HW 1: Reconstruction and Membership Inference Attack

CS 5510 Data privacy and security, Fall 2025

September 18, 2025

1. **Reconstruction Attack**
   In the course Github repo, we have provided a fake healthcare dataset[1] on 100 individual patients. Among the variables in the dataset is `result` indicating whether or not the patient's tests came back normal (indicated by a 0 value) or abnormal (indicated by a 1 value).

   This is a sensitive piece of information, since it might reveal whether or not a given patient has an underlying health condition. The full data card for the dataset is given below.

Attribute	**Description**
`Age`	Integer in the range 0–100
`Sex`	0: male, 1: female
`Blood type`	0: A+, 1: A-, 2: B+, 3: B-, 4: AB+, 5: AB-, 6: O+, 7: O-
`Admission type`	0: elective, 1: urgent, 2: emergency
`Test results`	0: normal, 1: abnormal

   The example dataset represents the type of information a hospital might wish to share with medical researchers. To uphold strong data security practices, the hospital could restrict access to the raw data and instead provide access to the data through a controlled query interface.

   In this problem, you will run experiments to evaluate the performance of the reconstruction attack on determining patients' `result` status. Treat the following variables in the dataset as public (so as an attacker you know them for all of the individuals in the dataset): $=$ (`age`, `sex`, `blood`, `admission`) Each query in your attack should specify a "random" boolean predicate q on the public variables (e.g. $q(_i) = [\texttt{age}_i > 34$ && $\texttt{admission}_i == 1]$), and receive as an answer an approximation to the subset sum query:

$$\sum_{i:q(_i)=1} \texttt{result}_i, \tag{1}$$

   where i ranges over the 100 individuals in the healthcare dataset that we have provided.

   We have provided you with some (optional) starter code in `hw2_starter.py` [2] in the GitHub repo.

   This includes definitions of:

   - `data`, a DataFrame containing the dataset you will be attacking.
   - `pub`, the names of the columns we are treating as public knowledge.
   - `execute_subsetsums_exact(predicates)`, a function that takes as input a list `predicates` on the `pub` variables and returns the list of (exact) answers to the corresponding subset sum queries on `data`, computed as in Equation (1).
   - An example of using `execute_subsetsums_exact` to count both the number of female patients in the dataset and the number of emergency admissions.

[1] `https://github.com/opendp/cs208/blob/main/spring2025/data/fake_healthcare_dataset_sample100.csv`
[2] Starter code at `https://github.com/opendp/cs208/blob/main/spring2025/homeworks/ps2/ps2_starter.py`

- A function `make_random_predicate()` that returns a (pseudo)random predicate q, which you can use to emulate the random subset sums that are used in the reconstruction attack as presented in class.

Carry out your attack and experiments in the following steps:

(a) Write a function `reconstruction_attack(data_pub,predicates,answers)` that takes as input `data_pub`, a DataFrame restricted to public columns, a list `predicates` of predicates on the public attributes, and a list of (possibly approximate) `answers` to the queries, and returns an attempted reconstruction of the sensitive (`result`) column as an array of boolean values of length n, where n is the number of rows in `data_pub`. Test your attack against `data` using $2n$ random queries generated by `make_random_predicate` and answered by `execute_queries_exact`. It should, with high probability, reconstruct all of the sensitive bits correctly.

(b) Implement the following defenses by modifying `execute_subsetsums_exact`:

i. `execute_subsetsums_round(R,predicates)`: round each result to the nearest multiple of R.

ii. `execute_subsetsums_noise(sigma,predicates)`: add independent Gaussian noise of mean zero and variance σ^2 to each result.

iii. `execute_subsetsums_sample(t,predicates)`: given a parameter $t \in \{1, \ldots, n\}$, randomly subsample a set T consisting of t out of the n rows and calculate all of the answers using only the rows in T (scaling up answers by a factor of n/t).

(c) Finally, run experiments on how your attack performs against the three defenses above.

i. Create functions to compute the accuracy of the answers returned by each of the `execute_subsetsums_*` functions (root-mean-squared-error between answers and exact values) and success of the attack (average fraction of values $\texttt{result}_i$ that are successfully reconstructed).

ii. Vary parameters R, σ, and t as integers from 1 to n. For each parameter setting, run 10 experiments with fresh randomness and plot the averages of the accuracy and reconstruction success fractions.

iii. Compare the trade-off between accuracy and success of the attack. Make sure to identify the regime where your attack transitions from near-perfect reconstruction (fraction close to 1) to failed reconstruction (fraction reconstructed is no higher than the proportion of the majority value).

2. **A Bayesian Interpretation of MIAs**

In formulating MIAs as frequentist hypothesis tests, we condition on Alice either being in or not in the dataset. In a Bayesian formulation, we might instead assume that the adversary has a prior belief that Alice is in the dataset with some probability p. A convenient measure of the adversary's belief is the *odds* $O_{prior} = p/(1-p)$, which tends to ∞ as the certainty that Alice is in the dataset increases and tends to 0 as it decreases.

(a) Suppose an attacker carries out a Membership Inference Attack on Alice and receives an "In" result. Let O_{post} be the odds corresponding to Alice's belief conditioned on "In" result from the MIA. Write a formula for O_{post} in terms of O_{prior} and the TPR and FPR of the MIA (on the same data distribution).

(b) Using your formula, briefly discuss (in a sentence or two) the significance of having a very small FPR, even when the TPR is very large (e.g. TPR=1).

Collaborators and AI tools for HW

Homework should be completed individually. If you discuss the assignment with others, please acknowledge them in your report. ChatGPT and other AI tools should be treated the same way as collaboration with your classmates. You may use these tools to help you understand the material or support your brainstorming process, but you should not rely on them to directly solve the homework problems. If you do use such tools, you must cite them and include both the prompts you entered and the responses you received.