CIS399: The Science of Data Ethics **Lecture 19: March 28, 2019**
Instructor: Michael Kearns & Ani Nenkova **Scribes: Anushka Makhija, Nishant Agrawal**

1 Recap

K-anonymity: a privatized/anonymized version of a particular database, which is k-anonymous if for any row in the database, there are k copies of that row. We think about the operational costs associated with reaching a k-anonymous database and how to minimize the number of operations. This is an NP-complete problem.

Operations: redacting (erasing one of the columns) and coarsening (for columns with numerical values, turning them into ranges rather then exact numbers).

Problem with K-Anonymity
Real problem with k-anonymity is that it might be a reasonable definition to provide a privacy guarantee for just this database in isolation, but it's never just this database. Even if I have multiple k-anonymized databases, combining them can "triangulate" and de-anonymize. Therefore this definition of privacy, one that only considers just the particular dataset in front of you in isolation, is *fundamentally broken.*

2 Differential Privacy

2.1 Introduction

Thought-provoking example: Why is it a privacy problem that your medical record is one of several thousand medical records in a database, especially when they're only reporting aggregated information?

Answer: Problem is that you're once again assuming that this is the only data analysis that's ever going to be done.

Suppose:

1. data = $x_1, ..., x_n \in \{0, 1\}$.
2. goal: compute $a = f(x_1, ..., x_n) = \frac{1}{n}\sum_i x_i$

Now say we have: - $b = f(x_1, ..., x_{n-1}) = \frac{1}{n-1}\sum_i x_i$

If you know both a ad b, then you can take $na - (n-1)b = \frac{1}{n}\sum_i x_i - \frac{1}{n-1}\sum_i x_i = x_n$. This basically tells us, that if you have one computation with the row and one without it, the difference tells us what the computation would yield on the individual data point. Your data is no longer private.

Sensitivity: how much can removing your data from the dataset or changing your data change the value of a particular function. A function with low sensitivity is necessary for differential privacy, but it's not sufficient.

Let's say we want to compute the average of n numbers while avoiding high sensitivity. We can do this using random noise introduction. Instead of releasing a and b, you release $\hat{a} \in [a-r, a+r]$ and $\hat{b} \in [b-r, b+r]$. You want to choose an r that's small enough to preserve the findings, but not so small that you're not really fixing the problem. What could this r be?

Our options:

- one intuition - $r\ \frac{1}{n}$
- another - $r\ \frac{1}{\sqrt{n}}$

We know $\frac{1}{\sqrt{n}} >> \frac{1}{n}$, so you're still maintaining privacy while also maintaining accuracy especially when n gets large. This is the core concept behind differential privacy. DP requires randomization.

However, the problem with this is that if the attacker knows n, and your output is basically $1.0 + r$, the attacker can easily determine what a is. The flaw is one of determinism. We added noise, but we did it in a brute force way.

2.2 Definition

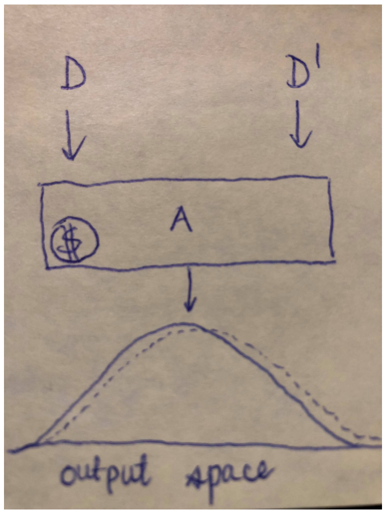
The definition of DP is a definition about algorithms. An algorithm either is differentially private or it's not.

Let A be some algorithm mapping any database D to some output space O, with some "goal" in mind.

Some examples of this include:

1. D = list of s/bits, O = single s, *goal* = compile/approximate average of D
2. D = database of medical records, O = neural networks, *goal* = predict disease from symptoms in database

Preliminary Definition: Say two databases D and D' are neighboring if they differ in/by only a a single row. A necessarily randomized algorithm A satisfies ϵ-DP if for *any* paid of neighboring databases D and D':



Because A is a randomized algorithm, it's output is not deterministic. Schematically it looks like a normal distribution on the page. Dashed line is A(D'). When you only change the input by a little bit, the algorithm has to output "very similar" distributions.

Formally, for any neighboring databases D and D' and for any subset $S \subseteq O$: $\Pr[A(D) \in S]$. This is the probability that the output of the algorithm given the database D falls within S. We also look at $\Pr[A(D') \in S]$: $\Pr[A(D) \in S] \leq e^{\epsilon} * \Pr[A(D') \in S]$. When $\epsilon = 0$, you're saying you want the two quantities to be identical.

If $\epsilon = 1$, you're saying that you want the two quantities to differ by a factor of 3. This promises that you're only 3 times more likely to get a bad outcome if you give your data than if you don't.

2.3 Randomized Response

Developed in the social sciences for a survey method for questions for which their is a social stigma associated with the answer. For example, asking Penn UGrads if they've ever cheated on an exam at Penn. This is an introduction into the idea:

Protocol: First, flip a fair coin. If it's tails, then answer question truthfully. If it's heads, flip again. Based on outcome of second coin flip - answer yes if heads and no if tails.

Why it works: The appeal to tell the truth here is that if you answer yes, you have plausible deniability for saying why this is true or not.

We can map this algorithm to Differential Privacy too. It's a distributed algorithm, where the input database $\in \{y, n\}^n$ and response is $\in \{\ddot{y}, \ddot{n}\}^n$. From the responder's perspective:

- $\Pr\{"y"/y\} = 1/2 + (1/2)(1/2) = 3/4$
- $\Pr(\ddot{y}/n\} = 0 + (1/2)(1/2) = 1/4)$
- $\Pr\{"y"/y\}/\Pr(\ddot{y}/n\} = (3/4)/(1/4) = 3$

Therefore, your lying to the algorithm doesn't change the output of the algorithm by more than 3. Randomized response obeys $ln(3)$ differential privacy, where 3 is ϵ.

Utility: If all follows RR, $E(\text{fraction "yes"}) = 3/4 * p + 1/4 * (1 - p)$ where p is the true fraction of cheaters and $\hat{p}$ is the empirical fraction of yes. Then w.h.p. $\hat{p} \in [p - c/sqrt(n), p + c/\sqrt(n)]$. As n gets larger and larger, you're getting better and better estimates of the true value of p.