Outlier Detection Based on Leave-One-Out Density Using Binary Decision Diagrams

Takuro Kutsuna[1,2] and Akihiro Yamamoto[2]

[1] Toyota Central R&D Labs. Inc., Nagakute, Aichi, 480-1192, Japan
kutsuna@mosk.tytlabs.co.jp
[2] Kyoto University, Sakyo-ku, Kyoto, 606-8501, Japan

Abstract. We propose a novel method for detecting outliers based on the leave-one-out density. The leave-one-out density of a datum is defined as a ratio of the number of data inside a region to the volume of the region after the datum is removed from an original data set. We propose an efficient algorithm that evaluates the leave-one-out density of each datum on a set of regions around the datum by using binary decision diagrams. The time complexity of the proposed method is near linear with respect to the size of a data set, while the outlier detection accuracy is still comparable to other methods. Experimental results show the usefulness of the proposed method.

Keywords: Outlier detection, binary decision diagram.

1 Introduction

In this paper, we propose a novel and efficient method for outlier detection, which is an important task in data mining and has been applied to many problems such as fraud detection, intrusion detection, data cleaning and so on [7]. The goal of outlier detection is to find an unusual datum (*outlier*) from a given data set. Although many kinds of notion have been proposed to define an outlier, we consider a datum as an outlier if the *leave-one-out density* is lower than a given threshold for a set of regions around the datum. The leave-one-out density is a ratio of the number of data inside a region to the volume of the region, in which the focused datum is removed from the original data set. Generally, a leave-one-out like method is time consuming because a learning procedure is repeated N-times, where N is the cardinality of a data set. However, the proposed method enables us to evaluate the leave-one-out density efficiently without repeating a learning procedure N-times.

We employ the initial region method proposed in [10], in which a data set is encoded into a Boolean formula and represented as a binary decision diagram. Although a one-class classifier is proposed based on the initial region method in [10], it is not applicable to outlier detection, because the classifier is estimated as an over-approximation of the data set and never classify a datum in the data set as an outlier. We extend the work of [10] to outlier detection by introducing the notion of leave-one-out density and developing an efficient algorithm to evaluate it.

V.S. Tseng et al. (Eds.): PAKDD 2014, Part II, LNAI 8444, pp. 486–497, 2014.

The proposed method is compared to other well-known outlier detection methods, the one-class support vector machine [13] and the local outlier factor [4], with both synthetic data sets and realistic data sets. The experimental results indicate that the computation time of the proposed method is shorter than those of the other methods, keeping the outlier detection accuracy comparable to the other methods.

The outlier detection problem addressed in this paper is formally defined in Sect. 2. We review the initial region method in Sect. 3. In Sect. 4, the outline of the proposed method is first stated, and then, its efficient implementation based on binary decision diagrams is proposed. Section 5 shows experimental results. In Sect. 6, we conclude this paper by discussing limitations and future work.

1.1 Related Work

Kutsuna [10] proposed a one-class classifier that over-approximates a training data set. The approximation is done quite efficiently by manipulating a binary decision diagram that is obtained by encoding the training data set. The situation considered in [10] is that both a training data set and a test data set are given: A classifier is first learned from the training dataset, then the test data set is classified by the classifier. It may seem that we can detect outliers within a data set by using the data set as both the training data set and the test data set simultaneously. However, no datum is detected as an outlier in such a setting, because the classifier is estimated as an over-approximation of the training data set. Therefore, the method in [10] cannot be applied to outlier detection directly.

Schölkopf et al. [13] extended the support vector machine (SVM) to outlier detection, which was originally invented for binary classification. Their method estimates a hyperplane that separates the origin and a data set with maximum margin, in which the hyperplane can be nonlinear by introducing kernel functions. The data that are classified to the origin side are detected as outliers. The SVM has an advantage that various nonlinear hyperplanes are estimated by changing kernel parameters. Some heuristics are proposed to tune kernel parameters, such as [6].

Breunig et al. [4] proposed the local outlier factor (LOF) that is calculated based on the distance to the k-nearest neighbor of each datum and has an advantage that it can detect local outliers, that is, data that are outlying relative to their local neighborhoods. The LOF has been shown to perform very well in realistic problems [12]. An efficient calculation of the k-nearest neighbors is essential in the LOF. Some techniques are proposed to accelerate the k-nearest neighbors calculation, such as [2].

2 Problem Setting

Let D be a data set that includes N data. The i-th datum in D is denoted by $x^{(i)} \in \mathbb{R}^u$ $(i = 1, \ldots, N)$. We assume that there is no missing value in D and all the data in D are unlabeled. In this paper, we regard a datum as an outlier

if the *leave-one-out density* is lower than a threshold for a set of regions around the datum. The leave-one-out density ρ_{LOO} of the i-th datum is defined as:

$$\rho_{\mathrm{LOO}}(i, \mathcal{D}) = \frac{\#\left(\left\{x^{(j)} \;\middle|\; x^{(j)} \in \mathcal{D},\ j = 1, \ldots, N,\ j \neq i\right\}\right)}{vol(\mathcal{D})}$$

where $\mathcal{D}$ is a u-dimensional region such that $x^{(i)} \in \mathcal{D}$, $\#(\cdot)$ is the cardinality of a set and $vol(\cdot)$ is the volume of the region. The *outlier score* S of the i-th datum is defined as:

$$S(i) = \max_{\mathcal{D} \in \tilde{\mathcal{D}}(i)} \rho_{\mathrm{LOO}}(i, \mathcal{D})$$

where $\tilde{\mathcal{D}}(i)$ is a set of regions around $x^{(i)}$ defined as:

$$\tilde{\mathcal{D}}(i) = \left\{ u\text{-dimensional region } \mathcal{D} \;\middle|\; x^{(i)} \in \mathcal{D} \right\}.$$

A datum is detected as an *outlier* if the outlier score of the datum is less than a given threshold. Note that $\tilde{\mathcal{D}}(i)$ is not the set of all possible regions, but rather a fixed family, which is defined in Sect. 4.1. In the following sections, we propose an efficient algorithm that enables us to evaluate the outlier score in near linear time with respect to N.

3 Preliminaries

In this section, we briefly review the initial region method [10] and define notations. Let $\mathcal{H}$ be a u-dimensional hypercube defined as $\mathcal{H} = [0, 2^m)^u$, where m is an arbitrary positive integer. And let σ be an example normalizer such that $\sigma(x^{(i)}) \in \mathcal{H}$ for every $x^{(i)}$ in D. An example of σ is given in [10] as a simple scaling function. The *neighborhood function* ν, which is defined as $\nu(z) := [\lfloor z_1 \rfloor, \lfloor z_1 \rfloor + 1) \times \ldots \times [\lfloor z_u \rfloor, \lfloor z_u \rfloor + 1)$, returns a u-dimensional unit hypercube that subsumes $z = \sigma(x)$, where $\lfloor \cdot \rfloor$ is the floor function. The *initial region* G is a u-dimensional region inside $\mathcal{H}$ that subsumes all the projected data, which is defined as:

$$G = \bigcup_{j=1,\ldots,N} \nu\left(z^{(j)}\right).$$

For example, we consider a data set in $\mathbb{R}^2$. We set $m = 3$, then the data set is projected into $\mathcal{H} = [0, 2^3)^2$ by σ. The projected data z are shown as x-marks and the initial region G is shown as the gray region in Fig. 1(a).

The initial region G is expressed as a Boolean function by using the coding function `CodeZ`. `CodeZ` first truncates each element of z to an integer, and then, code them into a logical formula in the manner of an unsigned-integer-type coding. The set of Boolean variables $\mathbb{B} := \{b_{ij} \mid i = 1, \ldots, u;\ j = 1, \ldots, m\}$ is used to code z, where b_{i1} and b_{im} represent the most and the least significant bit of the i-th element of z, respectively. The *initial Boolean function* F is given as a disjunction of logical formulas such as:

$$F = \bigvee_{i=1,\ldots,N} \texttt{CodeZ}(z^{(i)}).$$

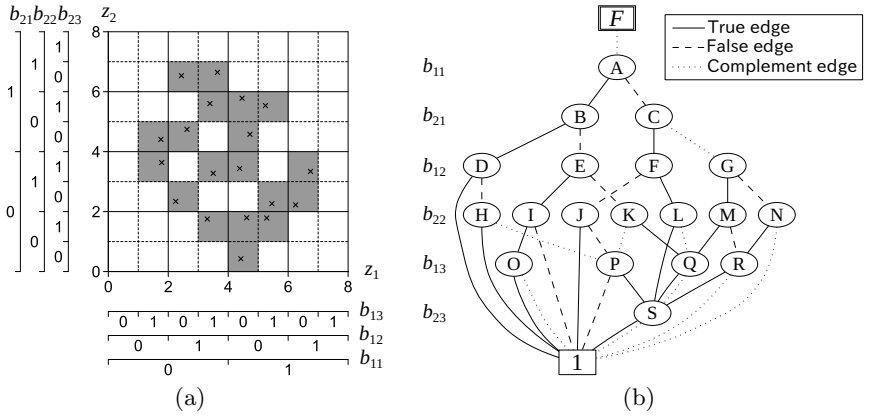

Fig. 1. X-marks mean a normalized data set and the gray region is the initial region G (*left*). A BDD that represents the initial Boolean function F (*right*).

It is shown that the initial Boolean function F is informational equivalent of the initial region G [11]. Let R be a function that decodes a Boolean function defined on $\mathbb{B}$ into the corresponding u-dimensional region. In particular, $R(1) = \mathcal{H}$ and $R(F) = G$. Binary decision diagrams (BDDs) [5] are used to efficiently construct and represent the initial Boolean function F. The order of Boolean variables is set to hold $[b_{11}, \ldots, b_{u1}] \prec \cdots \prec [b_{1m}, \ldots, b_{um}]$, where the variables inside the square brackets can be in arbitrary order, on constructing F as a BDD. For example, Fig. 1(b) shows a BDD that represents the initial Boolean formula F that is obtained from the data set in Fig. 1(a). In Fig. 1(b), square nodes, ellipsoidal nodes and double-squared nodes are referred to as terminal nodes, variable nodes and function nodes, respectively. Boolean variables that variable nodes represent are on the left side. Solid lines, dashed lines and dotted lines are true edges, false edges and complement edges, respectively. A path from a function node to the terminal 1 corresponds to a conjunction of literals. If the path contains an even number of complement edges, the conjunction is included in the function.

4 Proposed Method

4.1 Outline

In the proposed method, we calculate the leave-one-out density based on the normalized data $z^{(i)} = \sigma\left(x^{(i)}\right)$ as follows:

$$\rho_{\mathrm{LOO}}\left(i, \mathcal{C}\right) = \frac{\#\left(\left\{z^{(j)} \;\middle|\; z^{(j)} \in \mathcal{C},\ j = 1, \ldots, N,\ j \neq i\right\}\right)}{vol\left(\mathcal{C}\right)} \tag{1}$$

where $\mathcal{C}$ is a region such that $z^{(i)} \in \mathcal{C}$. The outlier score is given as:

$$S\left(i\right) = \max_{\mathcal{C} \in \tilde{\mathcal{C}}(i)} \rho_{\mathrm{LOO}}\left(i, \mathcal{C}\right) \tag{2}$$

where $\tilde{\mathcal{C}}(i)$ is a set of hypercubes defined as:

$$\begin{aligned} &\tilde{\mathcal{C}}(i) = \{R(\mathcal{L}_l(i)) \mid l = 0, \ldots, m\}, \\ &\mathcal{L}_0(i) = 1, \quad \mathcal{L}_l(i) = \bigwedge_{i'=1,\ldots,u} \bigwedge_{j'=1,\ldots,l} \tilde{b}_{i'j'} \quad (l = 1, \ldots, m) \end{aligned} \tag{3}$$

where $\tilde{b}_{i'j'}$ are assignments of Boolean variables that is obtained by coding $z^{(i)}$ with `CodeZ`. For example, the datum in the region $[5,6) \times [2,3)$ in Fig. 1(a) is coded as $(\tilde{b}_{11}, \tilde{b}_{21}, \tilde{b}_{12}, \tilde{b}_{22}, \tilde{b}_{13}, \tilde{b}_{23}) = (1,0,0,1,1,0)$. From (3), the set $\tilde{\mathcal{C}}(i)$ for this datum is derived as

$$\tilde{\mathcal{C}}(i) = \{[0,8) \times [0,8),\ [4,8) \times [0,4),\ [4,6) \times [2,4),\ [5,6) \times [2,3)\}$$

which are shown as bold squares in Fig. 2.

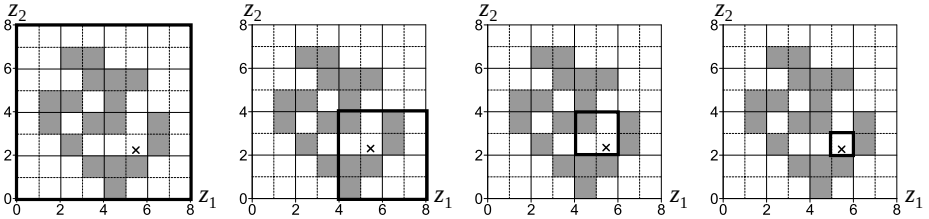

Fig. 2. Bold squares show $\tilde{\mathcal{C}}(i)$ for the data shown as the x-mark. The leave-one-out density ρ_{LOO} of the datum are $\frac{18}{64}$, $\frac{6}{16}$, $\frac{1}{4}$ and $\frac{0}{1}$ for each hypercube. The outlier score of the datum is evaluated as $\max\left(\frac{18}{64}, \frac{6}{16}, \frac{1}{4}, \frac{0}{1}\right) \approx 0.38$.

We assume that there is no duplicate in D, that is, at least one of the attributes has a different value for $x^{(i)}$ and $x^{(j)}$ if $i \neq j$. Then, we can construct the initial region G so that each datum in D is allocated in a distinct unit hypercube by setting m large enough and using an appropriate example normalizer. In this case, the number of data inside a region can be calculated as the volume of the region unless the boundary of the region goes across a unit hypercube in which a datum exists. Therefore, the leave-one-out density defined as (1) can be calculated based on the initial region as follows for $\mathcal{C} \in \tilde{\mathcal{C}}(i)$:

$$\rho_{\mathrm{LOO}}(i, \mathcal{C}) = \frac{vol\left(G_{\mathrm{LOO}}(i) \cap \mathcal{C}\right)}{vol\left(\mathcal{C}\right)} \tag{4}$$

where G_{LOO} is the leave-one-out region defined as:

$$G_{\mathrm{LOO}}(i) = G \setminus \nu\left(z^{(i)}\right). \tag{5}$$

For example, Fig. 2 illustrates the calculation of the leave-one-out density for the datum shown as the x-mark.

4.2 BDD Based Implementation

The Number of Minterms Calculation. A *minterm* is a conjunction of Boolean variables in which each Boolean variable in the domain appears once. Let $\#_A$ be a function that returns the number of minterms of a Boolean function on the assumption that the domain of the function is A. For example, $\#_a(1) = 2$ and $\#_{\{a,b\}}(1) = 4$ where a and b are Boolean variables.

Lemma 1. For a Boolean formula A that is defined on $\mathbb{B}$, it holds that:

$$vol\left(R\left(A\right)\right) = \#_{\mathbb{B}}\left(A\right).$$

Proof. Since a minterm represents a unit hypercube in the initial region method, the number of minterms equals to the volume of the region that A represents.

Let $\mathcal{N}_\alpha^+$ be a Boolean function that node α in a BDD represents being connected with a non-complement edge. Also, let $\mathcal{N}_\alpha^-$ be a Boolean function being connected with a complement edge. Both of $\mathcal{N}_\alpha^+$ and $\mathcal{N}_\alpha^-$ represent regions inside $\mathcal{H}$. For example, Fig. 3 shows $R\left(\mathcal{N}_{\mathrm{E}}^-\right)$ and $R\left(\mathcal{N}_{\mathrm{G}}^+\right)$ where E and G are nodes in Fig. 1(b). It is possible to efficiently calculate the number of minterms of $\mathcal{N}_\alpha^+$ and $\mathcal{N}_\alpha^-$ for each node α in a BDD in a depth-first manner [15]. For example, Table 1 shows the number of minterms of $\mathcal{N}_\alpha^+$ and $\mathcal{N}_\alpha^-$ for each node in Fig. 1(b). We can see that $\#_{\mathbb{B}}\left(\mathcal{N}_{\mathrm{E}}^-\right)$ and $\#_{\mathbb{B}}\left(\mathcal{N}_{\mathrm{G}}^+\right)$ are equal to the volumes of regions which are shown in Fig. 3.

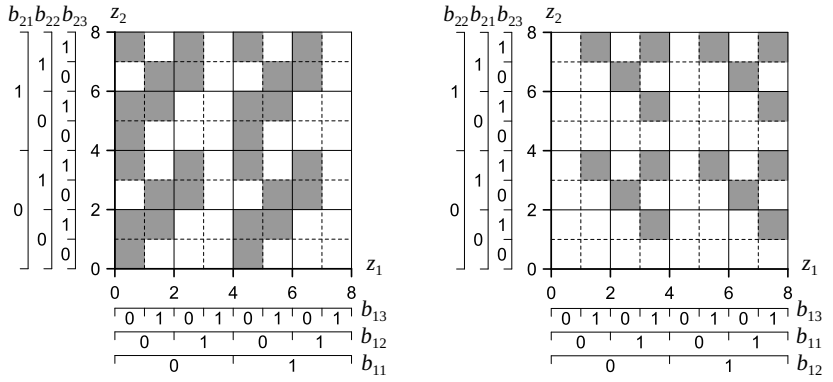

Fig. 3. Examples of regions that BDD nodes represent: $R\left(\mathcal{N}_{\mathrm{E}}^-\right)$ (*left*) and $R\left(\mathcal{N}_{\mathrm{G}}^+\right)$ (*right*) where E and G are nodes in Fig. 1(b)

The Leave-One-Out Density Calculation. Because of the fact that $\nu\left(z^{(i)}\right) \subseteq G$ and $\nu\left(z^{(i)}\right) \subseteq \mathcal{C} \in \tilde{\mathcal{C}}(i)$, it is derived from (4) and (5) that the following equation holds for $\mathcal{C} \in \tilde{\mathcal{C}}(i)$:

$$\rho_{\mathrm{LOO}}\left(i, \mathcal{C}\right) = \frac{vol\left(G \cap \mathcal{C}\right) - 1}{vol\left(\mathcal{C}\right)}. \tag{6}$$

Table 1. The number of minterms of $\mathcal{N}_\alpha^+$ and $\mathcal{N}_\alpha^-$ for each node α in Fig. 1(b)

α	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	1
$\#_{\mathbb{B}}\left(\mathcal{N}_\alpha^+\right)$	45	44	46	52	36	44	16	40	48	56	24	32	24	8	32	48	32	16	32	64
$\#_{\mathbb{B}}\left(\mathcal{N}_\alpha^-\right)$	19	20	18	12	28	20	48	24	16	8	40	32	40	56	32	16	32	48	32	0

By replacing $\mathcal{C}$ in (6) with $R\left(\mathcal{L}_l\left(i\right)\right)$, the following equation is derived.

$$\rho_{\mathrm{LOO}}\left(i, R\left(\mathcal{L}_l\left(i\right)\right)\right) = \frac{vol\left(G \cap R\left(\mathcal{L}_l\left(i\right)\right)\right) - 1}{vol\left(R\left(\mathcal{L}_l\left(i\right)\right)\right)} = \frac{vol\left(R\left(F \wedge \mathcal{L}_l\left(i\right)\right)\right) - 1}{vol\left(R\left(\mathcal{L}_l\left(i\right)\right)\right)} \tag{7}$$

From Lemma 1, (7) is transformed as follows:

$$\rho_{\mathrm{LOO}}\left(i, R\left(\mathcal{L}_l\left(i\right)\right)\right) = \frac{\#_{\mathbb{B}}\left(F \wedge \mathcal{L}_l\left(i\right)\right) - 1}{\#_{\mathbb{B}}\left(\mathcal{L}_l\left(i\right)\right)} = \frac{\#_{\mathbb{B}}\left(F \wedge \mathcal{L}_l\left(i\right)\right) - 1}{2^{(m-l)u}} \tag{8}$$

Lemma 2. In a BDD that represents F, let $\alpha_{i,l}$ be the node that can be reached from the function node through the path defined by $\mathcal{L}_l\left(i\right)$. Let c be the number of complement edges on the path. Then, it holds that:

$$\#_{\mathbb{B}}\left(F \wedge \mathcal{L}_l\left(i\right)\right) = \begin{cases} \#_{\mathbb{B}}\left(\mathcal{N}_{\alpha_{i,l}}^+\right)/2^{lu} & \text{if } c \text{ is even,} \\ \#_{\mathbb{B}}\left(\mathcal{N}_{\alpha_{i,l}}^-\right)/2^{lu} & \text{if } c \text{ is odd.} \end{cases}$$

Proof. $F \wedge \mathcal{L}_l\left(i\right)$ means that $l \times u$ Boolean variables that appear in $\mathcal{L}_l\left(i\right)$ are fixed to specific values in F. On the other hand, $\mathcal{N}_{\alpha_{i,l}}^+$ ($\mathcal{N}_{\alpha_{i,l}}^-$) means F with $l \times u$ Boolean variables in $\mathcal{L}_l\left(i\right)$ smoothed[1] if c is even (odd). Therefore, the number of minterms of $\mathcal{N}_{\alpha_{i,l}}^+$ ($\mathcal{N}_{\alpha_{i,l}}^-$) is 2^{lu} times larger than that of $F \wedge \mathcal{L}_l\left(i\right)$.

Theorem 1. The leave-one-out density ρ_{LOO} is evaluated based on a BDD that represents the initial Boolean function F as follows:

$$\rho_{\mathrm{LOO}}\left(i, R\left(\mathcal{L}_l\left(i\right)\right)\right) = \begin{cases} \left(\#_{\mathbb{B}}\left(\mathcal{N}_{\alpha_{i,l}}^+\right) - 2^{lu}\right)/2^{mu} & \text{if } c \text{ is even,} \\ \left(\#_{\mathbb{B}}\left(\mathcal{N}_{\alpha_{i,l}}^-\right) - 2^{lu}\right)/2^{mu} & \text{if } c \text{ is odd.} \end{cases}$$

Proof. It follows from (8) and Lemma 2 immediately.

It is worth mentioning that we can evaluate the leave-one-out density from the initial Boolean function F straightforwardly without any leave-one-out operation by using Theorem 1. For example, we consider a data set in Fig. 1(a) and the datum in $[5, 6) \times [2, 3)$. The path of the datum is $F \bullet \mathrm{A} \circ \mathrm{B} \circ \mathrm{E} \circ \mathrm{K} \circ \mathrm{Q} \circ \mathrm{S} \bullet 1$ in Fig. 1(b), where $\circ$ and $\bullet$ mean non-complement and complement edges, respectively. The leave-one-out density of the datum is calculated from Theorem 1 and Table 1 as follows:

$$\left(\#_{\mathbb{B}}\left(\mathcal{N}_{\mathrm{A}}^-\right) - 2^0\right)/2^6 = 18/64, \quad \left(\#_{\mathbb{B}}\left(\mathcal{N}_{\mathrm{E}}^-\right) - 2^2\right)/2^6 = 6/16,$$
$$\left(\#_{\mathbb{B}}\left(\mathcal{N}_{\mathrm{Q}}^-\right) - 2^4\right)/2^6 = 1/4, \quad \left(\#_{\mathbb{B}}\left(\mathcal{N}_1^+\right) - 2^6\right)/2^6 = 0.$$

We can see that these values are equal to those in Fig. 2.

[1] Smoothing a Boolean function f with respect to x means $(f \wedge x) \vee (f \wedge \overline{x})$.

4.3 The Proposed Algorithm and Computational Complexity

We propose Algorithm 1 that calculates the outlier score of each datum in D.In Algorithm 1, the time complexity of constructing the initial Boolean function F is approximately $O(MN)$, where M is the number of nodes of the created BDD, because logical operations between BDDs are practically almost linear to the size of the BDDs [3]. The size of the created BDD depends on the characteristics of the data set and can be exponentially large in the worst case, but, it is compact for realistic data sets used in our experiments. The time complexity of calculating the number of minterms is $O(M)$ as mentioned in Sect. 4.2. The time complexity of calculating the outlier score is $O(muN)$ because the depth of the BDD is mu. Consequently, the time complexity of Algorithm 1 is $O((M+mu)N)$. Therefore, the proposed method can deal with a large data set efficiently unless the number of Boolean variables and the created BDD are intractably huge.

Algorithm 1: The outlier score calculation.

Input: A data set D.
Output: The outlier score S of each datum in D.
1 Construct the initial Boolean function F as a BDD.;
2 Calculate the number of minterms of each node of the BDD.;
3 **for** $i \leftarrow 1$ **to** N **do**
4 Search the path that `CodeZ(`$z^{(i)}$`)` represents in the BDD and evaluate $\rho_{\mathrm{LOO}}\left(i, R\left(\mathcal{L}_l\left(i\right)\right)\right)$ for $l = 0, \ldots, m$ by using Theorem 1.;
5 $S\left(i\right) \leftarrow \max_{l=0,\ldots,m} \rho_{\mathrm{LOO}}\left(i, R\left(\mathcal{L}_l\left(i\right)\right)\right)$;

4.4 Dealing with Categorical Attributes

The proposed method can be extended in order to deal with a data set that consists of both continuous attributes and categorical attributes. Let $y^{(i)}$ be a vector of categorical attributes of the i-th datum. We extend the leave-one-out density defined as (1) as follows:

$$\rho_{\mathrm{LOO}}\left(i, \mathcal{C}\right) = \frac{\#\left(\left\{z^{(j)} \;\middle|\; z^{(j)} \in \mathcal{C},\ y^{(j)} = y^{(i)},\ j = 1, \ldots, N,\ j \neq i\right\}\right)}{vol\left(\mathcal{C}\right)}$$

Then, the outlier score defined as (2) can be evaluated very efficiently in the same manner as mentioned in the previous sections. The details are skipped because of the page limit.

5 Experimental Results

We compare the proposed method with existing methods, the one-class support vector machine (OCSVM) and the local outlier factor (LOF). The proposed method is referred to as ODBDD. We implemented ODBDD as a C program

with the help of CUDD [14]. The `ksvm` function in the `kernlab` package [9] is used for OCSVM and the `lofactor` function in the `DMwR` package [16] is used for LOF. The parameter m that defines the size of the hypercube $\mathcal{H}$ is fixed to $m = 16$ in ODBDD. In OCSVM, the Gaussian kernel is used and the kernel parameter γ is set to one of the 10%, 50% and 90% quantiles of the distance between samples [6], which are referred to as $\gamma_{0.1}$, $\gamma_{0.5}$ and $\gamma_{0.9}$, respectively. The parameter ν is fixed to $\nu = 0.1$ in OCSVM. In LOF, the number of neighbors is set to either $k = 10$ or $k = 50$. In OCSVM and LOF, continuous attributes are scaled and categorical attributes are coded by using dummy variables. The accuracy is evaluated in terms of the *area under an ROC curve* (*AUC*) [8]. The experiment was performed on a Microsoft Windows 7 machine with an Intel Core i7 CPU (3.20 GHz) and 64 GB RAM.

5.1 Evaluation with Synthetic Data Sets

Ten data set is a synthetic data set that consists of two continuous attributes and no categorical attribute. The 95 % data of Ten data set distributes inside the shape "10" randomly. The remaining 5 % data distributes outside randomly, which are regarded as outliers. The number of data is set to 10^3, 10^4, 10^5 or 10^6. An example of Ten-10^3 data set is shown in the left side of Fig. 4.

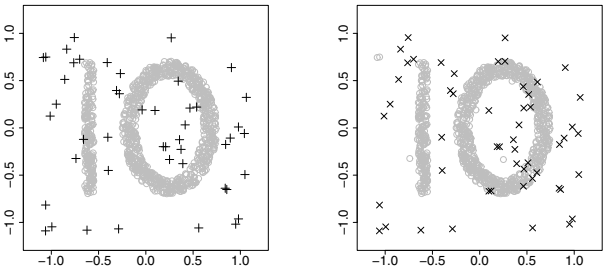

Fig. 4. An example of Ten-10^3 data set in which true outliers are shown as cross-marks (*left*). X-marks mean outliers detected by the proposed method (*right*).

Table 2 shows the mean and the standard deviation of computation time over 10 random trials for Ten data set. We can see that the computation time of the proposed method increases moderately compared to the other methods.

Table 3 shows the mean and the standard deviation of AUC values over 10 random trials for Ten data set. From Table 3, the accuracy of the proposed method is comparable to those of the other methods. For example, the outliers detected by ODBDD is shown in the right side of Fig. 4, in which the top 5 % of the data are detected as outliers based on the outlier score of ODBDD.

Table 2. The mean and the std. dev. of comp. time for Ten data set (sec)

		OCSVM ($\nu = 0.1$)			LOF	
Data set	ODBDD	$\gamma = \gamma_{0.1}$	$\gamma = \gamma_{0.5}$	$\gamma = \gamma_{0.9}$	$k = 10$	$k = 50$
Ten-10^3	0.1 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	0.5 (0.0)
Ten-10^4	0.3 (0.0)	3.0 (0.1)	2.7 (0.1)	2.6 (0.1)	22.6 (0.1)	24.8 (0.3)
Ten-10^5	2.3 (0.0)	294.5 (5.7)	267.3 (4.0)	250.6 (4.0)	2942.0 (38.4)	3052.1 (34.3)
Ten-10^6	30.1 (0.9)	timeout	timeout	timeout	timeout	timeout

The timeout limit is 3600 seconds.

Table 3. The mean and the std. dev. of AUC values for Ten data set

		OCSVM ($\nu = 0.1$)			LOF	
Data set	ODBDD	$\gamma = \gamma_{0.1}$	$\gamma = \gamma_{0.5}$	$\gamma = \gamma_{0.9}$	$k = 10$	$k = 50$
Ten-10^3	0.97 (0.02)	0.79 (0.02)	0.80 (0.03)	0.95 (0.01)	0.99 (0.01)	0.97 (0.02)
Ten-10^4	0.99 (0.00)	0.80 (0.02)	0.82 (0.02)	0.97 (0.00)	0.82 (0.02)	1.00 (0.00)
Ten-10^5	0.99 (0.00)	0.80 (0.00)	0.82 (0.00)	0.97 (0.00)	0.60 (0.01)	0.77 (0.01)
Ten-10^6	1.00 (0.00)	timeout	timeout	timeout	timeout	timeout

5.2 Evaluation with Realistic Data Sets

We use seven data sets from UCI machine learning repository [1] as shown in Table 4, where N_a is the size of the original data set. All of these data sets are originally arranged for the classification task. In order to apply these data sets to the evaluation of outlier detection algorithms, we randomly picked out data from each data set to generate a new data set as follows: 1) Pick out all the data whose class are C_m where C_m is the class of the maximum data size. Let N_m be the number of data that belong to class C_m. 2) Pick out $N_o = round(0.01N_m)$ data randomly from the remaining data set, which are regarded as outliers.

Table 4. An overview of UCI data sets used in the experiment

Data set	N_a	N_m	N_o	# of cate. attr.	# of cont. attr.
abalone	4177	689	7	1	7
adult	32561	24720	248	8	6
bank	45211	39922	400	9	7
ionosphere	351	225	3	0	34
magic	19020	12332	124	0	10
shuttle	43500	34108	342	0	8
yeast	1484	463	5	0	8

Table 5 shows the mean and the standard deviation of computation time over 10 random trials for UCI data sets. The computation time varies drastically depending on the size of the data set in both OCSVM and LOF. On the other hand, ODBDD works quite fast for all of the data sets.

Table 5. The mean and the std. dev. of comp. time for UCI data sets (sec)

		OCSVM ($\nu = 0.1$)			LOF	
Data set	ODBDD	$\gamma = \gamma_{0.1}$	$\gamma = \gamma_{0.5}$	$\gamma = \gamma_{0.9}$	$k = 10$	$k = 50$
abalone	0.2 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.3 (0.0)	0.4 (0.0)
adult	3.1 (0.1)	26.5 (0.2)	24.5 (0.1)	24.2 (0.1)	2688.9 (320.0)	2935.3 (459.5)
bank	6.0 (0.0)	66.2 (0.4)	61.5 (0.5)	60.0 (0.3)	2784.8 (672.6)	2328.6 (155.7)
ionosphere	0.2 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.1 (0.0)	0.1 (0.0)
magic	2.4 (0.0)	4.6 (0.1)	4.1 (0.1)	4.0 (0.1)	52.3 (0.7)	56.6 (0.1)
shuttle	2.9 (0.1)	35.6 (0.4)	32.3 (0.3)	31.0 (0.3)	398.4 (10.9)	424.4 (12.5)
yeast	0.2 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.2 (0.0)	0.2 (0.0)

Table 6 shows the mean and the standard deviation of AUC values over 10 random trials. Although some results of ODBDD are not as good as the best result of the other methods, ODBDD achieves similar accuracy to the others.

Table 6. The mean and the std. dev. of AUC values for UCI data sets

		OCSVM ($\nu = 0.1$)			LOF	
Data set	ODBDD	$\gamma = \gamma_{0.1}$	$\gamma = \gamma_{0.5}$	$\gamma = \gamma_{0.9}$	$k = 10$	$k = 50$
abalone	0.59 (0.13)	0.59 (0.08)	0.57 (0.08)	0.58 (0.08)	0.61 (0.11)	0.66 (0.15)
adult	0.59 (0.02)	0.62 (0.01)	0.62 (0.01)	0.61 (0.01)	0.46 (0.02)	0.52 (0.03)
bank	0.61 (0.02)	0.62 (0.01)	0.62 (0.01)	0.62 (0.01)	0.62 (0.01)	0.69 (0.01)
ionosphere	0.87 (0.12)	0.79 (0.08)	0.87 (0.09)	0.64 (0.13)	0.94 (0.07)	0.95 (0.08)
magic	0.79 (0.02)	0.64 (0.03)	0.67 (0.03)	0.71 (0.03)	0.85 (0.03)	0.83 (0.03)
shuttle	0.93 (0.01)	0.78 (0.01)	0.89 (0.01)	0.93 (0.00)	0.44 (0.02)	0.69 (0.03)
yeast	0.65 (0.15)	0.66 (0.11)	0.64 (0.10)	0.55 (0.06)	0.69 (0.15)	0.73 (0.12)

6 Conclusions

In this work, we proposed a novel approach for outlier detection. A score of being an outlier is defined based on the leave-one-out density, which is evaluated very efficiently by processing a binary decision diagram that represents a data set in a logical formula. The proposed method can deal with a large data set efficiently, because the time complexity is near linear unless the created BDD gets intractably huge.

This work can be extended in several ways. First, the region set $\tilde{\mathcal{D}}(i)$ is enriched by using various normalizers. Then, the accuracy of outlier detection is expected to improve. A simple approach to generate various normalizers is to incorporate a random rotation into a normalizer. Another extension is to employ nonlinear normalizers. If we use nonlinear normalizers, a hypercube in a projected space corresponds to a nonlinear region in the original space, which may lead to more precise outlier detection.

The proposed method may suffer from the curse of dimensionality when a data set has many attributes and the number of data is not enough, because the leave-one-out density is zero in almost every subregion of the whole hypercube

in such a situation. A simple solution is to embed some dimension reduction method into a normalizer.

Although we have conducted experiments with several data sets mainly to compare the proposed method to other methods, it is necessary to apply the proposed method to other real world problems in order to examine the practical usefulness of the proposed method and reveal problems to tackle in future.

References

1. Bache, K., Lichman, M.: UCI machine learning repository (2013), http://archive.ics.uci.edu/ml
2. Beckmann, N., Kriegel, H., Schneider, R., Seeger, B.: The R*-tree: an efficient and robust access method for points and rectangles. SIGMOD Rec. 19(2), 322–331 (1990)
3. Brace, K., Rudell, R., Bryant, R.: Efficient implementation of a BDD package. In: The 27th ACM/IEEE Design Automation Conference, pp. 40–45 (1990)
4. Breunig, M., Kriegel, H., Ng, R., Sander, J.: LOF: Identifying density-based local outliers. In: SIGMOD Conference, pp. 93–104 (2000)
5. Bryant, R.: Graph-based algorithms for boolean function manipulation. IEEE Trans. Computers 35(8), 677–691 (1986)
6. Caputo, B., Sim, K., Furesjo, F., Smola, A.: Appearance-based object recognition using svms: Which kernel should I use? In: NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision (2002)
7. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Computing Surveys (CSUR) 41(3), 15 (2009)
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters 27(8), 861–874 (2006)
9. Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A.: kernlab – an S4 package for kernel methods in R. Journal of Statistical Software 11(9), 1–20 (2004)
10. Kutsuna, T.: A binary decision diagram-based one-class classifier. In: The 10th IEEE International Conference on Data Mining, pp. 284–293 (December 2010)
11. Kutsuna, T., Yamamoto, A.: A parameter-free approach for one-class classification using binary decision diagrams. Intelligent Data Analysis 18(5) (to appear, 2014)
12. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A comparative study of anomaly detection schemes in network intrusion detection. In: Proceedings of SIAM Conference on Data Mining (2003)
13. Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R.: Estimating the support of a high-dimensional distribution. Neural Computation 13(7), 1443–1471 (2001)
14. Somenzi, F.: CUDD: CU decision diagram package, http://vlsi.colorado.edu/~fabio/CUDD/
15. Somenzi, F.: Binary decision diagrams. In: Calculational System Design, vol. 173, pp. 303–366. IOS Press (1999)
16. Torgo, L.: Data Mining with R, learning with case studies (2010)