



Privacy Preserving Machine Learning

Machine Learning Theory (MLT)

Edinburgh

Rik Sarkar

Coursework

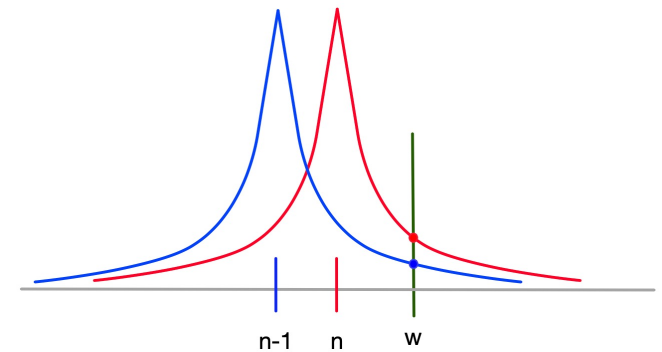
- Keep an eye on piazza
- Note that submission deadline is 12 march at 12:00.
 - Submit early. Don't keep till last minute!

Today: Differential privacy

- Sensitivity, utility
- Local DP
- Approximate DP
- Gaussian mechanism
- Differentially private machine learning
- Differentially private stochastic gradient descent

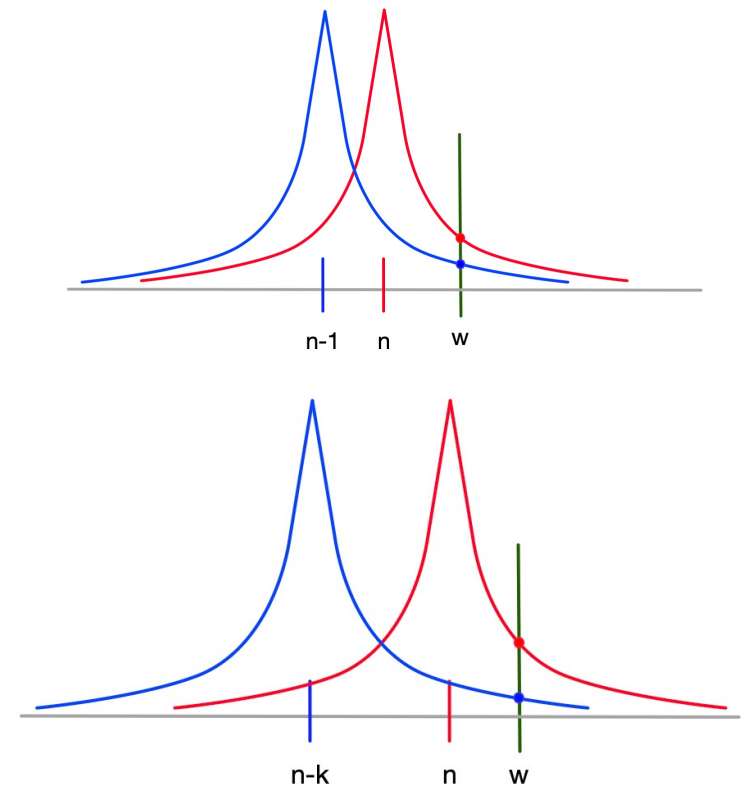
Recap: DP example: Make output probabilities similar for n and $n-1$

- Problem: count the number of people in the meeting
- Solution:
 - Find real count n
 - Publish $n + y$ where y is noise (a random number)
 - Draw y from Laplace distribution
 - $\Pr[y] = \frac{1}{2b} e^{-\frac{|y-\mu|}{b}}$
 - μ is the mean, $2b^2$ is variance
- We will write $Lap(b)$ to mean Laplace distribution with mean 0 and variance $2b^2$

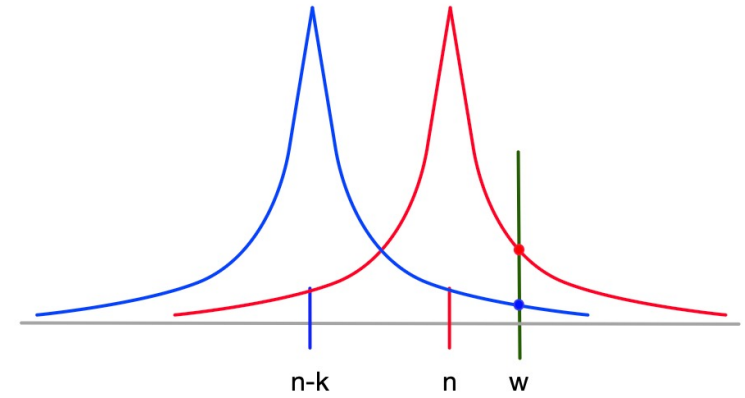


What about privacy other than one person?

- What about a group of k ?
- Suppose there is a group of k people who may not have attended.
 - Neighboring databases differ in count of k
 - A more general notion of neighboring databases
- And we want to hide this from the observer/adversary
 - Should have similar probabilities for n and $n-k$
- Exercise : Show that an ϵ -DP algorithm is $k\epsilon$ -DP for any group of size k



- Any published w can occur in two ways:
- $n + y$
- $(n - k) + (y + k)$
- The ratio $\frac{\Pr[\text{noise}=y]}{\Pr[\text{noise}=y+k]} = ?$

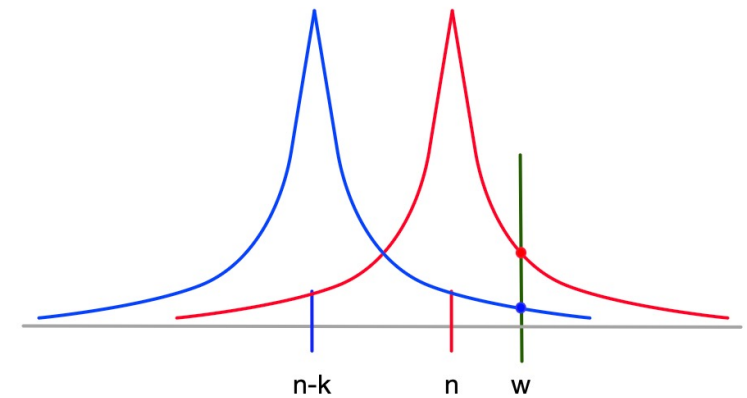


- Any published w can occur in two ways:

- $n + y$

- $(n - k) + (y + k)$

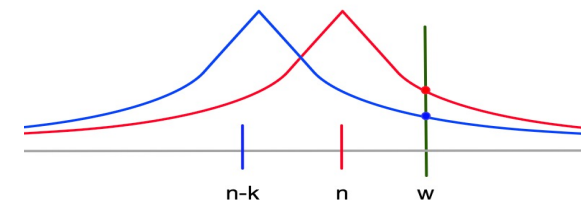
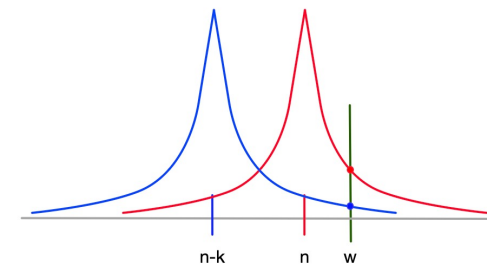
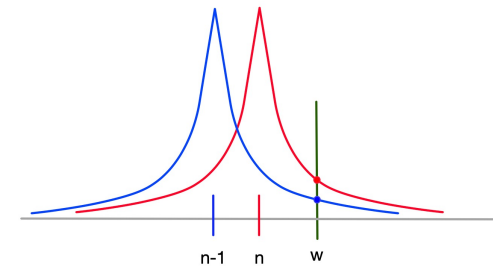
- The ratio $\frac{\Pr[\text{noise}=y]}{\Pr[\text{noise}=y+k]} = \frac{e^{-|y|\epsilon}}{e^{-|y+k|\epsilon}} \leq e^{k\epsilon}$



- Remember that $k\epsilon$ is worse privacy than ϵ
 - Larger groups are harder to privatise!
 - How can we get ϵ -DP for k people?

ϵ -DP for a group of k -people

- Add a noise of $Lap\left(\frac{k}{\epsilon}\right)$
- The larger variance makes the red and blue distributions more similar
 - Similar output probabilities
- Attains ϵ -DP
- Observe that it works to hide *any* group of k people not attending



How much noise do we need?

- The *variance* of the noise depends on how much f can change due to a unit change between neighboring databases
- If $f(D)$ and $f(D')$ are every close, we need only a little noise to make them indistinguishable
 - If they are very different, we need a large noise
- *Sensitivity* of f is written as Δf :
 - $\Delta f = \max_{D, D'} |f(D) - f(D')|$
 - The maximum possible difference in f between neighboring datasets.

Laplace mechanism

- Given a database D and a function f ,
- Output $A(D) = f(D) + y$, where $y \sim \text{Lap}\left(\frac{\Delta f}{\epsilon}\right)$
- This ensures ϵ differential privacy.

Privacy and utility

- Adding noise gains privacy
- Adding more noise (from a distribution of higher variance) gains more privacy
- But more noise decreases accuracy of the eventual use of the value, so it is less useful
 - We say it has less *utility*
- The challenge is to balance utility and privacy
 - By adding the minimum necessary noise
- If sensitivity is larger, noise is larger. Utility is less.

Examples

- When f is the count of number of people
 - $\Delta f = 1$
- Suppose the max height of a person is assumed to be 7ft.
 - If f is the sum of heights of a group in feet, then sensitivity is 7.
- Suppose the max salary in a company is capped at £50K
 - We are analysing total salaries in groups of size 4. That is, a group is either present or absent. Then the sensitivity is £200K.

Another example

- Suppose $f = \frac{1}{n} \sum X_i$, where $X_i \in [0,1]$
- Sensitivity is $\frac{1}{n}$ (if we replace one number by another)
 - Note that we are using the replacement version for convenience.
- Thus, the noise required is $y \sim \text{Lap}\left(\frac{1}{\epsilon n}\right)$

Utility of Laplace mechanism

- We add noise $y \sim \text{Lap}\left(\frac{\Delta f}{\epsilon}\right)$
- But noise hurts the utility.
 - How much does it affect the utility?
- We can show that the error from Laplace mechanism is limited
- Laplace method has an error bound (utility guarantee):
 - Expected error $\mathbb{E}[|A(D) - f(D)|] = \frac{\Delta f}{\epsilon}$
 - So the error induced is determined by sensitivity, but is not much larger

Local differential privacy

- Suppose X_1, X_2, \dots represents how many times person i used their phone in a day
- And we want to achieve privacy at the level of a single use. That is, any particular use of the phone should not be detectable from the data
- Suppose they are recorded by mobile phones of person $1, 2, \dots$ and transmitted to a server
- The problem: Each person wants their data to be private
 - E.g. when they do not trust the server collecting and computing
- Solution:
 - Add noise *before* transmitting the variables
 - I.e. Each phone i Transmits $X_i + y_i$, where $y_i \sim \text{Lap}\left(\frac{1}{\epsilon}\right)$
 - Either the server, or anyone snooping the line cannot be sure of the data transmitted
 - Data is protected by DP right from the source.
 - Disadvantage: A lot more noise than adding noise once

Approximate differential privacy

- Algorithm A is (ϵ, δ) -differentially private if :
 - For every neighbouring D, D'
 - $\Pr[A(D) \in S] \leq e^\epsilon \cdot \Pr[A(D') \in S] + \delta$
- In theory, δ must be very small to be useful, like $o\left(\frac{1}{n}\right)$.
 - In practice, it is sometimes a bit larger.
- When $\delta = 0$, we have pure DP (the version we saw already)

Gaussian mechanism

- Gaussian distribution: $\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{(y-\mu)^2}{2\sigma^2}\right)}$
- Compute $f(D)$ and Δf
- Draw noise from Gaussian distribution $y \sim \mathcal{N}(0, \sigma^2)$
 - Where $\sigma = \frac{\left(\sqrt{2 \ln\left(\frac{1.25}{\delta}\right)} \cdot \Delta f\right)}{\epsilon}$
- Output $f(D) + y$
- Theorem: Gaussian mechanism satisfies (ϵ, δ) -DP

Composition

- Suppose we run algorithms A_1, A_2, \dots
 - Where A_i is (ϵ_i, δ_i) -DP
- E.g. we perform different computations on the same data and publish the results
- Then the end result $(A_1(D), A_2(D), \dots A_n(D))$ is
- (ϵ, δ) - DP for
 - $\epsilon = \sum \epsilon_i,$
 - $\delta = \sum \delta_i$
- Thus, ϵ, δ increase and we lose privacy as more computations are released

Repeated computation

- The same idea holds if we carry out the same operation multiple times
- E.g. if A_1, A_2, \dots are all the same computation and each adds a noise, then, while each may be (ϵ, δ) DP, the overall privacy loss can be large for many repetitions

With k repetitions of the same query

- How can we get ϵ -DP?

Resilience to post processing

- Differential privacy holds for further use of the output
- E.g. if a model or output m is obtained with an ϵ -DP algorithm, then we can use m as many times as we wish without any additional loss of privacy
 - We are not using the raw database D any more, and using m again does not lose more privacy
 - If a computation uses D again, then it loses more privacy.

Notes on differential privacy

- DP is a property of the randomized algorithm
- The algorithm depends on the assumptions and computation objective
 - Not on the dataset
 - Noise depends on sensitivity (from what is known/assumed about the data)
 - Note that sensitivity is not *computed* from the data, as this itself will leak privacy
 - It is assumed that whatever is needed to compute sensitivity is known to everyone
 - Noise is computed for the objective, for *all possible inputs*
 - Noise is not computed for specific input points
- DP is determined based on the assumptions about the data, but not on the actual data.
 - If your DP mechanism depends on anything computed from the data, be very careful!

Algorithms with multi-dimensional outputs

- Till now, we have considered algorithms with real valued outputs
- Now, consider algorithms with k -dimensional outputs
- E.g. ML models with k parameters
- Similar to applying k different algorithms

- Suppose $f: \mathcal{Z}^n \rightarrow \mathbb{R}^k$
 - Produces a k -dim output
- On changing the input D , all k different parameters may change
- How do we measure sensitivity?

L_p Norms

- L_p norms are defined as:
 - $|x|_p = (x_1^p + x_2^p + \dots + x_k^p)^{1/p}$
- Most relevant for us:
 - $|x - y|_1 = |x_1 - y_1| + |x_2 - y_2| + \dots + |x_k - y_k|$
 - $|x - y|_2 = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_k - y_k)^2}$

L_1 and L_2 Sensitivity

- $f: \mathcal{Z}^n \rightarrow \mathbb{R}^k$
 - Produces a k -dim output
- Sensitivity measured as the norms:
- $\Delta_1 f = \max_{D, D'} |f(D) - f(D')|_1$
- $\Delta_2 f = \max_{D, D'} |f(D) - f(D')|_2$

k -dim Laplace mechanism

- Compute $\Delta_1 f$
- For $i = 1, \dots, k$, draw $y_i \sim \text{Lap}\left(\frac{\Delta_1 f}{\epsilon}\right)$
- Output $f(D) + Y$, where $Y = (y_1, y_2, \dots, y_k)$

k -dim Gaussian mechanism

- Compute $\Delta_2 f$
- For $i = 1, \dots, k$, draw $y_i \sim \mathcal{N}(0, \sigma^2)$
 - Where $\sigma = \frac{\left(\sqrt{2 \ln\left(\frac{1.25}{\delta}\right)} \cdot \Delta_2 f \right)}{\epsilon}$
- Output $f(D) + Y$, where $Y = (y_1, y_2, \dots, y_k)$

Which is better Laplace or gaussian?

- Consider L1 and L2 norms for k dimensions
- Assume $x - y = [1, 1, 1, 1, 1, \dots]$
- $|x - y|_1 = k$
- $|x - y|_2 = \sqrt{k}$
- Thus, in high dimensions, L2 sensitivity is smaller.
 - Gaussian mechanism can use L2 sensitivity
- Since Sensitivity determines standard deviation of distribution, gaussian mechanism has smaller standard deviation (or variance).
- But remember that gaussian mechanism also has the additive δ

Perturbation mechanisms

- Input perturbation:
 - Modify the data (add noise to each point) before any computation is applied
 - Similar to local DP
- Output perturbation
 - Compute output, then add the noise
 - E.g. add noise to the aggregate, or to the ML model parameters
- Gradient or internal perturbation
 - Add noise to the operation of the algorithm
 - E.g. to the gradients of SGD

DP for machine learning

- General idea:
- A training algorithm produces a model with n parameters in a vector \mathbf{w} .
- We need to get another vector Y with n parameters of noise
- And add to \mathbf{w}
- To do that, we need to know the sensitivity of \mathbf{w}
 - Then we can draw noise according to sensitivity and add

Differentially Private Empirical Risk Minimisation

- Regularized ERM:
 - $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}} (L_S(\mathbf{w}) + R(\mathbf{w}))$: Optimal regularized model
 - $R(\mathbf{w})$ is a regularizer
 - Common choice: $R(\mathbf{w}) = \lambda ||\mathbf{w}||^2$ is 2λ -strongly convex
 - Makes $(L_S(\mathbf{w}) + R(\mathbf{w}))$ strongly convex
- Theorem: L_2 sensitivity for ERM:
 - If loss function L is λ -strongly convex
 - If L is convex, differentiable, G -Lipschitz
 - Then the L_2 sensitivity is at most $\frac{4G}{n\lambda}$

DP-ERM using output perturbation

- Optimal solution: $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}}(L_S(\mathbf{w}) + R(\mathbf{w}))$
- For $i = 1, \dots, k$, draw $y_i \sim \mathcal{N}(0, \sigma^2)$
 - Where $\sigma = \frac{\left(4G \sqrt{2 \ln\left(\frac{1.25}{\delta}\right)}\right)}{n\lambda\epsilon}$
- $\hat{\mathbf{w}} + Y$, where $Y = (y_1, y_2, \dots, y_k)$
- Apply gaussian mechanism with the right sensitivity
- Guarantees (ϵ, δ) -DP

Non-Convex optimization

- The problem: We don't know where the minimum could be, and how much things could be different with small changes in data
- We need a different method suitable for SGD
- Idea:
 - In every step (with a single batch) limit the gradient
 - This limits the influence (sensitivity) of any one point
 - Then add noise to the gradient

DP-SGD

- At every step:
 - Sample a point or batch z
 - Compute Gradient $g_t = \nabla f_{\mathbf{w}_t}(z)$ (privacy risk: gradient is a function of z)
 - clip g_t to length at most C : $\widehat{g}_t \leftarrow g_t \cdot \frac{1}{\max(1, \frac{\|g_t\|}{C})}$
 - Add noise $\widehat{g}_t = \widehat{g}_t + r$ where $r \sim \mathcal{N}(0, \sigma^2 C^2)$ (in suitable dimensions..)
 - Update model $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \widehat{g}_t$
- At end, return final model

Differential privacy of DP-SGD

- The impact of data point(s) used in any one step is limited to a vector of length C

- For $\sigma = \frac{\sqrt{2 \ln \frac{1.25}{\delta}}}{\epsilon}$, every step is (ϵ, δ) -DP

- Overall, for T steps with batches that are q fraction of data, we get $(qT\epsilon, qT\delta)$ -DP
- More tighter bounds available, e.g. $(q\epsilon\sqrt{T}, \delta)$
- Proofs omitted

Amplification by sampling

- If A is an (ϵ, δ) -DP algorithm for D
- If $S(D)$ returns a sample of D where each element is present with probability p
- Then $A(S(D))$ is $(\epsilon', p\delta)$ -DP, where
 - $\epsilon' = \ln(1 + p(e^\epsilon - 1))$
 - Note that $\ln(1 + p(e^\epsilon - 1)) \leq 2p\epsilon$
- Sampling improves the privacy of a private algorithm (e.g. in randomly selected batches)

DP machine learning

- Current research area, with connections to all aspects of ML
 - Complexity, optimization, stability, dimension reduction....
- For any datascience/ML algorithm it is possible to ask: What is the DP version of this algorithm? Can we get a good enough privacy – utility tradeoff?
- Current situation with DP-SGD and differentially private training
 - Works well on smaller models
 - On large neural networks, the extra noise with many parameters makes them inaccurate
 - Ongoing research in making differential privacy more useful
 - Typical approach: argue that overall small levels of noise added carefully actually suffices to get required levels of privacy

Non-numerical queries

- Our basic output perturbation mechanism: $A(D) = f(D) + Y$
 - Works only with numeric values
- How can we privatise non-numeric queries?
 - What is the most popular movie?
 - Which date is likely to work best to schedule a meeting?
 - What are other examples of non-numeric or selection queries?
 - How do we privatize them?

Non-numerical queries

- Suppose $f: \mathcal{Z}^n \rightarrow \mathcal{O}$ (for dataset $D \in \mathcal{Z}^n$)
 - \mathcal{O} is a set of discrete possibilities
 - E.g. set of movies, set of computers, set of web sites etc
- While \mathcal{O} is discrete, we can imagine a numerical score or utility to elements of \mathcal{O}
 - E.g. the popularity of a movie, the number of people that can attend the meeting on a date etc
 - $s: \mathcal{Z}^n \times \mathcal{O} \rightarrow \mathbb{R}$
 - $s(D, o)$ represents the score of o as an approximation of $f(D)$

Sensitivity of s

- For neighbouring D, D'
- $\Delta s = \max_{o \in \mathcal{O}} \max_{D, D'} |s(D, o) - s(D', o)|$
- Worst case difference in output between neighboring datasets

Exponential mechanism

- Task is to find $f(D)$
- Compute s and Δs
- Output o with probability

- $\Pr[o] = \frac{e^{\frac{s(D,o)\epsilon}{\Delta s}}}{\sum_{o' \in \mathcal{O}} e^{\frac{s(D,o')\epsilon}{\Delta s}}}$

- Sample o with probability proportional as above
 - Denominator is just normalisation
- Probability drops off exponentially with decreasing score
- Compare with Laplace mechanism

Theorem

- Exponential mechanism satisfies ϵ -DP

