COMS 4771 HW4 (Fall 2022)

Due: Sun Dec 04, 2022 at 11:59pm

This homework is to be done **alone**. No late homeworks are allowed. To receive credit, a typesetted copy of the homework pdf must be uploaded to Gradescope by the due date. You must show your work to receive full credit. Discussing possible approaches for solutions for homework questions is encouraged on the course discussion board and with your peers, but you must write your own individual solutions and **not** share your written work/code. You must cite all resources (including online material, books, articles, help taken from/given to specific individuals, etc.) you used to complete your work.

1 Privacy in data

Data and consumer privacy has been one of the major concerns in recent years. It has been found that even when anonymized, processed data may still be susceptible to vulnerability[1]. Researchers have put significant effort into properly defining and preserving privacy.

Among the efforts for designing good privacy criteria, a popular method is the so called **differential privacy**[2]. It is a powerful tool for providing privacy preserving answers to statistical queries over databases. By adding a bit of noise to the results, it guarantees that the distribution of *noisy* query answers changes very little with the addition or deletion of any individual datapoint. Differential privacy has been a gold standard for privacy in academia since being proposed and has been gradually adopted to some real-world applications in recent years[3]. In this question, we will explore the definition of differential privacy and study two simple ways to achieve it.

Before formally defining differential privacy, we first introduce the following notions.

Probability simplex: Given a discrete set B, the probability simplex over B, denoted $\Delta(B)$ is the set:

$$\Delta(B) = \{x \in \mathbb{R}^{|B|} : \forall i, x_i \geq 0, \text{ and } \sum_{i=1}^{|B|} x_i = 1\}.$$

Randomized Algorithms: A randomized algorithm $\mathcal{M}$ with domain A and range B is an algorithm associated with a total map $M : A \to \Delta(B)$. On input $a \in A$, the algorithm $\mathcal{M}$ outputs $\mathcal{M}(a) = b$ with probability $(M(a))_b$ for each $b \in B$. The probability space is over the coin flips of the algorithm $\mathcal{M}$.

For simplicity we will avoid implementation details and we will consider databases as histograms.

[1] See e.g. Robust De-anonymization of Large Datasets, How to Break Anonymity of the Netflix Prize Dataset by Arvind Narayanan and Vitaly Shmatikov).

[2] See e.g. The Algorithmic Foundations of Differential Privacy by Cynthia Dwork and Aaron Roth.

[3] e.g. Starting with IOS 10, Apple began to use Differential Privacy technology to help discover the usage patterns of a large number of users while preserving individual privacy.

Given a universe $\mathcal{X}$, a histogram over $\mathcal{X}$ is an object in $\mathbb{N}^{|\mathcal{X}|}$. Then we have the following.

Distance Between Databases: The l_1 norm of a database $x \in \mathbb{N}^{|\mathcal{X}|}$ is denoted $\|x\|_1$ and is defined as:

$$\|x\|_1 = \sum_{i=1}^{|\mathcal{X}|} |x_i|$$

The l_1 distance between two databases x and y is defined as $\|x - y\|_1$. We call two databases $x, y \in \mathbb{N}^{\mathcal{X}}$ adjacent if $\|x - y\|_1 \leq 1$.

With the above definitions, now we can formally define differential privacy:

Differential Privacy: Given a randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ and range R, then we say that $\mathcal{M}$ is (ϵ, δ)-differentially private if for any subset $S \subseteq R$ we have:

$$\mathbb{P}[\mathcal{M}(x) \in S] \leq \exp(\epsilon)\, \mathbb{P}[\mathcal{M}(y) \in S] + \delta,$$

for every adjacent $x, y \in \mathbb{N}^{|\mathcal{X}|}$.
Often we will consider the simpler case where $\delta = 0$, in this case notice that we can rewrite the requirement of differential privacy as requiring that for every adjacent $x, y \in \mathbb{N}^{|\mathcal{X}|}$ and for any $r \in R$ (assuming R is discrete) we have

$$\exp(-\epsilon) \leq \frac{\mathbb{P}[\mathcal{M}(x) = r]}{\mathbb{P}[\mathcal{M}(y) = r]} \leq \exp(\epsilon)$$

The quantity

$$\ln \frac{\mathbb{P}[\mathcal{M}(x) = r]}{\mathbb{P}[\mathcal{M}(y) = r]}$$

is often called the privacy loss of the algorithm $\mathcal{M}$. If R is continuous, the same simplification holds by replacing probabilities ($\mathbb{P}$) by densities.

The usefulness of this definition stems from noticing that it becomes more difficult to identify the presence of a single individual element in the database if this ratio is small. With this definition, the next question to ask is how can we achieve it without sacrificing too much accuracy?

The most intuitive idea is to introduce some randomness to the data being queried or add some noise to the data. The trade-off between privacy and accuracy is intuitive: the more "complicated" noise we add or more randomness we introduce to the query, the less information the user will get and thus achieve a higher level of privacy.

(i) First let's consider a scenario where a user is answering some sensitive yes/no question and we want to preserve privacy while collecting useful information.
Let $k \in [0, 1]$ and sensitive_question(x) : $\mathbb{N}^{|\mathcal{X}|} \to \{$**True**, **False**$\}$, consider the following Randomized Response (**RR**) procedure:

```
**Randomized Response**$(x)$
uniformly sample a real number $m$ from $[0, 1]$.
**if** $m < k$ **then**
    **if** sensitive_question$(x)$ **then**
```

```
        answer yes
    **else**
        answer no
    **end if**
**else**
    flip an unbiased coin
    **if** heads **then**
        answer yes
    **else**
        answer no
    **end if**
**end if**
```

(a) **Privacy Guarantee of RR**: What value should k take if we want to make RR to be $(\epsilon, 0)$ differentially private?

Hint: Consider two neighboring databases x and y such that sensitive_question(x) $=$ yes and sensitive_question(y) $=$ no.

(b) **Accuracy Guarantee of RR:** What accuracy can we achieve in this case? In other words, what is $\mathbb{P}$[sensitive_question(x) $= RR(x)$]?

(ii) We will now consider the case when the output domain (i.e. R) is continuous. An idea is to add some noise to our original output. Before we proceed, we first define a notion to capture the possible influence that a single individual can have on the result of a numeric query. This influence can be captured by the notion of global sensitivity.

l_1 **global sensitivity:** The l_1 global sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}$ is:

$$\Delta f := \max\left\{|f(x) - f(y)| \mid x, y \in \mathbb{N}^{|\mathcal{X}|} \text{ adjacent}\right\}.$$

Now, let's consider the following procedure **A** for privacy preservation:
Given any function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}$, define:

$$\mathcal{M}_L(x, f(\cdot), \epsilon) = f(x) + Y$$

where Y is a random variable drawn from $L(y; \frac{\Delta f}{\epsilon})$, where

$$L(y; b) = \frac{1}{2b} \exp\left(-\frac{|y|}{b}\right)$$

is the distribution's probability density function (PDF).

(a) **Privacy Guarantee of A**: What level of differential privacy (in terms of ϵ, δ) can this procedure achieve? Justify your answer with a detailed derivation/proof.

(b) **Accuracy Guarantee of A**: Let $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}$. Show that $\forall p \in (0, 1]$:

$$\mathbb{P}\left[\left|f(x) - \mathcal{M}_L(x, f(\cdot), \epsilon)\right| \geq \left(\frac{\Delta f}{\epsilon}\right) \ln(1/p)\right] = p.$$

Hint: First show that the above distribution has a tail bound that guarantees that if Z is drawn from $L(y;b)$ then

$$\mathbb{P}\big[|Z| \geq bt\big] = \exp(-t).$$

Then, use this tail bound to prove the problem's statement.

2 Studying k-means

Recall that in k-means clustering we attempt to find k cluster centers $c_j \in \mathbb{R}^d, j \in \{1, \ldots, k\}$ such that the total (squared) distance between each datapoint and the nearest cluster center is minimized. In other words, we attempt to find $c_1, \ldots, c_k$ that minimizes

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \|x_i - c_j\|^2, \tag{1}$$

where n is the total number of datapoints. To do so, we iterate between assigning x_i to the nearest cluster center and updating each cluster center c_j to the average of all points assigned to the jth cluster (aka Lloyd's method).

(i) **[it is unclear how to find the best k, i.e. estimate the correct number of clusters!]** Instead of holding the number of clusters k fixed, one can think of minimizing (1) over both k and c. Show that this is a bad idea. Specifically, what is the minimum possible value of (1)? what values of k and c result in this value?

(ii) **[suboptimality of Lloyd's method]** For the case $d = 1$ (and $k \geq 2$), show that Lloyd's algorithm is *not* optimal. That is, there is a suboptimal setting of cluster assignment for some dataset (with $d = 1$) for which Lloyd's algorithm will not be able to improve the solution.

(iii) **[improving k-means quality]** k-means with Euclidean distance metric assumes that each pair of clusters is linearly separable (see part b below). This may not be the desired result. A classic example is where we have two clusters corresponding to data points on two concentric circles in the $\mathbb{R}^2$.

   (a) Implement Lloyd's method for k-means algorithm and show the resulting cluster assignment for the dataset depicted above. Give two more examples of datasets in $\mathbb{R}^2$, for which optimal k-means setting results in an undesirable clustering. Show the resulting cluster assignment for the two additional example datasets.

   (b) Show that for $k = 2$, for any (distinct) placement of centers c_1 and c_2 in $\mathbb{R}^d$, the cluster boundary induced by minimizing the k-means objective (i.e. Eq. 1) is necessarily linear.

One way to get a more *flexible* clustering is to run k-means in a transformed space. The transformation and clustering is done as follows:

- Let G_r denote the r-nearest neighbor graph induced on the given dataset (say the dataset has n datapoints), that is, the datapoints are the vertices (notation: v_i is the vertex associated with datapoint x_i) and there is an edge between vertex v_i and v_j if the corresponding datapoint x_j is one of the r closest neighbors of datapoint x_i.
- Let W denote the $n \times n$ edge matrix, where

$$w_{ij} = \mathbf{1}[\exists \text{ edge between } v_i \text{ and } v_j \text{ in } G_r].$$

- Define $n \times n$ diagonal matrix D as $d_{ii} := \sum_j w_{ij}$, and finally define $L = D - W$.
- Compute the *bottom* k eigenvectors/values of L (that is, eigenvectors corresponding to the k smallest eigenvalues). Let V be the $n \times k$ matrix of of the bottom eigenvectors. One can view this matrix V as a k dimensional representation of the n datapoints.
- Run k-means on the datamatrix V and return the clustering induced.

We'll try to gain a better understanding of this transformation V (which is basically the lower order spectra of L).

(c) Show that for any vector $f \in \mathbb{R}^n$,

$$f^{\mathsf{T}} L f = \frac{1}{2} \sum_{ij} w_{ij} (f_i - f_j)^2.$$

(d) L is a symmetric positive semi-definite matrix.

(e) Let the graph G_r have k connected components $C_1, \ldots, C_k$. Show that the $n \times 1$ indicator vectors $\mathbb{1}_{C_k}, \ldots, \mathbb{1}_{C_k}$ are (unnormalized) eigenvectors of L with eigenvalue 0. (the ith component of an indicator vector takes value one iff the vertex v_i is in the connected component)

Part (e) gives us some indication on why the transformation V (low order spectra of L) is a reasonable representation. Basically: (i) vertices belonging to the same connected component/cluster (ie, datapoints connected with a "path", even if they are located far away or form odd shapes) will have the same value in the representation $V = [\mathbb{1}_{C_1}, \ldots, \mathbb{1}_{C_k}]$, and (ii) vertices belonging to different connected component/cluster will have distinct representation. Thus making it easier for a k-means type algorithm to recover the clusterings.

(f) For each of the datasets in part (a) (there are total three datasets), run this flexible version of k-means in the transformed space. Show the resulting clustering assignment on all the datasets. Does it improve the clustering quality? How does the choice of r (in G_r) affects the result?

(You must submit your code for parts (a) and (f) to receive full credit. All plots and analysis must be included in the pdf document.)

3 Non-linear Dimensionality Reduction

Here is a simple way to accomplish non-linear dimensionality reduction:

Input: High-dimensional dataset $X = x_1, \ldots, x_n \in \mathbb{R}^D$, target dimension d
Output: $y_1, \ldots, y_n \in \mathbb{R}^d$ as the low-dimensional mapping of the given dataset

- Construct a k-nearest neighbor graph[4] G on X
- Let π_{ij} denote the shortest path between datapoints x_i and x_j according to G.

[4] A k-nearest neighbor graph is simply a graph where the nodes correspond to the datapoints, and edges correspond to the Euclidean distance between the corresponding datapoints. Important: For each node/datapoint, only the k closest nodes are connected, with edge weight being the Euclidean distance between the nodes.

- Select $y_1, \ldots, y_n \in \mathbb{R}^d$ according to the following minimization problem

$$\text{minimize}_{y_1,\ldots,y_n} \sum_{i,j} \big(\|y_i - y_j\| - \pi_{ij}\big)^2$$

(i) What is the derivative of the optimization function above with respect to a fixed y_i?

(ii) Is the optimization above convex with respect a fixed y_i? Why or why not.

(iii) Write a program in your preferred language to find a low-dimension embedding of any given input dataset. You must submit your code to receive full credit.

(iv) For the two datasets provided, compute your two dimensional embedding. Plot the original 3D data, its 2D PCA projection and the results obtained from your 2D embedding.

Analyze the quality of results you obtain. Under what circumstances would

- this non-linear embedding fail/succeed?
- PCA will perform better/worse than this non-linear embedding?