Outlier Analysis of Categorical Data using Artificial Intelligence

Jyotsna Diliprao Thakre[1] Gurudev B Sawarkar[2]
[2]Assistant Professor
[1,2]Department of Computer Science and Engineering
[1,2]V.M. Institute of Engineering & Technology (VMIT), Nagpur, India

Abstract— Outlier Mining is an essential job of discovering the data records that have a special performance comparing with other records in remaining dataset. Outliers do not follow with other data objects in the dataset. There are many practical approaches to detect outliers in numerical data. Most of the earliest work on outlier detection was performed by the statistics community on numeric data. But for categorical dataset there are limited approaches by using Memory efficient Incremental Local Outlier (MiLOF) detection algorithm and ROAD (Ranking-based Outlier Analysis and Detection algorithm).

Keywords: Outlier Detection, Stream Data Mining, Local Outlier, Memory Efficiency

I. Introduction

Outlier detection is the method of obtaining instances with rare performance which happens in the system. Effective detection of outliers leads to the discovery of valuable information in data. In many years, mining for outliers got major attention due to its wide applicability in areas such as obtaining fraudulent usage of credit cards, unauthorized access in computer networks, weather prediction & environmental monitoring.

Many existing methods are designed for obtaining outliers in continuous data. Most of these methods use distances between data points to detect outliers. In the case of data with categorical attributes, attempts are often made to map categorical features to numerical values. Such mappings impose arbitrary ordering of categorical values and may cause unreliable result.

An outlier is a data point that is significantly another from the remaining data. Hawkins formally defined [205] the concept of an outlier as follows:

“An outlier is an observation that deviates so much from the other observations as to arouse suspicions that it was created by a another mechanism.”

Outliers are also referred to as abnormalities, discordant, deviants or anomalies in the data mining and statistics literature. In most applications, the data is prepared by one or more generating processes that could either reflect activity in the system or interpretations collected about entities. When the generating procedure behaves in a rare way, it results in the formation of outliers. So, an outlier often contains useful data about abnormal characteristics of the systems and entities that impact the data generation method. The recognition of such rare characteristics gives useful application-specific insights

Few examples are given below:

Intrusion Detection Systems- In many host-based or networked computer systems, other kinds of data are collected about the operating system calls, network traffic or other activity in the system.

This data may show rare performance because of malicious activity. The detection of such activity is referred to as intrusion detection.

Outlier detection (also known as unusual event or anomaly detection) has got considerable attention in the field of data mining, since it need to detect rare events in variety of applications such as fake detection, human gait analysis & intrusion detection. A variety of outlier detection algorithms has proposed for the use on static data sets that have finite number of samples.

However, outlier detection on streaming data is particularly demanding, since the huge data to be analyzed is effectively unbounded and cannot be stored for ever in memory for processing [1]. Data streams are also generated at a high data rate, hence making the job of outlier detection even more demanding. This is a significant problem that arises in many real applications.

For example, outlier detection system in wireless sensor networks should work with the limited memory in each sensor node in order to find rare events in near real time. The cost of communication is an essential factor in such systems that limit the scope for downloading data to a central server for archiving and analysis. So a memory efficient outlier detection algorithm is needed for streaming data since it satisfies the memory constraint in sensor nodes & prevents the communication cost of having to transfer data to external storage.

II. Literature survey

In paper "Fast Distributed Outlier Detection in Mixed-Attribute Data Sets"[11] has discussed that a fundamental issue arrives that outlier detection algorithms attempt to obtain data points that are different from the rest of the data points in a given data set. The problem is of considerable importance, arising regularly in many real-world applications, for data mining researchers. Lots of practical applications concerning outlier detection occur in other domains such as fraud detection, cyber-intrusion detection, medical anomaly detection, image processing and textual anomaly detection [1].

In “Fine particles, thin films and exchange anisotropy,” and “Introduction to Data Mining” [2, 3] has discussed that Statistics-based approaches were first used for outlier detection based on an assumption that the distributions of data-sets are known. A data point was explained as an outlier if it deviates from existing distribution. With adequate information about the data-set, statistics-based methods work effectively. But in real-world, unfortunately, distributions of data-sets are unknown; signify all points that belong to clusters. The success of this approach depends on the clustering algorithm. Knorr and Ng [3] proposes to detect an outlier based on its distances from neighboring data points. Many other variations of distance-based approaches have been discussed in the literature.

In "Introduction to Data Mining" [4] has a discussion that each data point 'of the given data-set' should be assigned a degree of outlierness. In their view, as in other recent studies, a data point's degree of outlierness should be measured relative to its neighbors; hence they refer to it as the "/Local Outlier Factor" (LOF) of the data point. "LOF: Identifying density based local outliers" [4] argued that an outlier doesn't always have to be of lower density and lower density is not a compulsory condition to be an outlier. They changed LOF to obtain the connectivity-based outlier factor (COF) which they discussed is more efficient when a cluster and a neighboring outlier have similar neighborhood densities. Local density of a is normally measured in terms of k-nearest neighbors; LOF and COF both exploit characteristics associated with k-nearest neighbors of a given object in the data set. However, it is possible that an outlier lies in a location between objects from a sparse and a denser cluster. To account for such possibilities, Jin Etal [12] proposed another modification, called INFLO, that is based on a symmetric neighborhood relationship, i.e. the proposed modification considers neighbors and `reverse neighbors' of a data point when assessing its density distribution impacting the working of these methods. To avoid this obstacle clustering-based algorithms have been proposed to detect outliers (14,15).

In "Distance-based outliers: Algorithms and applications" [3] has argued that, the basic idea is that a data point is an outlier if it does not belong to any cluster. An outlier is a data object that differs significantly from the rest of the data. Detection of outliers is aimed at finding such rare objects and exceptions in a given data-set. It is a popular data mining job with applications in various domains such as fraud detection and intrusion detection in computer networks [1]. As a result, many methods have been developed for outlier detection employing various detection strategies. According to the distance-based methods [3], a data object is considered rare if it has very few neighbors in its proximity. On the other hand, the clustering-based methods try to identify various groups of objects based on their intrinsic similarity there by isolating objects with outlier characteristics. In "FP-Outlier: Frequent Pattern Based Outlier Detection" [6] has discussed this.

Additionally, many application settings like fraud detection and anomaly detection give themselves to be posed as unsupervised issues due to lack of prior knowledge about the nature of many outlier objects. This emphasizes the requirement for developing efficient unsupervised methods for outlier detection [5]. In many data mining applications, the data objects are explained using qualitative (categorical) attributes. The acceptable values of such a qualitative attribute are represented by many categories. The information on the occurrence frequencies of various categories of a categorical attribute in a given data set is very useful for many data-dependent tasks such as outlier detection.

Though there are many methods [1], [2], [6] for outlier detection in numerical data, the problem of outlier detection in categorical data is still evolving. The fundamental challenge in resolving this problem is the difficulty in explaining a suitable similarity measures over the categorical values. This is due to the fact that many values that has categorical variable can assume are not inherently ordered [7]. As a result, huge data mining associated tasks like determining the nearest neighbor of a categorical object, turned out to be non-trivial. Few research efforts [9] in this direction are indicative of the importance of this issue.

III. PROPOSED METHODOLOGY

Proposed project includes new hybrid methods for outlier detection analysis for Categorical dataset by (MiLOF) and Ranking algorithm.

A. Data

A major problem during evaluating outlier detection methods is that there are very few real world data-sets where it is exactly known that objects are really behaving abnormally since it belongs to the another mechanism. Though there are many available multiple case studies on outlier detection, the question is whether an object is an outlier or not is often depending on the point of view. Another problem is that the list of possible outliers is often incomplete making it hard to evaluate whether the algorithm ranked all outliers in the database properly. Therefore, we decided to evaluate our processes on artificially produced data. Thus, we can produce outliers and ordinary data points with respect to the initial definition, i.e. an outlier is a point being produced by another mechanism than the majority of data objects. To exactly evaluate the functioning of our new method for another dimensionalities and database sizes, we generated multiple data sets having 25, 50 and 100 dimensions. As database sizes (db-size) we selected 500, 1000, 5000 and 10,000 data objects.

In order to obtain data-sets having well-defined but not obvious outliers, we proceeded as follows. First of all, we randomly generated a Gaussian mixture model consisting of the equally weighted processes having random mean and variance values. This mixture model now describes the ordinary data points, i.e. the none-outlier data points. To build the outliers corresponding to other mechanism, that does not assign all the outliers to an additional cluster, we employed a uniform distribution on the complete data space.

This way we can generate 10 outliers for each data set that are fully independent on the mixture model describing the general data distribution. Let us note that it is possible that few outliers might be generated in an area being populated by none-outlier objects drawn from the Gaussian mixture model. Thus, even if an outlier detection mechanism works well, it does not necessarily have to rank all outliers into top positions.

B. Data Processing

1) Extract nominal feature attribute values from data files
Protocol_type=icmp,udp,tcp
Attack=phf,buffer_overflow,teardrop,guess_passwd,multihop,loadmodule,smurf,spy,normal,land,back,portsweep,warezclient,ftp_write,nmap,satan,rootkit,perl,imap,neptune,warezmaster,ipsweep, pod
Flag=RSTR,S3,SF,RSTO,SH,OTH,S2,RSTOS0,S1,S0,REJ
Service=vmnet,smtp,ntp_u,shell,kshell,aol,imap4,urh_i,netbios_ssn,tftp_u,mtp,uucp,nnsp,echo,tim_i,ssh,iso_tsap,time,netbios_ns,systat,hostnames,login,efs,supdup,http_8001,courier,ctf,finger,nntp,ftp_data,red_i,ldap,http,ftp,pm_dump,exec,klogin,auth,netbios_dgm,other,link,X11,discard,private,rem

ote_job,IRC,daytime,pop_3,pop_2,gopher,sunrpc,name,rje,domain,uucp_path,http_2784,Z39_50,domain_u,csnet_ns,whois,eco_i,bgp,sql_net,printer,telnet,ecr_i,urp_i,netstat,http_443,harvest

This has been done once and won't be required to be done again.

2) Changes of the nominal values to the numeric value as explained in paper A Practical Guide to Support Vector Classification Algorithm."We recommend using m numbers to represent an m-category attribute. Only one of the m numbers is one, and others are zero. For example, a three-category attribute such as red, green, blue can be represented as (0,0,1), (0,1,0), and (1,0,0).

Our experience indicates that if the number of values in an attribute is not too large, the coding may be more stable than using a single number."
If input file name is
Kddcup.data_10_percent_corrected-1000
then result of this step would be saved in
Kddcup.data_10_percent_corrected-1000-transformed

3) Scaling - as described in the same paper, we scale between 0.0 to 1.0. Result of this step would be saved in- kddcup.data_10_percent_corrected-1000-transformed-scaled

C. *Memory Efficient Incremental Local Outlier (MiLOF)*

If any data-set consist of outliers, then it deviates from its original performance and this data-set gives wrong results in any analysis. The MiLOF algorithm proposed the idea of finding a small subset of the data records that contribute to eliminate the disturbance of the data-set. This disturbance is also called 'entropy or uncertainty'. We can also called it as 'let us take a data-set D with m attributes A1, A2--- Am and d(Ai) is the domain of distinct values in the variable Ai, then the entropy of single attribute Aj is

$$E(Aj) = - \sum_{X \varepsilon\, d(Aj)} p(x) \log2(p(x)) \qquad (1)$$

Because all attributes are independent to each other, Entropy of the entire dataset
D={ A1, A2-------- Am} is equal to the sum of the entropies of each one of the m attributes and is defined as follows

$$E9A1, A2\text{---}Am)=E(A1)+E(A2)+\text{----}E(Am) \qquad (2)$$

When we want to obtain entropy the MiLOF algorithm takes k outliers as input. All records in the set are first designated as non-outliers. Initially all attribute value's frequencies are computed. Using these frequencies the initial entropy of the dataset is calculated.

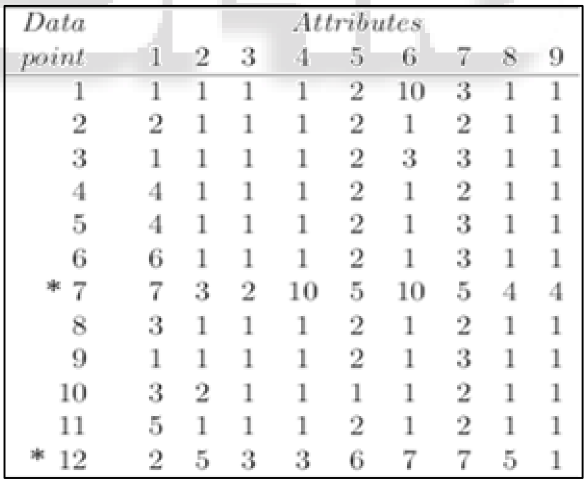
Then, MiLOF algorithm scans k times over the data to determine the top k outliers keeping aside one non-outlier each-time. While scanning each-time every single non-outlier is temporarily removed from the data-set once and the total entropy is recalculated for the remaining dataset. For any non-outlier point that results in the maximum reduction for the entropy of the balance data-set is the outlier data-point removed by the algorithm. The MiLOF algorithm complexity is O(k *n*m*d), where k is the required number of outliers, n is the number of objects in the dataset D, m is the number of attributes in D, and d is the number of distinct attribute values, per attribute.
Pseudo code for the MiLOF Algorithm is as follows

This proposed model has been de-fined as an optimal number of outliers in a single instance to get optimal precision in any classification model with good precision and low recall value. This method calculates 'k' value itself based on the frequency. Let us take the data set 'D' with 'm' attributes A1, A2----- Am and d (Ai) is the domain of distinct values in the variable Ai. kN is the number of outliers that are normally distributed. To get 'kN' this model used Gaussian theory. If any object frequency is less than "mean-3 S.D" then this model treats those objects as outliers. This method uses AVF score formula to obtain AVF score but no k-value is required. Let D be the Categorical dataset, contains 'n' data points, xi, where i= 1…n. If each data point has 'm' attributes, we can write xi = [xi1, …..xil,…..xim], where xil is the value of the 1th attribute of xi.

IV. Algorithm

A. *Memory Efficient Incremental Local Outlier (MiLOF) Algorithm*

Steps
1) Read data set D
2) Label all the Data points as non-outliers
3) calculate normalized frequency of each attribute value for each point x_i
4) Calculate the frequency score of each record x_i as, Attribute Value Frequency of x_i is
AVF Score $(x_i)=F_i= 1/m\sum^{m}_{j=1} f\,(xi\ j)$
5) compute N-seed values a and b as
b=mean (x_i), a=b-3*std (x_i), if max $(F_i) > 3$*std (F_i)
6) If $F_i< a$, then declare x_i as outlier
7) Return K_N detected outliers.



Data	*Attributes*								
point	1	2	3	4	5	6	7	8	9
1	1	1	1	1	2	10	3	1	1
2	2	1	1	1	2	1	2	1	1
3	1	1	1	1	2	3	3	1	1
4	4	1	1	1	2	1	2	1	1
5	4	1	1	1	2	1	3	1	1
6	6	1	1	1	2	1	3	1	1
* 7	7	3	2	10	5	10	5	4	4
8	3	1	1	1	2	1	2	1	1
9	1	1	1	1	2	1	3	1	1
10	3	2	1	1	1	1	2	1	1
11	5	1	1	1	2	1	2	1	1
* 12	2	5	3	3	6	7	7	5	1

Fig. 1: Example of normal and outlier points

Example of normal and outlier points from Sensor Network Dataset. Outlier points are denoted by *****.

Let's consider above dataset having 12 data point and every data point is having attribute.
We label all the data point as non-outliner first.

Let D is the Categorical dataset, which contains 'n' data points, xi, where i= 1...n. If each data point has 'm' attributes, we can write xi = [xi1, ...xil,.....xim], where xil is the value of the lth attribute of xi.
For example from above table for first row we can consider xi =1
[xi1, ...xil,.....xim]=[1 , 1, 1, 1, 2, 10, 3, 1, 1]

We then calculate normalized frequency for this attribute [1, 1, 1, 1, 2, 10, 3, 1, 1] that frequency we can denote as Fi.

Normalized frequency is calculated as ((freq of attribute) / (no of attribute)) * 1000

Let's say it has been calculated 5 for above attribute.

Then we compute N-seed value as a and b

1) b=mean (xi) = mean(1)
2) a=b-3*std (xi)= mean(1)-3*std (1)
3) if max (Fi) > 3*std (Fi) = max(5) > 3*std(5)
4) Then Fi > a value then xi is outliner.

1) Step 1: Read data set D
2) Step 2: Label all the Data points as non-outliers
3) Step 3: calculate normalized frequency of each attribute value for each point xi
4) Step 4: calculate the frequency score of each record xi as, Attribute Value Frequency of xi is:...
5) Step 5: compute N-seed values a and b as b=mean (xi), a=b-3*std (xi), if max (Fi) > 3*std (Fi)
6) Step 6: If Fi< a, then declare xi as outlier
7) Step 7: return KN detected outliers.

B. Ranking-based Outlier Analysis and Detection Algorithm

Given a data-set D which consists of n objects, which are described using m categorical attributes. The aim is to determine the likely set indicating the objects that are most likely outliers.

As per the definition for outliers, we use a two-phase algorithm for unsupervised detection of outliers.

The first-phase does the object density computation and also explores a clustering of the given data set.

Using the "resulting clustering structure", the set of big cluster '1' is identified in order to determine the distance between various data objects and their corresponding nearest big clusters.

In the second-phase, the frequency-based rank and the clustering-based rank of each data object are calculated.

Subsequently, a unified set of the most likely outliers is prepared using these two individual rankings. So, we name the proposed method as Ranking-based Outlier Analysis and Detection (ROAD) algorithm.

This algorithm tells us the issue of dealing with categorical data for outlier detection by providing a novel definition for outliers.

As per our novel approach, a data object turns out to be an outlier in two scenarios: either the categorical values describing that object are relatively infrequent (denoted as Type-1), or the combination of the categorical values describing that the object is relatively infrequent; though each one of these values are frequent individually (denoted as Type-2). These scenarios can be depicted pictorially as shown in Figure 2, for a simple data set described using two categorical attributes. In this figure 2, the object O1 turns out to be an outlier of Type-1 as its value corresponding to the second attribute is infrequent.

On the subsequently, although both the attribute values of object O2 are frequently individual, their combination is not frequent. This makes it an outlier of Type-2.

For object O3, though it qualifies to be of Type-2, it is primarily of Type-1 due to its infrequent attribute values. Hence, we make no distinction between objects like O1 and O3 in our methodology.

As brought out in [9], it is more meaningful to rank the data objects based on their degree of deviation; instead of making a binary decision, on whether or not an object is an outlier.

Also, in many applications, domains dealing with large data it are more sensible to identify the set of most likely outliers, since it enables in carrying out further analysis more efficiently.

Because of this insight, the algorithm proposed here, leverages the ranking concept for determining the 'set of most likely outliers' in the given data-set.

The computational steps involved in the proposed two phase ROAD algorithm are presented in the below Figure 2.

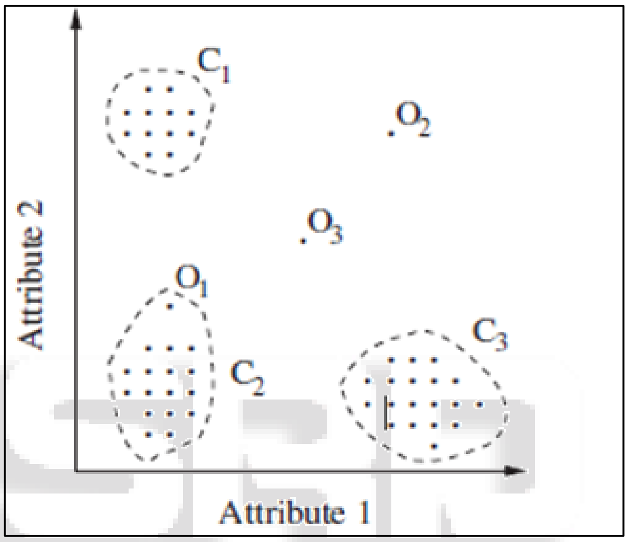


Fig. 2: Various scenarios of outlier occurrence

The computational complexity of the proposed algorithm is mainly contributed by the first three steps.

1) If the maximum number of unique values of an attribute is s, then the first step requires O(nms) computations. Typically, s is known to be a small quantity compared to n.
2) The next step requires O(nmk2) computations, as reported in [16].
3) The third step is the k-modes algorithm, that requires O(nmkt) computations, where t is the number of iterations required for convergence.

As we are using the initialization method proposed by [16], the k-modes algorithm required only a few iterations making 't' a very small value. Similarly, we work with very small number of clusters (k).

Finally, the ranking phase requires O(nlog(n)) effort.

Thus, the computational complexity of the proposed algorithm turns out to be O(nm + nlog(n)). It is must to note that the computational complexity of this algorithm is not affected by the number of outliers to be detected.

V. CONCLUSION

Here, we introduced a novel, parameter-free approach to outlier detection based on the variance of angles between pairs of data points. This idea alleviates the effects of the \curse of dimensionality" on mining high-dimensional data where distance-based approaches often fail to offer high quality results.

In addition to the basic approach memory efficient incremental local outlier (MiLOF) detection algorithm, we proposed two variants:

1) RANK as an acceleration suitable for low-dimensional but big data sets, and
2) Hybrid, a liter refinement approach as acceleration suitable also for high-dimensional data.

In a systematic evaluation, we expound the ability of my new approach to rank the best candidates for being an outlier with high precision and recall.

Moreover, the evaluation discusses an efficiency issues and enlighten the influence of the sample size to the runtime of the introduced methods.

REFERENCES

[1] M. E. Otey, A. Ghoting and A. Parthasarathy, "Fast Distributed Outlier Detection in Mixed-Attribute Data Sets," Data Mining and Knowledge Discovery He, Z., Deng, S., Xu, X., “A Fast Greedy algorithm for outlier mining”, Proc. of PAKDD, 2006.
[2] I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
[3] E. Knorr, R. Ng, and V. Tucakov, "Distance-based outliers: Algorithms and applications," VLDB Journal, 2000.
[4] M. M. Breunig, HP Kriegel, R. T. Ng and J. Sander, "LOF: Identifying density based local outliers," presented at ACM SIGMOD International Conference on Management of Data, 2000
[5] P. Tan, M. Steinbach, and V. Kumar, Introduction to Data Mining: Pearson Addison-Wesley, 2005
[6] S. Papadimitriou, H. Kitawaga, P. Gibbons, and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral," presented at International Conference on Data Engineering, 2003
[7] Z. He, X. Xu, J. Huang, and S. Deng, "FP-Outlier: Frequent Pattern Based Outlier Detection”, Computer Science and Information System (ComSIS'05)," 2005
[8] S. Wu and S. Wang, “Information-Theoretic Outlier Detection for Large-Scale Categorical Data, IEEE Transactions on Knowledge Engineering and Data Engineering, 2011
[9] Frank, & A. Asuncion, (2010). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
[10] E. Muller, I. Assent, U. Steinhausen, and T. Seidl, “Outrank: ranking outliers in high dimensional data,” in IEEE ICDE Workshop, Cancun, Mexico, 2008, pp. 600–603.
[11] K. Das and J. Schneider, “Detecting anomalous records in categorical datasets,” in ACM KDD, San Jose, California, 2007, pp. 220–229.
[12] Z. He, X. Xu, and S. Deng, “A fast greedy algorithm for outlier mining,” in PAKDD, Singapore, 2006, pp. 567–576.
[13] Koufakou, E. Ortiz, and M. Georgiopoulos, “A scalable and efficient outlier detection strategy for categorical data,” in IEEE ICTAI, Patras, Greece, 2007, pp. 210–217.
[14] S. Guha, R. Rastogi, and S. Kyuseok, “ROCK: A robust clustering algorithm for categorical attributes,” in ICDE, Sydney, Australia, 1999, pp. 512–521.
[15] Z. Huang, “A fast clustering algorithm to cluster very large categorical data sets in data mining,” in SIGMOD DMKD Workshop, 1997, pp. 1–8.
[16] K. Jain, “Data clustering: 50 years beyond K-means,” Pattern Recognition Letters, vol. 31, pp. 651–666, 2010.
[17] F. Cao, J. Liang, and L. Bai, “A new initialization method for categorical data clustering,” Expert Systems with Applications, vol. 36, pp. 10 223–10 228, 2009.
[18] Asuncion and D. J. Newman. (2007) UCI machine learning repository. [Online] http://archive.ics.uci.edu/ml
[19] E.M. Knorr and R T. Ng. Algorithms for mining distance based outliers in large datasets. In Proc. VLDB, 1998.
[20] E. M. Knorr and R. T. Ng. Finding intensional knowledge of distance-based outliers. In Proc. VLDB, 1999.
[21] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchthold. Efficient biased sampling for approximate clustering and outlier detection in large datasets. IEEE TKDE, 15(5):1170{1187, 2003.
[22] D. Newman, S. Hettich, C. Blake, and C. Merz. UCI repository of machine learning databases, 1998.
[23] S. Papadimitriou, H. Kitagawa, P. Gibbons, and C. Faloutsos. LOCI: Fast outlier detection using the local correlation integral. In Proc. ICDE, 2003.
[24] S. Ramaswamy, R. Rastogi, and K. Shim. Ealgorithms for mining outliers from large data sets. In Proc. SIGMOD, 2000.
[25] P. Rousseeuw and K. Van Driessen. A fast algorithm for the minimum covariance determinant estimator. Technometrics, 41:212{223, 1999}