
3. Barnett, V., Lewis, T., others: Outliers in statistical data, vol. 3. Wiley New York (1994). https://doi.org/10.1057/jors.1995.142
4. Breunig, M., Kriegel, H.P., Ng, R., Sander, J.: LOF: Identifying Density-Based Local Outliers. vol. 29, pp. 93–104 (Jun 2000). https://doi.org/10.1145/342009.335388
5. Dekking, F.M., Kraaikamp, C., Lopuhaä, H.P., Meester, L.E.: Exploratory data analysis: numerical summaries. In: Dekking, F.M., Kraaikamp, C., Lopuhaä, H.P., Meester, L.E. (eds.) A Modern Introduction to Probability and Statistics: Understanding Why and How, pp. 231–243. Springer, London (2005). https://doi.org/10.1007/1-84628-168-7_16
6. Guyon, I.: Design of experiments for the NIPS 2003 variable selection benchmark. In: undefined (2003). https://doi.org/10.1007/978-3-540-35488-8_10, https://www.semanticscholar.org/paper/Design-of-experiments-for-the-NIPS-2003-variable-Guyon/b979fa88ca448fb08633f961131f45214b1cf109
7. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-Based Anomaly Detection. ACM Transactions on Knowledge Discovery from Data **6**(1), 3:1–3:39 (Mar 2012). https://doi.org/10.1145/2133360.2133363
8. Sadik, S., Gruenwald, L.: Research issues in outlier detection for data streams. ACM SIGKDD Explorations Newsletter **15**(1), 33–40 (Mar 2014). https://doi.org/10.1145/2594473.2594479
9. Souiden, I., Omri, M.N., Brahmi, Z.: A survey of outlier detection in high dimensional data streams. Computer Science Review **44**, 100463 (May 2022). https://doi.org/10.1016/j.cosrev.2022.100463, https://www.sciencedirect.com/science/article/pii/S1574013722000107
10. Thung, F., Wang, S., Lo, D., Jiang, L.: An Empirical Study of Bugs in Machine Learning Systems. In: 2012 IEEE 23rd International Symposium on Software Reliability Engineering. pp. 271–280 (Nov 2012). https://doi.org/10.1109/ISSRE.2012.22, iSSN: 2332-6549
11. Vinutha, H.P., Poornima, B., Sagar, B.M.: Detection of Outliers Using Interquartile Range Technique from Intrusion Dataset. In: Satapathy, S.C., Tavares, J.M.R., Bhateja, V., Mohanty, J.R. (eds.) Information and Decision Sciences. pp. 511–518. Springer, Singapore (2018). https://doi.org/10.1007/978-981-10-7563-6_53