不确定数据基于密度的局部异常点检测

曹科研[1)] 栾方军[1)] 孙焕良[1)] 丁国辉[2)]

[1)](沈阳建筑大学信息与控制工程学院 沈阳 110168)

[2)](沈阳航空航天大学计算机学院 沈阳 110136)

摘 要 不确定数据作为一种新型的数据模型,被广泛应用于金融、基于位置的服务、移动物体监测、传感器网络等许多类型应用领域.近年来出现的面向不确定数据的分析处理技术已成为数据库、数据挖掘等领域的研究热点.许多传统的数据挖掘技术已经被扩展并应用到不确定数据的分析和管理,异常点检测是数据挖掘领域重要的技术,用来发现行为或特征不同于其他对象的数据对象.当数据对象的性质和行为明显区别于它的近邻时,则被为视为异常点.异常点检测在许多方面有着广泛的应用,如网络入侵检测、信用卡诈骗、环境监测等.该文研究不确定数据基于密度的局部异常点检测,每个不确定数据由几个离散的可能实例组成.首先,提出了基于特定不确定数据模型的局部异常点定义.为了能够快速地检测局部异常点,在不展开可能世界的前提下,提出了基础算法 UDOL(Uncertain Density-based Local Outlier).然后,又提出在不精确计算概率的情况下,通过估计局部异常点因子的检测算法 PUDOL(Pruning on Uncertain Density-based Local Outlier),可以有效地减少计算量.最后,通过大量的实验验证该文提出算法的性能.实验结果证明,该文所提出的算法是解决不确定数据基于密度的局部异常点检测的有效方法.

关键词 不确定数据;异常点;可能世界;概率;密度

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2017.02231

Density-Based Local Outlier Detection on Uncertain Data

CAO Ke-Yan[1)] LUAN Fang-Jun[1)] SUN Huan-Liang[1)] DING Guo-Hui[2)]

[1)](*Information and Control Engineering Faculty*, *Shenyang Jianzhu University*, *Shenyang* 110168)

[2)](*College of Computer Science*, *Shenyang Aerospace University*, *Shenyang* 110136)

Abstract As a new data type, uncertain data is widely used in many applications, such as finance, location-based services, mobile monitoring, sensor networks, and so on. In recent years, analysis and processing technology based on uncertain data has become a hot research in databases, data mining and other fields. Many techniques have been developed for mining and managing uncertain data. Outlier detection is one of the key problems in the data mining area which can reveal rare phenomena and behaviors. An object can be recognized as an outlier because its properties and behavior are different from those in its neighborhoods. The techniques of outlier detection can be applied to many fields such as network intrusion detection, credit card fraud detection, environment monitoring and so on. In this paper, we will examine the problem of density-based local outlier detection on uncertain data sets described by some discrete instances. Firstly, we propose a new density-based local outlier concept based on uncertain data. In order to quickly detect outliers, the Uncertain Density-based Local Outlier (UDLO) basic algorithm is proposed that does not require the unfolding of all possible worlds. Furthermore, we

收稿日期:2015-10-18;在线出版日期:2016-05-10.本课题得到国家自然科学基金(611602323)、中国博士后科学基金(2016M591455)、辽宁省博士启动基金(201601209)资助.**曹科研**,女,1981 年生,博士,副教授,主要研究方向为数据挖掘、不确定数据管理、数据流管理.E-mail: caokeyan@gmail.com. **栾方军**,男,1971 年生,博士,教授,主要研究领域为数据库、数据挖掘、大数据管理. **孙焕良**,男,1969 年生,博士,教授,主要研究领域为数据库、数据挖掘、大数据管理. **丁国辉**,男,1982 年生,博士,讲师,主要研究方向为数据管理、查询处理与优化、概率数据库.

propose the Pruning on Uncertain Density-based Local Outlier (PUDLO) algorithm, an estimator of the Local Outlier Factor (LOF) without calculating the probability. It can effectively reduce the amount of the calculations. Finally, the performance of our method is verified through a number of simulation experiments. The experimental results show that our method is an effective way to solve the problem of density-based local outlier detection on uncertain data.

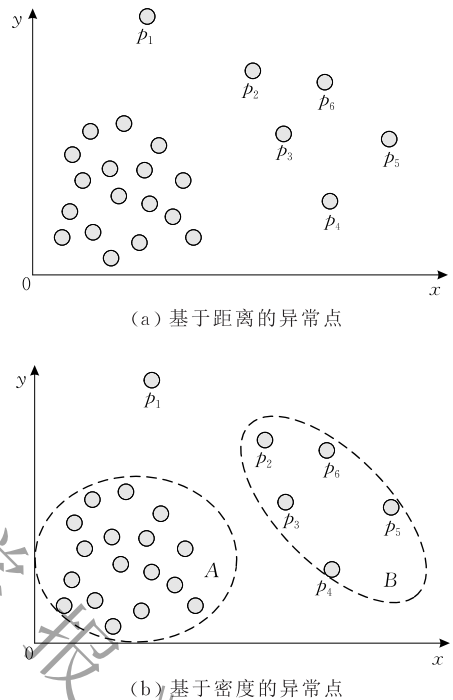
Keywords uncertain data; outlier; possible world; probability; density

1 引 言

近年来,随着人们对数据采集和处理技术理解的不断深入,不确定数据得到了广泛的重视.在许多现实的应用中,普遍存在着不确定数据[1],例如基于位置的服务、传感器网络、射频识别等领域.由于不确定数据中存在着不确定性,传统的数据分析技术无法有效地处理不确定数据,这就引发了学术界和工业界对研究新型的不确定数据分析技术的兴趣.

目前大多数应用中,将不确定数据分为存在级不确定性(existential uncertainty)和属性级不确定性(attribute level uncertainty).存在级不确定性:用概率描述元组的存在与否,这种模型较为普遍.属性级不确定数据:不涉及整个元组的不确定性,而是用概率密度函数(Probability Density Function,PDF)或统计参数(例如方差等)来描述特定属性的不确定性.然而有些情况,很难得到这样的概率密度函数.例如,利用传感器监测周围环境的温度时,通常记录的格式为:50%的概率是 29 度,30%的概率是 28 度,20%的概率是 25 度,元组表示为(〈29,50%〉,〈28,30%〉,〈25,20%〉).这样的记录方式也是属性级不确定性,相当于概率密度函数表示方式的抽样.本文的研究则是基于这种属性级不确定数据模型.

异常点检测一直是数据挖掘中重要的研究问题,可以发现行为明显不同于其他对象的数据对象[2].现有的算法多是基于距离检测异常点,而许多实际应用中,异常点的判定需要基于该对象和周围对象的关系才能确定[3-4].如图 1 所示,如果从基于距离的角度,R 选取的足够小,那么图 1(a)中 $p_1 \sim p_6$ 都是异常点,而从基于局部密度的角度考虑如图 1(b)所示,虽然 $p_2 \sim p_6$ 在 R 半径范围内近邻的数量比较少,但是它的近邻的情况也是如此,所以它们并不是异常点,从基于局部密度的角度只有 p_1 是异常点.例如在对某公司员工的工资做统计时,会发现领导的工资高于普通职工工资,但并不能因此就将领导视为异常点,因为在同岗位级别的领导中,他的工资属于正常,并不是异常点.



(a) 基于距离的异常点

(b) 基于密度的异常点

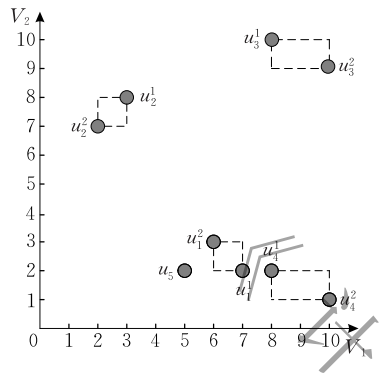
图 1 异常点的例子

图 2 所示为不确定数据的映射,每个不确定数据对象包含若干个可能实例,以下简称实例,如数据对象 u_1 包含两个可能实例 u_1^1 和 u_1^2,它们的存在概率分别是 0.8 和 0.2,每个实例有两维属性 V_1 和 V_2.

挑战 1. 什么是不确定数据基于密度的局部异常点?在确定数据中,LOF[5](Local Outlier Factor)定义为它的密度和近邻平均密度的比值,LOF 值最大的前 k 个数据对象为异常点.数据对象的密度定义为它到 k 近邻的距离的平均值的倒数.k 近邻是到当前对象最近的前 k 个数据对象.可见判断某个数据对象是否为异常点,要考虑它的 k 近邻.那么对于不确定数据而言,如何找到每个数据对象的 k 近邻成为首要问题.例如图 2 中的 5 个不确定对象,每个对象包含若干个实例,对于每个数据对象 u,用 V_1 和 V_2 表示它的两维属性,$P(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的存

位置	传感器 ID	属性 1	属性 2	概率
u_1	u_1^1	7	2	0.8
	u_1^2	6	3	0.2
u_2	u_2^1	3	8	0.7
	u_2^2	2	7	0.3
u_3	u_3^1	8	10	0.8
	u_3^2	10	9	0.2
u_4	u_4^1	8	2	0.5
	u_4^2	10	1	0.5
u_5		5	2	1

(a) 不确定数据对象表格



(b) 不确定数据对象的映射

图 2　不确定数据异常点的例子($k=2$)

在概率,假设所有的数据对象没有依赖关系.对于实例 u_3^1,它的 $k(k=1)$近邻是哪些数据对象?如果 u_2^1 存在,则它就是 u_3^1 的 k 近邻,但 u_2^1 的存在概率为 0.7,意味着有 0.3 的可能性它不存在,那么谁又是 u_3^1 的 k 近邻呢?通过这个例子证明不确定数据上的近邻不能根据确定数据的方法去定义,所以不确定数据上的异常点检测不能用现有的基于确定数据的方法.

贡献 1. 本文研究不确定数据上基于密度的局部异常点检测,每个不确定数据由若干个离散的可能实例组成.保留确定数据上的基于密度的异常点检测的主要思想,用概率描述 k 近邻,再计算密度,从而得到 LOF,然后根据每个不确定数据对象的 LOF 值降序排列,则前 top-n 个数据对象为异常点.

挑战 2. 现有的基于不确定数据异常点检测的研究,大多是基于距离进行检测,而基于距离和局部密度是从不同的角度看待异常点,检测算法不可以通用.如何能快速有效地检测基于密度的局部异常点?天真的想法是展开所有的可能世界,为每个实例在每个可能世界实例中找到它的 k 近邻,然后在每个可能世界中计算它到每个 k 近邻的距离和,再根据每个可能世界的概率,就可以得到每个实例的密度.然而可能世界的数量是根据实例的数量指数级增长,这样的计算代价是无法接受的.

贡献 2. 本文提出了有效的方法解决这个问题.首先在避免展开所有可能世界的前提下提出了精确的算法,通过扫描不确定数据对象的近邻列表,可以找到它的 k 近邻.基于数据对象到它的 k 近邻的距离以及每个 k 近邻对象的概率,可以得到每个数据对象精确的 LOF 值,从而确定异常点.为了提高算法的效率,又提出了通过估计 LOF 的上界和下界而剪枝的方法,这样可以减少一些不必要的计算,从而提高算法的效率.

本文第 2 节介绍国内外相关工作;第 3 节是问题定义;第 4 节介绍精确算法和剪枝算法以及算法的性能分析;在第 5 节进行广泛的实验评估;最后第 6 节总结全文.

2　相关工作

2.1　确定数据异常点检测

异常点[6]是观测值,它与其他观测值的差别如此之大,以至于怀疑它是由不同的机制产生的.根据异常点检测主要技术分为以下 4 种[7-8]:

(1) 基于距离

基于距离的异常点定义是根据局部近邻,或是 kNN[9](k-Nearest Neighbors),不考虑任何的数据分布,只是根据对象自身的性质进行判断.文献[10]是第一个提出了基于距离的异常点概念:如果以对象 x 为中心,半径为 R 的范围内,数据对象的个数少于阈值 K,则 x 是异常点.文中提出了两种方法用于异常点检测,首先返回精确的查询结果,但是使用了很大的存储空间.第二种方法是根据前面的结果进行推断,从而降低了存储代价.这种方法具有很高的伸缩性,很容易扩展到多维空间.

(2) 基于密度

当数据分布密度相差很大时,基于距离的异常点检测已经不能得到理想的结果[11-13].而基于密度的异常点检测完全从不同的角度看待异常点[5].文献[5]第一个提出了基于局部密度的异常点检测算法,可以检测到任意形状的簇.基于数据集中每个数据对象的异常点因子 LOF 降序排列,前 n 个数据对象即为基于密度的局部异常点.

(3) 基于统计分布[14-15]

基于统计分布的异常点检测对要检测的数据集假设了分布或概率模型[16],然后采用不和谐检验(discordancy test)进行异常点检测.基于统计的异常点检测大致可分为两类:参数法和非参数法,主要

的区别在于参数法假定给定数据的基本分布，并估计数据的分布模型参数，而非参数法不假设任何数据分布特征[17].

(4)基于偏差

基于偏差的异常点检测方法通过检测一组对象的主要特征来识别异常点，不符合这种特征的对象被视为异常点，根据检测技术主要分为两类：顺序异常技术(sequential exception technique)和 OLAP 数据立方体技术. 顺序异常技术模仿人类从一系列推测类似的对象中发现异常点的方法. OLAP 数据立方体技术使用数据立方体识别大型多维数据中的异常区域，为了提高效率，使偏差检测过程与立方体的计算重叠.

2.2 不确定数据异常点检测

文献[18]是第一个提出在不确定数据上进行异常点检测的概念，每个不确定数据对象由一个概率密度函数表示. 对不确定数据进行采样，用不确定数据出现在某特定范围内的概率来确定该对象是否是离群点. 然而，该研究没有考虑到不确定数据的另外一种数据模型，即元组级不确定数据.

文献[19]提出了不确定数据上的基于距离的异常点检测，基于这种数据模型上的异常点的定义是根据可能世界模型被提出. 如果某个对象近邻的存在概率都很低，那么该对象可以被视为基于距离的异常点. 该算法利用网格进行剪枝，提出了基础检测算法和动态剪枝方法，虽然得到了很好的效率，但该算法并没有考虑到参数变化的查询.

文献[20]研究了较为复杂的不确定数据模型，对象和实例都具有不确定性. 假设每个具有相似属性的不确定对象具有相似的实例，每个不确定对象的实例可以利用对象的相似性检测得到.

文献[21]将基于密度的局部异常点检测算法扩展到不确定数据上，研究的是每个数据对象由概率密度函数表示的数据模型，提出了基于这种特定数据模型的异常点因子的定义，并提出了优化算法，两个对象间距离的定义用到了 DBSCAN 中的直接密度可达定义等等，但该算法无法检测到位于聚类内部的异常点，所以导致实验结果的准确性受到严重的影响.

文献[22]是基于元组级不确定数据模型，提出不确定数据流上参数可变的异常点检测，满足用户在不同的时刻对异常点检测阈值不同的查询需求. 但该算法是检测基于距离的异常点，不适用于距离局部密度的局部异常点检测. 文献[23]是本文的前序工作，在基于确定数据的局部异常点定义的基础上，提出了不确定数据基于密度的局定异常点的概念. 而本文是基于文献[23]定义的基础上，提出了可以在不精确计算每个 k 近邻概率的前提下，通过估计每个对象异常因子的上界和下界，进行空间剪枝，生成远小于原始数据集的异常点候选集 S，只需要对候选集 S 里面的数据对象精确计算异常点因子，从而提高算法的效率.

3 问题定义

本文的研究是基于属性级不确定数据，每个数据对象由若干个可能实例组成，每个实例有 d 维属性. 两个实例 $\tilde{u}_i$ 和 $\tilde{u}_j$ 之间的距离表示为 $d(\tilde{u}_i,\tilde{u}_j)$，本文使用欧基里德距离，本文的技术可以扩展到其他的距离空间，表 1 列出了本文频繁使用的参数.

表 1 频繁使用的参数列表

参数	解释
U	不确定对象的集合
u_i^j	不确定对象 u_i 的第 j 个实例
$P(u_i^j)$	实例 u_i^j 的存在概率
$dis_{k(w)}(\tilde{u}_i)$	在可能世界 W 中，$\tilde{u}_i$ 的 k 近邻距离和
$n_k^W(\tilde{u}_i)$	$\tilde{u}_i$ 在可能世界 W 中的 k 近邻的集合
$P_k(\tilde{u}_j\rightarrow\tilde{u}_i)$	$\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻的概率
$den(\tilde{u}_i)$	实例 $\tilde{u}_i$ 的密度
$den_{\text{upper}}(\tilde{u}_i)$	$den(\tilde{u}_i)$ 的上界
$den_{\text{lower}}(\tilde{u}_i)$	$den(\tilde{u}_i)$ 的下界
u_i	集合 U 中第 i 个不确定对象
$\tilde{u}_i$	不确定对象 u_i 中任意一个实例
$P(W)$	可能世界 W 的概率
$\mathbb{N}$	所有可能世界的集合
$N_k(\tilde{u}_i)$	$\tilde{u}_i$ 在所有可能世界中的 k 近邻的集合
$N_{u_j\rightarrow\tilde{u}_i}$	$\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻的可能世界的集合
$LOF(\tilde{u}_i)$	$\tilde{u}_i$ 的局部异常点因子
$LOF_{\text{upper}}(\tilde{u}_i)$	$LOF(\tilde{u}_i)$ 的上界
$LOF_{\text{lower}}(\tilde{u}_i)$	$LOF(\tilde{u}_i)$ 的下界
$den_{k(w)}(\tilde{u}_j)$	可能世界 W 中，近邻 $\tilde{u}_j$ 密度

不确定对象：不确定对象的集合 $U=\{u_1,u_2,\cdots,u_i,\cdots,u_\mu\}$. 每个不确定对象 u_i 有 d 维属性. 本文考虑离散型，每个不确定数据对象 u_i 由 m 个实例组成 $\{u_i^1,u_i^2,\cdots,u_i^j,\cdots,u_i^m\}$，其中 $1\leq j\leq m$，$p(u_i^j)$ $(0\leq p(u_i^j)\leq 1)$ 表示实例 u_i^j 的存在概率.

可能世界：不确定对象的集合 $U=\{u_1,u_2,\cdots,u_i,\cdots,u_\mu\}$，$\tilde{u}_i$ 表示不确定数据对象 u_i 的任意一个可能实例，可能世界由每个数据对象中任意一个实例的集合组成，$W=\{\tilde{u}_1,\tilde{u}_2,\cdots,\tilde{u}_\mu\}$. 可能世界的数量是：

$$|\mathbb{N}|=\prod_{u\in U}|u|.$$

假设不确定对象间相互独立，不存在依赖关系，可能世界的概率：

$$P(W)=\prod_{u\in U,|\tilde{u}\cap W|=1}P(\tilde{u})\prod_{u\in U,|\tilde{u}\cap W|=\varnothing}(1-P(\tilde{u})).$$

$\mathbb{N}$ 表示所有可能世界的集合，则 $\sum_{W\in\mathbb{N}}P(W)=1.0$.

文献[13]提出了确定数据上的基于密度的局部异常点检测算法，数据对象 o 的 k 近邻集合由到 o 距离最近的 k 个数据对象组成，表示为 $n_k(o)$；数据对象 o 的 k 近邻集合中数据对象到 o 最远的距离定义为 k-距离，表示为 $d_{k\text{-}distance}(o)$. 如果对象 p 是对象 o 的 k 近邻，对象 o 的密度定义为：

$$den(o)=1\Big/\left\{\frac{\sum_{p\in n_k(o)}d(o,p)}{|n_k(o)|}\right\}.$$

局部异常点因子定义：

$$LOF(o)=\frac{\sum_{p\in n_k(o)}den(p)}{den(o)\times k}.$$

本文考虑的不确定数据是根据可能世界语义模型[24]，即需要在所有可能世界中找到每个实例的所有 k 近邻对象.

定义 1(k 近邻的距离和). 假设在可能世界实例 W 中，实例 $\tilde{u}_j$ 是实例 $\tilde{u}_i(i\neq j)$ 的 k 近邻($k<|U|$，$|U|$ 为数据集 U 中数据对象的数量)，$n_k^W(\tilde{u}_i)$ 表示在可能世界 W 中实例 $\tilde{u}_i$ 的 k 近邻的集合，$|n_k^W(\tilde{u}_i)|$ 表示集合 $n_k^W(\tilde{u}_i)$ 中数据对象的数量，则 $|n_k^W(\tilde{u}_i)|=k$. $dis_{k(w)}(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 在可能世界 W 中 k 近邻距离和：

$$dis_{k(w)}(\tilde{u}_i)=\sum_{\tilde{u}_j\in n_k^W(\tilde{u}_i)}dis(\tilde{u}_j,\tilde{u}_i).$$

定义 2(实例的密度). 实例 $\tilde{u}_j$ 和实例 $\tilde{u}_i$ 属于不同数据对象，即 $i\neq j$，$den(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的密度，在可能世界 W 中，实例 $\tilde{u}_j$ 是实例 $\tilde{u}_i$ 的 k 近邻，实例 $\tilde{u}_i$ 的密度定义为：

$$den(\tilde{u}_i)=\frac{k}{\sum_{W\in\mathbb{N}}dis_{k(W)}(\tilde{u}_i)P(W)}.$$

定义 3(实例的 k 近邻集合). $\tilde{u}_i$ 表示不确定数据对象 u_i 的任意一个实例，$u_i\in U$. $n_k^W(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 在可能世界 W 中的 k 近邻集合，$N_k(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 在所有可能世界中的 k 近邻集合：

$$N_k(\tilde{u}_i)=\bigcup_{W\in\mathbb{N}}n_k^W(\tilde{u}_i).$$

$|N_k(\tilde{u}_i)|$ 表示集合 $N_k(\tilde{u}_i)$ 中数据对象的数量，$|n_k^W(\tilde{u}_i)|$ 表示实例 $\tilde{u}_i$ 在可能世界 W 中的 k 近邻集合中数据对象的数量，则 $|n_k^W(\tilde{u}_i)|=k$，所以 $|N_k(\tilde{u}_i)|\geq k$.

定义 4(实例的局部异常点因子). 假设实例 $\tilde{u}_j$ 是实例 $\tilde{u}_i$ 的 k 近邻，$den(\tilde{u}_i)$ 和 $den(\tilde{u}_j)$ 分别表示实例 $\tilde{u}_i$ 和 $\tilde{u}_j$ 的密度，$den_{k(w)}(\tilde{u}_j)$ 表示在可能世界 W 中近邻 $\tilde{u}_j$ 的密度，$LOF(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的局部异常点因子：

$$LOF(\tilde{u}_i)=\frac{\sum_{W\in\mathbb{N},\tilde{u}_j\in n_k^W(\tilde{u}_i)}den_{k(w)}(\tilde{u}_j)\times P(W)}{k\times den(\tilde{u}_i)}.$$

定义 5(不确定对象的局部异常点因子). $\tilde{u}_i$ 表示不确定数据对象 u_i 的任意一个实例，$u_i\in U$，$P(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的概率，$LOF(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的异常点因子，$LOF(u_i)$ 表示不确定数据对象 u_i 的异常点因子：

$$LOF(u_i)=\sum_{\tilde{u}_i\in u_i}LOF(\tilde{u}_i)P(\tilde{u}_i).$$

定义 6(基于密度的局部异常点). 不确定数据对象以 LOF 值降序排列，n 是用户指定的阈值，则前 top-n 不确定对象是基于密度的局部异常点.

表 2 列出了图 2 例子中实例 u_3^1 的所有可能世界($W_0\sim W_7$)，可能世界 W_0 的概率 $P(w_0)=0.8\times0.7\times0.5\times1=0.28$. 如果 $k=2$，实例 u_3^1 在可能世界 W_0 中的 k 近邻集合 $n_k^{w_0}(u_3^1)=\{u_2^1,u_4^1\}$，同时根据定义 1 可知 k 近邻距离和：

$$dis_k(u_3^1)=dis(u_2^1,u_3^1)+dis(u_4^1,u_3^1)=13.39.$$

表 2　可能世界($k=2,n=2$)

可能世界	$P(W)$	k 近邻
$W_0=\{u_1^1,u_2^1,u_4^1,u_5^1\}$	0.28	u_2^1,u_4^1
$W_1=\{u_1^1,u_2^2,u_4^1,u_5\}$	0.12	u_2^2,u_4^1
$W_2=\{u_1^2,u_2^1,u_4^1,u_5\}$	0.07	u_2^1,u_1^2
$W_3=\{u_1^1,u_2^1,u_4^2,u_5\}$	0.28	u_2^1,u_1^1
$W_4=\{u_1^1,u_2^2,u_4^2,u_5\}$	0.12	u_2^2,u_1^1
$W_5=\{u_1^2,u_2^2,u_4^1,u_5\}$	0.03	u_2^2,u_1^2
$W_6=\{u_1^2,u_2^1,u_4^2,u_5\}$	0.07	u_2^1,u_1^2
$W_7=\{u_1^2,u_2^2,u_4^2,u_5\}$	0.03	u_2^2,u_1^2
$W_4=\{u_1^1,u_2^2,u_4^2,u_5\}$	0.12	u_2^2,u_1^1
$W_5=\{u_1^2,u_2^2,u_4^1,u_5\}$	0.03	u_2^2,u_1^2
$W_6=\{u_1^2,u_2^1,u_4^2,u_5\}$	0.07	u_2^1,u_1^2
$W_7=\{u_1^2,u_2^2,u_4^2,u_5\}$	0.03	u_2^2,u_1^2

从定义 2 可知，实例 u_3^1 的密度：

$$den(u_3^1)=\frac{2}{dis_{k(w_0)}(u_3^1)p(w_0)+\cdots+dis_{k(w_7)}(u_3^1)p(w_7)}=0.15.$$

根据定义 4,实例 u_3^1 的局部异常点因子:

$$LOF(u_3^1)=2.12.$$

同理可得到 $LOF(u_3^2)=2.77$,根据定义 5:

$$LOF(u_3)=2.12\times0.8+2.77\times0.2=2.25.$$

同理可得到

$$LOF(u_1)=0.724;\ LOF(u_2)=1.618;$$

$$LOF(u_4)=1.32;\ \ LOF(u_5)=1.21.$$

根据定义 6,如果 $n=2$,将 $u_1\sim u_5$ 根据它们的 LOF 降序排列,前 2 个数据对象为数据集 U 中基于密度的局部异常点,即数据对象 u_3 和 u_2 是异常点.

4 异常点检测算法

在本节中,首先介绍不确定数据基于密度的局部异常点检测的基础算法(UDLO),它将可能世界的问题转换成概率的问题,继而提出基于 UDLO 的空间剪枝算法 PUDLO,通过估计异常点因子的上界和下界排除一些数据从而进行剪枝.

4.1 异常点检测算法

不确定数据基于密度的局部异常点可以根据定义计算得到,但那是天真的想法,因为可能世界的数量是随着实例的数量指数级增长,这样的计算代价是不能接受的.为了解决这个问题,本文提出了新的算法 UDLO.对于实例 $\tilde{u}_i(\tilde{u}_i\in u_i\in U)$,将数据集中除了与 $\tilde{u}_i$ 属于相同不确定数据对象的实例以外其他实例 $\tilde{u}_j$ 按到 $\tilde{u}_i$ 的距离升序排列,形成列表 $L_1=\bar{t}_1,\bar{t}_2,\cdots,\bar{t}_n,n=\sum_{u_j\in U}|u_j|$,其中 $i\neq j$,$|u_j|$ 表示数据对象 u_j 里包含的实例的数量.

图 3 所示为例子 1 中 u_3^1 近邻列表 L_1,根据每个实例到 u_3^1 的距离升序排列,$k=2$,可见 u_3^1 的近邻集合为 $N_k(u_3^1)=\{u_2^1,u_2^2,u_1^2,u_4^1,u_1^1\}$.在 u_3^1 近邻列表 L_1 中,u_2^1 和 u_2^2 同属于不确定数据对象 u_2,虽然在任何一个可能世界 W 中,u_2^1 和 u_2^2 不会同时存在,但无论在哪个可能世界中,u_3^1 的最近邻如果不是 u_2^1,就一定是 u_2^2,所以可以换种说法,u_3^1 的最近邻是 u_2.

实例	u_2^1	u_2^2	u_1^2	u_4^1	u_1^1
距离	5.39	6.71	7.28	8	8.06
L_1	$\bar{t}_1$	$\bar{t}_2$	$\bar{t}_3$	$\bar{t}_4$	$\bar{t}_5$

图 3 实例 u_3^1 的 k 近邻集合($k=2$)

定义 7(k 近邻的概率). 已知两个实例 $\tilde{u}_j$ 和 $\tilde{u}_i$ 属于不同的两个数据对象,$i\neq j$,实例 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻,$N_{\tilde{u}_j\to\tilde{u}_i}$ 表示实例 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻所在的可能世界的集合.$P_k(\tilde{u}_j\to\tilde{u}_i)$ 表示 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻的概率,则 $P_k(\tilde{u}_j\to\tilde{u}_i)$ 是 $N_{\tilde{u}_j\to\tilde{u}_i}$ 集合中可能世界的概率和:

$$P_k(\tilde{u}_j\to\tilde{u}_i)=\sum_{W\in N_{\tilde{u}_j\to\tilde{u}_i}}P(W).$$

异常点概率的计算相似于不确定 top-n 的概率问题[21,25].实例 $\tilde{u}_i$ 的 k 近邻列表 $L=t_1,t_2,\cdots,t_m$ $(m=|N_k(\tilde{u}_i)|)$,根据每个实例到 $\tilde{u}_i$ 的距离升序排列.由于 k 小于整个数据集中数据的数量,可知 L 是列表 L_1 的一部分.假设在可能世界 W 中,$\tilde{u}_j$ 在实例 $\tilde{u}_i$ 的 k 近邻列表 L 中的位置是 t_i $(1\leq i\leq|N_k(\tilde{u}_i)|)$,只有在列表 L 中 $S=\{t_1,\cdots,t_{i-1}\}$ 中有 $\kappa-1$ 个实例存在,$\tilde{u}_j$ 才有可能是它的第 $\kappa(\kappa>0)$ 个近邻.$P(\tilde{u}_j,\kappa)$ 表示 $\tilde{u}_j$ 是第 $\kappa(\kappa>0)$ 个近邻的概率:

$$P(\tilde{u}_j,\kappa)=P(\tilde{u}_j)P(S_{t_{i-1}},\kappa-1),$$

其中,i 是指在列表 L 中第 i 个位置,$S_{t_{i-1}}$ 则表示列表 L 中前 $i-1$ 个对象的集合,$P(S_{t_{i-1}},\kappa-1)$ 表示集合 $S_{t_{i-1}}$ 中有 $\kappa-1$ 个对象存在的概率.

$\tilde{u}_j$ 是 k 近邻的概率:

$$P_k(\tilde{u}_j\to\tilde{u}_i)=\sum_{\kappa=1}^{\kappa=k}P(\tilde{u}_j,\kappa).$$

如果 $i<\kappa$,则 $P_\kappa(\tilde{u}_j\to\tilde{u}_i)=P(\tilde{u}_j)$.

$\tilde{u}_j$ 是 k 近邻的概率证明详见附录 A.

性质 1. $N_k(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的 k 近邻集合,如果不确定数据对象 u_j 中的所有实例都在 $\tilde{u}_i$ 的 k 近邻集合中,则数据对象 u_j 称为实例 $\tilde{u}_i$ 的完全 k 近邻数据对象,$N_{k(c)}(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 完全 k 近邻数据对象的数量:

$$N_{k(c)}(\tilde{u}_i)=k.$$

证明. 假设存在数据对象 u_j,是实例 $\tilde{u}_i$ 第 $k+1$ 个完全 k 近邻,集合 S_{t_i} 表示包含实例 $\tilde{u}_i$ 的 $k+1$ 个完全 k 近邻集合,数据对象 u_j 成为 k 近邻的概率:

$$P_k(u_j\to\tilde{u}_i)=$$

$$P(S_{t_{i-1}},k-1)P(u_j)+P(S_{t_{i-1}},k-1)(1-P(u_j)),$$

$P(S_{t_{i-1}},k-1)$ 表示集合 $S_{t_{i-1}}$ 中的数据有 $k-1$ 个对象存在的概率,而已知集合 $S_{t_{i-1}}$ 中有 k 个完全实例,每个完全实例的存在概率为 1,所以 $P(S_{t_{i-1}},k-1)=0$.同时已知 u_j 是安全 k 近邻,即 $P(u_j)=1$.根据以上分析可知:

$$P_k(u_j\to\tilde{u}_i)=0,$$

可见实例 k 近邻集合中有且只有 k 个完全 k 近邻.

图 3 列出了实例 u_3^1 的 k 近邻列表 L_1,$k=2$,计算列表中每个 k 近邻数据对象的 k 近邻概率,初始化 $P(\varnothing,0)=0$,$P(\varnothing,1)=0$,对于 $t_i(i\leq k)$,$P_k(t_i\to u_3^1)=$

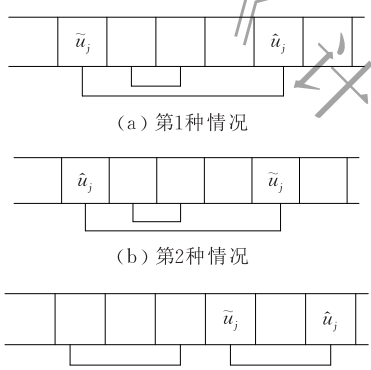
$P(t_i)$，故 $P_2(u_2^1\to u_3^1)=P(u_2^1)=0.7$，$P_2(u_2^2\to u_3^1)=P(u_2^2)=0.3$. 为了得到 $P_2(u_1^2\to u_3^1)$，首先要计算 $P(u_1^2,0)=0$，$P(S(t_2),1)=1$.

$\hat{u}_i$ 和 $\tilde{u}_i$ 表示同一个不确定数据对象的两个不同的实例，如果某个数据对象至少有两个实例在近邻列表中的位置都在 $\hat{u}_j$ 前面，则称它为复合对象. 为了正确的计算子集的概率 $P(S_{t_i},\kappa)$，如图 4 所示，计算 $P_k(\tilde{u}_j\to\tilde{u}_i)$ 一定符合下面的情况之一：

(1) 如果没有复合对象并且 $\tilde{u}_j$ 排在 $\hat{u}_j$ 的前面，可以忽略这种情况；

(2) 如果前面有复合对象并且 $\hat{u}_j$ 排在 $\tilde{u}_j$ 前面，将属于同一个对象的实例的概率合并，$\hat{u}_j$ 不能存在.

(3) 如果前面有复合对象排在 $\tilde{u}_j$ 前面，但前面没有和 $\tilde{u}_j$ 属于同一对象的其他实例，合并 $\tilde{u}_j$ 前面复合对象中实例概率，忽略 $\hat{u}_j$ 的存在.



(a) 第1种情况

(b) 第2种情况

(c) 第3种情况

图 4　计算 k 近邻概率

在图 2 的例子中，$k=2$. 根据不确定数据模型的性质，可知 u_2^1 和 u_2^2、u_1^2 和 u_1^1 互斥，表示为 $u_2^1\oplus u_2^2$，$u_1^2\oplus u_1^1$. 下面考虑如何计算 $P_2(u_1^2\to u_3^1)$，$P_2(u_4^1\to u_3^1)$ 和 $P_2(u_1^1\to u_3^1)$. 由于 u_1^2 排序在完全对象 u_2 之后，符合第 3 种情况，合并 $P_2(u_2^1\to u_3^1)$ 和 $P_2(u_2^2\to u_3^1)$，所以：

$$P_2(u_1^2\to u_3^1)=P(u_1^2)P(S_{t_2},1)=0.2.$$

同理：

$$P_2(u_4^1\to u_3^1)=P(u_4^1)(P(S_{t_3},1)+P(S_{t_3},0))$$
$$=0.5\times(1-0.2)=0.4.$$

由于 u_1^1 排在完全对象 u_2 和 u_1^2 的后面，属于第 2 种情况，合并 $P_2(u_2^1\to u_3^1)$ 和 $P_2(u_2^2\to u_3^1)$，同时使 u_1^2 不要存在，可得 $P_2(u_1^1\to u_3^1)=0.4$. 图 5 列出了 u_3^1 的 k 近邻的概率.

L	t_1	t_2	t_3	t_4	t_5
实例	u_2^1	u_2^2	u_1^2	u_4^1	u_1^1
p_2	0.7	0.3	0.2	0.4	0.4

图 5　实例 u_3^1 的 k 近邻概率($k=2$)

性质 2.　已知两个实例 $\tilde{u}_j$ 和实例 $\tilde{u}_i$ 属于不同的两个不确定数据对象，$i\neq j$，实例 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻，$P_k(\tilde{u}_j\to\tilde{u}_i)$ 表示 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻的概率，$den(\tilde{u}_i)$ 表示实例 $\tilde{u}_i$ 的密度：

$$den(\tilde{u}_i)=\frac{k}{\sum\limits_{\tilde{u}_j\in N_k(\tilde{u}_i)}dis(\tilde{u}_j,\tilde{u}_i)P_k(\tilde{u}_j\to\tilde{u}_i)}.$$

证明.　根据密度的定义 2，可知：

$$den(\tilde{u}_i)=\frac{k}{\sum\limits_{W\in\mathbb{N}}dis_{k(W)}(\tilde{u}_i)P(W)}.$$

通过定义 1，可知：

$$den(\tilde{u}_i)=\frac{k}{\sum\limits_{W\in\mathbb{N},\ \tilde{u}_j\in n_k^W(\tilde{u}_i)}dis(\tilde{u}_j,\tilde{u}_i)P(W)}.$$

再根据定义 7：

$$den(\tilde{u}_i)=\frac{k}{\sum\limits_{\tilde{u}_j\in N_k(\tilde{u}_i)}dis(\tilde{u}_j,\tilde{u}_i)P_k(\tilde{u}_j\to\tilde{u}_i)}.$$

性质 3.　实例 $\tilde{u}_j$ 和实例 $\tilde{u}_i$ 属于不同的两个不确定数据对象，$i\neq j$，实例 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻，$P_k(\tilde{u}_j\to\tilde{u}_i)$ 表示 $\tilde{u}_j$ 是 $\tilde{u}_i$ 的 k 近邻的概率，$den(\tilde{u}_i)$ 和 $den(\tilde{u}_j)$ 分别表示实例 $\tilde{u}_i$ 和 $\tilde{u}_j$ 的密度，则可以得到实例 $\tilde{u}_i$ 的 LOF：

$$LOF(\tilde{u}_i)=\frac{\sum\limits_{\tilde{u}_j\in N_k(\tilde{u}_i)}den(\tilde{u}_j)P_k(\tilde{u}_j\to\tilde{u}_i)}{k\times den(\tilde{u}_i)}.$$

证明过程和性质 2 相似，这里不再详述. 重新考虑图 2 中的例子，根据定义 5 可得：

$$LOF(u_3^1)=\frac{den(u_2^2)P_k(u_2^2\to u_3^1)+\cdots+den(u_1^1)P_k(u_1^1\to u_3^1)}{2\times den(u_3^1)}=0.15.$$

以上是对 UDLO 算法的介绍，要查询基于密度的局部异常点，首先为每个不确定数据对象的实例找到 k 近邻同时计算出它们成为的 k 近邻概率，然后精确地计算出每个实例的密度，根据实例自身和它的近邻的密度的关系，计算出实例的异常点因子，从而得到数据对象的异常点因子，即 LOF 值. 最后根据每个数据对象的 LOF 值降序排列，前 top-n 个不确定数据对象即为基于密度的局部异常点，具体过程描述详见算法 1.

算法 1.　UDLO 算法.

输入：U：不确定数据集

　　k：选取近邻的数量

　　n：异常点的数量

输出：O：异常点

1. FOR 每个不确定数据实例 $\tilde{u}_i$ 对象

2.　进行 k 近邻查询；

3. 返回 k 近邻集合 $N_k(\tilde{u}_i)$及每个近邻概率 $P_k(\tilde{u}_j \to \tilde{u}_i)$；
4. 根据性质 3,计算每个实例的密度；
5. 根据性质 4,计算每个实例的异常点因子 $LOF(\tilde{u}_i)$；
6. END FOR
7. 根据定义 5,计算每个不确定对象的局部异常点因子 $LOF(u_i)$；
8. 根据不确定对象的局部异常点因子 $LOF(u_i)$进行降序排列；
9. 前 n 个不确定数据对象即为异常点,并输出.

4.2 剪枝算法

根据 UDLO 算法描述,可见这部分的难点是计算每个不确定数据对象的 LOF 值.在实际应用中,异常点只占数据集中很小的一部分,如果计算所有数据对象的 LOF 值后再判断是很大的浪费,本文提出可以通过对每个对象 LOF 值估计上界和下界的方法直接排除一些不可能是异常点的数据对象,在估计的过程中不需要计算异常点的概率值,直接避开了基础算法的最耗资源的计算,从而提高了算法的效率.通过本小节介绍的剪枝算法 PUDLO,可以得到异常点候选集 S,然后只需要按照算法 UDLO 里面的第 1 步,计算候选集 S 里面的每个不确定数据,而不是整个不确定数据集中的数据,从而减少了计算的代价.

首先为每个实例组建 k 近邻的集合 N_k,而并不计算它们的 k 近邻概率,则问题从不确定数据被转化为确定数据上.性质 2 中,参数 k 是由用户指定,根据定义可知,实例的密度与它到 k 近邻的距离和以及 k 近邻的概率有关.对于实例 $\tilde{u}_i$的每个 k 近邻,它的 k 近邻的概率:$0 \leq P_k(\tilde{u}_j) \leq 1$.根据 L 列表,每个完全对象中只保留最靠前的一个实例,同一个完全对象中的其他可能实例将被删除,如图 6 所示,则形成列表 $L'=t'_1,t'_2,\cdots,t'_j,\cdots,t'_m(1 \leq j \leq m)$.将实例 $\tilde{u}_i$的 k 近邻列表按每个近邻到实例 $\tilde{u}_i$的距离降序排列,同理,只保留每个完全对象中最靠前的实例,属于同一个完全对象中的其他实例将被删除,如图 7 所示,形成列表 $L''=t''_1,t''_2,\cdots,t''_j,\cdots,t''_m$.

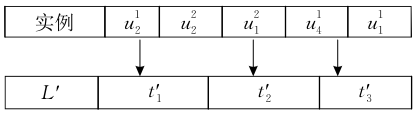


图 6 实例 u_3^1的 L'列表

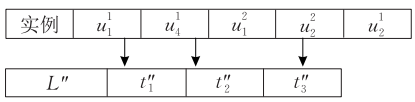


图 7 实例 u_3^1的的 L''列表

性质 4. $den_{\text{upper}}(\tilde{u}_i)$表示实例 $\tilde{u}_i$密度的上界,当 $\tilde{u}_i$的 L'列表中 $t'_1 \sim t'_k$都存在,实例 $\tilde{u}_i$的密度能取到最大值:

$$den_{\text{upper}}(\tilde{u}_i)=\frac{k}{dis(t'_1,\tilde{u}_i)+\cdots+dis(t'_k,\tilde{u}_i)}=\frac{k}{\sum_{j=1}^{j=k} dis(t'_j,\tilde{u}_i)}.$$

性质 5. $den_{\text{lower}}(\tilde{u}_i)$表示实例 $\tilde{u}_i$密度的下界,当 $\tilde{u}_i$的 L''列表中 $t''_1 \sim t''_k$都存在,实例 $\tilde{u}_i$的密度能取到最小值:

$$den_{\text{lower}}(\tilde{u}_i)=\frac{k}{dis(t''_1,\tilde{u}_i)+\cdots+dis(t''_k,\tilde{u}_i)}=\frac{k}{\sum_{j=1}^{j=k} dis(t''_j,\tilde{u}_i)}.$$

仍然以图 2 为例,当 $k=2$,图 6 所示为实例 u_3^1的 L'列表,实例 u_2^1在表中的位置为 t'_1,由于 u_2^1和 u_2^2属于同一个数据对象,所以 u_2^2不进入表 L'.u_1^2在表 L'中的位置是 t'_2.图 7 所示为实例 u_3^1的 L''列表,u_1^1的位置是 t''_1,实例 u_4^1的位置是 t''_2.由于 u_1^2和 u_1^1属于同一个数据对象,所以 u_1^2不进入表 L''.根据性质 4 和性质 5,实例 u_3^1的密度的上界和下界如下:

$$den_{\text{upper}}(u_3^1)=\frac{k}{dis(u_2^1,u_3^1)+dis(u_1^2,u_3^1)}=\frac{2}{5.39+7.28}=0.16,$$

$$den_{\text{lower}}(u_3^1)=\frac{k}{dis(u_1^1,u_3^1)+dis(u_1^4,u_3^1)}=\frac{2}{8.06+8}=0.12.$$

根据本文异常点的定义,异常点因子 LOF 的值不仅仅与实例的密度有关,也与它的 k 近邻密度有关.根据性质 3,实例的 k 近邻的密度越大,它的 LOF 值越大.将实例 $\tilde{u}_i$的 k 近邻根据每个近邻密度的上界降序排列,每个完全近邻对象中只取位置排在最前面的实例,其他完全近邻对象中的实例不进入列表 $L'_{den}=d'_1,d'_2,\cdots,d'_j,\cdots,d'_m(1 \leq j \leq m)$.同样,$L''_{den}=d''_1,d''_2,\cdots,d''_j,\cdots,d''_m(1 \leq j \leq m)$是根据近邻密度的下界按升序排列,每个完全近邻对象中只取位置排在最前面的实例,其他同一完全近邻对象中的实例不进入列表 L'_{den}.

性质 6. $LOF_{\text{upper}}(\tilde{u}_i)$表示 $LOF(\tilde{u}_i)$的上界,当实例 $\tilde{u}_i$的 k 近邻的密度都取到最大值,而实例 $\tilde{u}_i$的密度最小值时,将得到 $LOF(\tilde{u}_i)$的最大值:

$$LOF_{\text{upper}}(\tilde{u}_i)=\frac{den_{\text{upper}}(d'_1)+\cdots+den_{\text{upper}}(d'_k)}{|N_k(\tilde{u}_i)|\times den_{\text{lower}}(\tilde{u}_i)}=\frac{\sum_{j=1}^{j=|N_k(\tilde{u}_i)|}den_{\text{upper}}(d'_j)}{|N_k(\tilde{u}_i)|\times den_{\text{lower}}(\tilde{u}_i)}.$$

性质 7.　$LOF_{\text{lower}}(\tilde{u}_i)$ 表示 $LOF(\tilde{u}_i)$ 的下界，当实例 $\tilde{u}_i$ 的 k 近邻的密度都取到最小值，而实例 $\tilde{u}_i$ 的密度最大值时，将得到 $LOF(\tilde{u}_i)$ 的最小值：

$$LOF_{\text{lower}}(\tilde{u}_i)=\frac{den_{\text{lower}}(d''_1)+\cdots+den_{\text{lower}}(d''_{|N_k(\tilde{u}_i)|})}{|N_k(\tilde{u}_i)|\times den_{\text{upper}}(\tilde{u}_i)}=\frac{\sum_{j=1}^{j=|N_k(\tilde{u}_i)|}den_{\text{lower}}(d''_j)}{|N_k(\tilde{u}_i)|\times den_{\text{upper}}(\tilde{u}_i)}.$$

性质 8.　$P(\tilde{u}_i)$ 表示不确定实例 $\tilde{u}_i$ 的概率，$LOF_{\text{upper}}(\tilde{u}_i)$ 和 $LOF_{\text{lower}}(\tilde{u}_i)$ 分别表示 $LOF(\tilde{u}_i)$ 的上界和下界：

$$LOF_{\text{upper}}(u_i)=\sum_{\tilde{u}_i\in u_i}LOF_{\text{upper}}(\tilde{u}_i)P(\tilde{u}_i),$$

$$LOF_{\text{lower}}(u_i)=\sum_{\tilde{u}_i\in u_i}LOF_{\text{lower}}(\tilde{u}_i)P(\tilde{u}_i).$$

继续以图 1 为例，$k=2$，图 8 所示为实例 u_3^1 的 L'_{den} 列表，表中 u_1^1 的位置是 d'_1，由于 u_1^2 和 u_1^1 属于同一个完全近邻对象，所以 u_1^2 不进入列表 L'_{den}，u_4^1 的位置是 d'_2. 图 9 所示为实例 u_3^1 的 L''_{den} 列表，同理 u_2^2 不进入列表 L'_{den}，u_2^1 和 u_1^2 的位置分别是 d''_1 和 d''_2. 基于性质 7 和性质 8，实例 u_3^1 的上界和下界分别是：

$$LOF_{\text{upper}}(u_3^1)=\frac{den_{\text{upper}}(u_1^1)+den_{\text{upper}}(u_4^1)}{2\times den_{\text{lower}}(u_3^1)}=\frac{0.67+0.5}{2\times 0.12}=4.88,$$

$$LOF_{\text{lower}}(u_3^1)=\frac{den_{\text{lower}}(u_2^1)+den_{\text{lower}}(u_2^2)}{2\times den_{\text{upper}}(u_3^1)}=\frac{0.15+0.3}{2\times 0.16}=1.41,$$

$$LOF_{\text{upper}}(u_3)=LOF_{\text{upper}}(u_3^1)P(u_3^1)+LOF_{\text{upper}}(u_3^2)P(u_3^2)=4.814.$$

实例	u_1^1	u_1^2	u_4^1	u_2^1	u_2^2
den_{lower}	0.67	0.55	0.50	0.18	0.17
L'_{den}		d'_1	d'_2	d'_3	

图 8　实例 u_3^1 的 L'_{den} 列表

实例	u_2^1	u_2^2	u_1^2	u_4^1	u_1^1
den_{lower}	0.15	0.15	0.30	0.38	0.39
L''_{den}		d''_1	d''_2	d''_3	

图 9　实例 u_3^1 的 L''_{den} 列表

通过异常点定义 6 可知，根据每个不确定对象的 LOF 值降序排列，那么前 top-n 个对象即为基于密度的局部异常点. 用 S 表示候选集合，存储可能是异常点的数据对象. 通过基于密度的局部异常点的性质，估计每个对象的 LOF 值的上界和下界. 根据数据集内每个对象 LOF 值的下界降序排列，形成列表 $L_{LOF_{\text{lower}}}=l_1,l_2,\cdots,l_j,\cdots,l_\mu(1\leq j\leq\mu)(\mu=|U|)$. $L_{LOF_{\text{lower}}}$ 表中位置在 $l_1\sim l_n$ 位置的对象可能是异常点，将其存储到候选集 S，而位置为 $l_{n+1}\sim l_\mu$ 的数据对象根据每个对象的 LOF 值上界降序排列，形成列表 $L_{LOF_{\text{upper}}}=l'_1,l'_2,\cdots,l'_j,\cdots,l'_{|U|-n}(1\leq j\leq|U|-n)$. 对于 $L_{LOF_{\text{upper}}}$ 中 LOF 值的上界大于列表 $L_{LOF_{\text{lower}}}$ 中位置为 l_n 的数据对象 LOF 值的下界的数据对象存储到候选集合 S，其他数据对象将不进入候选集合 S. 然后仅需要计算候选集合 S 中每个不确定数据对象的 LOF 值. 这个例子中，只有 5 个不确定数据，选取其中的 top-2 为异常点，使用这个剪枝算法只淘汰一个数据对象，效果不是很明显，但在实际的应用中，异常点只占数据集中很小的一部分，这个剪枝算法的效果体现出更多的优点，具体数据在实验中给出.

图 10 列出了图 2 中的例子产生的候选集合 S，基于每个数据对象 LOF 值的上界和下界，可以得到列表 $L_{LOF_{\text{lower}}}$. 设 $k=2$，$n=2$，数据对象 u_2 和 u_3 被添加到候选集 S. 数据对象 u_4、u_1 和 u_5 根据它们的 LOF 值的上界重新降序排列，产生列表 $L_{LOF_{\text{upper}}}$. 由于 $LOF_{\text{upper}}(u_4)$ 和 $LOF_{\text{upper}}(u_5)$ 大于 $LOF_{\text{lower}}(u_3)$，所以将 u_4 和 u_5 添加到候选集 S. 而数据对象 u_1 一定不会是异常点，所以不需要精确计算它的异常点概率.

实例	LOF_{upper}	LOF_{lower}
u_1	1.275	0.595
u_2	3.450	1.173
u_3	4.814	1.384
u_4	2.070	0.895
u_5	2.090	0.570

(a) 例子中数据对象的异常点因子上界和下界

实例	u_2	u_3	u_4	u_1	u_5
$L_{LOF_{\text{lower}}}$	l_1	l_2	l_3	l_4	l_5

候选集合S⇐n个不确定对象

实例	u_5	u_4	u_1
$L_{LOF_{\text{upper}}}$	l'_1	l'_2	l'_3

候选集合S　删除

(b) 候选集合的产生

图 10　候选集合 $S(k=2,n=2)$

4.3　算法分析

UDLO 算法首先为每个实例找到 k 近邻，计算

代价为 $O(m^2 \cdot \log m)$，其中 m 表示不确定数据集中实例的数量. 然后基于 k 近邻的查询结果，计算 k 近邻集合中每个实例成为 k 近邻的概率，计算代价为 $O(m \cdot l \cdot k^3)$，其中 $l=\max|N_k(\tilde{u}_i)|$，$\tilde{u}_i \in u_i \in U$. 在计算每个实例密度的过程当中，计算代价为 $O(m \cdot l)$，计算得到 $LOF(\tilde{u}_i)$ 的代价同样为 $O(m \cdot l)$，得到 $LOF(u_i)$ 的代价为 $O(m)$，最终根据每个不确定数据对象的 $LOF(u_i)$ 值进行降序排列，从而得出该数据集中基于密度的局部异常点的计算代价是 $O(|U| \cdot \log n)$，其中 $|U|$ 表示不确定数据集中数据对象的数量，n 表示是用户指定的阈值，则前 top-n 不确定对象即是基于密度的局部异常点. 根据以上的分析，算法 UDLO 的时间复杂度为 $O(m^2 \cdot \log m + m \cdot l \cdot k^3 + |U| \cdot \log n)$.

PUDLO 算法计算每个实例 k 近邻集合的过程与 UDLO 算法相同，复杂度同为 $O(m^2 \cdot \log m)$，其中 m 表示不确定数据集 U 中实例的数量. P 表示算法 PUDLO 的剪枝率，$|U|$ 表示不确定对象数据集合 U 中数据对象的数量，$|S|$ 表示经过算法 PUDLO 剪枝后产生的候选集 S 中数据对象的数量，则

$$P=\frac{|U-S|}{|U|},$$

其中，$|U-S|=\{o|o \in U, o \notin S\}$，$k$ 近邻的存在概率计算的代价为 $O(m \cdot k \cdot P)$，其中 $l=\max|N_k(\tilde{u}_i)|$，$\tilde{u}_i \in u_i \in U$. 同理，实例的异常点因子和不确定数据对象的异常点因子计算代价为 $O(m \cdot l \cdot k^2 \cdot (1-P))$. 最后将候选集中的数据对象根据异常点因子值降序排列，数据集中基于密度的局部异常点的计算代价是 $O(|U| \cdot \log n \cdot P)$. 根据以上分析，剪枝算法 PUDLO 的时间复杂度为 $O(m^2 \cdot \log m + m \cdot k \cdot P + m \cdot l \cdot k^2 \cdot (1-P) + |U| \cdot \log n \cdot P)$.

5 算法性能

实验测试环境采用 3.3GHz Inter Core 3 CPU 和 4 GB RAM 的主机上，Microsoft Visual C++. 实验是基于两个真实数据集和一个合成数据集，测试算法的剪枝率，效率和准确性. 由于本文和文献[20]是基于两种不同的不确定数据模型，文献[20]的研究是基于对象和实例都具有不确定性的数据模型基础上，本文则是基于实例具有不确定性的模型，同时文献[20]并不是基于局部密度检测异常点，所以本文在实验部分不与文献[20]作对比. 为了证明本文提出算法 UDOL 和基于它的剪枝算法 PUDLO 的有效性，实现了文献[21]中的算法 BULOF 和 ULOF 作为对比算法. 真实数据集采用文献[21]中实验使用的数据集. 合成数据集使用高斯分布，形成 3 个不同的类，包含 10^5 个数据对象，每个类的密度不同. 再产生 100 个不属于这 3 个类的数据对象，由于这 100 个数据对象在不同的机制产生，可被作为异常点. 使用 LDPA[23] 数据集和 MAGIC[26] 数据集作为真实数据集，其中 LDPA 数据集是人活动的位置信息，包含 7 个属性值，155 952 个数据对象，MAGIC 数据集包含 10 个属性，19 020 个数据对象. 为了模拟不确定数据，将每个实例对象用若干个实例表表示，每个对象包含实例的数量范围[1,10]，所属同一数据对象的实例的概率和等于 1.

5.1 剪枝率测试

本文提出用剪枝率 P 衡量算法的剪枝效果，P 定义为通过剪枝算法被排除的数据对象的数量和算法数据集中数据量的比值. 通常 P 越大，证明算法的剪枝效果越好. 首先，测试剪枝率和异常点在数据集中所占比例的关系，如图 11 所示，异常点的数量在整个数据集中的比例是从 0.5%～3%. 从图中可见，算法 PUDLO 的剪枝率在 3 个数据集中都远高于算法 ULOF，这是由于算法 PUDLO 的剪枝是根据用户指定的阈值 n 来确定候选集合 S，即在不确定数据集中对象的数量固定的前提下，n 越小，进入候选集 S 中的对象数量越少，剪枝率则越大，所以剪枝率将随着 n 的增大而降低. 图 12 所示为剪枝率

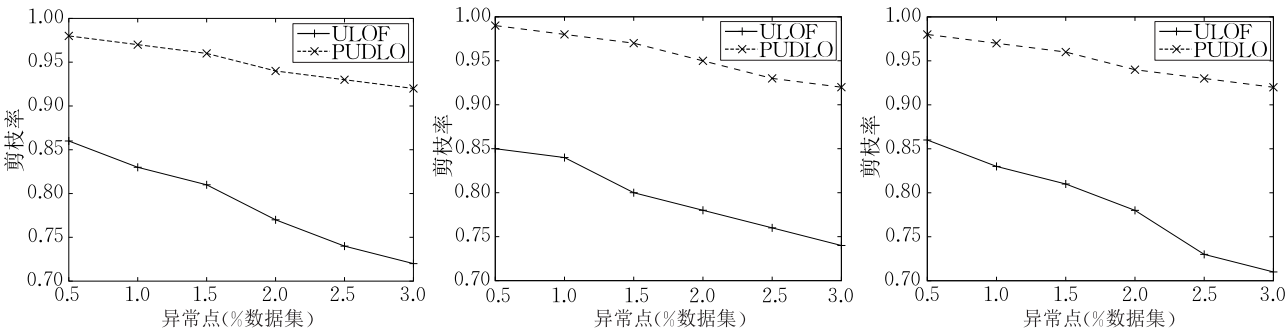


图 11 剪枝率与异常点数量

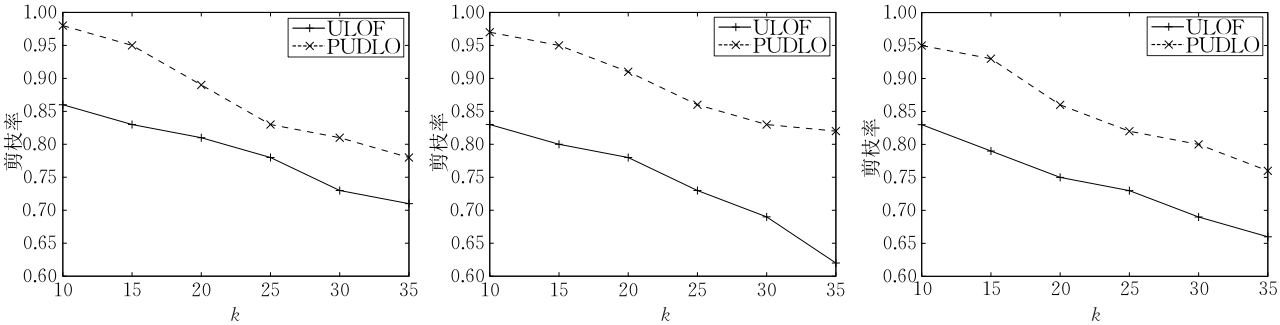


图 12　剪枝率与 k

和 k 值的关系，k 值的变化范围是[10,35]，两个算法的剪枝率都随着 k 值的增大而减小，但无论 k 值如何变化，PUDLO 算法的剪枝率始终高于 ULOF 算法.

5.2　效率测试

效率是衡量算法性能的重要参数，图 13 所示为算法的运行时间和不确定数据对象的数量的关系，数据的数量范围[10^2,10^5]，可以发现数据集的大小对算法的运行时间有着明显的影响. 数据集越大，算法需要处理的数据量就越大，从而导致算法的运行时间变长. 当仅数据量增大，而异常点的数量并没有改变，致使异常点在数据集中所占的比例会降低，利用本文中剪枝算法，可以降低数据集的改变对运行时间的影响. 但是从图 13 可见，无论数据量如何变化，由于算法 PUDLO 的剪枝率更好一些，排除了更多不必要的计算，它的运行时间始终少于算法 ULOF.

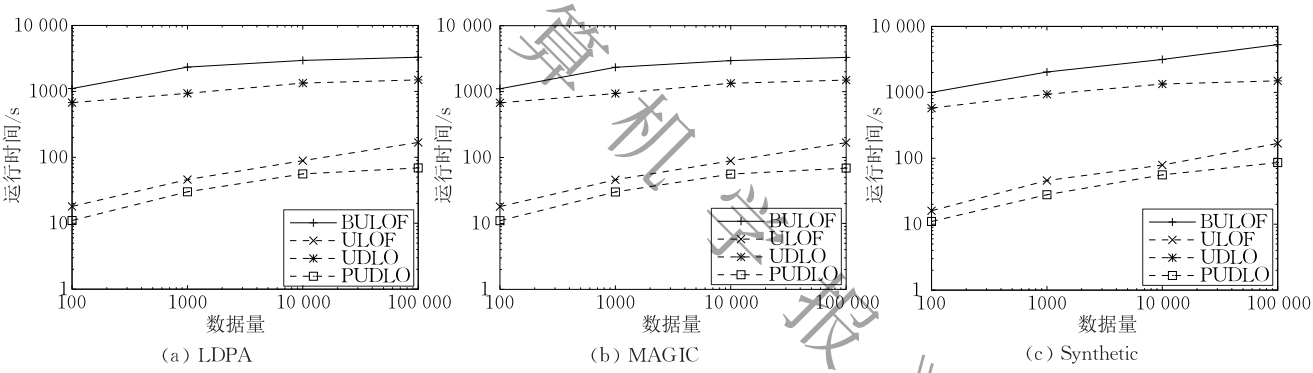


图 13　运行时间(s)与数据量

图 14 列出了 4 种算法的运行时间，参数 n 表示数据集中异常点的数量，n 变化范围是[20,100]. 算法 UDLO 中，需要精确计算出每个数据对象的异常点概率，根据概率再进行排序，取前 n 个对象即为异常点，由于算法的运行时间主要用来计算每个数据对象的异常点概率，所以参数 n 对算法 UDLO 的运行时间影响并不大. 而在算法 PUDLO 中，n 值越大意味着剪枝率就会随之变小，那么需要精确计算异常点概率的数据的数量就会增大，从而使算法的运行时间变长. 但从图中可见，无论 n 如何变化，算法 PUDLO 的运行时间始终小于 ULOF.

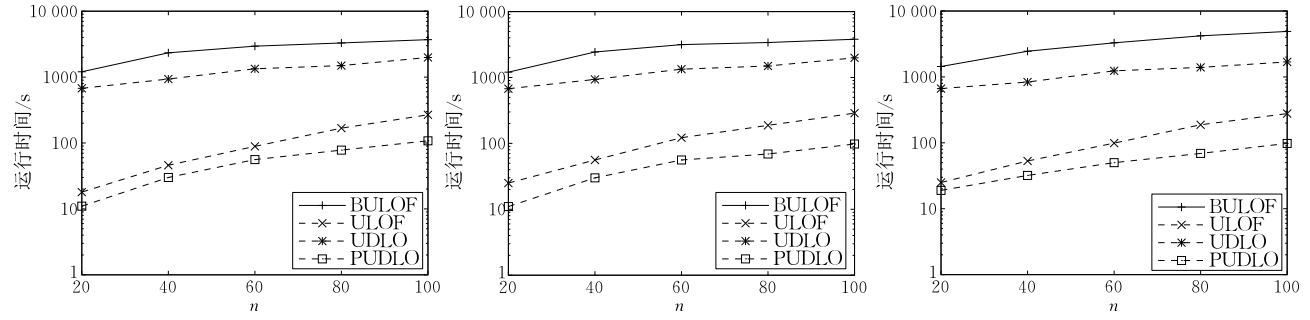
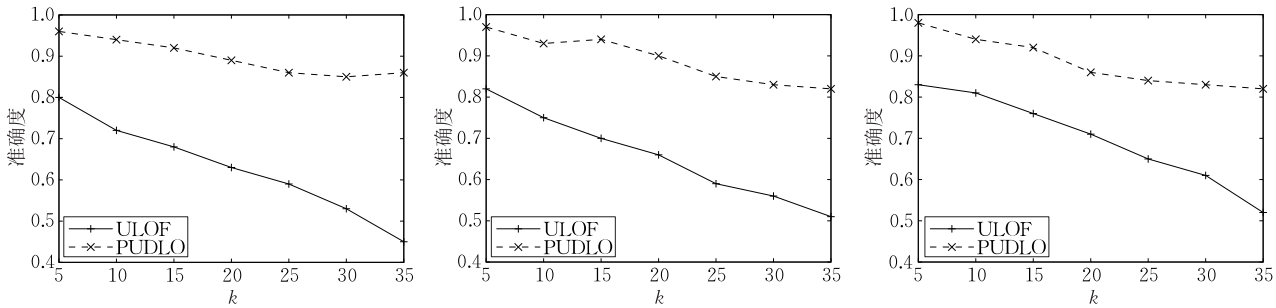


图 14　运行时间与 n

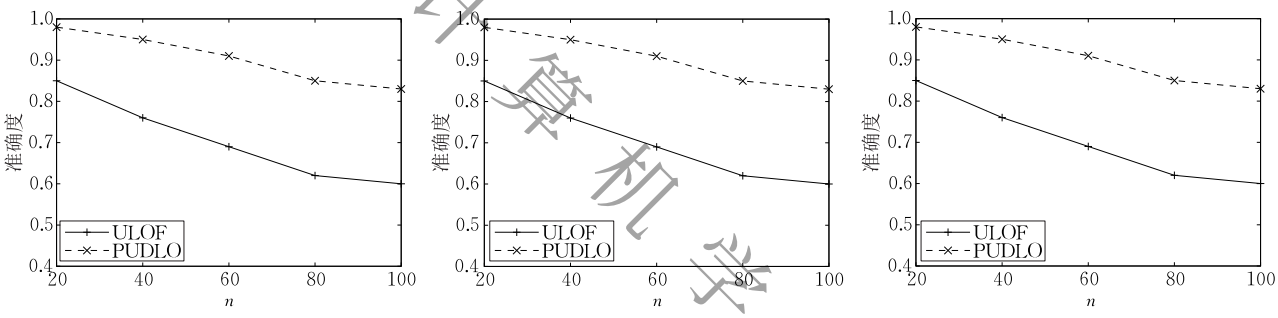
5.3 准确度测试

本节测试算法的准确度和参数 k 的关系，如图 15 所示. 生成合成数据集时，使用不同的机制生成一些明显是异常点的数据对象，以作为准确度度量参考. 由于算法 ULOF 很难发现在聚类中间的异常点，从而使算法的准确度受到严重的影响，而本文提出的算法 UDLO 严格按照局部异常点的定义进行查询，所以算法 PUDLO 的准确度远高于算法 ULOF. 图 16 所示为两种算法的准确度和 n 的关系，从图中可见，准确度会随着 n 的增大而降低，但无论 n 如何变化，算法 PUDLO 的准确度始终高于 ULOF 算法. 可见，PUDLO 算法虽然是对算法 UDLO 的剪枝，但由于是严格遵守定义进行剪枝，所以在提高效率的同时并没有损害算法的准确度.



(a) LDPA (b) MAGIC (c) Synthetic

图 15 准确度与 k



(a) LDPA (b) MAGIC (c) Synthetic

图 16 准确度与 n

6 结 论

不确定数据上的异常点检测有着重要的应用，许多应用要求异常点的检测要基于局部信息，即根据数据对象和它的近邻的关系判断是否为异常点. 本文首先提出了不确定数据上基于密度的局部异常点的定义，然后提出基础算法 UDLO，可以在不展开可能世界的前提下，有效地检测基于密度的局部异常点. 为了提高算法的效率和降低存储代价，又提出通过估计 *LOF* 值上界和下界而进行剪枝的算法 PUDLO，从而算法的性能得到进一步提高.

参 考 文 献

[1] Zhou Ao-Ying, Jin Che-Qing, Wang Guo-Ren, Li Jian-Zhong. A survey on the management of uncertain data. Chinese Journal of Computers, 2009, 32(1): 1-16(in Chinese)
(周傲英，金澈清，王国仁，李建中. 不确定性数据管理技术研究综述. 计算机学报，2009，32(1)：1-16)

[2] Cao L, Wei M, Yang D, Elke A. Rundensteiner: Online outlier exploration over large dataset//Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Sydney, Australia, 2015: 89-98

[3] Breuning M M, Kriegel H-P, Ng R T, Sander J. LOF: Identifying Density-based local outliers//Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. Dallas, USA, 2000: 93-104

[4] Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. ACM Computing Surveys, 2009, 41(3):75-79

[5] Aggarwal C C. On density based transforms for uncertain data mining//Proceedings of the 23rd International Conference on Data Engineering. Istanbul, Turkey, 2007: 866-875

[6] Knorr E M, Ng R T, Tucakov V. Distance-based outliers: Algorithms and applications. The International Journal on Very Large Data Bases, 2000, 8(3-4): 237-253

[7] Zhang J. Advancements of outlier detection: A survey. ICST Transactions on Scalable Information Systems, 2013, 13(1): 1-26

[8] Chandola V, Banerjee A, Kumar V. Outlier detection: A survey. Minnesota, USA: Department of Computer Science and Engineering, University of Minnesota, Technical Report: TR 07-017, 2007

[9] Wang W, Zhang J, Wang H. Grid-ODF: Detecting outliers effectively and efficiently in large multi-dimensional databases //Hao Yue, Liu Jiming, Wang Yuping, et al, eds. Computational Intelligence and Security. Lecture Notes in Computer Science 3801. Berlin, Germang: Springer-Verlag, 2005: 765-770

[10] Knorr E M, Ng R T. Algorithms for mining distance-based outliers in large datasets//Proceedings of the 24th International Conference on Very Large Data Bases. New York, USA, 1998: 392-403

[11] Han J-W, Kamber M. Data Mining: Concepts and techniques. 2nd Edition. New York, American: Morgan Kaufmann, Elsevier Inc., 2006

[12] Zhang J. Advancements of outlier detection: A survey. EAI Endorsed Transactions on Scalable Information Systems, 2013, 1(1): 1-26

[13] Breuning M M, Kriegel H-P, Ng R, Sander J. LOF: Identifying density-based local outliers//Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. Dallas, USA, 2000, 93-104

[14] Horn P S, Feng L, Li Y, Pesce A J. Effect of outliers and nonhealthy individuals on reference interval estimation. Clinical Chemistry, 2001, 47(12): 2137-2145

[15] Solberg H E, Lahti A. Detection of outliers in reference distributions: Performance of Horn's algorithm. Clinical Chemistry, 2005, 51(12): 2326-2332

[16] Branch J W, Giannella C, Szymanski B, et al. In-network outlier detection in wireless sensor networks. Knowledge and Information Systems, 2013, 34(1): 23-54

[17] Petrovskiy M I. Outlier detection algorithms in data mining systems. Programming and Computer Software, 2003, 29(4): 228-237

[18] Aggarwal C C, Yu P S. Outlier detection with uncertain data //Proceedings of the 5th Secure Data Management. Auckland, New Zealand, 2008: 483-493

[19] Wang B, Xiao G, Yu H, Yang X. Distance based outlier detection on uncertain data//Proceeding of the 9th International Conference on Computer and Information Technology. Xiamen, China, 2009: 293-298

[20] Jiang B, Pei J. Outlier detection on uncertain data: Objects, instances, and inferences//Proceedings of the 27th International Conference on Data Engineering. Hannover, Germany, 2011: 422-433

[21] Liu J, Deng H. Outlier detection on uncertain data based on local information. Knowledge-Based Systems, 2013, 51: 60-71

[22] Cao Ke-Yan, Wang Guo-Ren, Han Dong-Hong, et al. Continuous outlier monitoring on uncertain data streams. Journal of Computer Science & Technology, 2014, 29(3): 436-448

[23] Cao Ke-Yan, Shi Ling-Xu, Wang Guo-Ren, et al. Density-based local outlier detection on uncertain data//Proceedings of the 15th International Conference on Web-Age Information Management. Macau, China, 2014: 67-71

[24] Sarma A D, Benjelloun O, Halevy A Y, Widom J. Working models for uncertain data//Proceedings of the 22nd International Conference on Data Engineering. Atlanta, USA, 2006: 7-7

[25] Hua M, Pei J, Zhang W-J, Lin X-M. Ranking queries on uncertain data: A probabilistic threshold approach//Proceedings of the ACM SIGMOD International Conference on Management of data. Vancouver, Canada, 2008: 673-686

[26] Kaluža B, Mirchevska V, Dovgan E, et al. An agent-based approach to care in independent living//Proceedings of the 1st International Conference on Ambient Intelligence. Malaga, Spain, 2010: 177-186

附录 A.

如果 $\tilde{u}_j$ 在列表 L 中位置是 $t_i(1\leq i)$，S_{t_i} 表示列表 L 中前 i 个数据对象的集合，$P(S_{t_i},\kappa)$ 表示 S_{t_i} 中有 κ 个实例存在的概率：

$$P(S_{t_i},0)=P(S_{t_{i-1}},0)(1-P(t_i))=\prod_{j=0}^{j=i}(1-P(t_j)),$$

$$P(S_{t_i},\kappa)=P(S_{t_{i-1}},\kappa-1)P(t_i)+P(S_{t_{i-1}},\kappa)(1-P(t_i)).$$

CAO Ke-Yan, born in 1981, Ph. D., associate professor. Her research interests include data mining, uncertain data management and data stream management.

LUAN Fang-Jun, born in 1971, Ph. D., professor. His research interests include database, data mining, big data management.

SUN Huan-Liang, born in 1969, Ph. D., professor. His research interests include database, data mining, big data management.

DING Guo-Hui, born in 1982, Ph. D., lecturer. His research interests include data management, query processing and optimization, probabilistic database.

Background

Uncertainty is inherent to many important applications, such as location-based services (LBS), sensor monitoring and radio-frequency identification (RFID). In these applications, outlier detection often is essential when analyzing uncertain data. In many real-world applications, determining whether the object is an outlier, not only its distance to neighbors is considered, but also the density of surrounding neighbors should be considered. As such density-based outlier detection can consider local information.

Motivation: Sensors are often used to monitor remote environments, such as forests, oceans. Due to limitations of the sensors employed, oftentimes, the acquired data is faulty or inaccurate. Multiple sensors are set to monitor the same area to improve the accuracy of the acquired data, each sensor reporting its readings to the central node, and a confidence in the measurement is often estimated, leading to uncertain data.

This research is supported by the National Natural Science Foundation of China (Grant No. 611602323), the National Postdoctoral Fund (Grant No. 2016M591455), the Liaoning Province Doctor Start-Up Fund (Grant No. 201601209).