

A comparative Study of Outlier Mining and Class Outlier Mining

Motaz K. Saad¹ and Nabil M. Hewahi
Computer Science Department, Islamic University of Gaza – Palestine

Abstract

Outliers can significantly affect data mining performance. Outlier mining is an important issue in knowledge discovery and data mining and has attracted increasing interests in recent years. Class outlier is promising research direction. Few researches have been done in this direction. The paper theme has two main goals: the first one is to show the significance of *Class Outlier Mining* by discussing a comparative study between a *Class Outlier* detection method called *Class Outlier Distance Based (CODB)* and a conventional Outlier detection method. The second goal is to introduce *Enhanced Class Outlier Distance Based (ECODB)* algorithm which is enhancement of *CODB* algorithm. *ECODB* reduces *CODB* parameters using a heuristic approach. The experimental results show that *CODB* can detect *Class Outliers* that cannot be detected using conventional *Outlier* detection methods. The experiments also show that *ECODB* works efficiently as *CODB*.

Keywords: Outlier; Class Outlier; Distance Based approach.

1. Introduction

Outlier Mining is the problem of detecting rare events, deviant objects, and exceptions. It is an important data mining issue in knowledge discovery; it has attracted increasing interests in recent years. Outliers are observations that deviate so much from other observations as to arouse suspicion that they were generated by a different mechanism. They do not comply with the general behavior of the data. Some of very well known outlier detection methods are statistical based (distribution based) [3, 7, 21], clustering [2, 6, 13], depth based [12], distance based [4, 13-17, 20], density based [5], and model based (Neural Networks) [8].

¹ Corresponding Author: Motaz K. Saad

Email: msaad@iugaza.edu

© 2009-2012 All rights reserved. ISSR Journals

A Class Label is an attribute chosen among a set of attributes in a given database based on the request of the user and type of the application. A class label could be medical diagnoses, credit or loan approval decision, customer classification... etc.). The conventional Methods (*Outlier Mining*) is to find exception or rare cases in a dataset irrespective of the class label of these cases, they are considered rare events with respect to the whole dataset. *Class Outlier Mining* is to find suspicious instances taking into account the class label [9, 10, 11, 19]. *Outlier Mining* cannot detect cases which behave (deviate) differently from its class, whereas the *Class Outlier Mining* can do [11].

During getting acceptance for publication of the paper [11], one of the reviewers claimed that conventional outlier detection methods (distance based outlier detection methods) can be adopted to detect class outliers by looking for outlier instances of each class separately and applying this approach repeatedly for the subsets of instances of each class. We shall call this claim hypothetical (HYPO). In this paper, we refute this HYPO and show the significance of *Class Outlier Mining* by discussing a comparative study between *CODB* (*Class Outlier Mining* method) and Distance Based Outlier detection (*Outlier Mining* method). Beyond the comparative study discussion, we introduce *ECODB* which is an enhancement to Class Outlier Distance Based (*CODB*) Algorithm proposed by Hewahi and Saad [11]. The enhancement is focused on removing certain parameters included in the main formula of *CODB* algorithm. Those parameters should be entered by the user based on previous runs (trial and error).

The rest of the paper is organized as follows: section 2 reviews related works, section 3 discusses a comparative study between *Outlier Mining* and *Class Outlier Mining*. Section 4 presents the enhanced *CODB* algorithm (*ECODB*), its steps and its experimental results compared to *CODB*. Finally, we draw the conclusion in section 5.

2. Related Work

The problem of “given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels” has only been explicitly considered in [9, 10, 11, 19]. The proposed methods are Semantic outlier [9], Cross-Outlier [19], Class Outlier [10, 11]. It is obvious that other outliers detection methods cannot detect such type of outlieriness as we shall show in the next section.

He, et al. [9] tried to find meaningful outliers that called Semantic Outlier Factor (*SOF*). The approach is based on applying a clustering algorithm on a dataset with a class label, it is expected that the instances in every output cluster are to be identified with the same class label. However, this is not always true. The Semantic outlier definition 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.

Papadimitriou and Faloutsos [19] tried to solve the problem: Given two sets (or classes) of objects, find those which deviate with respect to the other set. Those points are called Cross-outlier, and the problem is identified by *Cross-outlier* detection. The approach assumes a primary set P in which we want to discover *cross-outliers* with respect to a reference set R (detecting outlying observations: discover points $p \in P$ that “arouse suspicions” with respect to points $r \in R$). The proposed solution is to use a statistically intuitive criterion for outlier flagging (the local neighborhood size differ more than three standard deviation from the local average). Papadimitriou and Faloutsos [19] considered that some single class approaches may be modified to deal with multiple classes, but the

task is nontrivial. The authors generally considered the existing approaches for the single-set problem, are not immediately extensible to cross-outlier detection. Also several outlier definitions themselves cannot be extended.

He, et al. [10] tried to find a general framework to contributions presented in [24, 13] by proposing a practical solution and extending existing outlier detection algorithms. The generalization does not consider only outliers that deviate with respect to their own class, but also outliers that deviate with respect to other classes. In addition, potential applications of customer relationship management (*CRM*) are introduced.

Hewahi and Saad [11] proposed a novel definition for class outlier and new method for mining class outliers based on distance-based approach and nearest neighbors. This method is called *CODB* algorithm and is based on the Concept of Class Outlier Factor (*COF*) which represents the degree of being a class outlier for a data object. The main key factors of computing *COF* for an instance are the probability of the instance's class among its neighbors, the deviation of the instance from the instances of the same class, and the distance between the instance and its *K* nearest neighbors.

3. Outlier Mining vs. Class Outlier Mining

A popular method of identifying outliers is by examining the distance to an example's nearest neighbors [4, 13-17, 20]. In this approach, one looks at the local neighborhood of points for an example typically defined by the *K* nearest examples (also known as neighbors). If the neighboring points are relatively close, then the example is considered normal; if the neighboring points are far away, then the example is considered unusual. The advantages of distance-based outliers are that, no explicit distribution needs to be defined to determine unusualness, and it can be applied to any feature space for which we can define a distance measure. Distance based outliers can be defined as top *n* examples whose distance to the *Kth* nearest neighbor is greatest [20].

Class Outlier Mining is the problem of “given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels [9, 10, 11, 19]. Based on *CODB* algorithm, Class Outliers are the top *N* instances which satisfy the following [11]:

1. The *K*-Distance (*KDist*) to its *K* nearest neighbors is the least.
2. Its Deviation is the greatest.
3. Has different class label of its *K* nearest neighbors' class.

The Class Outlier Factor (*COF*) for the instance *T* based on *CODB* concept is defined as [11]:

$$COF(T) = K \times PCL(T, K) + \alpha \times \frac{1}{Deviation(T)} + \beta \times KDist(T) \quad (1)$$

Where *PCL(T, K)* is the Probability of the class label of the instance *T* with respect to the class labels of its *K* Nearest Neighbors. *Deviation(T)* is how much the instance *T* deviates from instances of the same class, and computed by summing the distances between the instance *T* and every instance belong to the same class of the instance. *KDist(T)* is the summation of distance between the instance *T* and its *K* nearest neighbors. α and β are factors to control the importance and the effects of *Deviation(T)* and *KDist(T)*, and they determined by trial and error. The term $K \times PCL(T, K)$ value rang

fall into $[1 - K]$. α and β should be chosen to make $Deviation(T)$ and $KDist(T)$ terms fall into $[0 - 1]$. The main concept of *CODB* algorithm is to rank each instance in the given dataset using formula (1), and keep only top N instances least *COF* value.

The main difference between *Outlier Mining* and *Class Outlier Mining* is the consideration of class label attribute. To prove that the *HYPO* mentioned above is false, and to show the significance of class outlier mining, we conduct a comparative study between the *CODB* algorithm and a well known outlier mining method based on distance measure proposed by Ramaswamy, Rastogi and Shim [20], which is primarily a statistical outlier search based on a distance measure similar to the $DB(p,D)$ -Outlier Search from Knorr and Ng [14]. The distance based outlier method will utilize a distance search through the K^{th} nearest neighborhood, so it implements some sort of locality as well [18]. The distance based outlier detection will be applied based on *HYPO* and then its results are compared with those obtained by *CODB* algorithm.

Both *CODB* [11] and Distance Based Outlier detection [20] algorithms are applied on various data set including vote-84, credit-a, and hepatitis. All datasets are available publically at *UCI-ML* repository [1]. All missed values are replaced with mean and mode for numeric and nominal attributes respectively.

The dataset of house-vote-84 [1] includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes. There are 16 attributes, all Boolean valued with Yes (denoted as "y") and No (denoted as "n"), there are 435 instances belonging to two classes, i.e. democrats or republicans. The class distribution is 61.38% Democrats, 38.62% Republicans. Missing attribute values are denoted by "?" in the original dataset. It is important to recognize that "?" in this database does not mean that the value of the attribute is unknown or missed, it means simply, that the value is neither "y" nor "n" [1]. So we replaced all "?" by "noVote" value to represent the real position of the voter, and to avoid handling it as a missed value. The dataset of credit approval (credit-a) [1] contains 690 instances belonging to two classes, i.e. "+" or "-" for credit approval, described by 16 attributes among which 6 attributes are continuous and the remaining 10 attributes are categorical. The class distribution is 44.49% "+", 55.51% "-". All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data [1]. The dataset of hepatitis [1] contains 155 instances belonging to two classes, i.e. positive or negative for hepatitis, described by 20 attributes, among which 6 attributes are continuous and the remaining 13 attributes are categorical. The class distribution is 20.67% DIE, 79.35% LIVE.

Based on the suggested approach of adopting conventional outlier mining to detect class outliers, we divided each dataset to many sub datasets based on the number of classes in each dataset, where each sub dataset includes all instances that belong to one of the class values of the class attribute.

Table 1 shows the top 20 class outliers detected by *CODB* for vote vote-84, credit-a, and hepatitis datasets. Only 3 of 20, and 1 of 20 instances from top 20 Classed Outliers are detected using conventional Distanced based Outlier detection method for vote84 and credit-a dataset respectively. However, 16 of 20 instances from top 20 Classed Outliers are detected using Distanced based method for hepatitis dataset. Instance that detected using *CODB* and *Distance Based Outlier* detection are shaded in table 1.

Referring to table 1, instance 407 in vote-84 dataset (1st rank) is not detected in top 20 outliers using adopted conventional distance based outlier detection method in democrat subset although it has the highest deviation value with $PCL(407, 7)$ value of 1/7. Table 2 shows the 7NN of instance 407 (including itself (shaded in the table)) in democrat sub dataset and vote-84 dataset. The 7NN of instance 407 in democrat subset are close enough to it which makes it not in the top 20 outliers. But the 7NN of the same instance in vote-84 dataset which belong to republican class are closer to it than its 7NN in democrat subset, the same instance also has the highest deviation, so all of that make it the 1st rank Class Outlier.

Table 1: The top 20 class outliers detected by *CODB* for vote84, credit-a, and hepatitis datasets, Instances that detected using *CODB* and *Distance based Outlier* method are shaded.

vote-84				credit-a				hepatitis			
Rank	#	Class	COF	Rank	#	Class	COF	Rank	#	Class	COF
1	407	democrat	1.71	1	115	-	1.20	1	31	DIE	1.66
2	375	democrat	1.92	2	523	-	1.23	2	35	DIE	1.82
3	388	democrat	1.92	3	110	-	1.29	3	134	DIE	1.90
4	161	democrat	1.97	4	99	-	1.34	4	98	DIE	1.97
5	267	republican	2.04	5	320	+	1.41	5	128	LIVE	2.74
6	71	republican	2.13	6	319	+	1.42	6	120	DIE	2.89
7	77	democrat	2.16	7	48	+	1.43	7	71	DIE	2.92
8	325	democrat	2.97	8	348	-	1.50	8	30	DIE	3.04
9	160	democrat	2.97	9	546	-	1.56	9	76	DIE	3.12
10	382	democrat	3.02	10	116	-	1.57	10	126	LIVE	3.76
11	176	republican	3.04	11	13	+	1.59	11	121	LIVE	3.84
12	384	democrat	3.05	12	78	-	1.67	12	109	DIE	3.88
13	365	democrat	3.06	13	598	+	1.68	13	94	DIE	3.97
14	6	democrat	3.09	14	10	+	1.69	14	146	DIE	3.98
15	355	republican	3.10	15	103	-	1.73	15	103	LIVE	3.98
16	164	democrat	3.13	16	531	-	1.74	16	147	DIE	3.98
17	402	republican	3.30	17	529	-	1.75	17	67	DIE	3.99
18	151	democrat	3.31	18	537	-	1.75	18	39	LIVE	4.02
19	173	democrat	3.93	19	105	-	1.75	19	135	LIVE	4.05
20	75	democrat	4.06	20	227	+	1.76	20	153	LIVE	4.10

Table 2: the 7NN of instance 407 in democrat subset and vote84 dataset

7NN of instance 407 in democrat subset		7NN of instance 407 in vote-84 dataset	
Instance	Distance	Instance	Distance
n,y,y,y,y,n,n,n,y,n,y,y,n,n, democrat	2.06	y,n,n,y,y,y,n,n,n,n,y,y,y,n,n, republican	1.0
n,n,n,n,y,y,y,n,n,n,n,y,y,y,n,y, democrat	2.0	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, republican	1.0
n,n,n,n,y,y,n,n,n,y,y,y,y,y,n,y, democrat	1.73	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, republican	1.0
n,y,y,y,y,n,n,n,n,y,y,noVote,y,n,n, democrat	1.73	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, republican	1.0
n,n,noVote,n,y,y,n,n,n,n,y,y,y,y,n,y, democrat	1.73	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, republican	1.0
n,y,n,y,y,n,n,n,n,y,y,n,y,n,n, democrat	1.41	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, republican	1.0
n,n,n,y,y,y,n,n,n,n,y,y,y,y,n,n, democrat	0.0	n,n,n,y,y,y,n,n,n,n,n,y,y,y,n,n, democrat	0.0

The experimental results show the significance of *CODB* and its advantages to detect Class outliers that cannot be detected using Distanced based outlier detection method.

4. Enhanced Class Outlier - Distance Based Algorithm *ECODB*

To get rid of trial and error for specifying α and β values and still have the same performance of *CODB* algorithm, we introduce *ECODB* algorithm. The Class Outlier Factor (*COF*) for an instance T based on *ECODB* algorithm eliminates α and β parameters by normalizing *Deviation*(T) and *KDist*(T) terms. From the Class Outlier definition, the new *COF*(T) formula is defined as follows:

$$COF(T) = K \times PCL(T, K) - norm(Deviation(T)) + norm(KDist(T)) \quad (2)$$

Where $K \times PCL(T, K)$ value fall into $[1 - K]$. $norm(Deviation(T))$ and $norm(KDist(T))$ are the normalized value of *Deviation*(T) and *KDist*(T) respectively and their value fall into the range $[0 - 1]$ and they computed as follows

$$norm(Deviation(T)) = (Deviation(T) - MinDev) / (MaxDev - MinDev) \quad (3)$$

$$norm(KDist(T)) = (KDist(T) - MinKDist) / (MaxKDist - MinKDist) \quad (4)$$

Where,

MaxDev is the highest deviation value for top N class outliers,

MinDev is the lowest deviation value for top N class outliers,

MaxKDist is the highest *KDist* value for top N class outliers,

And *MinKDist* is the lowest *KDist* value for top N class outliers.

The user can normalize *Deviation*(T) and *KDist*(T) into different ranges to give more importance to a specific factor based on the application domain. *min-max* normalization can be used for this purpose. *min-max* normalization performs a linear transformation on the original data. Suppose that *Min* and *Max* are the minimum and the maximum values of set of values. *min-max* normalization maps a value v to v' in the range $[newMin, newMax]$ by computing

$$v' = \frac{v - Min}{Max - Min} \times (NewMax - NewMin) + NewMin \quad (5)$$

The *ECODB* algorithm steps are as follows:

1. For a given dataset, compute *PCL*(T, K) for all instances.
2. Maintain a list of top N instance with least *PCL*(T, K) value.
3. For each instance in the top N list, compute *Deviation*(T) and *KDist*(T), and maintain *MaxDev*, *MinDev*, *MaxKDist*, and *MinKDist* values.
4. Compute *COF* value for all instances in the top N list according to formula 2.
5. Resort the top N list in ascending order according to their *COF* value.

Both *ECODB* and *CODB* are implemented using *Weka* [22] and applied on three datasets: vote-84, credit-a, and hepatitis dataset. Table 3 shows the top 20 class outlier detected by *ECODB* for vote-84, credit-a, and hepatitis datasets. 18 of 20, 16 of 20, and 20 of 20 instances have the same/similar rank (we mean by similar close to the same rank with maximum difference of five ranks) obtained by *CODB* for vote84, credit-a, and hepatitis datasets respectively. Instances that have same/similar rank are shaded in table 3.

Instances which not detected in top 20 list are detected in later rank. For example, instance 215 in vote-84 dataset has the rank 22 using *CODB*. It is to be noted that our obtained results by *ECODB* is very close to that of *CODB* algorithm with top 20, we expect to have also still better results if the top N is more widen. The experimental results reflect that *ECODB* work efficiently as *CODB* without the need to input α and β . The only needed inputs for *ECODB* are K (number of nearest neighbors) and N (the number of top class outliers), which are also required in the conventional outlier method.

Table 3: The top 20 class outliers detected by *ECODB* for vote84, credit-a, and hepatitis datasets Instance that have same/similar rank using *ECODB* and *CODB* are shaded

vote-84				credit-a				hepatitis			
Rank	#	Class	COF	Rank	#	Class	COF	Rank	#	Class	COF
1	407	democrat	0.00	1	115	-	0.22	1	31	DIE	0.99
2	375	democrat	0.38	2	523	-	0.22	2	35	DIE	1.27
3	388	democrat	0.43	3	99	-	0.28	3	134	DIE	1.43
4	161	democrat	0.58	4	116	-	0.35	4	98	DIE	1.55
5	77	democrat	0.95	5	105	-	0.37	5	128	LIVE	1.59
6	267	republican	1.24	6	110	-	0.38	6	120	DIE	2.41
7	71	republican	1.37	7	78	-	0.41	7	71	DIE	2.46
8	325	democrat	1.52	8	348	-	0.46	8	126	LIVE	2.53
9	160	democrat	1.56	9	103	-	0.65	9	121	LIVE	2.64
10	382	democrat	1.60	10	318	+	0.77	10	30	DIE	2.68
11	384	democrat	1.70	11	321	+	0.87	11	103	LIVE	2.82
12	365	democrat	1.71	12	227	+	0.90	12	76	DIE	2.83
13	6	democrat	1.73	13	48	+	0.98	13	135	LIVE	2.94
14	164	democrat	1.79	14	206	+	1.04	14	39	LIVE	2.97
15	151	democrat	2.04	15	58	+	1.10	15	109	DIE	3.38
16	176	republican	2.25	16	320	+	1.11	16	94	DIE	3.55
17	355	republican	2.34	17	270	+	1.15	17	146	DIE	3.57
18	402	republican	2.74	18	13	+	1.22	18	147	DIE	3.57
19	215	democrat	2.78	19	10	+	1.23	19	67	DIE	3.59
20	242	republican	3.94	20	269	+	1.25	20	6	DIE	3.97

5. Conclusion

In this paper, a comparative study has been discussed between *CODB* and Distance Based Outlier detection method. The Comparative discussion showed the significance of *ECODB* in detection Class Outliers that cannot be detected using conventional outlier detection method. Beyond the comparative study, we introduced Enhancement to *Class Outlier Distance based (CODB)* algorithm which called *Enhanced Class Outlier Distance Based ECODB*. The enhancement includes reducing *CODB* parameters using normalization techniques. The experimental results show that *ECODB* work

efficiently as *CODB*. The future research directions include building a hybrid formula that might serve both *Outlier Mining* and *Class Outlier Mining* detection method.

References

- [1] Asuncion, A. & Newman, D.J. *UCI Machine Learning Repository* [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, School of Information and Computer Science. 2007.
- [2] Barbarà, D., Chen, P.: *Using the fractal dimension to cluster datasets*, In: Proc. KDD, pp. 260–264, 2000.
- [3] Barnett, V., Lewis, T.: *Outliers in Statistical Data*, John Wiley, 1994.
- [4] Bay, S. D., and Schwabacher, M.: *Mining Distance-Based Outliers in Near Linear Time with Randomization and a Simple Pruning Rule*, Proc. of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003.
- [5] Breunig, M., Kriegel, H., Ng, R., Sander, J.: *LOF: Identifying density based local outliers*, In: Proc. SIGMOD Conf, pp. 93–104, 2000.
- [6] Ester M., Kriegel H.-P., Sander J., Xu X.: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96), Portland, OR. pp. 226-231, 1996.
- [7] Hawkins, D.: *Identification of Outliers*, Chapman and Hall, 1980.
- [8] Hawkins, S., He, H. X., Williams, G. J., Baxter, R. A.: *Outlier detection using replicator neural networks*, In Proc. of the Fifth Int. Conf. and Data Warehousing and Knowledge Discovery (DaWaK02), 2002.
- [9] He, Z., Deng, S., Xu., X.: *Outlier detection integrating semantic knowledge*, In: Proc. of WAIM'02, pp. 126-131, 2002.
- [10] He, Z., Xu, X., Huang, J., Deng, S.: *Mining Class Outliers: Concepts, Algorithms and Applications in CRM, Expert Systems with Applications (ESWA'04)*, 27(4): pp. 681-697, 2004.
- [11] Hewahi, N. Saad, M.: *Class Outliers Mining: Distance Based-Approach*, International Journal of Intelligent Systems and Technologies, Vol. 2, No. 1, pp 55-68, 2007.
- [12] Johnson, T., Kwok, I., Ng, R.: *Fast computation of 2-dimensional depth contours*, In: Proc. KDD. pp. 224–228, 1998.
- [13] Knorr E. M., Ng. R. T.: *Finding intensional knowledge of distance based outliers*, In Proc. of the 25th VLDB Conference, 1999.
- [14] Knorr, E., Ng, R., Tucakov, V.: *Distance-based outliers: Algorithms and applications*, VLDB Journal 8, pp. 237–253, 2000.
- [15] Knorr, E., Ng, R.: *A unified notion of outliers: Properties and computation*, In: Proc. KDD. pp. 219–222, 1997.
- [16] Knorr, E., Ng, R.: *Finding intentional knowledge of distance-based outliers*, In: Proc. VLDB. pp. 211–222, 1999.
- [17] Knorr, E.M., Ng, R.: *Algorithms for mining distance-based outliers in large datasets*, In: Proc. VLDB pp. 392–403, 1998.
- [18] Mierswa, Ingo and Wurst, Michael and Klinkenberg, Ralf and Scholz, Martin and Euler, Timm: *YALE: Rapid Prototyping for Complex Data Mining Tasks*, in Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06), 2006.

- [19] Papadimitriou, S., Faloutsos C.: *Cross-outlier detection*, In: Proc. Of SSTD'03, pp. 199-213, 2003.
- [20] Ramaswamy, S., Rastogi, R., Shim, K.: *Efficient algorithms for mining outliers from large data sets*, In Proc. of the ACM SIGMOD Conference, pp. 427-438, 2000.
- [21] Rousseeuw, P., Leroy, A.: *Robust Regression and Outlier Detection*, John Wiley and Sons, 1987.
- [22] Witten I. H. and Frank E.: *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.