Expert Systems with Applications 27 (2004) 681–697

Expert Systems with Applications

www.elsevier.com/locate/eswa

# Mining class outliers: concepts, algorithms and applications in CRM

Zengyou He[a,*], Xiaofei Xu[a], Joshua Zhexue Huang[b], Shengchun Deng[a]

[a]*Department of Computer Science and Engineering, Harbin Institute of Technology, 92 West Dazhi Street, P.O. Box 315, Harbin 150001, China*
[b]*E-Business Technology Institute, The University of Hong Kong, Pokfulam, Hong Kong, China*

**Abstract**

Outliers, or commonly referred to as exceptional cases, exist in many real-world databases. Detection of such outliers is important for many applications and has attracted much attention from the data mining research community recently. However, most existing methods are designed for mining outliers from a single dataset without considering the class labels of data objects. In this paper, we consider the *class outlier detection problem* 'given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels'. By generalizing two pioneer contributions [Proc WAIM02 (2002); Proc SSTD03] in this field, we develop the notion of class outlier and propose practical solutions by extending existing outlier detection algorithms to this case. Furthermore, its potential applications in CRM (customer relationship management) are also discussed. Finally, the experiments in real datasets show that our method can find interesting outliers and is of practical use.

*Keywords:* Outlier; Data mining; CRM; Direct marketing

## 1. Introduction

In contrast to traditional data mining task that aims to find the general pattern applicable to the majority of data, outlier detection targets the finding of the rare data whose behavior is very exceptional when compared with rest large amount of data. Studying the extraordinary behavior of outliers helps uncovering the valuable knowledge hidden behind them and aiding the decision makers to make profit or improve the service quality. Thus, mining for outliers is an important data mining research with numerous applications, including credit card fraud detection, discovery of criminal activities in electronic commerce, weather prediction, and marketing.

A well-quoted definition of outliers is firstly given by Hawkins Hawkins (1980). This definition states, "An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism". With increasing awareness on outlier detection in data mining literature, more concrete meanings of outliers are defined for solving problems in specific domains. Nonetheless, most of these definitions follow the spirit of the Hawkins-Outlier. That is, the majority of the past research efforts has been focused on the problem of detecting deviants in a single dataset without considering the class labels of data objects. i.e.

**Problem 1 (Traditional Outlier Detection)**. Given a set of objects, find these that deviate significantly from the rest regardless of class labels.

However, as shown in He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003), there are several important practical situations where we have collections of points with different class labels. Consider the following illustrative example of customer analysis in CRM:

**Example 1**. Nowadays, analysts usually employ the segment-level view marketing. Customer segmentation is the division of the entire customer population into smaller groups, called *customer segments*. The key idea is that each segment is fairly homogeneous from a certain perspective, though not necessarily from other perspectives. Thus, the customer base is first segmented by the value they represent to an organization, and then by the needs they may have for specified products and services.

* Corresponding author. Tel.: +86-451-8641-4906x8002.
*E-mail addresses:* zengyouhe@yahoo.com (Z. He), xiaofei@hit.edu.cn (X. Xu), jhuang@eti.hku.hk (J.Z. Huang), dsc@hit.edu.cn (S. Deng).

0957-4174/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2004.07.002

As time goes on, customers shift among segments. One can move from the loyal segment to the vulnerable segment in terms of profits. Therefore, it is critical to identify these potential churners. From the viewpoint of outlier detection, even though everything may look 'normal' when we ignore segment types in the whole customer base, while they may deviate with respect to the loyal segment and look normal with respect to the vulnerable segment. Therefore, it is critical to identify these potential churners in real applications.

Therefore, we consider the class outlier detection problem.

**Problem 2 (Class Outlier Detection)**. Given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels.

To our knowledge, the class outlier detection problem has only been explicitly considered in He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003). Unfortunately, both He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003) only consider a special case of class outlier. In He, Deng, and Xu (2002), the class outlier is defined as 'semantic outlier'. A semantic outlier is a data point, which behaves differently with other data points in the same class, while looks normal with respect to data points in another class. In contrast, the authors of Papadimitriou and Faloutsos (2003) propose the cross-outlier detection problem. That is, given two sets (or classes) of objects, find those objects in one set that deviate with respect to the other set.

In this paper, we develop the notion of class outlier by generalizing two pioneer works (He, Deng, & Xu, 2002; Papadimitriou and Faloutsos, 2003). For mining class outliers, we propose practical solutions by extending existing outlier detection algorithms. Furthermore, how to apply it in CRM is discussed and its effectiveness is evaluated on real datasets. The main contributions of our work can be summarized as follows:

- We identify the problem of class outlier detection. Compared with He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003), our generalization not only considers outliers that deviate with respect to its own class, but also outliers that deviate with respect to other classes, so that the all kinds of class outliers can be fully identified.
- We discuss the potential applications of class outlier detection techniques in analytic CRM.
- Practical methods are provided for extending the traditional outlier detection algorithms to class outlier detection problem which are very efficient in practice.
- Experimental results on real marketing dataset show that the class outlier detection method is of practical use and provides at least the same level of effectiveness as other popular techniques.

The remainder of this paper is organized as follows. Section 2 presents the concept of class outlier. In Section 3, we propose algorithms for class outlier mining. Section 4 discusses the potential applications of class outlier in customer migration and direct marketing. Experimental results are given in Section 5. Related work is discussed in Section 6 and Section 7 concludes the paper.

## 2. Class outlier concepts

In this section, we describe the class outlier detection problem and introduce our definitions. In general, the class outlier detection problem is stated as:

**Definition 1: (Class Outlier Detection)**. Given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels.

**Remarks**.

1. The intuitive Definition 1 implies that we are discussing a more general problem beyond He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003), for which only consider special cases of class outlier.
2. The problem becomes more challenging when the dataset contains multiple class labels (more than 2), that is, how to effectively define and detect outliers when multiple class labels exist.

Firstly, we introduce two basic types of class outliers: *local class outlier* and *reference class outlier*. As we will show later in this section, other more complicated class outliers can be derived from the combinations of them.

The notions to be used are listed in Table 1.

Table 1
Notations

| Notion | Meaning |
|---|---|
| $DB$ | Set of objects $DB=\{C_1,C_2,\ldots,C_k)$, $C_i \cap C_j = \phi$ and $C_1 \cup C_2 \cup,\cdots,\cup C_k = DB$ |
| $\lvert DS \rvert$ | The number of elements in set $DS$ |
| $Sp$ ($DB$) | We let the span of a set $DB$, denoted by $Sp$ ($DB$), be the set of all unions of subsets of DB. The is, we define $Sp(\mathrm{DB}) = \{\cup C_{C \in B} \lvert B \subseteq \mathrm{DB}\}$ |
| $k$ | The number of sets with different class labels |
| $C_i$ | Set of objects with class label $cl_i$ |
| $cl_i$ | The class label for $C_i$ |
| $T$ | The target set for class outlier mining, $T \in DB$ |
| $p$ | A data object in the target set $T$, $p \in T$ |
| $OF$ ($p$) | The outlier factor of an object |
| $LCOF$ ($p$) | The local class outlier factor of an object |
| $RCOF$ (p, $C_i$) | The reference class outlier factor of an object with respect to $C_i$ |
| $SUCOF$ ($p$, $R$) | The set union class outlier factor of an object with respect to $R$, $R \in Sp$ ($DB$) |
| $LSUCOF$ ($p$, $R$) | The local set union class outlier factor of an object with respect to $R$, $R \in Sp$ ($DB$) |
| $RSUCOF$ ($p$, $R$) | The reference set union class outlier factor of an object with respect to $R$, $R \in Sp$ ($DB$) |
| $CCOF$ ($p$, $\{R_1,R_2,\ldots,R_m\}$) | The combined class outlier factor of an object |

**Definition 2: (Local Class Outlier Detection)**. Given *DB* and target set *T*, for an object $p \in T$, the local class outlier factor of *p*, denoted as *LCOF* (*p*), captures the degree to which we call *p* an outlier with respect to *T*. The local class outlier detection problem is to find those suspicions objects according to the value of *LCOF* (*p*).

**Definition 3: (Reference Class Outlier Detection)**. Given *DB* and target set *T*, for an object $p \in T$, the reference class outlier factor of *p*, denoted as *RCOF* ($p$,$C_i$), captures the degree to which we call *p* an outlier with respect to $C_i$. The reference class outlier detection problem is to find those suspicions objects according to the value of *RCOF* ($p$,$C_i$).

In Definition 2, we consider the problem of detecting outlying observations from a target set of objects *T*, with respect to itself. It is named as 'local class outlier' because even though everything may look normal in *DB* when we ignore object types, there is still the possibility of finding 'suspicious' objects when we only consider objects in *T*.

In Definition 3, we want to discover points in *T* that 'arouse suspicions' with respect to points in set $C_i$. Note that our notion of reference class outlier is same with that of cross-outlier proposed in Papadimitriou and Faloutsos (2003). As pointed out in Papadimitriou and Faloutsos (2003), local class outlier is a special case of reference class outlier, when we let $C_i = T$. However, the following observations motivate us to define them separately:

1. Firstly, from the application viewpoint, in spite of their similarities in representation, their meanings differ significantly. Continuing example 1, a local class outlier in the loyal segments is a possible churning customer that needs to be retained. While a reference class outlier with respect to vulnerable segment may be a loyal customer, for which shares little common characteristics with customers in the vulnerable segment. Hence, explicitly distinguishing these two kinds of outliers is of great advantage for business users.
2. Secondly, from the viewpoint of algorithm design, some traditional approaches (or local class outlier detection methods) may be modified to deal with reference class outlier detection problem, but the task is non-trivial Papadimitriou and Faloutsos (2003). The general problem is open and provides promising future research directions.

So far, we only provide the concepts of local class outlier and reference class outlier, and assume that both of them can mined by extending existing methods. Implementation details will be discussed in the next section. We further assume that the outlier factor values are normalized into the interval [0,1]. For each object's *degree* of being an outlier is measured by its outlier factor, without loss of generality, it is assumed that the higher is the outlier factor value of an object, the higher of its outlier-ness is.

In the sequel, we will introduce *set union class outlier*, where the reference set is the union of some sets.

**Definition 4: (Set Union Class Outlier)**. Given *DB* and target set *T*, for an object $p \in T$, the set union class outlier factor of *p* with respect to *R*, denoted as *SUCOF* (*p*, *R*), where $R \in Sp(DB)$. Moreover, set union class outlier can be divided into two kinds: *local set union class outlier* and *reference set union class outlier*. That is, if $T \subseteq R$, *p* is called a local set union class outlier; its outlier factor is denoted as *LSUCOF* (*p*, *R*); Otherwise, *p* is called a reference set union class outlier; its outlier factor is denoted as *RSUCOF* (*p*, *R*).

According to Definition 4, both local class outlier and reference class outlier are special cases of set union class outlier. They are distinguished by the reference set. More concise, if $R = T$, $LSUCOF\ (p,\ R) = LCOF\ (p)$; if $R = C_i(C_i \neq T)$, $RSUCOF\ (p,\ R) = RCOF\ (p,\ R)$; More importantly, when $R = DB$, the problem of set union class outlier detection is equivalent to traditional single-class outlier detection problem. That is, given a set of objects, find those objects that deviate significantly from the rest regardless of class labels. Therefore, all aforementioned outlier detection problems can be studied in the framework of set union class outlier.

We now turn to another kind of outlier-*combined class outlier*, which can be regarded as a further extension on set union class outlier and is of practical use.

**Definition 5: (Combined Class Outlier)**. Given *DB* and target set *T*, for an object $p \in T$, the combined class outlier factor of *p* is defined as:

$$\mathrm{CCOF}(p, \{R_1, R_2, \ldots, R_m\}) = \oplus(v_1, v_2, \ldots, v_m)$$

where $\oplus$ is the *combiner* function, which computes the outlier factor value form $v_1$,$v_2$,…,$v_m$, $R_i \in Sp(DB)$ and $v_i = SUCOF(p,\ R_i)$ or $v_i = 1 - SUCOF(p,\ R_i)$.

If $m > 1$, we call *p* true combined class outlier.

From Definition 5, the value of combined class outlier factor is the result of applying the *combiner* function $\oplus$ on some set of values. Each $v_i \in (v_1,v_2,\ldots,v_m)$ is the set union class outlier factor of *p* or its reciprocal.

Compared to other types of class outlier, we refer to multiple sets instead of a single set. Note that the set union class outlier is also a special case of combined class outlier, that is,

$$\mathrm{CCOF}(p, \{R_i\}) = \oplus(v_i) = v_i = \mathrm{SUCOF}(p, R_i) \quad or$$

$$1 - \mathrm{SUCOF}(p, R_i).$$

To complete the description of combined class outlier, we discuss the following issues: our choice of combining operators and the usefulness of combined class outlier in real applications.

Choosing a combining operator: Our potential choices for $\oplus$ are the followings.[1] We offer some additional insights on these choices in Section 5.

- The product operator $\prod$: $\oplus(v_1,v_2,\ldots,v_m)=v_1v_2,\ldots,v_m$.
- The addition operator $+$: $\oplus(v_1,v_2,\ldots,v_m)=v_1+v_2+\cdots+v_m$.
- A generalization of addition operator—it is called the $S_q$ combining rule, where $q$ is an odd natural number. $S_q(v_1,v_2,\ldots,v_m)=(v_1^q+v_2^q+,\cdots,+v_m^q)^{(1/q)}$. Note that the addition is simply the $S_1$ rule.
- A 'limiting' version of $S_q$ rules, denoted as $S_\infty$. $S_\infty$ $(v_1,v_2,\ldots,v_m)$ is defined to be equal to $v_i$, where $v_i$ has the largest absolute value among $(v_1, v_2,\ldots,v_m)$.

Thus, the $S_1$ combining rule is *linear* with respect to the component outlier factors. The $\prod$ and $S_q$ rules for $q>1$, on the other hand, involve a *non-linear* term for each individual outlier factor. $S_\infty$ is an especially appealing rule, since it is non-linear, particularly fast to compute, and also appears to have certain useful 'sum-like' properties.

Practical use of combined class outlier. According to Hawkins (1980), an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. We introduce a set of class outlier factors for each object in the dataset, indicating its degree of outlier-ness. We have assumed that the outlier factor values are normalized into the interval [0,1] and the higher is the outlier factor value of an object $p$, the higher of its outlier-ness (For set union class outlier, its outlier factor is denoted as *SUCOF* $(p, R_i)$) is. Therefore, we can get the following statement:

> The lower is the outlier factor value of an object $p$, the lower of its outlier-ness is. More concise, objects with lower values of outlier factor are more likely to be generated by the same mechanism.

Based on the above statement and assumption, higher *SUCOF* $(p, R_i)$ value or lower $(1-$*SUCOF* $(p, R_i))$ value indicates higher outlier-ness. In contrast, lower *SUCOF* $(p, R_i)$ value or higher $(1-$*SUCOF* $(p, R_i))$ value indicates lower outlier-ness.

Furthermore, given two sets $R_1$ and $R_2$, *SUCOF* $(p, R_1)$ and *SUCOF* $(p, R_2)$ denote the outlier factor values of an object $p$ with respect to $R_1$ and $R_2$, respectively. From the outlier detection viewpoint, all possible combinations using combining operator $\oplus$ are listed in Table 2.

As shown in Table 2, taking *SUCOF* $(p, R_1)\oplus$*SUCOF* $(p, R_2)$ as an example, we can derive the fact that: "The higher is the value *SUCOF* $(p, R_1)\oplus$*SUCOF* $(p, R_2)$ of the object $p$, the higher of its outlier-ness with respect to $R_1$ and $R_2$ is". Other combinations can be illustrated in a similar way. To complete the description of the usefulness of combined class outlier, considering the following example.

Table 2
Combinations and their meanings

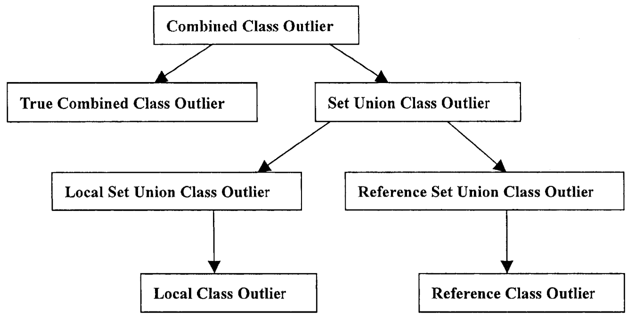
| Combinations | Meanings |
|---|---|
| *SUCOF* $(p, R_1)\oplus$*SUCOF* $(p, R_2)$ | The object $p$ deviates from both $R_1$ and $R_2$ |
| *SUCOF* $(p, R_1)\oplus(1-$*SUCOF* $(p, R_2))$ | The object $p$ deviates from $R_1$ and looks normal in $R_2$ |
| $(1-$*SUCOF* $(p, R_1))\oplus$*SUCOF* $(p, R_2)$ | The object $p$ deviates from $R_2$ and looks normal in $R_1$ |
| $(1-$*SUCOF* $(p, R_1)\oplus(1-$*SUCOF* $(p, R_2))$ | The object $p$ looks normal in both $R_1$ and $R_2$ |

**Example 2**. Continuing Example 1, a potential customer that transferred from loyal segment (denoted as $L$) to vulnerable segment (denoted as $V$) would satisfy the following desired properties. (1) His (Her) behavior is very exceptional when compared with rest large amount of customers in loyal segment, and (2) he (she) looks very normal with respect to customers in vulnerable segment. Hence, for our purpose, to simultaneously incorporate these two measures into an integrated measure, *SUCOF* $(p, L)\oplus$ $(1-$*SUCOF* $(p, V))$ will be a good choice. That is, to detect potential switched customers, our task is to detect combined class outliers according to the value of *SUCOF* $(p, L)\oplus(1-$*SUCOF* $(p, V))$.

In general, for combined class outlier with outlier factor CCOF $(p, \{R_1,R_2,\ldots,R_m\})$, which refers to more than two sets, its practical use in applications can also be illustrated in the similar way.

The 'semantic outlier' proposed in He, Deng, and Xu (2002) is defined informally as: "A semantic outlier is a data point, which behaves differently with other data points in the same class, while looks normal in another class". Obviously, the 'semantic outlier' is a special case of combined class outlier.

In summary, all aforementioned class outliers can be studied in the framework of combined class outlier, their relationships are described in Fig. 1.


Combined Class Outlier
True Combined Class Outlier
Set Union Class Outlier
Local Set Union Class Outlier
Reference Set Union Class Outlier
Local Class Outlier
Reference Class Outlier
Fig. 1. Classification of class outliers.

[1] The choices for $\oplus$ have been widely studied in dynamic systems. In this paper, we borrow some interesting idea from Gibson, Kleiberg, and Raghavan (1998).

## 3. Algorithms

In this section, we propose our methods for mining class outliers. We begin with identifying the difference on score model building between class outlier detection and traditional outlier detection. Consequently, we show how to modify existing outlier detection algorithms for class outlier mining.

### 3.1. Difference on score model building

In Section 2, we have seen that the outlier factor of combined class outlier is result of applying the *combiner* function $\oplus$ on some set of values of set union class outlier factors. Therefore, the main changeling task is to compute the set union class outlier factor for each object in the target set. That is, we can focus on the problem of detecting outlying observations from a target set of points $T$, with respect to a reference set of points $R$. We want to discover points $p \in T$ that 'arouse suspicions' with respect to points $r \in R$. As have been discussed in Section 2, traditional single-set outliers are a special case, where $R = T$.

Since traditional outlier detection algorithms first build a scoring model using the data points in $T$ and the outlier factor for each point is computed with derived model, we can describe those algorithms using the following schema $A1$ (See Fig. 2).

The algorithmic schema $A2$ for class outlier detection algorithms is shown in Fig. 3. Compared with schema $A1$, the training set for building scoring model is $R + \{p\}$. Thus, for each point $p$, the model $SM$ has to be re-constructed. For real data mining applications, the size of dataset is commonly very large, the frequent re-constructing process will be time-consuming. Motivated by the above observation, as shown in Fig. 4, we propose a new algorithmic schema $A3$ by modifying algorithmic schema $A2$.

The algorithmic schema $A3$ can be regarded as a fast approximation schema for class outlier detection. In Sections 3.2 and 3.3, we modify two existing outlier detection algorithms according to schema $A3$.

### 3.2. Frequent pattern based class outlier detection

The frequent pattern based outlier detection method He, Xu, and Deng (2002b) is modified for mining class outlier. We will describe this method in detail.

```
Algorithmic Schema A1:
Build scoring model SM using T
foreach Point p ∈ T {
Outlier factor OF (p) = ComputeScore ( p, SM);
}
Sort points according to outlier factors
DoSomething ( ) ;
```

Fig. 2. The algorithmic schema $A1$ for traditional outlier detection algorithms.

```
Algorithmic Schema A2:
foreach Point p ∈ T {
Build scoring model SM using R+{p};
Outlier factor OF (p) = ComputeScore ( p, SM);
}
Sort points according to outlier factors
DoSomething ( ) ;
```

Fig. 3. The algorithmic schema $A2$ for class outlier detection algorithms.

Agrawal's statement of the problem of discovering frequent itemset in market basket databases is the following Agrawal and Srikant (1994).

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of $m$ literals called *items*. Let the database $D = \{t_1, t_2, \ldots, t_n\}$ be a set of $n$ transactions each consisting of a set of items from $I$. An itemset $X$ is a non-empty subset of $I$. The length of itemset $X$ is the number of items contained in $X$, and $X$ is called a $k$-itemset if its length is $k$. A transaction $t \in D$ is said to contain an itemset $X$ if $X \subseteq t$. The *support* of an itemset $X$ is the percentage of transactions in $D$ containing $X$: *support* $(X) = \|\{t \in D | X \subseteq t\}\| / \|\{t \in D\}\|$.

The problem of finding all *frequent itemsets* in $D$ is then traditionally defined as follows. Given user defined threshold for the permissible minimal support, find all itemsets with support greater or equal to *minisupport*. Frequent itemsets are also called *frequent patterns*.

From the viewpoint of knowledge discovery, frequent patterns reflect the *common patterns* that apply to many objects, or to large percentage of objects, in the dataset. In contrast, outlier detection focuses on a very small percentage of data objects. Hence, the idea of making use of frequent patterns for outlier detection is very intuitive.

**Definition 6: (FPOF—frequent pattern outlier factor).** Let the database $D = \{t_1, t_2, \ldots, t_n\}$ be a set of $n$ transactions with *items I*. Given threshold *minisupport*, the set of all frequent patterns is donated as: *FPS* (*D, minisupport*). For each transaction $t$, the *frequent pattern outlier factor of t* is defined as:

$$\mathrm{FPOF}(t) = \frac{\sum_X \mathrm{support}(X)}{\|\mathrm{FPS}(D, \mathrm{minisupport})\|}, \tag{1}$$

where $X \subseteq t$ and $X \in \mathrm{FPS}(D, \mathrm{minisupport})$

The interpretation of formula (1) is as follows. If a data object contains more frequent patterns, its *FPOF* value will be larger, which indicates that it is unlikely to be an outlier.

```
Algorithmic Schema A3:
Build scoring model SM using R
foreach Point p ∈ T {
Outlier factor OF (p) = ComputeScore ( p, SM);
}
Sort points according to outlier factors
DoSomething ( ) ;
```

Fig. 4. The modified algorithmic schema $A3$ for class outlier detection algorithms.

```
Algorithm FindCFPOF

Input:  R            // the reference transaction database
        T            //the target transaction database
        minisupport  // user defined threshold for the permissible minimal support
        top-n        // user defined threshold value for top n fp-outliers
        top-k        // user defined threshold value for top k contradict frequent patterns
Output: The values of FPOF for all transactions in T //indicates the degree of deviation
        The top-n FP-outliers with their corresponding TKCFPs

01 begin
02     Mining the set of frequent patterns on database R using minisupport
03     /* the set of all frequent patterns is donated as: FPS (R, minisupport) */
04     foreach transaction t in T do begin
05         foreach frequent pattern X in FPS (R, minisupport) do begin
06             if t contains X then
07                 FPOF (t) = FPOF (t)+ support (X)
08             end if
09         end
10         return FPOF (t)
11     end
12     Output the transactions in the ascending order of their FPOF values. Stop when it
       outputs top-n transactions
13     foreach transaction t in top-n outliers do begin
14         Finds its top-k contradict frequent patterns and outputs them
15     end
```

Fig. 5. The *FindCFPOF* Algorithm.

In contrast, objects with smaller *FPOF* values will have greater outlying-nesses. In addition, the FPOF value is between 0 and 1.

**Definition 7**. For each transaction $t$, the itemset $X$ is said to be contradict to $t$ if $X \not\subset t$. The contradict-ness of $X$ to $t$ is defined as:

$$\text{Contradict}-\text{ness}(X,t) = (\|X\| - \|t \cap X\|)^{*}\text{support}(X) \quad (2)$$

In our approach, the *frequent pattern outlier factor* given in definition 6 is used as the basic measure for *identifying* outliers. To *describe* the reasons why identified outliers are abnormal, the itemsets that are not contained in the transaction (it is said that the itemset is *contradict* to the transaction) are good candidates.

The consideration behind formula (2) is as follows. First, the greater the support of the itemset $X$, the greater the value of *contradict-ness* of $X$ to $t$ since larger support value of $X$ suggests a more strong deviation. Second, longer itemsets will give a better description than that of shorter ones.

With Definition 7, it is possible to identify the contribution of each itemset to the outlying-ness of specified transaction. However, it is not feasible to list all the *contradict itemset*, and it will be preferable to present only the top $k$ contradict frequent patterns to the end user, as shown in Definition 8.

**Definition 8: (TKCFP—top $k$ contradict frequent pattern)**. The meanings of $D$, $I$, minisupport and FPS ($D$, *minisupport*) are the same as given in Definition 6. For each transaction $t$, the itemset $X$ is said to be a top $k$ contradict frequent pattern if there exist[2] no more than $(k-1)$ itemsets whose *contradict-ness* is higher than that of X, where $X \in$ FPS ($D$, minisupport).

With the outlier factor FPOF, we can determine the degree of point's deviation. The algorithm *FindCFPOF* for detecting class outliers is listed in Fig. 5. The algorithm first gets the frequent patterns from the reference database $R$ using existing association rule mining algorithm with given mini-support (Step 2–3). Then, for every transaction in the target database $T$, the value of *FPOF* is computed with

[2] Since there could be more than one itemset having the same *contradict-ness* for a transaction, to ensure the set mined is independent of the ordering of the frequent patterns in *FPS* ($D$, *minisupport*), our method will mine all the itemsets whose *contradict-ness* is no less than the *k-th contradict frequent pattern*.

Definition 6 (Step 4–11). Finally, outputting the *top-n* FP-outliers with their corresponding *top-k* contradict frequent patterns (Step 12–15).

The *FindCFPOF* algorithm has three parts: (1) Mining the frequent patterns from the database $R$, (2) Computing the value of *FPOF* for each transaction in $T$ and (3) Finding the *top-n* FP-outliers with their *TKCFP* descriptions. The frequent-pattern mining algorithm determines the cost of part (1) and it is donated as $O$ (*FP*). We remark that many fast frequent-pattern mining algorithms are available Agrawal and Srikant (1994), Han, Pei, and Yin (2000) and Zaki (2000) and so the computation complexity of part (1) will be acceptable. As to the part (2), two 'for loop' are required. Therefore, cost of part (2) is $O\ (N\times S)$, where $N$ is number of the transactions in the database $T$ and $S$ is the size of the set of frequent patterns. Part (3) has the computation cost of $O\ (N\log N+S\times(\textit{top-n})\times(\textit{top-k})\times\log(\textit{top-k}))$, because finding the *top-n* outliers and the *top-k* contradict frequent patterns for each identified outlier needs a *sort* operation.

Thus, the overall computation complexity for the *FindCFPOF* algorithm is: $O\ (FP+N\times S+N\log N+S\times(\textit{top-n})\times(\textit{top-k})\times\log(\textit{top-k}))$.

### 3.3. Cluster based class outlier detection

The concept of cluster-based local outlier is introduced in He, Xu, and Deng (2003). This section modifies it for the purpose of class outlier detection.

**Definition 9**. Given a transaction database $D$, the results of a clustering algorithm executed on $D$ is denoted as: $G=\{G_1,G_2,\ldots,G_k\}$ where $G_i\cap G_j=\phi$ and $G_1\cup G_2\cup,\cdots,\cup G_k=D$. The number of clusters is $k$.

Here, the clustering algorithm used for partitioning the dataset into disjoint sets of records can be chosen freely. The only requirement for the selected clustering algorithm is that it should have the ability to produce good clustering results. The clustering algorithm adopted in this paper is the *Squeezer* algorithm He, Xu, and Deng (2002a).

A critical problem that must be solved before defining the *cluster-based local outlier* is how to identify whether a cluster is *large or small*. This problem is discussed in Definition 10.

**Definition 10: (large and small cluster)**. Suppose $G=\{G_1,G_2,\ldots,G_k\}$ is the set of clusters in the sequence that $|G_1|\ge|G_2|\ge\cdots\ge|G_k|$. Given two numeric parameters $\alpha$ and $\beta$, we define $b$ as the boundary of large and small cluster if one of following formulas holds.

$$\{(|G_1|+|G_2|+\cdots+|G_b|)\ge|D|^{*}\alpha \tag{C1}$$

$$\{|G_b|/|G_{b+1}|\ge\beta \tag{C2}$$

Then, the set of *large* cluster is defined as: $LC=\{G_b|i\le b\}$ and the set of *small* cluster is defined as: $SC=\{G_j|j>b\}$.

Definition 10 gives quantitative measure to distinguish *large* and *small* clusters. Formula (C1) considers the fact that most data points in the data set are not outliers. Therefore, clusters that hold a large portion of data points should be taken as *large* clusters. For example, if $\alpha$ is set to 90%, we intend to regard clusters contain 90% of data points as *large* clusters. Formula (C2) considers the fact that *large* and *small* clusters should have significant differences in size. For instance, it is easy to get that, if we set $\beta$ to 5, the size of any cluster in *LC* is at least 5 times of the size of the cluster in *SC*.

**Definition 11: (cluster-based local outlier factor)**. Suppose $G=\{G_1,G_2,\ldots,G_k\}$ is the set of clusters in the sequence that $|G_1|\ge|G_2|\ge\cdots\ge|G_k|$ and the meanings of $\alpha$, $\beta$, $b$, LC and SC are the same as they are formalized in Definition 10. For any record $t$, the cluster-based local outlier factor of $t$ is defined as:

$$\mathrm{CBLOF}(t)=\begin{cases}|G_i| * \max(\mathrm{similarity}(t,G_j)),\\ \quad \text{where}\quad t\in G_i,\quad G_i\in SC \text{ and } G_j\in LC\\ \quad \text{for } j=1 \text{ to } b\\ |G_i| * \ (\mathrm{similarity}(t,G_i)),\\ \quad \text{where } t\in G_i \quad \text{and} G_i\in LC\end{cases} \tag{C3}$$

From Definition 11, the *CBLOF* of a record is determined by the size of its cluster, and the similarity between the record and its closest cluster (if this record lies in *small* cluster) or the similarity between the record and the cluster it belongs to (if this record belongs to *large* cluster), which provides importance to the local data behavior.

For the computation of similarity between the record and the cluster, it is sufficient to adopt the similarity measure used in the clustering algorithm. Note that the similarity value is normalized into the interval [0,1].

With the outlier factor *CBLOF*, we can determine the degree of each point's deviation. The algorithm *FindCCBLOF* for detecting class outliers is listed in Fig. 6.

To detect class outlier using the cluster-based local outlier concept, in the algorithm *FindCCBLOF*, we make some changes on computing outlier factors. As shown in Fig. 6, for each object in the target set, we find its nearest cluster ($G_{\max}$) and the outlier factor is computed as $|G_{\max}|\times$ *similarity* ($t,G_{max}$) (Step 8).

Note that, similar to the frequent pattern based class outlier detection algorithm presented in Section 3.2, we can also sort the outlier factors and output *top-n* outliers.

## 4. Applications in CRM

Customer relationship management (CRM) is a strategy to acquire new Customers, to retain them and to recover them if they defected Arndt and Gersten (2001).

```
Algorithm FindCCBLOF

Input:   R                    // the reference data set
         T                    // the target data set
Output:  The values of CBLOF for all records in T //indicates the degree of deviation

01 begin
02     Clustering the dataset R using clustering algorithm
03     /* Produced clusters: $G = \{G_1, G_2, \ldots, G_k\}$ and $|G_1|>=|G_2|>= \ldots >=|G_k|$ */
04     foreach record t in the dataset T do begin
05         foreach cluster Gi in G
06             find the cluster that have max similarity with t, denoted as $G_{max}$
07         end
08         CBLOF= | $G_{max}$ |*similarity (t, $G_{max}$ ))
09         return CBLOF
10     end
12 end
```

Fig. 6. The *FindCCBLOF* Algorithm.

Operational CRM includes all activities concerning the *direct customer contact*, such as campaigns, hotlines or customer clubs. Every oCRM activity is generally implemented in one of the three enterprise processes: sales, marketing or service, since these are the processes concerned with direct customer contact (ECCS, 1999).

Analytical CRM (aCRM) provides all components to *analyze customer characteristics* (behaviors) in order to accomplish oCRM activities, with respect to the customers' needs and expectations Skinner (1990). There, the idealistic goal is to provide all information necessary to create a tailored cross-channel dialogue with each single customer on the basis of his or her actual reactions (Arndt & Gersten, 2001).

Before enterprises can develop marketing or CRM strategies, they must understand how consumers make their purchase decisions. This decision process is called customer buying cycle (CBC). We assume that the chain of all CBCs a single customer runs through is his or her customer life cycle. The process ends with the final stop of consumption. Fig. 7 illustrates the overall system and is described below (Arndt & Gersten, 2001).

In order to organize the oCRM and a CRM activities along the CRM process, they are implemented as separate programs (as illustrated in Fig. 7 on the level of CRM organization) with clear interfaces, special goals and corresponding direct marketing activities, like acquisition campaigns or road shows. Fig. 8 illustrates how

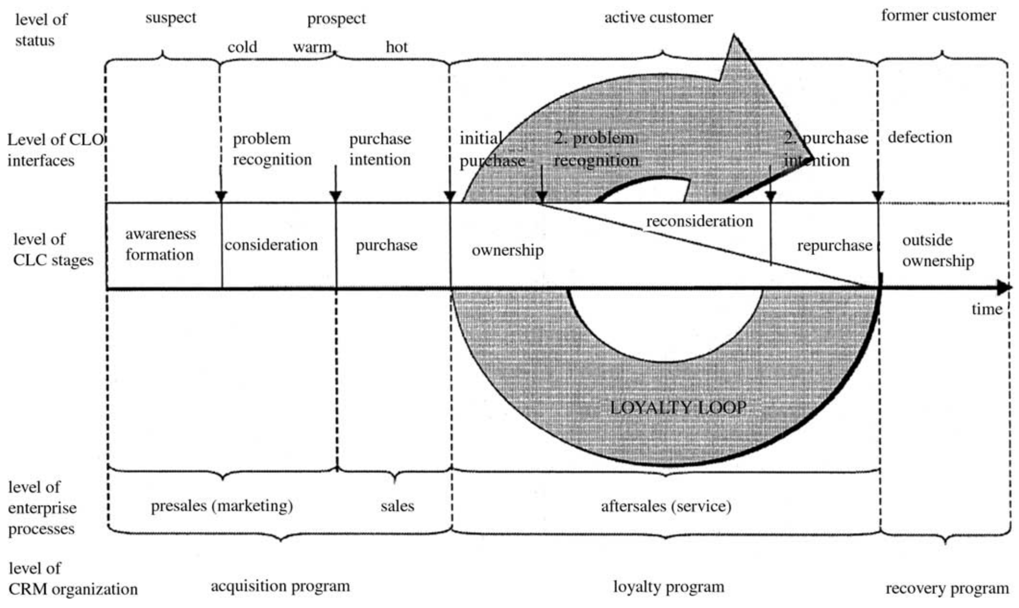


Fig. 7. The CRM-process based on Customer Life Cycle (Arndt & Gersten, 2001).

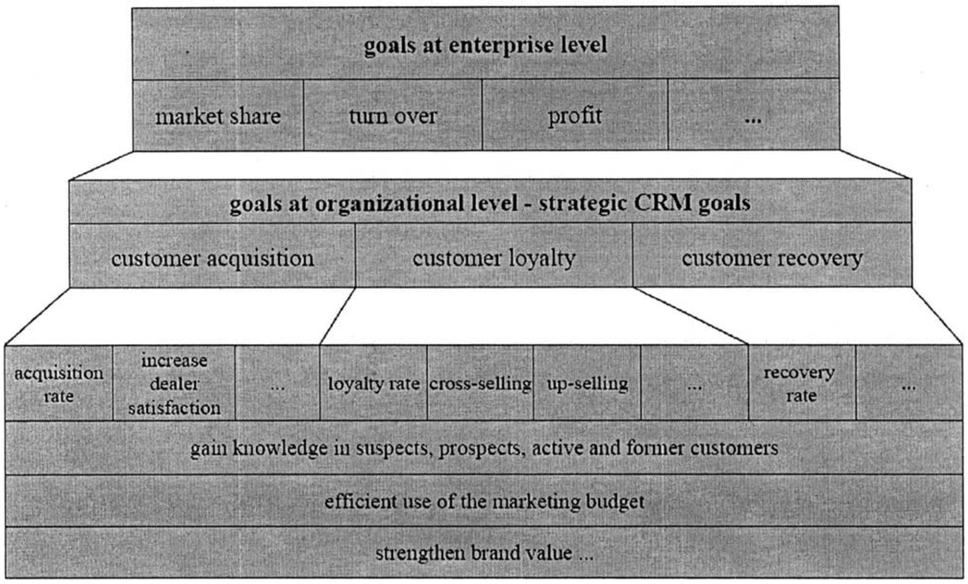

goals at enterprise level
market share
turn over
profit
...
goals at organizational level - strategic CRM goals
customer acquisition
customer loyalty
customer recovery
acquisition rate
increase dealer satisfaction
...
loyalty rate
cross-selling
up-selling
...
recovery rate
...
gain knowledge in suspects, prospects, active and former customers
efficient use of the marketing budget
strengthen brand value ...

Fig. 8. Systematic of CRM goals Arndt and Gersten (2001).

the program-related CRM goals fit into the general goal pyramid of the enterprise (Arndt & Gersten, 2001).

Among those CRM goals, we use 'customer loyalty' and direct marketing to illustrate the application of class outlier in CRM.

## 4.1. Loyalty

Loyalty is the key program of CRM. It is important to detect customers, which have the intension to leave from loyal segment (denoted as $L$) to vulnerable segment (denoted as $V$). As discussed in Section 2, these customers satisfy the following properties.

(1) His (Her) behavior is very exceptional when compared with rest large amount of customer in loyal segment.
(2) He (she) looks very normal with respect to customers in vulnerable segment.

Hence, within the framework of class outlier, we will have three alternative measures to identify those customers (Each customer is denoted as $p$).

(1) $SUCOF$ ($p$, $L$)—measuring the outlier-ness of $p$ with respect to $L$.
(2) $(1-SUCOF$ ($p$, $V$))—measuring the likeness of $p$ with respect to $V$.
(3) $SUCOF$ ($p$, $L$)$\oplus(1-SUCOF$ ($p$, $V$)—measuring both the outlier-ness of $p$ with respect to $L$ and the likeness of $p$ with respect to $V$.

In summary, we apply class outlier factors as loyalty scores to detect customers that have the possibility to leave the loyalty segment. For these identified customers (outliers), the marketer can explore suitable retention program to keep them.

## 4.2. Direct marketing

Direct marketing aims at obtaining and maintaining direct relations between suppliers and buyers within one or more product/market combinations. In marketing, there are two main different approaches to communication: mass marketing and direct marketing (Ling & Li, 1998).

Mass marketing uses mass media such as print, radio and television to the public without discrimination. While direct marketing involves the identification of customers having potential market value by studying the customers' characteristics and the needs (the past or the future) and selects certain customers to promote. Direct marketing becomes increasingly popular because of the increased competition and the cost problem. It is an important area of applications for data mining, data warehousing, statistical pattern recognition, and artificial intelligence.

In direct marketing, models (profiles) are generated to select potential customers (from the client database) for a given product by analyzing data from similar campaigns, or by organizing test mail campaigns (Setnes & Kaymak, 2001).

Let $U$ be a finite set of objects. Elements of $U$ may be customers or products we are interested in market oriented decision making (Yao, Zhong, Huang, Ou, & Liu, 2002). The set $U$ is divided into three pair-wise disjoint classes, i.e. $U=P\cup N\cup T$. The sets $P$, $N$ and $T$ are called *positive*, *negative* and targeting or do not know instances, respectively. $P$ is the set of current customers, which is often the minority class, e.g. the buyer class in direct marketing. $N$ is the set of people who had previously refused to become the customers, e.g. those refuse to buy products. $T$ is the targeting set of potential customers. The set $N$ may be empty. A direct marketing problem may be defined as finding elements from $T$, and possibly from $N$, that are similar to elements in $P$, and possibly dissimilar to elements in $N$. In other words, the goal is to identify elements from $T$ and $N$ that are more likely to become new members of $P$. We are interested in developing a scoring model with class outlier so that elements of $T$ can be ranked accordingly.

The analysis of a direct marketing campaign entails two stages. First, *feature selection* must determine the variables that are relevant for the specific target selection problem. Second, the rules for selecting the customers should be determined, given the relevant features (Setnes & Kaymak, 2001). At present, statistical tools like CHAID (SPSS Inc., 1993) are used to search for features that discriminate between respondents and non-respondents, and to build decision tree models for target selection.

The problem of direct marketing has received a great deal of attention (Ling & Li, 1998; Liu, Ma, Wong, & Yu, 2003; Setnes & Kaymak, 2001; Yao et al., 2002). In this paper, the class outlier factor is utilized as a scoring measure to score the data points in $T$. Since potential responders in $T$ are similar to elements in $P$, possibly dissimilar to elements in $N$, and possibly dissimilar to elements in its own set $T$. Thus, from the viewpoint of class outlier detection, there can be many alternative ways to design methods to score the data. For example, we can simply score each data case $p$ using $1-SUCOF(p, P)$. This method is reasonable because the set union class outlier factor value can be seen as a probability estimate indicating the likelihood that the data case belongs to the positive class.

To design the best scoring method based on class outlier, however, is very difficult. This is because there are a few possible methods. In Table 3, we list all the possible scoring functions and their names. $W_T$, $W_P$ and $W_N$ are the weights of target class, positive class and negative class, respectively.

The first problem should be discussed is which function should be used in practice. To answer this question, the following factors should be considered:

1. The availability of each dataset.
   For example, in some cases, the sets $P$ or $N$ are not available so that it appears that the *TSF* function will be the only choice for user.
2. The data quality of each dataset.

In real life datasets, there is no exception that some values are missing or they are incorrect. Data contains errors because it has been provided by a source that is not fully reliable (human, measurement, complex computation). Such data will affect the results of data mining algorithms significantly. Therefore, it will be better to select scoring functions that involved confident datasets only.

In general, the scoring function *PTSF* always produce reliable result, which is verified in the experimental evaluation section.

Now the problem is what should be the weights in those functions based on true combined class outlier. Since we must consider the two factors that affect the choice of functions discussed above, the weights should reflect their needs. In general, the weight term of less confident dataset should be small because we would like to reduce its effect on the scores. In contrast, the weight term of more confident dataset should be large.

Table 3
Class outlier based scoring functions for direct marketing

| Scoring functions using class outlier | Names |
|---|---|
| $1-SUCOF(p, P)$ | PSF |
| $SUCOF(p, N)$ | NSF |
| $SUCOF(p, T)$ | TSF |
| $W_N \times SUCOF(p, N) \oplus W_P \times (1-SUCOF(p, P))$ | PNSF |
| $W_N \times SUCOF(p, N) \oplus W_T \times SUCOF(p, T)$ | NTSF |
| $W_T \times SUCOF(p, T) \oplus W_P \times (1-SUCOF(p, P))$ | PTSF |
| $W_T \times SUCOF(p, T) \oplus W_P \times (1-SUCOF(p, P)) \oplus W_N \times SUCOF(p, N)$ | PNTSF |

## 5. Empirical evaluation

In order to show the practical use of our method, we applied the proposed class outlier detection method to two public available datasets: the Congressional Voting dataset from UCI Machine Learning Repository Merz and Murphy (1996) and the 'Coil Challenge 2000' dataset The Coil dataset can found at: http://www.liacs.nl/∼putten/library/cc2000/.

In particular, the Congressional Voting dataset has also been used in He, Deng, and Xu (2002). Through the experiments on this dataset, we would like to show that the concept of class outlier is indeed useful and our ideas can successfully reveal interesting new findings. The Coil Challenge 2000 dataset is used to verify the practical use of class outlier based method in direct marketing.

### *5.1. Results on votes data*

Congressional Voting dataset is the United States Congressional Voting Records in 1984. Each record represents one Congressman's votes on 16 issues. All attributes are boolean with Yes (donated as $y$) and No (donated as $n$) values, and missing value is donated as '?'. A classification label of Republican (the set is denoted as *Rep*) or Democrat (the set is denoted as *Dem*) is provided with each record. The dataset contains 435 records with 168 Republicans and 267 Democrats.

As discussed in He, Deng, and Xu (2002), the records with the same class label should be similar with each other since the data points in the same group should be similar. Therefore, it will be very interesting to detect those points behave differently with other points in the same class, while look 'normal' in another class. Obviously, such a record $p$ can be regarded as combined class outlier in *Rep* with outlier factor $SUCOF(p, Rep) \oplus (1-SUCOF(p, Dem))$ or combined class outlier in *Dem* with outlier factor $SUCOF(p, Dem) \oplus (1-SUCOF(p, Rep))$.

Table 4
Top 10 outliers in votes dataset

| Label | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Democrat | *n* | *y* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *n* | ? | *y* | *y* | *y* |
| Democrat | *n* | *y* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *n* | *y* | *y* | *n* | *n* |
| Democrat | *n* | *n* | *n* | *n* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* | *y* | *y* | *n* | *y* |
| Democrat | *n* | *y* | *n* | *n* | *y* | *y* | *n* | *n* | ? | *n* | *n* | *y* | *y* | *y* | *n* | *y* |
| Democrat | *n* | *y* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | ? |
| Democrat | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *y* | *n* | *n* |
| Republican | *y* | *y* | *y* | *y* | *n* | *n* | *y* | *y* | *y* | *y* | *y* | *n* | *n* | *y* | *n* | *y* |
| Republican | *n* | *n* | *y* | *y* | *n* | *n* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* |
| Republican | *y* | *n* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *n* | *y* |
| Republican | *y* | *n* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* |

Table 5
The value distribution of democrat and republican class

| Label | Value | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Democrat | *y* | 156 | 120 | 231 | 14 | 55 | 123 | 200 | 218 | 188 | 124 | 129 | 36 | 73 | 90 | 160 | 173 |
| | *n* | 102 | 119 | 29 | 245 | 200 | 135 | 59 | 45 | 60 | 139 | 126 | 213 | 179 | 167 | 91 | 12 |
| | ? | 9 | 28 | 7 | 8 | 12 | 9 | 8 | 4 | 19 | 4 | 12 | 18 | 15 | 10 | 16 | 82 |
| Republican | *y* | 31 | 75 | 22 | 153 | 157 | 148 | 39 | 24 | 19 | 92 | 21 | 135 | 136 | 158 | 14 | 96 |
| | *n* | 134 | 73 | 142 | 2 | 8 | 17 | 123 | 133 | 146 | 73 | 138 | 20 | 22 | 3 | 142 | 50 |
| | ? | 3 | 20 | 4 | 3 | 3 | 2 | 6 | 11 | 3 | 3 | 9 | 13 | 10 | 7 | 12 | 22 |

Table 4 contains top 10 class outliers detected from the Votes dataset.[3] To get a further insight into the properties of these detected outliers, we use Table 5 to describe the value distribution of democrat and republican class in the whole dataset. Table 5 clearly shows that the votes on most of the 16 issues, the attitude of the Democrat and the Republican differs significantly.

In order to have better understanding and explanation of these outliers, Figs. 9 and 10 present the content in Table 5 in a more vivid manner. From Table 4, the strongest outlier is the record (*republican*, $y, y, y, y, n, n, y, y, y, y, y, n, n, y, n, y$). His group reveals that his votes should be similar with other republicans. However, from Figs. 9 and 10 and his votes we can see that, only on the 4, 10, 13, 15, 16 issues he behaved like most of the republicans, while on the issues 1, 3, 5, 7, 8, 9, 12, 13, he acted as he is a democrat. He is so similar to democrats, not republicans.

Other outliers in Table 4 can be analyzed in the same manner. And we also develop an effective visualization method for such kinds of class outliers, as shown in Fig. 11.

In contrast, we implemented an traditional outlier mining algorithm described in Ramaswamy, Rastogi, and Kyuseok (2000) to find outliers according to their definitions. The top 10 outliers produced are shown in Table 6.

As shown in Tables 4 and 6, all the top 10 class outliers cannot be find out with the algorithm in Ramaswamy et al. (2000). It indicates that previous algorithms failed to find these meaningful and interesting outliers. Thus, it verifies that our concept of class outlier is of practical use.

### 5.2. Results on coil challenge 2000

The coil challenge 2000 data is based on a real world business problem. The training set contains over 5000 descriptions of customers, including the information of whether or not they have a caravan insurance policy. Each record consists of 86 attributes, containing socio demographic data (attribute 1–43) and product ownership (attributes 44–86). The socio demographic data is derived from zip codes. All customers living in areas with the same zip code have the same socio demographic attributes. Attribute 86, "CARAVAN: Number of mobile home policies", is the target variable A test set contains 4000 customers, and it has the same format as training set, only the target is missing.

The Coil Challenge 2000 offered the chance to work on two tasks based on an insurance domain data set. The tasks were given in[4] as:

1. Predict which customers are potentially interested in a caravan insurance policy.
2. Describe the actual or potential customers; and possibly explain why these customers buy a caravan policy.

For the prediction task, the underlying problem is to the find the subset of customers with a probability of having

[3] Since this special kind of combined class outliers can be effectively identified by the method presented in He et al. (2002). Thus, the experiments results are reproduced from He et al. (2002).

[4] http://www.dcs.napier.ac.uk/coil/challenge/thetasks.html

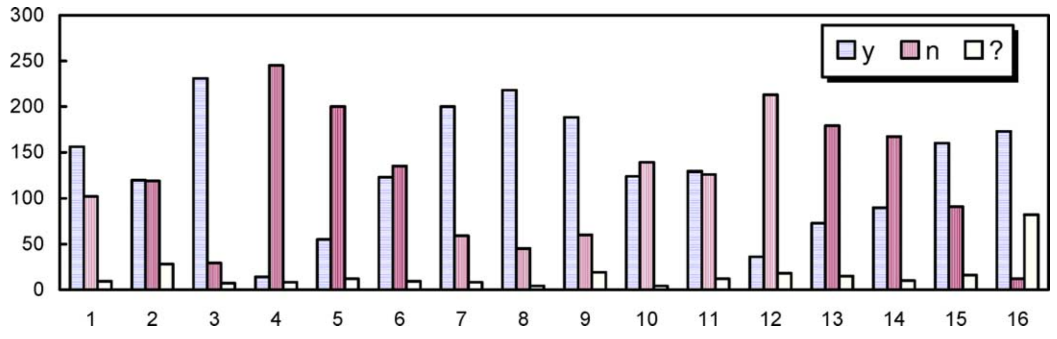


Fig. 9. The value distribution of democrat in the dataset.

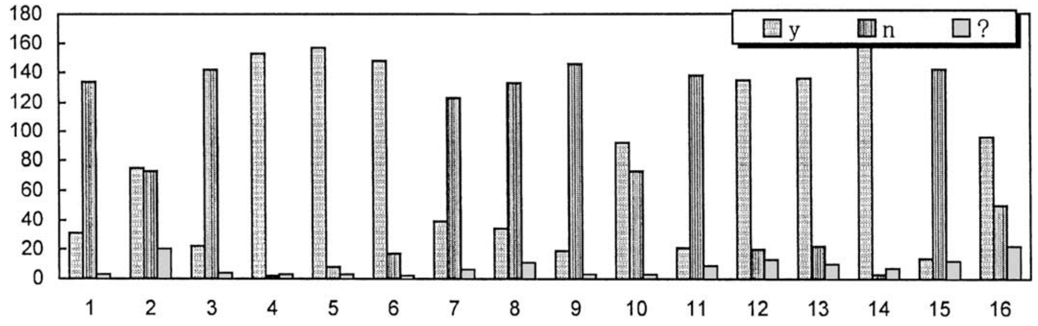


Fig. 10. The value distribution of republican in the dataset.

a caravan insurance policy above some boundary probability. The known policyholders can then be removed and the rest receives a mailing. The boundary depends on the costs and benefits such as of the costs of mailing and benefit of selling insurance policies. Task 1 was formulated more precisely as the demand for 20% of the 4000 test data record indexes. These 800 customers should be likely to be interested in a caravan insurance policy such that a virtual mail campaign for these addressees would be as successful as possible.

The following outline our class outlier based approach to solve the tasks of the Challenge, the experimental results, and some experience gained during the process.

The Coil data set encloses 85 possible, sometimes highly correlated input variables. Feature selection, finding a reliable subset of variables is the first important task. In this paper, we employ the feature selection method propose in Lewandowski (2000). The variables {59, 47, 25, 82, 65} are suggested as relevant variables for the problem in Lewandowski (2000). Therefore, only these five attributes are utilized in our mining process. As discussed in Section 4.2, the training set is divided into *P* and *N*, and the test dataset is taken as *T*. In our experiments, we let all class outlier based scoring methods to produce top 800 outliers (customers), and examine the number of policy owners in the chosen set of 800 customers. The class outlier detection algorithms *FindCCBLOF* and *FindCFPOF* that have been introduced in Sections 3.2 and 3.3 are tested to examine their effectiveness. For all the experiments, parameters

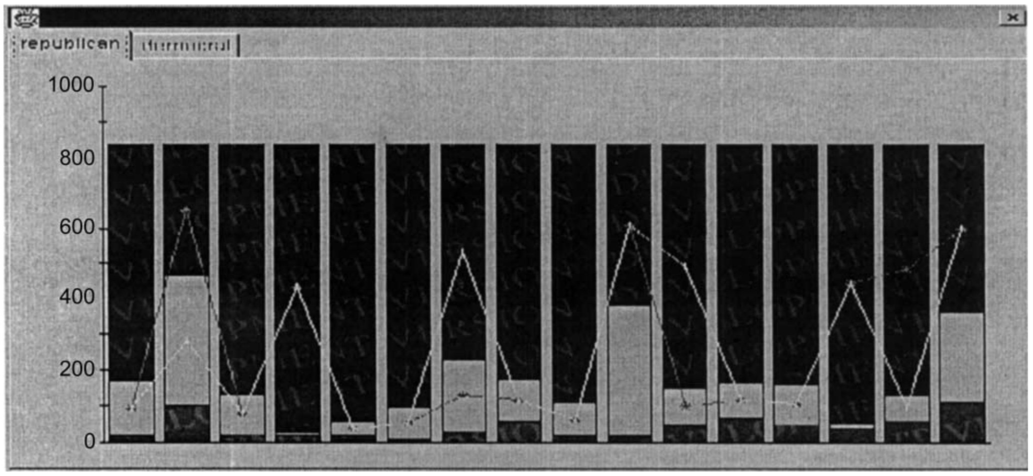


Fig. 11. Visualizing class outlier in *Rep*.

Table 6
Top 10 outliers produced by the algorithm in Ramaswamy et al. (2000)

| Label | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Democrat | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | ? | ? |
| Democrat | *y* | *n* | *y* | *n* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *n* | *y* |
| Democrat | *y* | ? | *y* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *y* | ? |
| Democrat | *n* | ? | *y* | *n* | ? | ? | *y* | *y* | *y* | *y* | ? | ? | *n* | *n* | *y* | *y* |
| Democrat | *n* | ? | *y* | *n* | *n* | *y* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *y* | *y* | *y* |
| Democrat | *y* | *n* | *y* | *n* | *n* | ? | *y* | *y* | *y* | *n* | ? | ? | *n* | ? | ? | ? |
| Republican | *y* | *y* | *n* | *y* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *y* |
| Republican | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* |
| Republican | *n* | ? | *n* | *y* | *y* | *y* | *n* | *n* | *n* | *n* | *n* | *y* | *y* | *y* | *n* | *n* |
| Republican | *n* | *y* | *n* | *y* | *y* | *y* | ? | ? | *n* | *y* | *n* | *y* | ? | ? | ? | ? |

needed by *FindCCBLOF* algorithm are set to default as suggested in He et al. (2003). For *FindCFPOF* algorithm, the parameter *mini-support* for mining frequent patterns is fixed to 2%, and the maximal number of items in an itemset is set to 5. Moreover, Table 7 shows the experimental results.

Some important observations from Table 7 is summarized as follows:

(1) Fig. 12, adapted from Elkan (2001), shows a histogram of the scores achieved by the 43 individuals or teams that submitted entries to the contest. The winning entry identified 121 caravan policyholders among its 800

Table 7
Experimental results on coil challenge 2000 dataset

| Algorithm | Scoring function | | Number of true policy owners |
|---|---|---|---|
| FindCFPOF | [*TSF*] $SUCOF\ (p, T)$ | | 62 |
| | [*PSF*] $1-SUCOF\ (p, P)$ | | 80 |
| | [*NSF*] $SUCOF\ (p, N)$ | | 76 |
| | [*PTSF*] $W_T \times SUCOF\ (p, T) \oplus W_P \times (1-SUCOF\ (p, P))$ | $\oplus=+$ | 105 |
| | | $\oplus=\prod$ | 110 |
| | | $\oplus=S_q,\ q=3$ | 101 |
| | | $\oplus=S_\infty$ | 86 |
| | [*NTSF*] $W_T \times SUCOF\ (p, T) \oplus W_N \times SUCOF\ (p, N)$ | $\oplus=+$ | 65 |
| | | $\oplus=\prod$ | 65 |
| | | $\oplus=S_q,\ q=3$ | 65 |
| | | $\oplus=S_\infty$ | 76 |
| | [*PNSF*] $W_N \times SUCOF\ (p, N) \oplus W_P \times (1-SUCOF\ (p, P))$ | $\oplus=+$ | 111 |
| | | $\oplus=\prod$ | 111 |
| | | $\oplus=S_q,\ q=3$ | 102 |
| | | $\oplus=S_\infty$ | 85 |
| | [*PNTSF*] $W_T \times SUCOF\ (p, T) \oplus W_P \times (1-SUCOF\ (p, P)) \oplus W_N \times SUCOF\ (p, N)$ | $\oplus=+$ | 104 |
| | | $\oplus=\prod$ | 106 |
| | | $\oplus=S_q,\ q=3$ | 89 |
| | | $\oplus=S_\infty$ | 84 |
| FindCCBLOF | [*TSF*] $SUCOF\ (p, T)$ | | 87 |
| | [*PSF*] $1-SUCOF\ (p, P)$ | | 86 |
| | [*NSF*] $SUCOF\ (p, N)$ | | 76 |
| | [*PTSF*] $W_T \times SUCOF\ (p, T) \oplus W_P \times (1-SUCOF\ (p, P))$ | $\oplus=+$ | 94 |
| | | $\oplus=\prod$ | 99 |
| | | $\oplus=S_q,\ q=3$ | 108 |
| | | $\oplus=S_\infty$ | 99 |
| | [*NTSF*] $W_T \times SUCOF\ (p, T) \oplus W_N \times SUCOF\ (p, N)$ | $\oplus=+$ | 92 |
| | | $\oplus=\prod$ | 94 |
| | | $\oplus=S_q,\ q=3$ | 91 |
| | | $\oplus=S_\infty$ | 94 |
| | [*PNSF*] $W_N \times SUCOF\ (p, N) \oplus W_P \times (1-SUCOF\ (p, P))$ | $\oplus=+$ | 97 |
| | | $\oplus=\prod$ | 99 |
| | | $\oplus=S_q,\ q=3$ | 97 |
| | | $\oplus=S_\infty$ | 94 |
| | [*PNTSF*] $W_T \times SUCOF\ (p, T) \oplus W_P \times (1-SUCOF\ (p, P)) \oplus W_N \times SUCOF\ (p, N)$ | $\oplus=+$ | 102 |
| | | $\oplus=\prod$ | 95 |
| | | $\oplus=S_q,\ q=3$ | 101 |
| | | $\oplus=S_\infty$ | 100 |

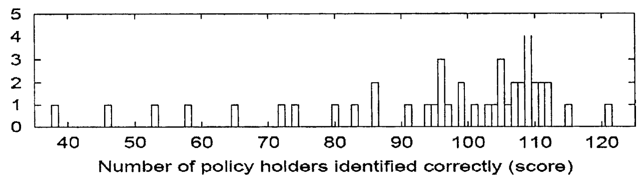


Fig. 12. Distribution of scores achieved by 43 entries.

top predictions. The next best methods identified 115 and 112 policyholders. The mean score was 95.4, with a standard deviation of 19.4. The distribution of scores is clearly not normal, with a long tail to the left. The most common score was 109, and the median score was 103 (Elkan, 2001).

Figs. 13 and 14 show the histograms of the scores achieved by the 19 scoring functions with *FindCFPOF* and *FindCCBLOF* algorithms, respectively. Our first claim is that the performance of class outlier based method is comparable to other popular techniques in the direct marketing applications. It should be noted that the results of scoring functions *NSF*, *PSF* and *TSF*, which only use one dataset, are also included in Figs. 13 and 14. Obviously, it can be known in advance that these functions will not produce satisfactory results, because they utilize only one dataset (Furthermore, the NSF function explores test dataset only). Thus, if we do not consider these functions, better average performance can be achieved.

(2) Which algorithm is better? From Figs. 13 and 14, the mean value of *FindCCBLOF* is larger than that of *FindCFPOF*, and standard deviation of *FindCCBLOF* is relatively smaller, which gives the evidence that *FindCC-BLOF* is better than *FindCFPOF*.

(3) How does the choice of scoring functions affect the results? From the results reported in Table 7, it is very clear that the scoring functions contain combine operator ($\oplus$) outperform those simple functions. This is because *true combined class outlier* based scoring functions utilize more available datasets. However, it is needed to point out that, simple scoring functions are also useful and interesting in their own rights in practice. For example, when the positive dataset (*P*) and negative dataset (*N*) are not available to the analyzer, that is, the direct marketing campaign is in its initial stage, the *TSF* function will be the only applicable function. It is also one important advantage of our class outlier based method that we can still detect potential customers when only the test dataset is available.

For those complicated scoring functions with combine operator ($\oplus$) as their component, they exhibit comparable performances. While in average, the *PNSF* function is a little better than the other functions. Hence, the *PNSF* function is more preferable than other alternatives.

(4) How does the choice of combine operator ($\oplus$) affect the results? From Table 7, the '+' operator and '$\prod$' operator are the clear winners. That is, for a specified

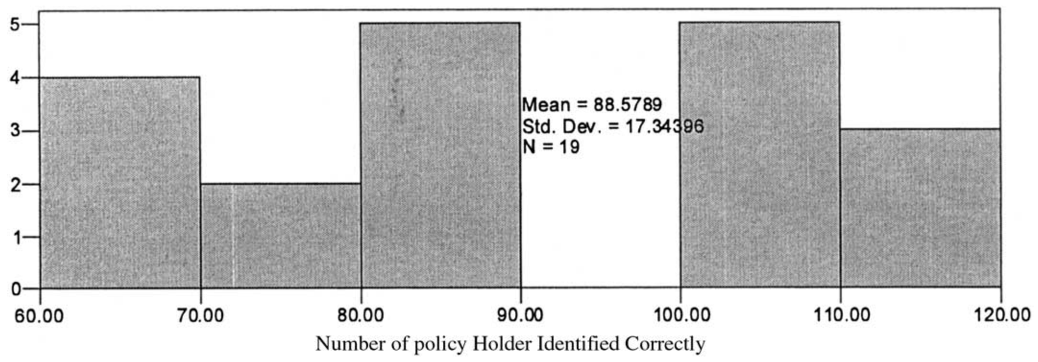


Fig. 13. Distribution of scores achieved by 19 entries with findCFPOF.

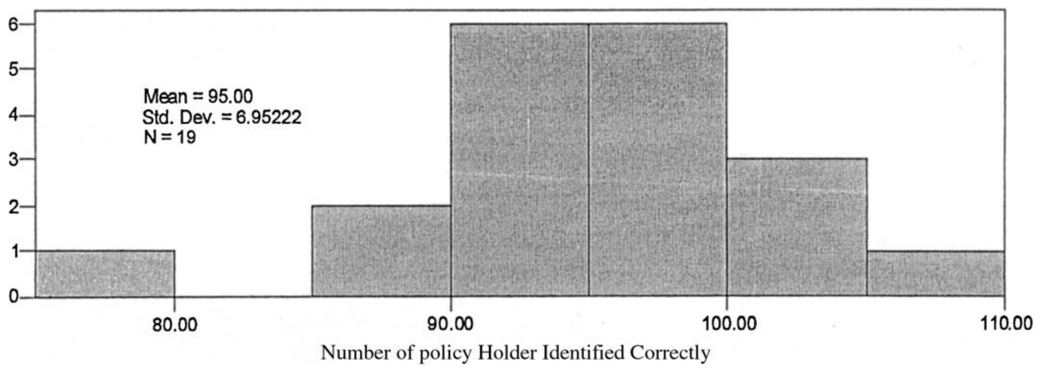


Fig. 14. Distribution of scores achieved by 19 entries with FindCCBLOF.

scoring function, the '+' operator and '$\prod$" operator outperform $S_q$ and $S_\infty$ in most cases. This observation suggests that the '+' operator and '$\prod$' operator will be better choices in practice for users. Moreover, further experiments show that with increase of $q$ in the $S_q$ operator, the performance of scoring functions will deteriorate.

In summary, the experimental results show that class outlier method is competitive in the direct marketing applications. Furthermore, it is more flexible than other popular techniques because different kinds of scoring functions are available for various situations, even for the situation that other method failed to work (It is the case that *only* test dataset is available).

## 6. Related work

In this section, we present prior work on the problem of outlier detection.

### 6.1. Traditional outlier detection

Previous methods for traditional outlier detection broadly fall into the following categories.

*Distribution based* methods in this category are previously conducted by the statistics community (Hawkins, 1980; Barnett & Lewis, 1994; Rousseeuw & Leroy, 1987). They deploy some standard distribution model (e.g. normal) and flag as outliers those points that deviate from the model. Recently, Yamanishi, Takeuchi and Williams (2000) used a Gaussian mixture model to present the normal behaviors and each datum is given a score based on changes in the model. High score indicates high possibility of being an outlier. This approach has been combined with a supervised-based learning approach to obtain general patterns for outlier (Yamanishi & Takeuchi, 2001). For arbitrary data sets without any prior knowledge of the distribution of points, we have to perform expensive tests to determine which model fits the data best, if any.

*Depth-based* is the second category for outlier mining in statistics (Johnson, Kwok, & Ng, 1998; Nuts & Rousseeuw, 1996). Based on some definition of depth, data objects are organized in convex hull layers in data space according to peeling depth, and outliers are expected to be finding out from data objects with shallow depth values.

*Deviation-based* techniques identify outliers by inspecting the characteristics of objects and consider an o bject that deviates these features as an outlier (Arning, Agrawal, & Raghavan, 1996).

*Distance based* method originally proposed by Knorr and Ng (1997), Knorr and Ng (1998), Knorr and Ng (1999) and Knorr, Ng, and Tucakov (2000). A distance-based outlier in a dataset $D$ is a data object with $pct\%$ of the objects in $D$ having a distance of more than $d_{\min}$ away from it. This notion generalizes many concepts from distribution-based approach and enjoys better computational complexity. It is further extended based on the distance of a point from its $k$th nearest neighbor Ramaswamy et al. (2000). After ranking points by the distance to its $k$th nearest neighbor, the top $k$ points are identified as outliers. Efficient algorithms for mining top-$k$ outliers are given. Alternatively, in the algorithm proposed by Angiulli and Pizzuti (2002), the outlier factor of each data point is computed as the sum of distances from its $k$ nearest neighbors. Bay and Schwabacher (2003) modify a simple algorithm based on nested loops to yield near linear time mining for distance-based outlier detection.

*Density based* This was proposed by Breunig, Kriegel, Ng, and Sander (2000). It relies on the local outlier factor (*LOF*) of each point, which depends on the local density of its neighborhood. In typical use, points with a high *LOF* are flagged as outliers. Tang, Chen, Fu and Cheung (2002) introduces a connectivity-based outlier factor (*COF*) scheme that improves the effectiveness of *LOF* scheme when a pattern itself has similar neighborhood density as an outlier. Three enhancement schemes over *LOF* are introduced in Chiu and Fu (2003). An effective algorithm for mining local outliers is proposed in Jin Tung, and Han (2001). The *LOCI* method Papadimitriou, Kitagawa, Gibbons and Faloutsos (2003) and low-density regularity method Hu and Sung (2003) further extended the density-based approach (Breunig et al., 2000).

*Clustering-based* outlier detection techniques regarded *small* clusters as outliers (Jiang, Tseng, & Su, 2001) or identified outliers by removing clusters from the original dataset (Yu, Sheikholeslami, & Zhang, 2002). The authors in He et al. (2003) further extended existing clustering based techniques by proposing the concept of *cluster-based local outlier*, in which a measure for identifying the outlier-ness of each data object is defined.

*Sub-Space based*. Multi- and high-dimensional data make the outlier mining problem more complex because of the impact of *curse of dimensionality* on algorithms' both performance and effectiveness. Aggarwal and Yu (2001) discussed a new technique for outlier detection, which finds outliers by observing the density distribution of projections from the data. Wei, Qian, Zhoul, Jin, and Yu (2003) introduces an outlier mining method based on a hypergraph model to detect outliers in categorical dataset.

*Support vector based*. Support vector novelty detector (*SVND*) was recently developed. The first *SVND* is proposed by Tax and Duin (1999), which estimates a sphere to contain all the normal data patterns with the smallest radius; the outliers can be identified from the normal data patterns by calculating the distance of a data pattern to the center of the sphere. Another alternative *SVND* is proposed by Schölkopf et al. (2001). Instead of a sphere, a function is estimated to separate the region of normal data patterns from that of outliers with maximal margin, thus detecting the outliers from the normal data patterns. Cao, Lee, and Chong (2003) further extended the *SVND* method. Petrovskiy (2003) combine kernel methods and fuzzy clustering methods.

*Neutral network based*. The replicator neutral network (*RNN*) is employed to detect outliers by Harkins, He, Willams, and Baster (2002) and Willams, Baster, He, Harkins and Gu (2002). The approach is based on the observation that the trained neutral network will reconstruct some small number of individuals poorly, and these individuals can be considered as outliers. The outlier factor for ranking data is measured according to the magnitude of the reconstruction error.

### 6.2. Class outlier detection

To our knowledge, the class outlier detection problem has only been explicitly considered in He, Deng, and Xu (2002) and Papadimitriou and Faloutsos (2003). A semantic outlier (He, Deng, & Xu, 2002) is a data point, which behaves differently with other data points in the same class, while looks 'normal' with respect to data points in another class. In contrast, Papadimitriou and Faloutsos (2003) propose the cross-outlier detection problem. That is, given two sets (or classes) of objects, find those that deviate with respect to the other set.

Both He, Deng, and Xu, (2002) and Papadimitriou and Faloutsos (2003) consider special cases of class outlier. To be more precise, semantic outlier is a kind of *true combined class outlier*, while cross-outlier is the *reference class outlier*.

## 7. Conclusions

In this paper, we present the problem of class outlier detection by generalizing two pioneer works (He, Deng, & Xu, 2002; Papadimitriou & Faloutsos, 2003) in this field. We argue that the generalization is a non-trivial. We present several kinds of class outliers and discuss their relationships in a systematic way.

Beyond introducing the problem, we present effective algorithms by extending existing outlier detection algorithms to this domain. Furthermore, its potential applications in customer migration and direct marketing are discussed. We demonstrate the effectiveness of our ideas on real datasets that have been widely used.

## Acknowledgements

The authors wish to thank Spiros Papadimitriou for sending us their SSTD'03 paper. This work was supported by the High Technology Research and Development Program of China (Grant No. 2002AA413310, Grant No. 2003AA4Z2170, Grant No. 2003AA413021), the National Nature Science Foundation of China (Grant No. 40301038) and the IBM SUR Research Fund.

## References

Agrawal, R., & Srikant, R. (1994). *Fast algorithms for mining association rules Proceedings of the VLDB94* pp. 478–499.

Aggarwal, C., & Yu, P. (2001). *Outlier detection for high dimensional data Proceedings of the SIGMOD01* pp. 37–46.

Angiulli, F., & Pizzuti, C. (2002). *Fast outlier detection in high dimensional spaces Proceedings of the PKDD02* pp. 25–36.

Arndt D, Gersten W. (2001). Data management in analytical customer relationship management. Data Mining for Marketing Applications Workshop at ECML/PKDD.

Arning, A., Agrawal, R., & Raghavan, P. (1996). *A linear method for deviation detection in large databases Proceedings of the KDD96* pp. 164–169.

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data*. New York: Wiley.

Bay, S. D., & Schwabacher, M. (2003). *Mining distance based outliers in near linear time with randomization and a simple pruning rule Proceedings of the KDD03*.

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). *LOF: identifying density-based local outliers Proceedings of the SIGMOD00* pp. 93–104.

Cao, L. J., Lee, H. P., & Chong, W. K. (2003). Modified support vector novelty detector using training data with outliers. *Pattern Recognition Letters*, *24*(14), 2479–2487.

Chiu, A. L., & Fu, A. W. (2003). *Enhancements on local outlier detection Proceedings of the IDEAS03*.

ECCS, (1999). CRM Defining customer relationship marketing and management. http://www.eccs.uk.com/crm/crmdefinitions/define.asp.

Elkan, C. (2001). *Magical thinking in data mining: lessons from CoIL challenge 2000 Proceedings of the KDD01*.

Gibson, D., Kleiberg, J., & Raghavan, P. (1998). *Clustering categorical data: an approach based on dynamic systems Proceedings of the VLDB98* pp. 311–323.

Han, J., Pei, J., & Yin, J. (2000). *Mining frequent patterns without candidate generation Proceedings of the SIGMOD00* pp. 1–12.

Harkins, S., He, H., Willams, G. J., & Baster, R. A. (2002). *Outlier detection using replicator neural networks Proceedings of the DaWaK02* pp. 170–180.

Hawkins, D. (1980). *Identification of outliers*. Reading, London: Chapman & Hall.

He, Z., Deng, S., & Xu, X. (2002). *Outlier detection integrating semantic knowledge Proceedings of the WAIM02* pp. 126–131.

He, Z., Xu, X., & Deng, S. (2003). Discovering cluster based local outliers. *Pattern Recognition Letters*, *24*(9–10), 1651–1660.

He, Z., Xu, X., & Deng, S. (2002a). Squeezer: an efficient algorithm for clustering categorical data. *Journal of Computer Science and Technology*, *17*(5), 611–624.

He, Z., Xu, X., & Deng, S. (2002b). *FP-outlier: Frequent pattern based outlier detection Technology Report*. Harbin Institute of Technology.

Hu, T., & Sung, S. Y. (2003). Detecting pattern-based outliers. *Pattern Recognition Letters*, *24*(16), 3059–3068.

Jiang, M. F., Tseng, S. S., & Su, C. M. (2001). Two-phase clustering process for outliers detection. *Pattern Recognition Letters*, *22*(6,7), 691–700.

Jin, W., Tung, A. K. H., & Han, J. (2001). *Mining top-n local outliers in large databases Proceedings of the KDD01* pp. 293–298.

Johnson, T., Kwok, I., & Ng, R. (1998). *Fast computation of 2-dimensional depth contours Proceedings of the KDD98* pp. 224–228.

Knorr, E., & Ng, R. (1997). *A unified notion of outliers: Properties and computation Proceedings of the KDD97* pp. 219–222.

Knorr, E., & Ng, R. (1998). *Algorithms for mining distance-based outliers in large datasets Proceedings of the VLDB98* pp. 392–403.

Knorr, E., & Ng, R. (1999). *Finding intentional knowledge of distance-based outliers Proceedings of the VLDB99* pp. 211–222.

Knorr, E., Ng, R., & Tucakov, T. (2000). Distance-based outliers: Algorithms and applications. *VLDB Journal*, *8*(3 and 4), 237–253.

Lewandowski, A. (2000). *How to detect potential customers CoIL Challenge 2000: The Insurance Company Case, Technical Report 2000–09*. Netherlands: Leiden Institute of Advanced Computer Science.

Ling, C. X., & Li, C. (1998). *Data mining for direct marketing: Problems and solutions Proceedings of the KDD98* pp. 73–79.

Liu, B., Ma, Y., Wong, C. K., & Yu, P. S. (2003). Scoring the data using association rules. *Applied Intelligence*, *18*(2), 119–135.

Merz, G., & Murphy, P. (1996). *Uci repository of machine learning databases Technical Report*. University of California: Department of Information and Computer Science http://www.ics.uci.edu/mlearn/MLRepository.html.

Nuts, R., & Rousseeuw, P. (1996). Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis*, *23*, 153–168.

Papadimitriou, S., & Faloutsos, C. (2003). *Cross-outlier detection Proceedings of the of SSTD03* pp. 199–213.

Papadimitriou, S., Kitagawa, H., Gibbons, P. B., & Faloutsos, C. (2003). *Fast outlier detection using the local correlation integal Proceedings of the ICDE03*.

Petrovskiy, M. (2003). *A hybrid method for patterns mining and outliers detection in the web usage log Proceedings of the AWIC03* pp. 318–328.

Ramaswamy, S., Rastogi, R., & Kyuseok, S. (2000). *Efficient algorithms for mining outliers from large data sets Proceedings of the SIGMOD00* pp. 93–104.

Rousseeuw, P., & Leroy, A. (1987). *Robust regression and outlier detection*. New York: Wiley.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high dimensional distribution. *Neural Computation*, *13*(7), 1443–1472.

Setnes, M., & Kaymak, U. (2001). Fuzzy modeling of client preference from large data sets: an application to target selection in direct marketing. *IEEE Transactions on Fuzzy Systems*, *9*(1), 153–163.

Skinner S. J., (1990). Marketing Houghton Mifflin, Boston et al.

SPSS Inc (1993). *SPSS CHAID for Windows 6.0*. Englewood Cliffs, NJ: Prentice-Hall.

Tang, T., Chen, Z., Fu, A. W., & Cheung, D. W. (2002). *Enhancing effectiveness of outlier detections for low density patterns Proceedings of the PAKDD02* pp. 535–548.

Tax, D. M. J., & Duin, R. P. W. (1999). Support vector data description. *Pattern Recognition Letters*, *20*(11–13), 1191–1199.

The Coil dataset can found at: http://www.liacs.nl/(putten/library/cc2000/.

Wei, L., Qian, W., Zhou, A., Jin, W., & Yu, J. X. (2003). *HOT: hypergraph-based outlier test for categorical data Proceedings of the PAKDD03* pp. 399–410.

Willams, G. J., Baster, R. A., He, H., Harkins, S., & Gu, L. (2002). *A comparative study of rnn for outlier detection in data mining Proceedings of the ICDM02* pp. 709–712.

Yamanishi, K., & Takeuchi, J. (2001). *Discovering outlier filtering rules from unlabeled data-combining a supervised learner with an unsupervised learner Proceedings of the KDD01* pp. 389–394.

Yamanishi, K., Takeuchi, J., & Williams, G. (2000). *On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms Proceedings of the KDD00* pp. 320–325.

Yao, Y., Zhong, N., Huang, J., Ou, C., & Liu, C. (2002). Using market value functions for targeted marketing data mining. *International Journal of Pattern Recognition and Artificial Intelligence*, *16*(8), 1117–1132.

Yu, D., Sheikholeslami, G., & Zhang, A. (2002). Findout: finding out outliers in large datasets. *Knowledge and Information Systems*, *4*(4), 387–412.

Zaki, M. J. (2000). Scalable algorithms for association mining. *IEEE Transactions On Knowledge and Data Engineering*, *12*(3), 372–390.