Due Date: Nov 4, 3pm

Please typeset all responses in LaTeX and submit a PDF through Markus (instructions to follow).

In responding to questions in Section 1, please provide *both* written responses *and* code[1] in your submission.

1 Coding Exercises

Remark: Helper code and data can be found at the following github repo:
https://github.com/ecreager/csc2541-f19/tree/master/assignment2

1.1 Preliminaries

This assignment explores variations on the theme of *private logistic regression*. We will provide code that performs non-private logistic regression, from which you can adapt your private algorithms. Each question uses a different set of privacy assumptions to reflect the variety of contexts that inform private machine learning in practice.

Dataset We will use a binarized version of the MNIST dataset, which uses only two of the usual ten classes. Specifically, the task is to classify between 1's and 7's. We will provide utilities for loading the train and test dataset, as well as a dataset that neighbors the train set by one random example.

1.2 Private Model Selection

Overview In this question we will give you K classifiers, and you will implement a private algorithm for answering the question "which classifier does best on the test data"? To report a private answer we return sample from a distribution over models using the Exponential Mechanism.

Assumptions The test data are considered private; the train data are not needed because the classifiers are pre-trained.

Deliverables

- What score function should we use to compare models on the test set? Briefly justify your answer and state the sensitivity of this score.

[1] For the coding questions please submit *all* of the python files included in the git repo. You should modify the files `model_selection.py`, `empirical_sensitivity.py`, and `output_perturbation.py` to include your solution code. Your PDF writeup should also address any of the qualitative deliverables (discussing trends, etc.) specified in the coding exercises.

- Implement the Exponential Mechanism using the score function chosen above. Note that the `exponential_mechanism` function takes ϵ as an argument. This is the ϵ your implementation should achieve, so you will need to figure out how to scale the scores appropriately[2] to achieve it.
- Once your implementation is working, run the main function with parameter values that with that sweep $\epsilon \in \{1, 2, 4\}$ and $n \in \{1, 10, 100, 1000\}$. In other words, there should be 12 plots altogether, one for each (ϵ, n) pair. Look over the plots and comment on any trends you notice.

1.3 Empirical Sensitivity

Overview Chaudhuri and Monteleoni (2009) states that the sensitivity of a logistic regressor that learns from n examples (with input norm at most 1) with regularizer λ is at most $\frac{2}{n\lambda}$. This is the theoretical upper bound. What values of ℓ_2 sensitivity do we see in practice for the binary MNIST classification task?

Assumptions Training data are considered private, but for the purposes of this question all of our exploratory work including plotting is considered "inside the privacy barrier", i.e., we do not need differential privacy algorithms in this question because we are not publishing our results "to the public".

Deliverables Generate a histogram of the ℓ_2 sensitivity of the logistic regression loss across 100 neighboring training sets $\{D, D'\}$ (use the same λ for all runs) by doing the following 100 times:

- Set $n = 100$.
- Generate a pair of neighboring training sets $\{D, D'\}$ using a data seed that hasn't been used so far.
- Train a λ-regularized logistic regressor on each of D and D', resulting in two weights w and w'.
- Compute $||w - w'||_2$.

We give you template code for this for-loop so you only need to implement the last two items. Plot a histogram of $||w - w'||_2$ across all 100 runs. Show the theoretical bound from Chaudhuri and Monteleoni (2009) as a vertical line on the same plot. We also provide code for the plotting. Briefly describe your observations, and report the training hyperparameters you use such as learning rate, momentum term, λ, etc.

Plot the neighboring data points that yield with the maximum $||w - w'||_2$. Do the same for the neighboring data points that yield the minimum $||w - w'||_2$. This is the only part of the assignment where you need to implement plotting code from scratch.

[2] Also, watch out for numerical issues when normalizing probabilities; you may look to `torch.distributions.categorical.Categorical` or `numpy.logsumexp` as helper code to side step these issues.

Finally briefly describe what a "worst case" neighboring dataset pair (one that approaches the theoretical limit) might look like (both the images and label values).

Bonus Describe how the histogram changes in response to changes in λ. What about in response to changes in n?

1.4 Private Training by Output Perturbation

Overview In this question we want publish a differentially private parameter vector by adding calibrated noise to parameters derived from non-private logistic regression training. This approach is called output perturbation.

Assumptions Training data are considered private, but test data are considered public

Deliverables Look over Algorithm 1 from Chaudhuri and Monteleoni (2009). Start by answering the following True/False propositions:

- With λ and ϵ fixed, increasing n requires Algorithm 1 to add more noise to w^* [True/-False]
- With λ and n fixed, increasing ϵ requires Algorithm 1 to add more noise to w^* [True/-False]
- With ϵ and n fixed, increasing λ requires Algorithm 1 to add more noise to w^* [True/-False]

To answer these questions, and for your implementation, it will be helpful to clarify the exact parameterization used by Chaudhuri and Monteleoni (2009) to denote the Gamma distribution[3]. Chaudhuri and Monteleoni (2009) use the shape-scale parameterization (the version with parameters k, θ on Wikipedia). On the other hand, PyTorch uses the concentration-rate parameterization (the version with parameters α, β on Wikipedia). In PyTorch the shape parameter is referred to as "concentration". We will give you helper functions that work with either parameterization, but you will need to be careful in your implementation to make sure you are adding the right amount of noise.

Now implement Algorithm 1 from Chaudhuri and Monteleoni (2009) to train a private logistic regressor on the train data with $\epsilon = 2, n = 1000$. You may choose the value for λ. Report the performance gap between this private logistic regressor and a standard non-private logistic regressor on the public test data.

Next, try tuning λ and n by hand. See if you can exceed 90% test accuracy with $\epsilon = 2$.

Bonus Implement Algorithm 2 from Chaudhuri and Monteleoni (2009) (the objective perturbation approach). How does this approach perform compared with parameter perturbation perturbation in terms of test accuracy for a given ϵ.

[3] https://en.wikipedia.org/wiki/Gamma_distribution

2 Written Exercises

2.1 Differential Privacy

For this question, let M be a (randomized) algorithm that takes as input a database $x \in \mathcal{X}^n$ and outputs, for each choice of random bits s, some element $r \in R$, where R is the range. (Thus we can view $M(x)$, the output of M on x, as a probability distribution over R.)

(2.1a) . Prove that the following two definitions of $(\epsilon, 0)$-DP are the same

- For every two neighboring databases x, y and for each element $r \in R$,

$$Pr[M(x) = r] \leq e^{\epsilon} Pr[M(y) = r].$$

- For every two neighboring databases x, y and for every subset $S \subseteq R$,

$$Pr[M(x) \in S] \leq e^{\epsilon} Pr[M(y) \in S].$$

(2.1b) Are the two definitions still equivalent for (ϵ, δ)-DP? Prove your answer.

2.2 Local Differential Privacy

Recall the Local Laplace Mechanism $\mathcal{M}$ seen in class. You will prove its privacy and accuracy guarantees. Each agent i holds a sample x_i drawn independently at random from an unknown distribution P on the sample space $\mathcal{X}$. Given a statistical query $\phi : \mathcal{X} \to [-1, +1]$, the aim is is to produce an estimate z of $\mathbb{E}_{x \sim P}[\phi(x)]$. To do so, each agent i releases $\mathcal{R}(x_i) = \phi(x_i) + w_i$ where w_i is drawn indepndently from $\text{Lap}(1/\varepsilon)$. The estimate z made by the mechanism is given by $z = \frac{1}{n} \sum_{i=1}^{n} \mathcal{R}(x_i)$.

(2.2a) (Privacy of Local Laplace Mechanism.) Suppose that the output of the mechanism $\mathcal{M}$ includes, not only z, but also the the values $\mathcal{R}(x_1), \ldots, \mathcal{R}(x_n)$ communicated by the individual agents. Explain why that mechanism would still be ε-differentially private.

You may assume that the central model version of the Laplace mechanism is ε-differentially private, and you may use the basic composition properties of differential privacy.

(2.2b) (Accuracy of Local Laplace Mechanism) Show, for some $n = O\left(\frac{1}{\tau^2 \varepsilon^2}\right)$, that

$$\Pr_{X,W}[|z - \mathbb{E}_{x \sim P}[\phi(x)]| \leq \tau] \geq 3/4.$$

where the probability is with respect to the randomness of $X = (x_1, \ldots, x_n)$ and $W = (w_1, \ldots, w_n)$.

References

Chaudhuri, K. and C. Monteleoni (2009). Privacy-preserving logistic regression. In *Advances in neural information processing systems*, pp. 289–296.