Building a Classifier with Differential Privacy

April 27, 2018

Due data & time 11:59pm on Apr. 27th, 2018. Email your code and report to the TA (geh@purdue.edu).
Late policy You have three extra days in total for all your projects. Any portion of a day used counts as one day; that is, you have to use integer number of late days each time. If you exhaust your three late days, any late project won't be graded.
Additional instructions (1) This project worths 10% of the course grade, You can work as a group of two or individually. If you work together, both students receive the same grade. (2) The submitted homework must be typed. Using Latex is recommended, but not required.

1 Background

In recent years, differential privacy (DP) has been increasingly accepted as the *de facto* standard for data privacy. In a DP setting, a data curator collects personal data from each individual, and produces outputs based on the dataset in a way that satisfies differential privacy. Informally, the DP notion requires any single element in a dataset to have only a limited impact on the output.

In this project, you will get familiar with the data processing pipeline which satisfies differential privacy. The project constructs with a written task (45 pts) and a programming task (55 pts). The written task helps you to deeply understand the theoretical problem of DP. In the programming task, you are asked to implement an algorithm that takes in a sensitive dataset and outputs a noisy histogram satisfying differential privacy. More specifically, you will need to implement several functions which completes the data processing pipeline which satisfies DP.

First, let us go over the concept of Differential Privacy.

Definition 1 (ϵ-Differential Privacy)**.** *An algorithm* $\mathbf{A}$ *satisfies* ϵ*-differential privacy (*ϵ*-DP), where* $\epsilon \geq 0$*, if and only if for any datasets* D *and* D' *that* differ on one element*, we have*

$$\forall T \subseteq \mathit{Range}(\mathbf{A}) : \ \mathsf{Pr}\left[\mathbf{A}(D) \in T\right] \leq e^{\epsilon} \mathsf{Pr}\left[\mathbf{A}(D') \in T\right], \tag{1}$$

where $\mathit{Range}(\mathbf{A})$ *denotes the set of all possible outputs of the algorithm* $\mathbf{A}$*.*

One way to understand the intuition of DP is the following "opting-out" analogy. We want to publish $\mathbf{A}(D)$, where D consists of data of many individuals. An individual objects to publishing $\mathbf{A}(D)$ because her data is in D and she is concerned about her privacy. In this case, we can address the individual's privacy concern by removing her data from D (or replacing her data with some arbitrary value) to obtain D' and publishing $\mathbf{A}(D')$. However, achieving privacy protection by removing an individual's data is infeasible. Since we need to protect everyone's privacy, following this approach means that we would need to remove everyone's data. DP tries to approximate the effect of opting out, by ensuring that any effect due to the inclusion of one's data is small. This is achieved by ensuring that for any output, one will see the same output with a similar probability even if any single individual's data is removed (unbounded DP), or replaced (bounded DP).

Bounded versus unbounded DP. When applying DP, an important choice is the precise condition under which D and D' are considered to be neighboring. In *Unbounded DP*, D and D' are neighboring if D can be obtained from D' by adding or removing one element. In *Bounded DP*, D and D' are neighboring if D can be obtained from D' by replacing one element in D' with another element. Throughout this project, we assume bounded DP is used.

Laplace Mechanism. To satisfy differential privacy, it requires to add some kind of noise during the process of data publishing. The Laplace mechanism is the first and probably most widely used mechanism for DP. It satisfies ϵ-DP by adding noise to the output of a numerical function. Specifically, to publish $f(D)$, one can publish $\tilde{f}(D) = f(D) + X$, where X is a random variable so that

$$\forall t, \frac{\mathsf{Pr}\left[\tilde{f}(D) = t\right]}{\mathsf{Pr}\left[\tilde{f}(D') = t\right]} = \frac{\mathsf{Pr}\left[f(D) + X = t\right]}{\mathsf{Pr}\left[f(D') + X' = t\right]} = \frac{\mathsf{Pr}\left[X = t - f(D)\right]}{\mathsf{Pr}\left[X' = t - f(D')\right]} \leq e^{\epsilon},$$

where X and X' are drawn from the same distribution. Let $d = f(D) - f(D')$, we need to ensure that

$$\forall x, \ \frac{\mathsf{Pr}\left[X = x\right]}{\mathsf{Pr}\left[X' = x + d\right]} \leq e^{\epsilon}. \tag{2}$$

We need to ensure that Eq. (2) holds for all possible d, and thus need the concept of the global sensitivity of f, which is the maximum change of f between two neighboring datasets D and D'.

Definition 2 (Global sensitivity)**.** *Let $D \simeq D'$ denote that D and D' are neighboring. The global sensitivity of a function f, denoted by Δ_f, is given below*

$$\Delta_f = \max_{D \simeq D'} |f(D) - f(D')|, \tag{3}$$

We want to ensure that Eq. (2) holds for all $d \leq \Delta_f$. In other words, the probability density function of the noise should have the property that if one moves no more than Δ_f units on the x-axis, the probability should increase or decrease by a factor of no more than e^{ϵ}, i.e., if one moves no more than 1 unit on the x-axis, the probability should change by a multiplicative factor of no more than e^{ϵ/Δ_f}.

The distribution that naturally satisfies this requirement is $\mathsf{Lap}\left(\frac{\Delta_f}{\epsilon}\right)$, the Laplace distribution, where $\mathsf{Pr}\left[\mathsf{Lap}(\beta) = x\right] = \frac{1}{2\beta} e^{-|x|/\beta}$. Note that

$$\frac{\mathsf{Pr}\left[\mathsf{Lap}(\beta) = x\right]}{\mathsf{Pr}\left[\mathsf{Lap}(\beta) = x + d\right]} \leq e^{d/\beta} \leq e^{\Delta_f/\beta} = e^{\epsilon}.$$

Theorem 1 (Laplace mechanism, scalar case)**.** *For any function f, the Laplace mechanism $\mathbf{A}_f(D) = f(D) + \mathsf{Lap}\left(\frac{\Delta_f}{\epsilon}\right)$ satisfies ϵ-DP.*

The Laplace mechanism can also be applied to a function f that outputs a vector, in which case, the global sensitivity Δ_f is the maximum L_1 norm of the difference between $f(D)$ and $f(D')$, i.e.:

$$\Delta_f = \max_{D \simeq D'} ||f(D) - f(D')||_1. \tag{4}$$

And noise calibrated to the global sensitivity should be added to all components of a vector.

Theorem 2 (Laplace mechanism, the vector case)**.** *The Laplace mechanism for a function f whose value is a k-dimensional vector, defined below, satisfies ϵ-DP.*

$$\mathbf{A}_f(D) = f(D) + \langle X_1, X_2, \cdots, X_k \rangle,$$

where $X_1, X_2, \cdots, X_k$ are i.i.d. random variables drawn from $\mathsf{Lap}\left(\frac{\Delta_f}{\epsilon}\right)$.

Sometimes, the algorithm is complicated and has multiple steps. To argue the whole algorithm is private, it is easy to proof each step is private, and use the following composition theorem:

Theorem 3 (General Sequential Composition)**.** *Let $\mathscr{A}_1, \mathscr{A}_2, \cdots, \mathscr{A}_k$ be k algorithms (that take auxiliary inputs) that satisfy ϵ_1-DP, ϵ_2-DP, $\cdots$, ϵ_k-DP, respectively, with respect to the input dataset. Publishing*

$$\mathbf{t} = \langle t_1, t_2, \cdots, t_k \rangle, \ \textit{where } t_1 = \mathscr{A}_1(D), t_2 = \mathscr{A}_2(t_1, D), \cdots, t_k = \mathscr{A}_k(\langle t_1, \cdots, t_{k-1} \rangle, D)$$

satisfies $(\sum_{i=1}^{k} \epsilon_i)$-DP.

Note that each of the $\mathscr{A}_i$ will read the whole dataset. If some operation only touches the noisy result, it will not contribute to ϵ.

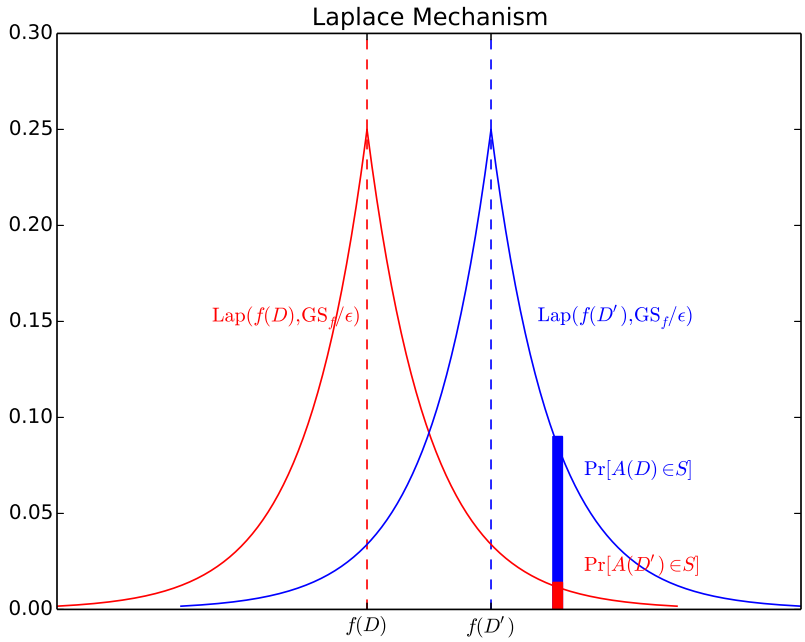

Figure 1: Differential Privacy via Laplace Noise.

2 Written Tasks (45 pts total)

To understand DP better, you are asked to answer the following theoretical problems. Note that all the questions are based on the bounded DP definition.

1. Suppose that all the n values are in the range $[a, b]$, what is the sensitivity for the following functions? (Hint: The questions on the slides are for unbounded DP.) (1) sum, (2) mean, (3) median. (**9 pts**)

2. Suppose that all the n values are either a or b. Suppose your DP algorithm outputs the number of a values and b values, respectively (with Laplace mechanism under privacy budget ϵ), and you now calculate the mean of these n values. What is the variance of this estimation? (Hint: Get familiar with Laplace distribution on wiki.) (**6 pts**)

3. Suppose that the n values each has two attributes, age and gender. You are going to publish a histogram (with Laplace mechanism and ϵ) of both attribute, with age bucketized into $[0-49]$ and $[50-100]$ (so there will be four numbers for: male-$[0-49]$, male-$[50-100]$, female-$[0-49]$, female-$[50-100]$). Now you want to estimate the number of male users, what is the variance of this estimation? What is the variance if you just use the gender attribute and ignore age when you publish the histogram? If each value has d binary attributes, what is the size of your histogram? (**10 pts**)

4. Suppose that all the n values are in the range $[a, b]$, and your task is to publish the 25th, 50th, and 75th percentiles (assume $n > 100$). Now you are given an algorithm that adds independent Laplace noise $\mathsf{Lap}(\beta_{25})$, $\mathsf{Lap}(\beta_{50})$, and $\mathsf{Lap}(\beta_{75})$, to the real answers, respectively ($\beta_{25} < \beta_{50} < \beta_{75}$). Your task is to find out (1) what is the sensitivity of this problem, (2) what is the final minimal ϵ this algorithm can achieve? (Hint: You should argue by a proof starting from Definition 1.) (**10 pts**)

5. If there is no public dataset available, and you instead use 10% of your sensitive data, sampled randomly, to find the desired histogram, without differential privacy. The remaining 90% of data is used to calculate the exact values in each cell of the histogram (and then add Laplace noise $\mathsf{Lap}\left(\frac{1}{\epsilon}\right)$). What will be the worst case ϵ for the whole process? (Hint: Look at each step separately, start from

Definition 1 with the histogram structure as the output; the sample probability for each value is thus 10%.) (**10 pts**)

3 Programming Tasks (55 pts total)

A data scientist in your company wants to share data with other companies by using a data processing pipeline. What the collaborating companies want is a classifier: they want to predict some behavior of their incoming customer. And in this specific task, they want to learn whether the customer is earning over $50K$ dollars a year. As the chief privacy officer, you understand that sharing raw data is dangerous, and suggest to enhance the existed data processing pipeline by implementing an algorithm which publishing a differentially private histogram.

3.1 Dataset

We use the UCI Machine Learning Adult Data Set (`https://archive.ics.uci.edu/ml/datasets/adult`) to predict whether the income of an individual exceeds $50K/yr based on census data. Also known as "Census Income" dataset. The dataset can be downloaded from `https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data`.

3.2 Differential Private Data Processing Pipeline

The sample code of the Data Processing Pipeline consists of 6 steps which are briefly described in the following:

1. **Parse the raw dataset**

   The program loads the UCI Adult data (CSV format) into the memory.

2. **Project the raw dataset into a histogram**

   Each record in the dataset has multiple attributes e.g., numerical attribute such as age and categorical attributes such as gender and marriage statues, etc. To project the dataset into a histogram, you need to select some buckets for each attribute. For example, the attribute marital-status has multiple different possible values: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse. You can bucketize this attribute into 'single' and 'married'. You can also have more buckets to carry more information, such as 'never-married', 'divorced', 'divorced-marry-again', etc. Furthermore, you can have unbalanced buckets such as 'divorced' and 'other'. If you want to ignore this attribute completely, you can have one 'any' bucket. The cells in the histogram are the Cartesian product of all buckets for all attributes. Once you have the cells, you can simply count the number of records for each cell (each record can only belong to one cell).

3. **Add noise to the histogram**

4. **Generate a synthetic dataset from the histogram**

   Suppose the generated histogram has N cells. Each cell represents a list of attributes and has count c_i, where $1 \leq i \leq n$. The synthetic dataset should contains c_i records for the i-th cell.

5. **Train the classification model based on the re-generated data**

   Split the synthetic dataset into training and test sets with 80/20. Train a logistic regression classification model with the split training data.

6. **Test the model prediction accuracy**

   Evaluate the model prediction accuracy against the test data and output the accuracy score.

The sample code provides steps 1 to 6, but it is not differential private. **Your task is to complete steps 2 and 3** to make the histogram to be differential private and also achieve a reasonable prediction accuracy. More specifically, after studying the dataset, you found three attributes, age, salary, and investment gain, are informative for annual income. And the average annual income for 40-year-olds are highest. You can keep the unbalanced buckets '$[40-50]$' and 'other' for age, and try different precision for salary and investment. By running at least 10 times for different ϵ value, find which histogram performs best. Regard to step 3, The noise you added in the histogram should satisfying the definition of bounded DP.

3.3 Programming Environment

Please use python 3 to implement your algorithm. You are allowed to use following python non-default packages: sklearn, pandas, numpy, scipy, and pickle (current version). Please email TA if your code needs to use other packages and explain the reason.

- **What's in the sample code?**
  - `exp.py` The main script for running data processing pipeline. You should not make any changes to this file.
  - `hist.py` You should make your implementation in this file. Make sure that your code does not change the function interface of `project_hist` and does not generate any command line output. You are free to add additional functions.
- **How to run the sample code?** Download the sample code and adult data. Put them under the same directory. Install packages in local if you do not have.

  ```
  pip3 install --user sklearn pandas numpy
  ```

  You can choose install package globally, requiring root privilege.

  ```
  sudo pip3 install sklearn pandas numpy
  ```

  To run the code, use the following command

  ```
  python3 exp.py --epsilon=0.1 --file=adult.data.txt
  ```

  For details of the sample code, please read the source code or check the command

  ```
  python3 exp.py --help
  ```

4 Submission

You should submit your implementation code `hist.py` and the project report to the TA. The project report should include both the written task and the description of your implementation.

5 Grading policy:

1. **Written Tasks**: **45 pts**
2. **Programming Tasks**: **55 pts**

   Given the histogram, we will synthesize a dataset and use logistic regression to train a model, which is used to test for mis-classification rate. The mis-classification rate is the average for three epsilon values: 0.1, 0.5, 1.0 (run 10 times for each value). Your grade will be depended on the prediction accuracy and the DP sanctification.

   Here are the specific distribution of the points:

- A brief report of the programming task is required. The report should describe your algorithm, the reason for your design, and the argument why your algorithm is ϵ-DP. (**10 pts**)
- As long as your algorithm works, you receive the implementation points (**10 pts**)
- As long as your algorithm satisfies DP (you should explain this in your report), you receive the privacy points. (**10 pts**)
- As long as your algorithm can beat majority vote (mis-classification rate $< 25\%$), you receive the base points (**10 pts**). Note that if the algorithm is not private, you cannot receive the points below.
- As long as your algorithm can achieve mis-classification rate $< 20\%$, you receive the advanced points. (**10 pts**)
- As long as your algorithm beat algorithms from other students, and the non-private example, you receive the top points. (**5 pts**)