CS-GY 6923: Lecture 7
A Brief Introduction to Differential Privacy

NYU Tandon School of Engineering, Prof. Christopher Musco

data leakage

As we saw in the text generation lab, machine learning algorithms are prone to leak information about their training data:

> arm towards the viewer. Gregor then turned to look out the window at a barren sister only needed to hear the visitor's first words of greeting and he knew who calm, "I'll get dressed straight away now, pack up my samples and set off. Will again, "seven o'clock, and there's still a fog like this." And he lay there sighing, harder than before, if that was possible, he felt that the lower part of his body a

Here, our generative model revealed entire sentences from the training input. This is a quality issue, but can also be a privacy issue.

data leakage

Many modern ML systems trained on user data.

- Smart Compose in Gmail (trained on user emails).
- Generative AI for medical record taking (trained on patient health data).
- Github Copilot trained on public and private repositories.

```
//apa.js
//create an AngularEvaporate instance

$scope.ae = new AngularEvaporate
({
  bucket: 'motoroller',
  aws_key:              ,
  signerUrl: '/signer',
  logging: false
});
```

data leakage

Even if models do not directly generate private data, it can sometimes be extracted from them.

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Nicholas Carlini[1,2] Chang Liu[2] Úlfar Erlingsson[1] Jernej Kos[3] Dawn Song[2]

[1]*Google Brain* [2]*University of California, Berkeley* [3]*National University of Singapore*

Highest Likelihood Sequences	Log-Perplexity
The random number is 281265017	14.63
The random number is 281265117	18.56
The random number is 281265011	19.01
The random number is 286265117	20.65
The random number is 528126501	20.88
The random number is 281266511	20.99
The random number is 287265017	20.99
The random number is 281265111	21.16
The random number is 281265010	21.36

Perplexity = inverse likelihood. So secret is not likely to be generated, but we can find it if we look for sequences with higher

the privacy challenge

How do we balance privacy concerns with the desire to train models on as much data as possible?

formalizing privacy

There have been many many attempts to formalize what it means for a machine learning algorithm or system to be private.

Calibrating Noise to Sensitivity in Private Data Analysis

Cynthia Dwork[1], Frank McSherry[1], Kobbi Nissim[2], and Adam Smith[3⋆]

[1] Microsoft Research, Silicon Valley. {dwork,mcsherry}@microsoft.com
[2] Ben-Gurion University. kobbi@cs.bgu.ac.il
[3] Weizmann Institute of Science. adam.smith@weizmann.ac.il

Differential Privacy has become the gold standard definition.

Clear theoretical founding, widely used in implemented systems (TensorFlow, US Census statistics, Apple User data, etc.)

differential privacy

Definition based on notation of neighboring datasets.

Definition: A dataset $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ is neighbors of a dataset $\mathbf{X}' = [\mathbf{x}'_1, \ldots, \mathbf{x}'_n]$ if:

$$\mathbf{x}_i = \mathbf{x}'_i \text{ for all but one value of } i \in \{1, \ldots n\}.$$

I.e., $\mathbf{x}_j \neq \mathbf{x}'_j$ for a single index j.

Alernative but closely related definition: $\mathbf{X}$ and $\mathbf{X}'$ are neighbors if $\mathbf{X}'$ can be obtained by adding or removing a single data point from $\mathbf{X}$.

differential privacy

Definition

An algorithm $\mathcal{A}$ satisfies ϵ-differential privacy if, for any two neighboring datasets $\mathbf{X}$, $\mathbf{X}'$, and any possible output of the algorithm $\mathbf{z}$,

$$\Pr[\mathcal{A}(\mathbf{X}) = \mathbf{z}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{X}') = \mathbf{z}].$$

In the context of machine learning, $\mathcal{A}$ could be the training procedure and $\mathbf{z}$ could be, e.g., the model weights.

In the context of databases/statistical applications, $\mathcal{A}$ might implement a simple statistic function like the mean:

$$\frac{1}{n} \sum_{i=1}^{n} x_i.$$

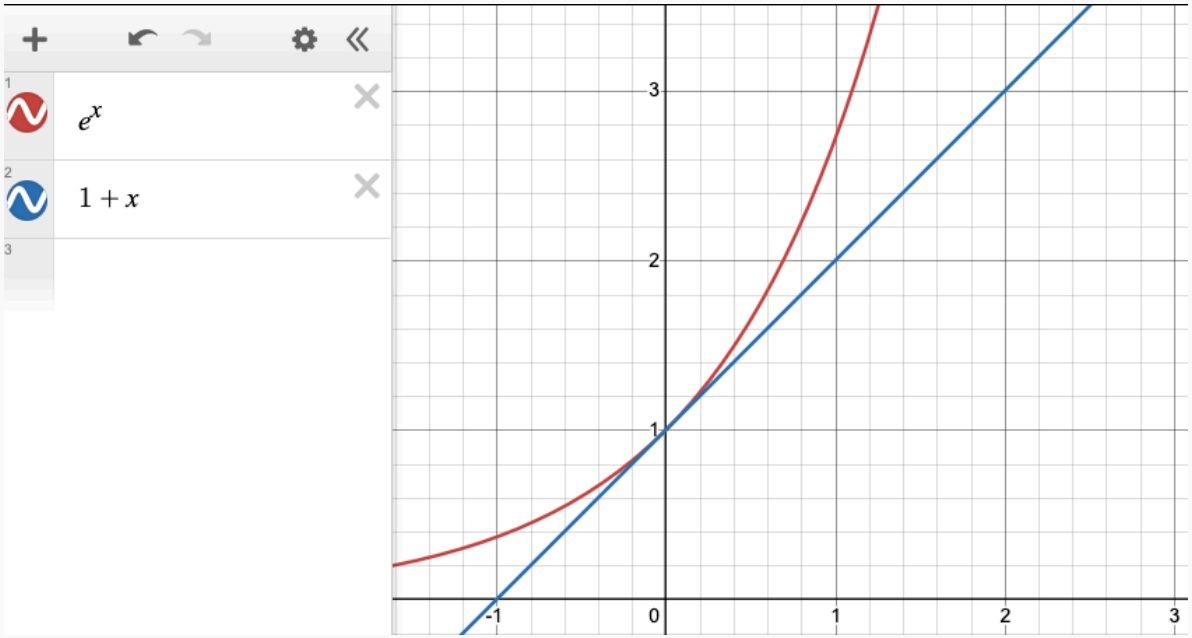
differential privacy

Definition

An algorithm $\mathcal{A}$ satisfies ϵ-differential privacy if, for any two neighboring datasets $\mathbf{X}$, $\mathbf{X}'$, and any possible output of the algorithm $\mathbf{z}$, $\Pr[\mathcal{A}(\mathbf{X}) = \mathbf{z}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{X}') = \mathbf{z}]$.

$\Pr[A(x') = z] \leq e^{\epsilon} \Pr[A(x) = z]$

Think of ϵ as a reasonably small constant. E.g. $\epsilon \in (0, 5]$. For small ϵ, $e^{\epsilon} \approx (1 + \epsilon)$.



differential privacy

Definition

An algorithm $\mathcal{A}$ satisfies ϵ-differential privacy if, for any two neighboring datasets $\mathbf{X}$, $\mathbf{X}'$, and any possible output of the algorithm $\mathbf{z}$, $\Pr[\mathcal{A}(\mathbf{X}) = \mathbf{z}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{X}') = \mathbf{z}]$.

In words, differential privacy says that including an individuals data in a dataset $\mathbf{X}$ can only increase or decrease the probability of observing any particular output by a small factor.

Inherently a property of randomized algorithms. Obtaining differentially private machine learning methods will require **adding randomness to the training process**.

differential privacy properties

Postprocessing property: If an algorithm $\mathcal{A}(\mathbf{X})$ is ϵ-DP, then $\mathcal{B}(\mathcal{A}(\mathbf{X}))$ is ϵ-DP for any (possibly non-private) algorithm $\mathcal{B}$.

Composition property: If an algorithm $\mathcal{A}_1$ is ϵ_1-DP and $\mathcal{A}_2$ is ϵ_2-DP, then $\mathcal{B}(\mathcal{A}_1(\mathbf{X}), \mathcal{A}_2(\mathbf{X}))$ is $(\epsilon_1 + \epsilon_2)$-DP.

differential privacy

There are many ways to add randomness. Perhaps the most common is noise injection.

Simple example: Suppose $\mathbf{X}$ contains scalar values $x_1, \ldots, x_n \in \{0, 1\}$. Suppose we want to compute the average, $Q(\mathbf{X}) = \frac{1}{n} \sum_{i=1}^{n} x_i$.

Naively, adding or removing a point from the dataset changes the average by $\pm\frac{1}{n}$ with probability 1, so, naively, a mean computation is not differentially private.

Differentially Private Estimate of Q:

- Generate an appropriate random number η.
- Return $Q(\mathbf{X}) + \eta$.

noise injection

Example $= \mathbf{X} = \{0, 1, 1, 0, 0, 0\}, \mathbf{X}' = \{0, 1, 1, 0, 1, 0\}$.

$\frac{2}{6} : \frac{1}{3}$ $\qquad$ $\frac{3}{6} : 1/2$

$A(x) = \frac{1}{3} + .2$ $\qquad$ $1/2 + .4$

$.1$ $\qquad$ $-.3$

$-.5$ $\qquad$ $.2$

Trade-off between privacy and accuracy.

what type of noise and how much?

Theorem (Laplace Mechanism)

For a function Q with sensitivity Δ_Q,

$$\mathcal{A}(\mathbf{X}) = Q(\mathbf{X}) + Lap(\Delta_Q/\epsilon)$$

is ϵ-differentially private.

Sensitiviy $\Delta_Q = \max_{\text{neighboring } \mathbf{x},\mathbf{x}'} |Q(\mathbf{X}) - Q(\mathbf{X}')|$.

What is Δ_Q for $Q(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} x_i$?

$x_1 \ldots x_n \in \{0, 1\}$

$Lap(b)$ is a Laplacian random variable with parameter b (which means variance $2b^2$). PDF is:

$$p_b(\eta) = \frac{1}{2b} e^{-|\eta|/b}$$

laplace mechanism analysis

Theorem (Laplace Mechanism)

For a function Q with sensitivity Δ_Q,
$\mathcal{A}(\mathbf{X}) = Q(\mathbf{X}) + Lap(\Delta_Q/\epsilon)$ is ϵ-differentially private.

Proof: For any possible output z,

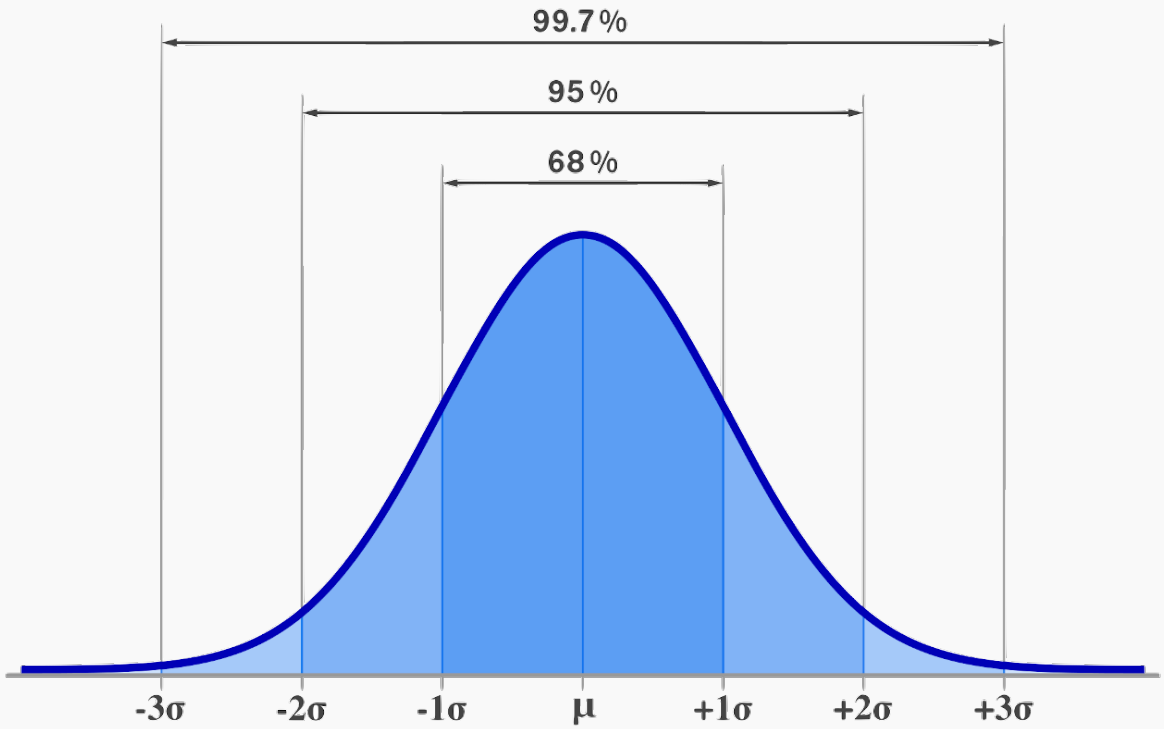
- $\Pr[\mathcal{A}(\mathbf{X}) = z] = \frac{1}{2(\Delta_Q/\epsilon)} e^{-|Q(\mathbf{X})-z|/(\Delta_Q/\epsilon)}$
- $\Pr[\mathcal{A}(\mathbf{X}') = z] = \frac{1}{2(\Delta_Q/\epsilon)} e^{-|Q(\mathbf{X}')-z|/(\Delta_Q/\epsilon)}$

$$\frac{\Pr[\mathcal{A}(\mathbf{X}) = z]}{\Pr[\mathcal{A}(\mathbf{X}') = z]} = e^{-(|Q(\mathbf{X})-z|-|Q(\mathbf{X}')-z|)/(\Delta_Q/\epsilon)}$$

$$\leq e^{\frac{|Q(\mathbf{X})-Q(\mathbf{X}')|}{\Delta_Q/\epsilon}} \leq e^{\epsilon}.$$

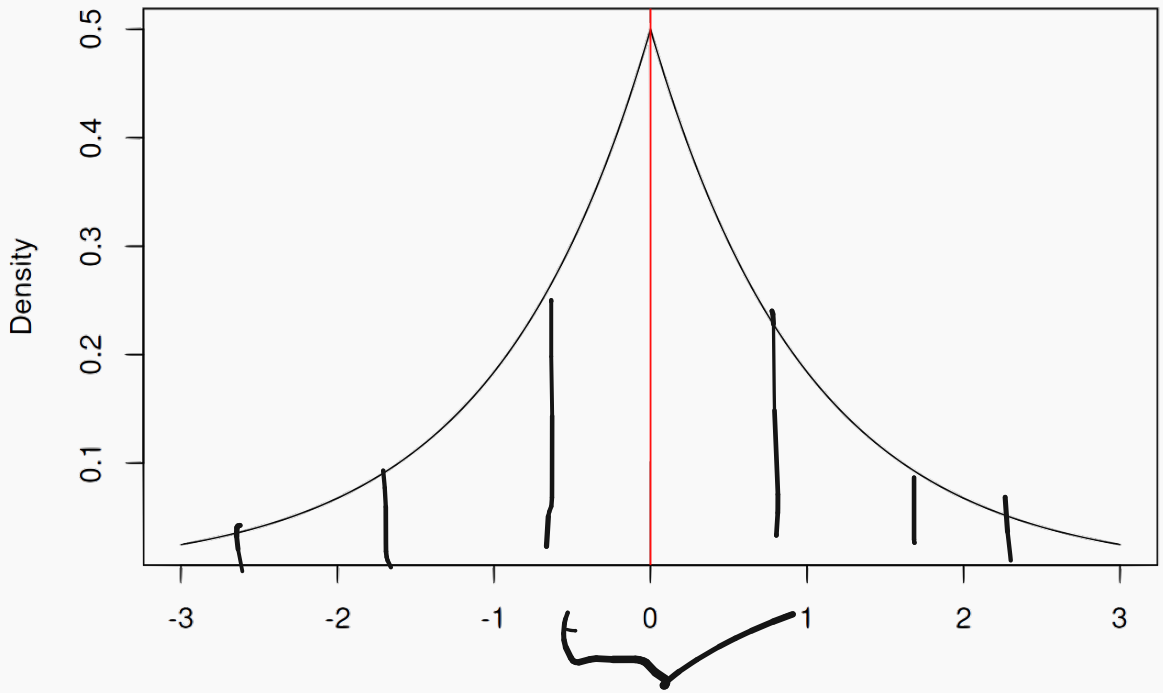
what do we pay in terms of accuracy?

$Lap(b)$ has standard deviation $\sqrt{2}b$. Like Gaussian distribution, Laplace random variables usually fall within a few standard deviations of the mean:



what do we pay in terms of accuracy?

$Lap(b)$ has standard deviation $\sqrt{2}b$. Like Gaussian distribution, Laplace random variables usually fall within a few standard deviations of the mean:



what do we pay in terms of accuracy?

Standard deviation $= \sqrt{2} \cdot \frac{\Delta_Q}{\epsilon}$.

For $x_1, \ldots, x_n \in [0,1]$, $Q(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} x_i$, we have that:

$$\Delta_Q = \frac{1}{n}.$$

Overall error from adding noise:

$$O\left(\frac{1}{\epsilon n}\right)$$

Very reasonable if n is large!

E.g., if $n = 10,000$ can get error roughly .001 on mean estimate with privacy parameter $\epsilon = .1$.

what about more complex functions?

In machine learning applications, Q is an entire training procedure, and the output is vector of parameters.

$$Q(\mathbf{X}, \mathbf{y}) \to \boldsymbol{\beta} \in \mathbb{R}^d.$$

Challenges:

- Very hard to estimate the sensitivity to figure out how much noise should be added.
- If some parameters are more sensitive to noise, we could change the models output drastically.

differentially private (stochastic) gradient descent)

Main idea: Typically $Q(\mathbf{X}, \mathbf{y})$ is computed by running gradient descent on a loss function $L(\beta)$. Instead of directly adding noise to $Q(\mathbf{X}, \mathbf{y})$, add noise at each step of gradient descent.

Basic Gradient descent algorithm:

- Choose starting point $\beta^{(0)}$.
- For $i = 0, \ldots, T$:
  - $\beta^{(i+1)} = \beta^{(i)} - \eta \nabla L(\beta^{(i)})$
- Return $\beta^{(T)}$.

differentially private (stochastic) gradient descent)

Typical loss function in machine learning have finite sum structure.

$$L(\beta) = \sum_{j=1}^{n} \ell(\beta, \mathbf{x}_j, y_j)$$

By linearity:

$$\nabla L(\beta) = \sum_{j=1}^{n} \nabla \ell(\beta, \mathbf{x}_j, y_j)$$

Looks just like our mean estimation problem! Can bound the contribution of each data example $(\mathbf{x}_j, y_j)$ to the gradient to get a sensitivity, then add noise.

differentially private (stochastic) gradient descent)

Due to a 2016 paper by Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, Li Zhang.

DP-SGD:

- Choose starting point $\beta^{(0)}$.
- For $i = 0, \ldots, T$:
  - $\beta^{(i+1)} = \beta^{(i)} - \eta(\nabla L(\beta^{(i)}) + \mathbf{r}_i)$
- Return $\beta^{(T)}$.

Above each $\mathbf{r}_i$ is a random Gaussian vector.

Leading way to incorperate privacy into training machine learning models. Implented natively, e.g., in TensorFlow.