剪烛漫记:Differential Privacy

差分隐私学习笔记

作者：谢天

时间：Dec 29, 2022, 大二寒假

版本：1.0

学校：中科大网安学院

临渊羡鱼，不如退而结网

目录

第一章 The Spark of Differential Privacy

内容提要

- 历史背景
- Dinur & Nissim
- reconstruction attack
- Blatant Non-Privacy(BNP)

1.1 简述

本节从回溯 Dinur 的一篇论文开始，该文被认为是启发了 Dwork 差分隐私的灵感源泉。我暂时不会阐释怎样使得一个算法有多隐私，恰恰相反，我想说的是大部分方案都是“全然非隐私”的（BNP）。同时，我们会重点分析一种称为重构攻击（Reconstruction Attacks）的攻击形式，并阐述了为了保护隐私，我们应该增加的噪声容限理论上的值应该是多少。

注 本节均参照此论文 Revealing Information while Preserving Privacy Policy

1.2 简化版模型与基本表述

我们从这里严谨完备一个数据库模型，以下做了部分简化以便描述

- 定义每条数据为一行
- 每列为一个属性/特征
- 我们假定 Name,Postal Code,Date of Birth,Sex 等属性是不敏感的，是公开数据
- 只认为一种属性是敏感的，即 Has Disease？我们将其简化为 1bit(取值只能为 0 或 1)
- 令 $d \in \{0,1\}^n$ 表示隐私 bit 的向量

Name	Postal Code	Date of Birth	Sex	Has Disease?
Alice	K8V7R6	5/2/1984	F	1
Bob	V5K5J9	2/8/2001	M	0
Charlie	V1C7J	10/10/1954	M	1
David	R4K5T1	4/4/1944	M	0
Eve	G7N8Y3	1/1/1980	F	1

图 1.1: 数据库模型实例

命题 1.1 (模型概述)

这里假定有攻守双方，一方是查询者，一方是管理者。

查询者被允许提的问题形如：“数据库里有多少行在条件 X 下满足’Has Disease=1’?”

条件 X 可以是”Name= Alice OR Name = Charlie OR Name = David”，那么问题答案就是 2

更抽象起见，我们令查询向量为 $S = \{0,1\}^n$,0 表示该行不满足条件，1 表示满足条件，例如条件 X 可以表示为 $S = \{1,0,1,1,0\}$. 真实结果用 A(S) 表示，$A(S) = d \cdot S$ (例中 $A(S) = \{1,0,1,0,1\} \cdot \{1,0,1,1,0\} = 2$)

当然，这极大侵犯了隐私，所以管理者的响应 r(S) 会在 A(S) 的基础上加上一些噪声，界限小于等于 E

$$|r(S) - A(S)| \leq E \tag{1.1}$$

1.3 重构攻击的数学表达

定义 1.1 (全然非隐私 BNP)

我们称如下算法是全然非隐私的（blatantly non-private）：
如果攻击者能够重构一个数据库 $c \in \{0,1\}^n$, 使得其与真实数据库 d 几乎完全匹配, 或者说偏移量不超过 $o(n)$.

♣

如果一个算法是 BNP 的，那么这个算法之下毫无隐私可言。这就是重构攻击（reconstruction attack），随后我们可以证明出一般的方案都是 BNP 的。

定理 1.1

如果查询者被允许进行 2^n 次子集查询，并且管理者添加了 E 的噪声，那么根据响应结果，查询者可以重构出偏移量为 4E 的数据库。

♡

证明

由于查询者可以进行 2^n 次子集查询，那么他可以得到 2^n 个响应, 记响应总体为 $r(S)$
遍历所有 $c \in \{0,1\}^n$，剔除掉所有的 $|\sum c_i - r(S)| > E$，剩下的 c 就是可能的情况。
显然，真正的数据库 d 不会被剔除，由是我们考察两个集合 $I_0 = \{i|d_i = 0\}, I_1 = \{i|d_i = 1\}$

$$\begin{cases} |\sum_{i\in I_0} c_i - r(I_0)| \le E \\ |\sum_{i\in I_0} d_i - r(I_0)| \le E \end{cases}$$

由三角不等式得 $|\sum_{i\in I_0}(d_i - c_i)| \le 2E$, 对于 I_1 同理，故总偏移量为 $4E$
简要来说，这证明了该算法就是全然不隐私的。当然一个现实的问题是，查询者不可能进行指数量级的查询，这是一种低效的攻击，下面的攻击更高效，更具有现实意义。

定理 1.2 (Dinur-Nissim attack)

如果查询者被允许进行 O(n) 次子集查询，管理者加入噪声界限 $E = O(\alpha\sqrt{n})$, 那么根据响应结果，查询者可以重构出偏移量为 $O(\alpha^2)$ 的数据库。

♡

证明 具体的证明是困难的，这里只给出一些直觉上的分析

由于查询者只能进行 O(n) 次随机且均匀子集查询，仿照上文的 $c \in \{0,1\}^n$，利用线性回归剔除掉不合理的值。此处我们做一个转换，令 $c, d, S \in \{-1,+1\}^n$...

假设可能的 c(candidates) 与真实的数据库 d 在 $\Omega(n)$ 的数据上是不重合的, 考察 c 与 d 的差异到底有多大，$(c-d)\cdot S = \sum (c_i - d_i)\cdot S_i$

1. 如果 $c_i = d_i$，那么 $(c-d)\cdot S$，不影响结果。
2. 如果 $c_i \neq d_i$，那么

$$(c-d)\cdot S = \begin{cases} 2 & \text{w.p.}\frac{1}{2} \\ -2 & \text{w.p.}\frac{1}{2} \end{cases}$$

所以 $(\sum (c_i - d_i)\cdot S_i) \sim Bin(\Omega(n), \frac{1}{2})$，放缩移位后它取得 $\Omega(\sqrt{n})$ 的概率还是很高的。

由于管理者加入的噪声被限制在 $E = O(\sqrt{n})$，这使得不少 c 可以被排除。将偏移量太远的 c 排出后，我们对可能情况取一个并集，这样就十分接近真实情况了。严谨的数学证明还是参照 Dwork书里定理 8.2 的证明吧。

让我们回顾一下本小节，第一种攻击需要 2^n 次的查询以消除 $O(n)$ 的噪音, 第二种攻击需要 $\Omega(n)$ 次的查询以消除 $O(\sqrt{n})$ 的噪音. 而差分隐私允许在 $O(\sqrt{n})$ 的噪声条件下，进行 $O(n)$ 次查询，一般是通过拉普拉斯或者高斯机制去实现。其实隐私上限还是有”缩水“的空间的。比方说，我们查询的数据量远小于 n, 例如仅请求 $m << n$ 次的查询，那么相应的，管理者只需要加入 $O(\sqrt{m})$ 噪音就可以了。

正式的讨论，还是留给我们进入差分隐私的时候再说吧。

1.4 一次啼笑皆非的悬赏

到目前为止，所有的讨论看起来都像是在纸上谈兵。你是否在想，在现实生活中，这些攻击是不是真的可以实现呢？Aircloak 公司表示，它可以一战。

2017 年，Aircloak 公司发布了新系统 Diffix，它是一个数据库查询系统。不过，它的工作方式，看上去完全违反了上一节为了安全加以的诸多限制，首先它允许不限次数的查询，其次，它加入的噪声大小远远小于差分隐私保护机制起效所需的噪声量。尽管如此，他们还是大大咧咧的开出了 5000 美元的悬赏：给出一个攻击 Diffix 的方法，并重构数据库。

与以前类似，数据分析人员可以进行子集查询。主要的区别是，随着每次查询进行，噪声的大小呈条件集本身大小的平方根形式增加。其他防止攻击的方法包括限制计数量，调整极端值并且，禁用了"OR" 操作符。

回顾我们此前攻击做法，Dinur-Nissim 要求进行随机式地查询。这里先是获取随机的查询子集总体，然后利用一些条件集使得查询子集的特征得以显现。即便我们忽略不得使用"OR" 操作符的限制，看上去为了使得 k 个查询子集得以区分，我们似乎不得不加入 k 个条件集 (诸如此类的其他要求)。

Cohen 和 Nissim 天才式地绕过了这些限制，他们提出一个有效的新思路：相比与之前先选择查询子集再加入条件集使之凸显，他们在查询时就加入了极少量的条件。经历了精心设计的查询条件后，得到响应数据的随机性足以重构整个数据库。

具体方法解释如下。每个数据库中的用户对应唯一的 client-id, 问题是是否有一个函数以 client-id 作为参数，并将它“足够随机地”地隐含在查询集中？他们使用由四个变量指定的函数：mult、exp、d、pred。前三个变量是数字，第四个变量是一个真伪判断 (T/F),。即 $(mult * client - id)^{exp}$ 这个表达式的 d 位是否满足某个条件？它的 SQL 实现如下：

```
SELECT count(clientId)
FROM loans
WHERE floor(100 * ((clientId * 2)^0.7) + 0.5) = floor(100 * ((clientId * 2)^0.7))
AND clientId BETWEEN 2000 and 3000
AND loanStatus = 'C'
```

最后一位 loanStatus 就是他们尝试攻击的隐私位，他们攻击的用户范围是 2000 到 3000 之间的用户。如上所示，他们只是使用了一个条件，这给每次查询得到响应数据里，只带来了常数量的噪声，比上文所需的 $O(\sqrt{n})$ 噪声要远远小得多。最后，他们稍作调整除杂就 100% 地重构了整个数据库，拿走了奖金！详情参照此论文