Privacy in Statistics and Machine Learning — Spring 2025
Homework 1: Due Sunday, February 16, 2025

Adam Smith (based on materials developed with Jonathan Ullman)

Collaboration and Honesty Policy Reminder: Collaboration in the form of discussion is allowed. However, all forms of cheating (copying parts of a classmate's assignment, plagiarism from papers or old posted solutions) are NOT allowed. A rough rule of thumb: you should be able to walk away from a discussion of a homework problem with no notes at all and write your solution on your own. Finding answers to problems on the Web or from other outside sources (these include anyone not enrolled in the class) is forbidden.

- *You must write up each problem solution by yourself without assistance, even if you collaborate with others to solve the problem.*
- You must identify your collaborators. If you did not work with anyone, you should write "Collaborators: none."
- Asking and answering questions in every forum the class provides (on Piazza, in class, and in office hours) is encouraged!
- Even though looking up answers is forbidden, using the web, generative AI, or similar resources to find alternative explanations of concepts you need for the homework *is* allowed, and encouraged. Asking Deepseek to solve a problem for you is not ok; asking it for an explanation of Chernoff boudns or for examples of encoding quardatic constraints in Gurobi is fine. You must **document your use of outside sources** and describe it at a high level in your solutions.

Problems to be handed in

1. (In-class Exercise 3 from Lecture 3) **More accurate reconstruction with more random queries**

   In this question we'll explore how to interpolate between the two reconstruction theorems we've seen. Specifically, we will prove a version of Theorem 2.5 that gives a more accurate reconstruction when we have $k \gg n$ queries. Suppose we have the following version of Claim 2.6 from the lecture notes:

   Claim 0.1. *Let $t \in \{-1, 0, +1\}^n$ be a vector with at least m non-zero entries and let $u \in \{0,1\}^n$ be a uniformly random vector. Then for every parameter $2 \le w \ll 2^m$*

   $$\mathbb{P}\left(|u \cdot t| \ge \frac{\sqrt{m \log w}}{10}\right) \ge \frac{1}{w} \tag{1}$$

   Using this claim, prove the following theorem

   Theorem 0.2. *If we ask $n^2 \ll k \ll 2^n$ queries, and all queries have error at most αn, then with extremely high probability, the reconstruction error is at most $O(\frac{\alpha^2 n^2}{\log(k/n)})$.*

   How does this theorem compare to the reconstruction attacks we've seen for $k \approx n^2$? What about $k \approx 2^{\sqrt{n}}$? What about $k \approx 2^n$?

2. (**Node sensitivity**) Imagine we have a data set that comes from a simple social network with n people. Each node in the graph is a person. For each person, we have the following data: a unique identifier u, their income $a_u \in [0,1]$ and their friend list F_u, which is a list of identifiers of other nodes. F_u can have any size—it can be empty, or consist of *all* othe nodes, or anything in between. We assume that friendship is symmetric: if Alice is on Bob's friend list, then Bob is on Alice's.

   We'll say two graphs are neighbors if they differ by changing the data for one node, including a_u and the list of edges connected to that node. Notice that changing one person's data can potentially affect everyone's else list of friends. Because of that, natural things we would like to compute can have high sensitivity. For example, the number of connected components in the graph can go from n (the current number of nodes) to 1 by changing the friend list of a single node.

   (a) As a function of n, what is the global sensitivity of each of the following functions? Give the best upper bounds and lower bounds that you can. (It should be possible to given an exact answer, like $n^2 - 1$), for each of these.)

   How does the sensitivity compare to the range of the function (that is, the difference between the largest and smallest possible values it can return)?

      i. the *number of edges* in the graph
      ii. the *number of triangles* in the graph
      iii. the *diameter of the graph* (this is the largest possible length of the shortest path betweent two nodes; we define the diameter of a disconnected graph to be the largest diamter of any of its connected components).
      iv. *Distance from bipartiteness*: this is the smallest number of nodes that must be changed for the graph to be bipartite.

   (b) *Income correlation*: How correlated are friends' incomes? Let μ be the average income in the graph $\mu = \frac{1}{n}\sum_u a_u$. The income correlation is

$$g(x) = \frac{1}{2\#(edges)} \sum_u \sum_{id' \in F_u} (a_u - \mu)(a_{id'} - \mu)\,.$$

   (We multiply the number of edges by 2 since each edge gets counted twice in the formula.)

   This quantity ranges between 0 and 1. Design a differentially private algorithm which approximates g with additive error $\pm O(1/n)$ on all graphs for which $\#(edges) \geq n^2/20$. (The constant 20 is arbitrary. In fact, one can get vanishing relative error under much weaker conditions.)

3. **(Histograms with Randomized Response)** We saw a randomized response protocol RR_ε where each participant inputs a single bit and announces a single output bit. What if each person's input is one of d different possibilities (say, the name of their favorite artist on Spotify) and we want to estimate a *histogram* of how many people have each possible input? Can we do this so that each person can do the randomization of their data entirely on their own?

   Consider the following approach, which we will call approach A:

   - Each person encodes their input $x_i \in [d]$ as a "one-hot" indicator vector $e_{x_i} = (0,0,\ldots,0,1,0,\cdots,0) \in \{0,1\}^d$ where the one is in entry x_i.
   - Each person applies the randomizer $RR_{\varepsilon/2}$ to each entry of this vector, to get a new vector $\vec{Y}_i \in \{0,1\}^d$, which they announce publicly.

Answer the following questions:

(a) Prove that the approach A is ε-differentially private.

(b) Demonstrate that, given the $\vec{Y}_i$'s, one can estimate the counts of every item with expected error $O\left(\frac{\sqrt{n \ln d}}{\varepsilon}\right)$. You many do this *either* by giving a proof, *or* by coding up the mechanism and the estimation algorithm, and plotting the accuracy it achieves for different values of n, d, ε. You should give three plots, each of which shows the effect of n, d, or ε.

(c) (*, optional for glory) Now consider approach B, where each person i does the following:

- Announce a random vector $u_i \in \{0,1\}^d$ (independent of their data)
- Announce $RR_\varepsilon(u_i(x_i))$ where $u_i(x_i)$ denotes entry number x_i of the vector u_i.

Show that approach B has similary utility to approach A (with the right estimation procedure). Also, show that it is ε-differentially private regardless of how u_i is chosen. In particular, we can choose the u_i's pseudorandomly and send only a seed used to generate u_i. This reduces the communication in the protocol dramatically! The pseudorandomness is only required for arguing accuracy.

4. **(Reconstruction from 2-way Marginals)** In class, we considered settings where the data set is binary. But we know that complex machine learning models sometimes encode even very complex inputs. Let's consider a setting where one can, somewhat surprisingly, reconstruct the entire data set from sufficiently rich statistics.

Suppose we have a data set consisting of n rows (one per person), with d binary attributes per row. A 1-way marginal describes the distribution in the data set of one particular attribute. Since the attributes are binary, a one-way marginal is the just count of how many 0's and 1's there are in a particular column. The (d-dimensional) vector of all one-way marginals is just the sum of the rows of the data.

	a	b	c	d
Alice	0	1	1	0
Bob	1	0	1	0
Carol	0	1	0	0

	a	b	c	d
One-way marginals (count of 1's)	1	2	2	0

Two-way marginal for a, b	
00	0
01	2
10	1
11	0

A *two-way marginal* describes the distribution of a pair of attributes. It consists of 4 counts, one for each of the pairs of bits 00, 01, 10, and 11. If we already know the one-way marginals for both attributes, we just need a single additional number such as the count of 11's. In the example above, if we reveal the one-way marginals together with the count of 11's for the attributes a, b (which is 0), then we can conclude the that 01 appears twice, 10 appears once, and 00 appears 0 times.

One- and two-way marginals are basic summary statistics about a dataset. The question is, when are they sufficiently rich to pin down the data set exactly?

Mathematically, if $X \in \{0,1\}^{0,1^{n\times d}}$ is the original sensitive table, then releasing the one- and two-way marginals is equivalent to releasing the matrix $X^T X$ and the parameters n and d. (Why? Check this for yourself.)

(a) How many different constraints does one learn by seeing $X^T X$? How does this compare to the number of variables in X? For which values of n and d would you expect to be able to recover X from $X^T X$?

(b) In Python, program the following experiment for values of n and d between 1 and 12. Specifcally, try all pairs n, d where $n \in \{3, 6, 12\}$ and $d \in \{1, 2, \ldots, 12\}$. Repeat the following 20 times for each pair (n, d):

   i. Choose X uniformly at random and compute $M = X^T X$.

   ii. Use Gurobi's free Python package API (see below) to find a binary matrix $\tilde{X}$ such that $\tilde{X}^T \tilde{X} = M$.

   iii. Check if $\tilde{X}$ and X are the same up to re-ordering of the rows. (How can you check this quickly?).

   For each setting of n and d, record the fraction of times that the experiment led to exact reconstruction.

(c) Do the results of Part 4b and 4a agree? Why or why not?

(d) (*, optional for glory) Gurobi's exact solver is too slow to solve this problem meaningfully for larger dimensions, at least on my laptop. Design a heuristic that scales to d and n in the hundreds, and carry out the experiment with $n = 200$ and different settings of d. For what values of d does your heuristic reconstruct the data exactly with reasonable probability (say at least 5% of the time)?

What to submit: You should submit a report answering the questions above, with your code attached as an appendix. (You have to submit a PDF on Gradescope, so you'll have to "print" your code to PDF and concatenate the two PDF files.)

Gurobi: To get Gurobi up and running, you will need to install it. This web page has instructions on getting Gurobi installed and up and running. I will also distribute an example Python notebook that might be helpful. As for the optional problem... feel free to use any standard Python packages. Your code should runnable on a typical laptop or PC. (For $n = 12$ and $d = 12$, my use of Gurobi was taking under 5–10 minutes to run all 20 trials.)