Outlier Detection Using k-Nearest Neighbour Graph

Ville Hautamäki, Ismo Kärkkäinen and Pasi Fränti
University of Joensuu, Department of Computer Science
Joensuu, Finland
{villeh, iak, franti}@cs.joensuu.fi

Abstract

We present an Outlier Detection using Indegree Number (ODIN) algorithm that utilizes k-nearest neighbour graph. Improvements to existing kNN distance -based method are also proposed. We compare the methods with real and synthetic datasets. The results show that the proposed method achieves resonable results with synthetic data and outperforms compared methods with real data sets with small number of observations.

1. Introduction

Outlier is defined as an observation that deviates too much from other observations that it arouses suspicions that it was generated by a different mechanism from other observations [6]. *Inlier*, on the other hand, is defined as an observation that is explained by underlying probability density function. In clustering, outliers are considered as noise observations that should be removed in order to make more reliable clustering [5]. In data mining, detection of anomalous patterns in data is more interesting than detecting inlier clusters. For example, a breast cancer detection system might consider inlier observations to represent healthy patient and outlier observation as a patient with breast cancer. Similarly computer security intrusion detection system finds an inlier pattern as representation of normal network behaviour and outliers as possible intrusion attempts [13].

The exact definition of an outlier depends on the context. Definitions fall roughly into five categories [7]: i) *distribution-based*, ii) *depth-based*, iii) *distance-based*, iv) *clustering-based* and v) *density-based*. Distribution-based methods originate from statistics, where observation is considered as an outlier if it deviates too much from underlying distribution. For example, in normal distribution outlier is an observation whose distance from the average observation is three times of the variance [4]. The problem is that in real world cases underlying distribution is usually unknown and cannot be estimated from data without outliers affecting the estimate, thus creating a chicken-egg problem. Distance-based methods [8] define outlier as an observation that is d_{min} distance away from p percentage of observations in the dataset. The problem is then finding appropriate d_{min} and p such that outliers would be correctly detected with a small number of false detections. This process usually needs domain knowledge [8]. In clustering-based methods, outlier is defined to be observation that does not fit to the overall clustering pattern [15].

In density-based methods, outlier is detected from local density of observations. These methods use different density estimation strategies. A low local density on the observation is an indication of a possible outlier. For example, Brito *et al.* [1] proposed a *Mutual k-Nearest Neighbor* (MkNN) graph based approach. MkNN graph is a graph where an edge exists between vectors v_i and v_j if they both belong to each others k-neighbourhood. MkNN graph is undirected and is a special case of *k-Nearest Neighbour* (kNN) graph, in which every node has pointers to its k nearest neighbours. Each connected component is considered as a cluster if, it contains more than one vector and an outlier, when connected component contains only one vector. Ramaswamy *et al.* [11] proposed a method, in which n largest kNN distances are considered as outliers. This can be seen as "sparseness estimate" of a vector, in which the n sparsest vectors are considered as outliers. We name the method RRS according to the original authors' initials.

In this paper, we propose two density-based outlier detection methods. In the first method, a vector is defined as an outlier if it participates in at most T neighbourhoods in kNN graph, where threshold T is a control parameter. To accomplish this we consider kNN graph as a directed proximity graph, where the vectors are vertices of the graph and edges are distances between the vectors. We classify a vector as outlier on basis of its *indegree number* in the graph. The second method, a modification of RRS, sorts all vectors by their average kNN distances, for which a global threshold T is defined. Vectors with large average kNN -distance are all marked as outliers.

Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)
1051-4651/04 $ 20.00 IEEE

2. Outlier Detection with kNN Graph

2.1. kNN Graph

We define k-nearest neighbour (kNN) graph as a weighted directed graph, in which every vertex represents a single vector, and the edges correspond to pointers to neighbour vectors. Every vertex has exactly k edges to the k nearest vectors according to a given distance function. Weight of the edge e_{ij} is the distance between vectors v_i and v_j. The problem of creating kNN graph is known in computational geometry as *all-k-nearest neighbours problem* [2]. The graph can be constructed by exhaustive search considering all pairwise distances at the cost of $O(N^2)$ time. Callahan and Kosaraju [2] have shown that all-k-nearest neighbour problem can be solved in $O(kN+N\log N)$ time. The kNN graph can be used for solving clustering problem as in [3].

2.2. Detecting Outliers with kNN Graph

The kNN graph can also be used for detecting outliers [1]. Mutual k-Nearest Neighbour (MkNN) uses a special case of kNN graph. It defines an *undirected* proximity graph, which has an edge between vertices v_i and v_j if kNN graph has an edge both from v_i to v_j and from v_j to v_i. Connected components form clusters in the data and connected component with just one vertex is defined as an outlier. Potential problem with this definition is that, an outlier that is too close to an inlier, can be missclassified. For example in Fig. 1, where 13 and 16 are neighbours of each other, and not outliers according to MkNN algorithm. Thus we need more flexibility in the outlier definition.

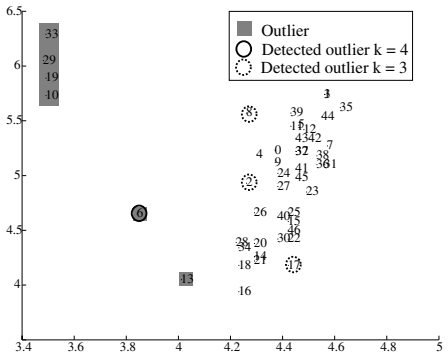

Figure 1. Outliers in HR dataset detected with ODIN, with threshold T = 0

Ramaswamy *et al.* [11] presented *RRS method*, which calculates kNN sparseness estimate for all vectors in dataset S. Vectors are sorted in an ascending order according to the distance from a vector to its k^{th} neighbour. Outliers are defined as the last n vectors in the ordered list. The intuitive idea is that when the distance to the k^{th} vector is large, vector is in sparse region, and is very likely to be an outlier. A drawback of the RRS is that user has to know in advance how many outliers there are in the dataset.

2.3. Proposed Methods

We propose the following definition of outlier using kNN graph:

Definition 1 *Given kNN graph G for dataset S, outlier is a vertex, whose indegree is less than equal to T.*

First, a kNN graph is created for datset S. Then, if vertex i has an indegree of T or less, mark it as an outlier and otherwise mark it as an inlier. The proposed method has two control parameters: the number of outgoing edges k and the indegree threshold T.

Algorithm 1 ODIN

```
T is indegree threshold
Calculate kNN graph of S
for i = 1 to |S| do
  if indegree of v_i ≤ T then
    Mark v_i as outlier
  end if
end for
```

Fig. 1 shows results of ODIN with indegree threshold set to $T = 0$ applied on the dataset Hertzsprung-Russell [12] with $k = 3$ and $k = 4$. The algorithm detects star 6 correctly as an outlier with $k \geq 4$, but star 13 is not detected with any k value. However, when using threshold $T = 1$ $k = 7$ we detect stars 6 and 13 correctly as outliers.

We extend the RRS method to specify cut point in the sorted list by considering adjacent differences as shown in Fig. 2. We consider two different variants of the RRS method, mean of kNN distances (MeanDIST) and maximun of kNN distances (KDIST). When scanning the ordered list from smaller to larger distances, we check if difference between adjancent distances is larger than a given threshold. We then define vectors after the cut point as outliers. We define the threshold as:

$$T = \max(L_i - L_{i-1}) * t, \tag{1}$$

where L_i is the KDIST or MeanDIST of i^{th} vector, and $t \in]0, 1[$ is a user defined parameter.

In Fig. 2 kNN distances are first sorted in descending order, and then the differences are taken from adjancent distances. We can see that differences grow fast when moving to right.

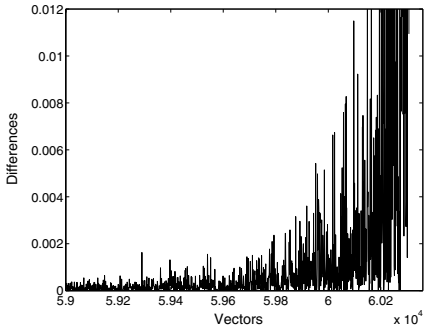

Figure 2. Differencies of distances for KDD dataset

Algorithm 2 MeanDIST

Compute T using Eq. 1 with t
Calculate kNN graph of S
$L \leftarrow$ Sort vectors in ascending order by kNN density
Find smallest i for which $L_i - L_{i-1} \geq T$
Mark $L_i, \ldots, L_{|S|}$ as outliers

3. Experiments

3.1. Description of Datasets

Experiments were run on *HR*, *KDD*, *NHL1*, *NHL2* and *synthetic* datasets, see Table 1. HR dataset in Fig. 1 is *Hertzsprung-Russell diagram* of the star cluster CYG OB1, where the first attribute is the logarithm effective temperature of the surface and the second the logarithm of light intensity.

Table 1. Datasets used in the experiments

Name	N	d	Outliers
HR [12]	47	2	2
KDD [9]	60318	3	486
NHL1 [8]	681	3	2
NHL2 [8]	731	3	1
synthetic	5165	2	165

KDD dataset was extracted from KDD Cup 1999 network intrusion dataset[1]. It was intended to be used as a training set for a supervised learning method. We follow the methodology described by Yaminishi *et al.* [14].

NHL1 and NHL2 dataset were selected from National Hockey League 96 player performance statistics [8]. In dataset NHL1, we selected attributes *games played*, *goals scored* and *shooting percentage*, and in NHL2 dataset *points scored*, *plus-minus statistics* and the *number of penalty minutes*. In NHL1, Chris Osgood and Mario Lemieux have been considered as outliers and Vladimir Konstantinov in NHL2 [8].

[1]Original data can be found from (http://kdd.ics.uci.edu/).

The synthetic dataset was made by generating cluster centers randomly so that they were not closer to each other than a predefined limit. Data points were then generated for each cluster with the limitation that points were not allowed to be farther than the given distance from the cluster center they belong to. We generated a GMM and using it computed minimum probability density for the points in data. Outliers were sampled from uniform distribution so that the probability density according to the GMM was at most half of the minimum of that of the data points. This ensured that the outliers were not inside the clusters.

3.2. Results

We use *Receiver Operating Characteristics* (ROC) as an evaluation method. It consists of *False Rejection* (FR) and *False Acceptance* (FA) rates. FR is number of detected outliers divided by all detections and FA is number of inliers detected as outliers divided by all detections. To combine FR and FA values we calculate *Half Total Error Rate* (HTER), defined as (FR + FA) / 2. Similar evaluation methodology has been used in [10].

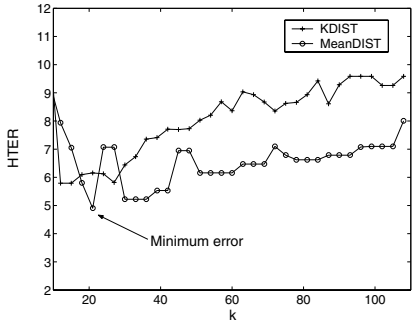

Figure 3. Error rate as a function of neighbourhood size for synthetic dataset

Table 2 summarizes parameters that give minimum error rate for each algorithm. HTER is used as the error rate and the values in the parenthesis are k and T. ODIN performs well on all datasets, but for synthetic dataset MeanDIST and KDIST perform better. Good performance can be explained by the generation method of data: outliers were drawn from uniform distribution and made sure that they lie far enough from normally distributed clusters. On the other hand, results for KDD dataset show that ODIN, MeanDIST and KDIST achieve practically the same error rate. For HR, NHL1 and NHL2 datasets ODIN achieves zero error, whereas other methods perform considerably worse. Reason for large HTER values is that given a small number of outliers even a few false acceptances increases the error rate greatly.

In Fig. 3, HTER is shown as a function of neighbourhood size for synthetic dataset. MeanDIST achieves lowest error

Table 2. Summary of results as error rate (k, threshold)

Method	synthetic	KDD	HR	NHL1	NHL2
MkNN [1]	50.0 (13)	77.0 (1)	25.0 (5)	25.0 (29)	44.4 (28)
ODIN	9.0 (190,26)	49.6 (1,2)	**0.0** (7, 1)	**0.0** (87, 9)	**0.0** (36, 2)
MeanDIST	**4.9** (21, 0.05)	49.6 (232, 0.19)	30.0 (1, 0.15)	16.7 (20, 0.05)	43.8 (1, 0.57)
KDIST [11]	5.7 (12, 0.06)	**48.6** (72, 0.40)	30.0 (1, 0.15)	30.0 (1, 0.02)	41.7 (7, 0.75)

and in general has lower error rate than KDIST for synthetic data.

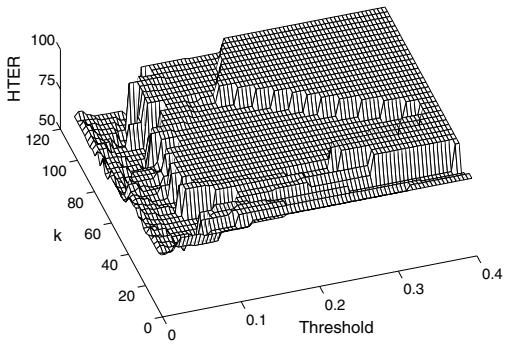

Figure 4. Error rate as a function of k and threshold for KDD dataset

Table 2 shows that optimal parameters for each dataset vary greatly, this leads to a problem of how to find the optimal parameter combiunation in the 2d parameter space. On the other hand, Fig. 4 shows the error rate as a function of k and T for MeanDIST with the KDD dataset. We can see that when T is below 0.1, the selection of the neighbourhood degree k is not critical. The problem of how to find the correct parameters is then just finding the best T.

4. Conclusions

We proposed a graph based outlier detection algorithm, which works well for the tested data sets. While MeanDIST and KDIST outperform the ODIN with the synthetic data, they give worse results for real data sets. This may be due to the small size of the data sets, in which case the density based methods may not obtain a reliable estimate. The proposed variant of RRS (MeanDIST) performs much better than KDIST with one dataset while the differences between the two are small in other cases.

References

[1] M. R. Brito, E. L. Chávez, A. J. Quiroz, and J. E. Yukich. Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Statistics & Probability Letters*, 35(1):33–42, August 1997.

[2] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the Association for Computing Machinery*, 42(1):67–90, 1995.

[3] P. Fränti, O. Virmajoki, and V. Hautamäki. Graph-based agglomerative clustering. In *Proceedings of The Third IEEE Int. Conf. on Data Mining*, pages 525–528, Melbourne, Florida, November 2003.

[4] D. Freedman, R. Purves, and R. Pisani. *Statistics*. W.W. Norton, New York, 1978.

[5] S. Guha, R. Rastogi, and K. Shim. CURE an efficient clustering algorithm for large databases. In *Proceedings of the 1998 ACM SIGMOD Int. Conf. on Management of Data*, pages 73–84, Seattle, Washington, June 1998.

[6] D. M. Hawkins. *Identification of Outliers*. Chapman and Hall, London, 1980.

[7] W. Jin, A. K. H. Tung, and J. Han. Finding top-n local outliers in large database. In *7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 293–298, 2001.

[8] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *Proceedings 24th Int. Conf. Very Large Data Bases*, pages 392–403, New York, USA, 1998.

[9] W. Lee, S. J. Stolfo, and K. W. Mok. A data mining framework for building intrusion detection models. In *IEEE Symposium on Security and Privacy*, pages 120–132, May 1999.

[10] J. Liu and P. Gader. Outlier rejection with mlps and variants of RBF networks. In *Proceedings of The 15th Int. Conf. on Pattern Recognition*, pages 680–683, 2000.

[11] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD Int. Conf. on Management of Data*, pages 427–438, Dallas, Texas, May 2000.

[12] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, New York, October 1987.

[13] G. Williams, R. Baxter, H. He, S. Hawkings, and L. Gu. A comparative study of RNN for outlier detection in data mining. In *Proceedings of the 2nd IEEE Int. Conf. on Data Mining*, Maebashi City, Japan, December 2002.

[14] K. Yamanishi, J. Takeuchi, G. Williams, and P. Milne. Online unsupervised outlier detection using finite mixtures with discounting learning algorithm. In *Proceedings The Sixth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 320–324, August 2000.

[15] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.