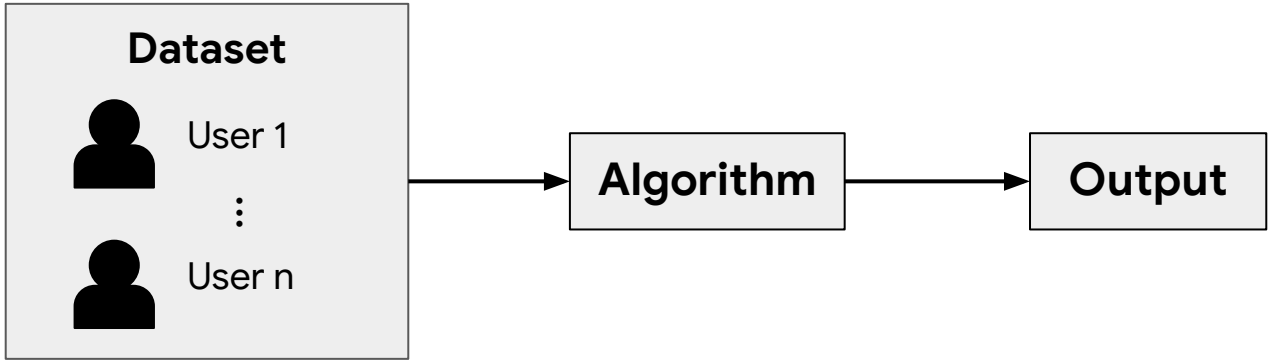
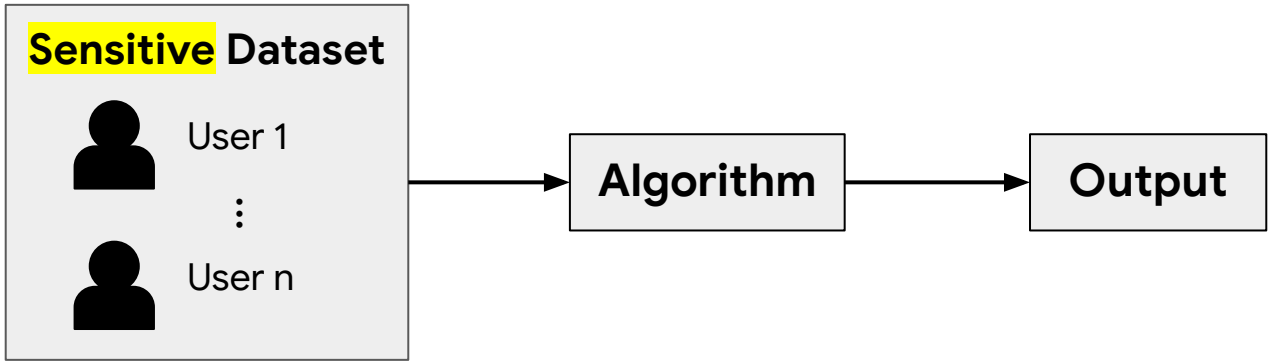
Privacy-preserving Machine Learning

Pasin Manurangsi
Google Research
Thailand

What this tutorial is about...



What this tutorial is about...



- Face images ⇒ ML model
- Database ⇒ Aggregated Statistics
- Medical datasets ⇒ Anonymized Dataset

Is it privacy-safe to share the output?

What this tutorial is *not* about...

Security

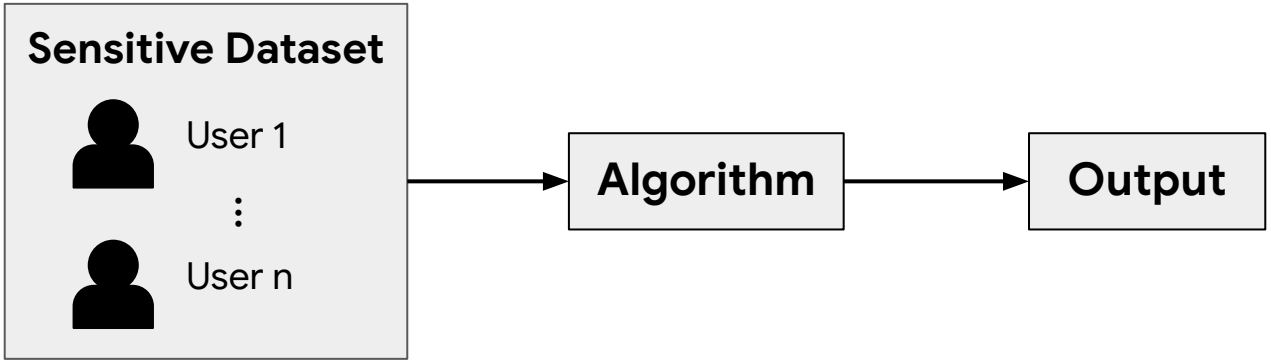
- Cryptography Primitives & Attacks
  - Encryptions
  - Secure Multiparty Computation
  - ...

Legislations

- Information flow & rights to data privacy:
  - PDPA
  - GDPR
  - DMA
  - ...

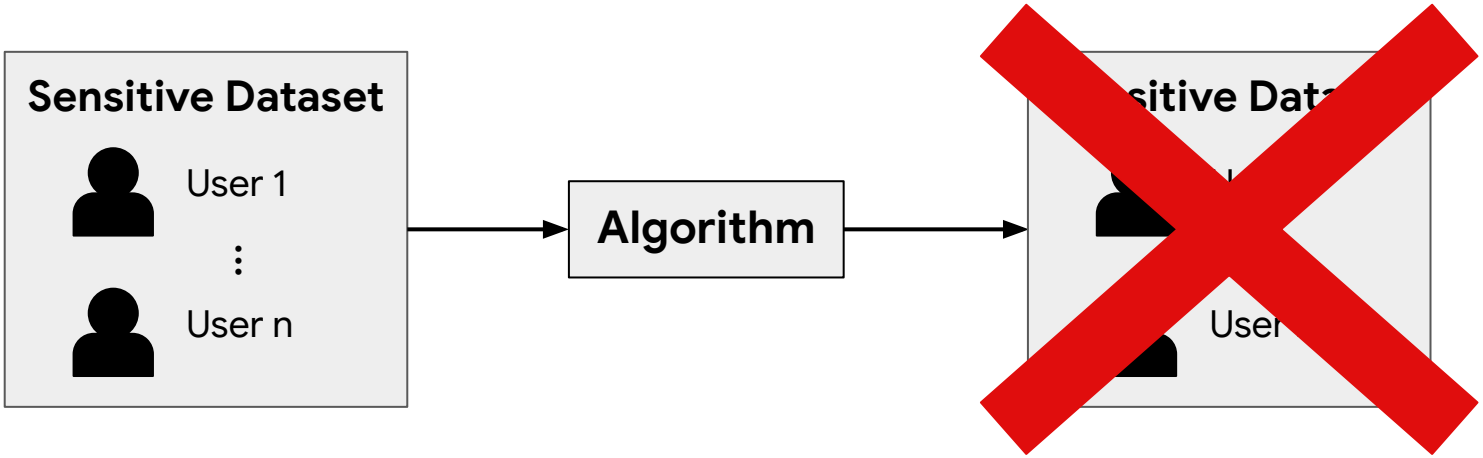
- Company's trade secret

What this tutorial is about...



Is the algorithm *not* privacy-preserving?

What this tutorial is about...



Is the algorithm *not* privacy-preserving?

Tutorial Outline

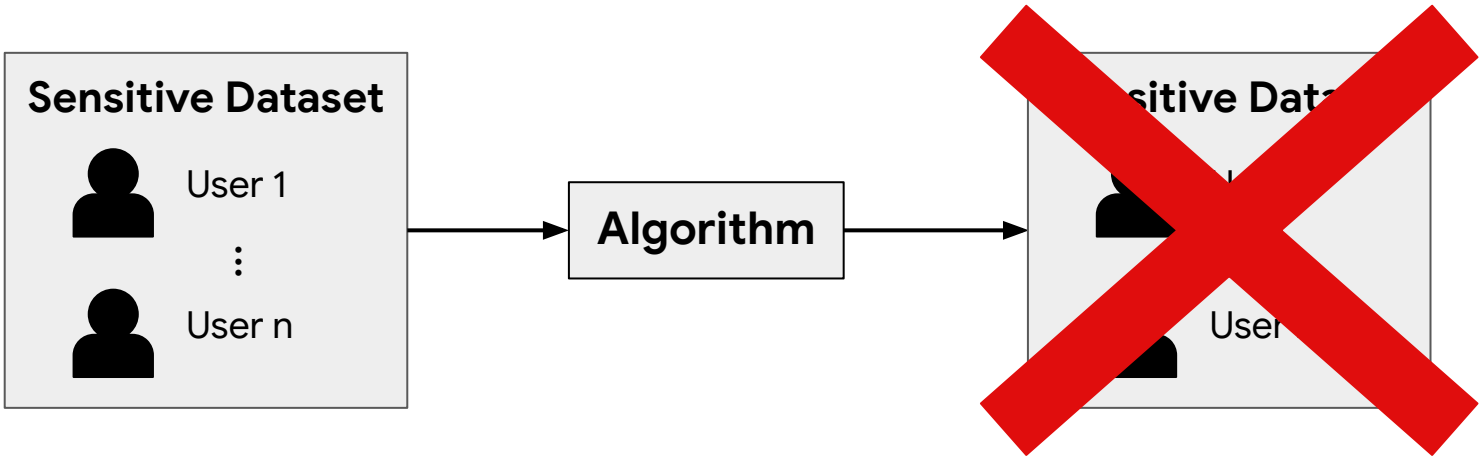
Part I

- Example of Privacy Attacks
  - Reconstruction
  - Membership Inference
  - Linkage
  - Model / Gradient Inversion
- What is privacy leakage?

Part II

- Differential Privacy (DP)
  - Definition
  - Basic Noise Addition Algorithms
  - Properties of DP
  - DP Machine Learning

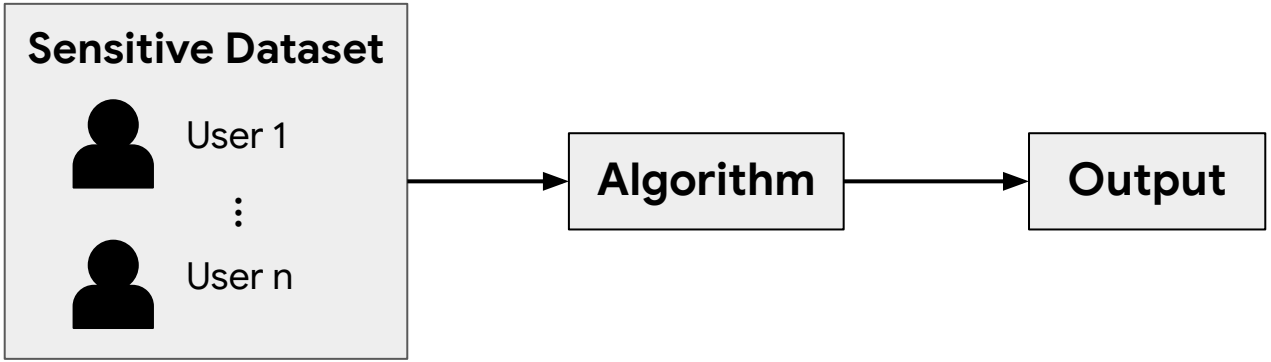
Privacy Attacks



not

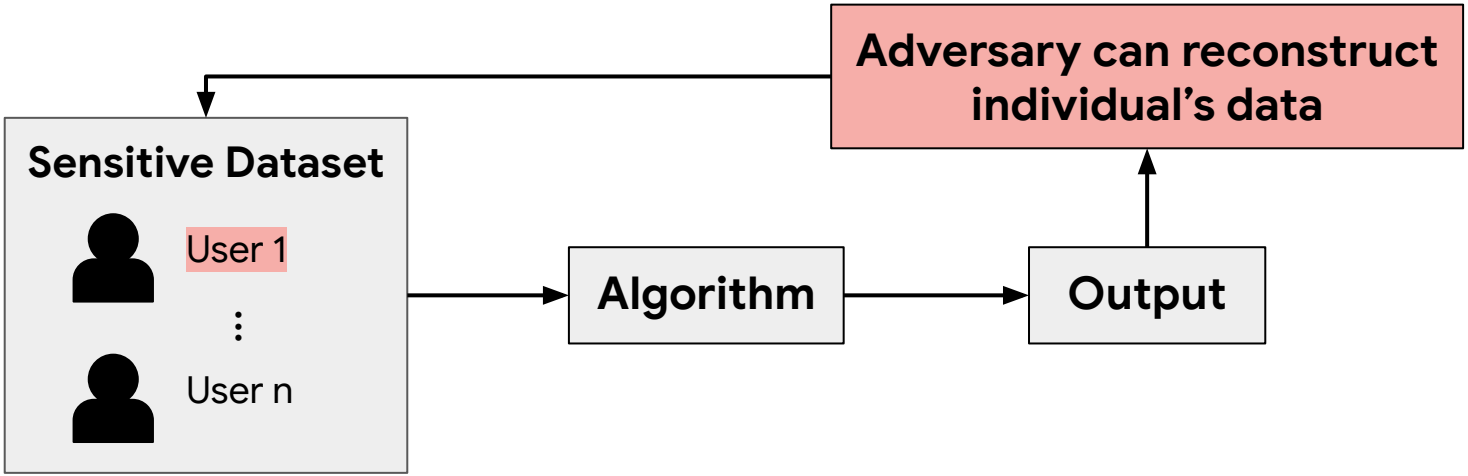
Is the algorithm privacy-preserving?

Reconstruction Attacks



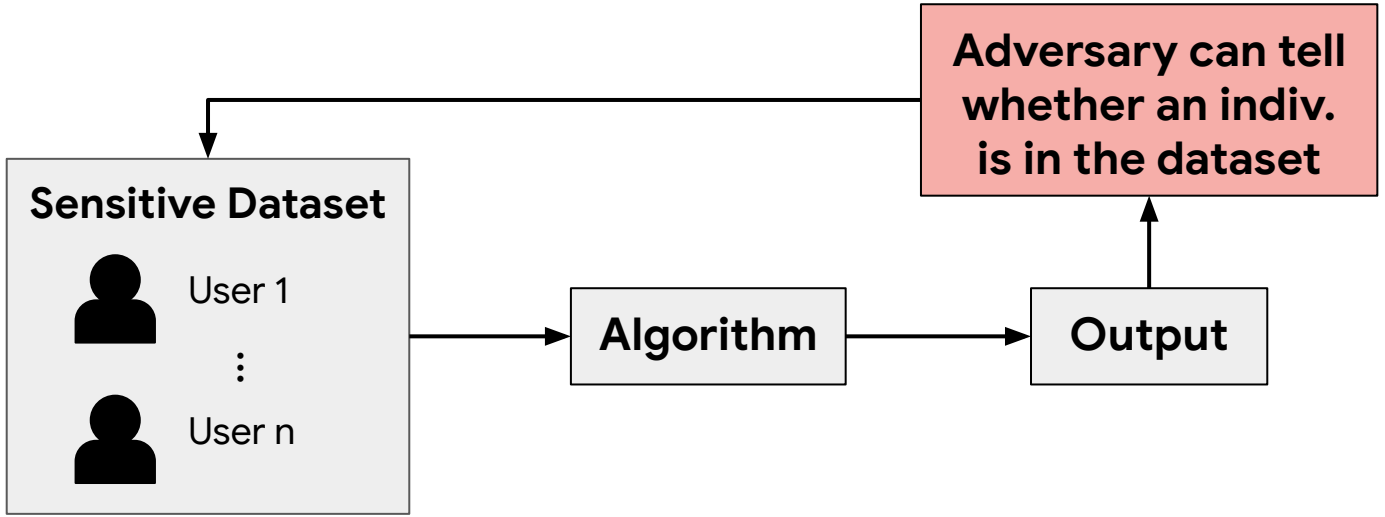
Is the algorithm *not* privacy-preserving?

Reconstruction Attacks



Is the algorithm *not* privacy-preserving?

Membership Inference Attacks



Can be harmful

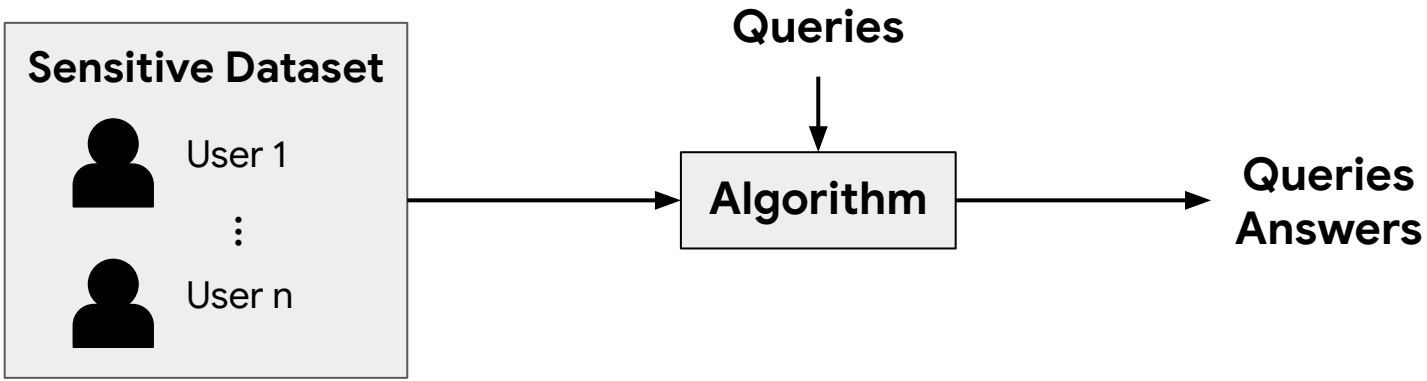
E.g. dataset contains only those with a certain disease

not

Is the algorithm privacy-preserving?

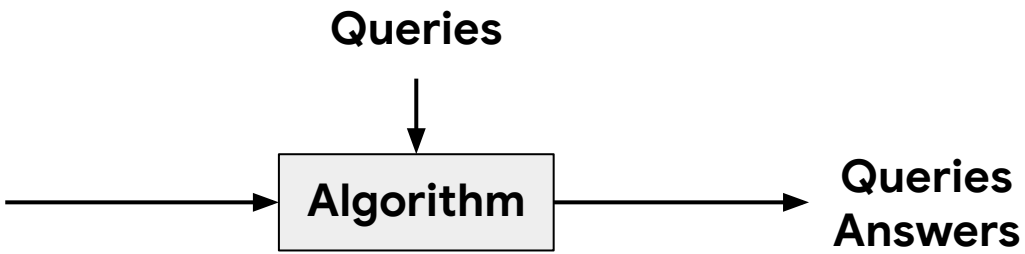
Attacks Against Query-Answering System

Query-Answering System



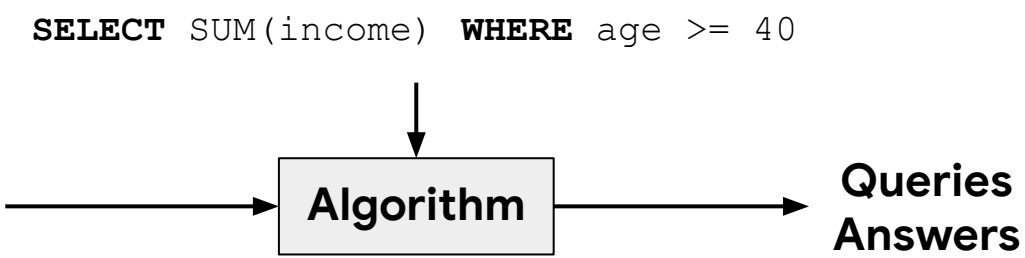
Query-Answering System

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



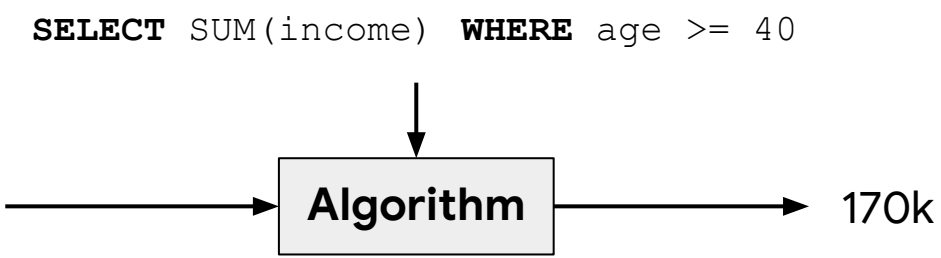
Query-Answering System

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



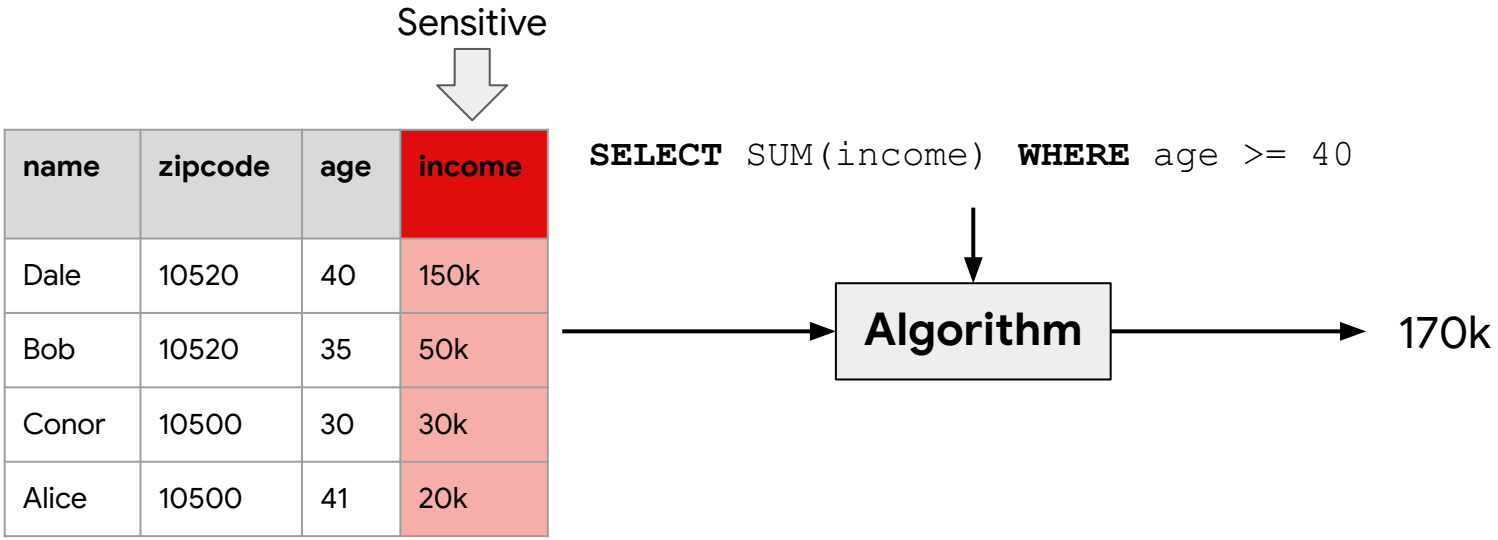
Reconstruction Attacks

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



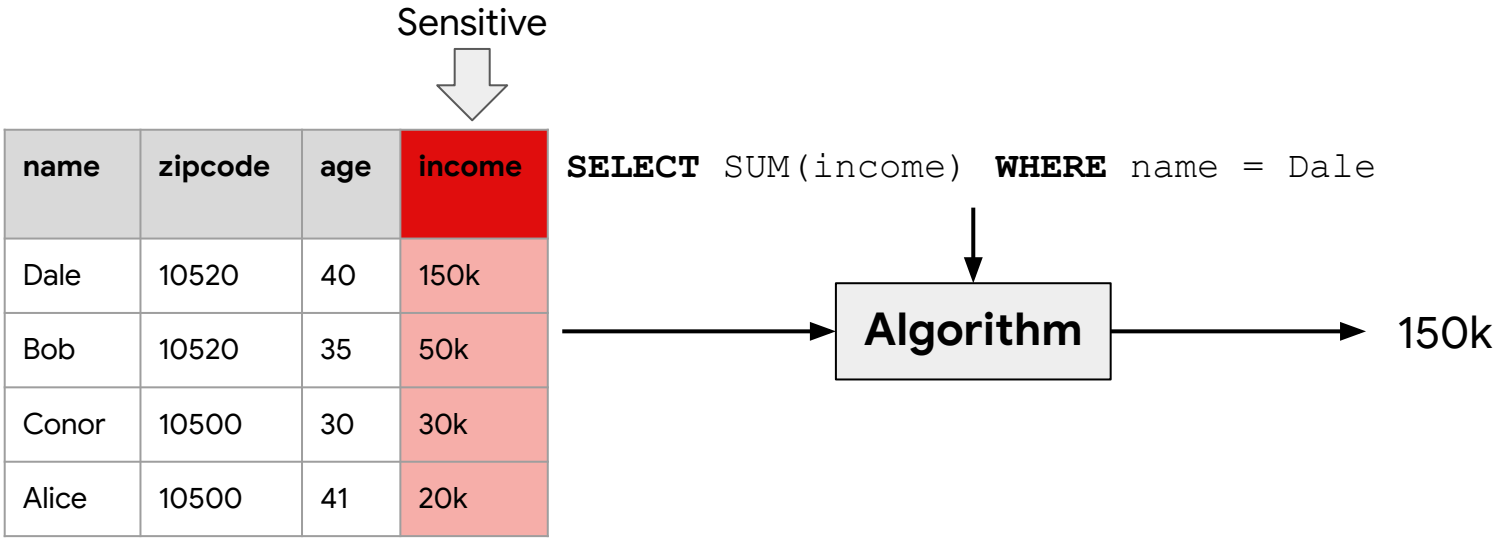
- Simplifying assumptions: only allow **SUM** (or **COUNT**) queries
- Only one sensitive column

Reconstruction Attacks

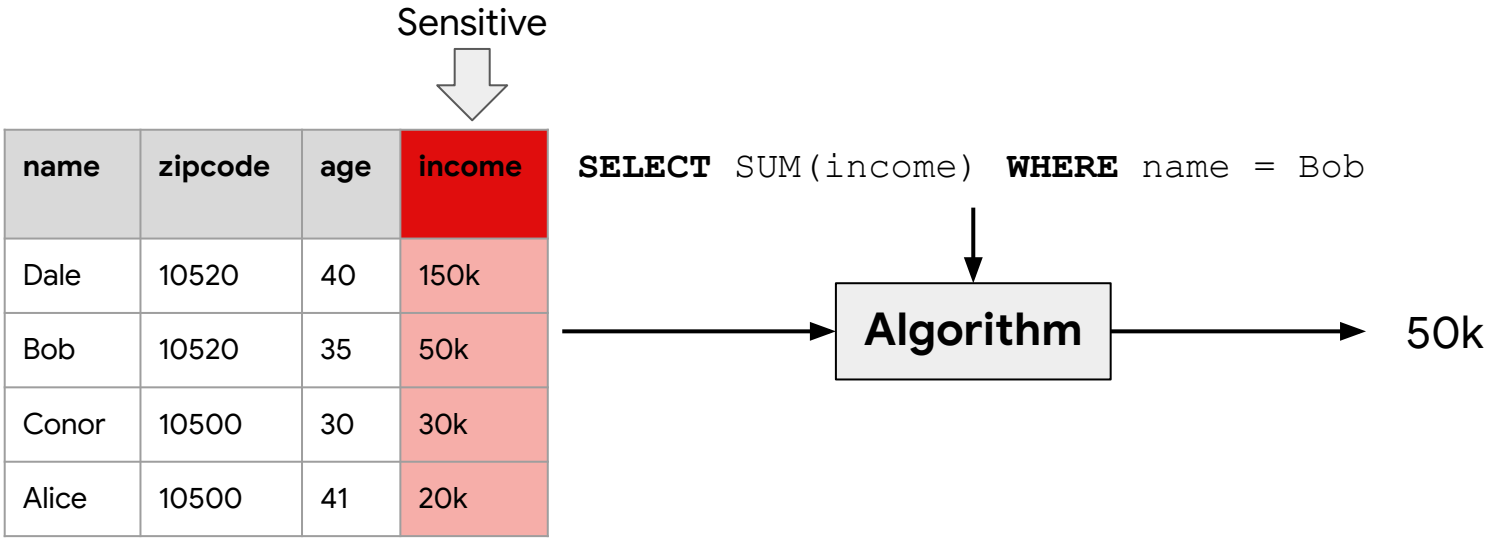


- Simplifying assumptions: only allow **SUM** (or **COUNT**) queries
- Only one sensitive column

Setting I: Adversary Can Select Queries

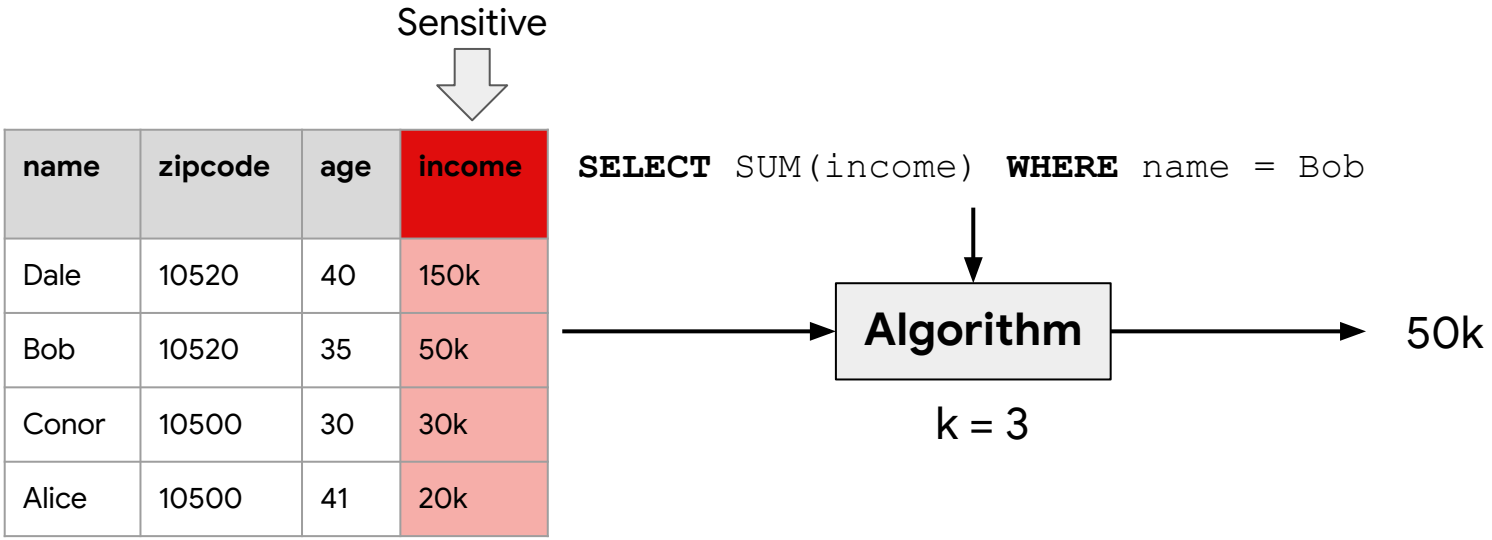


Setting I: Adversary Can Select Queries



The adversary can construct the entire dataset!

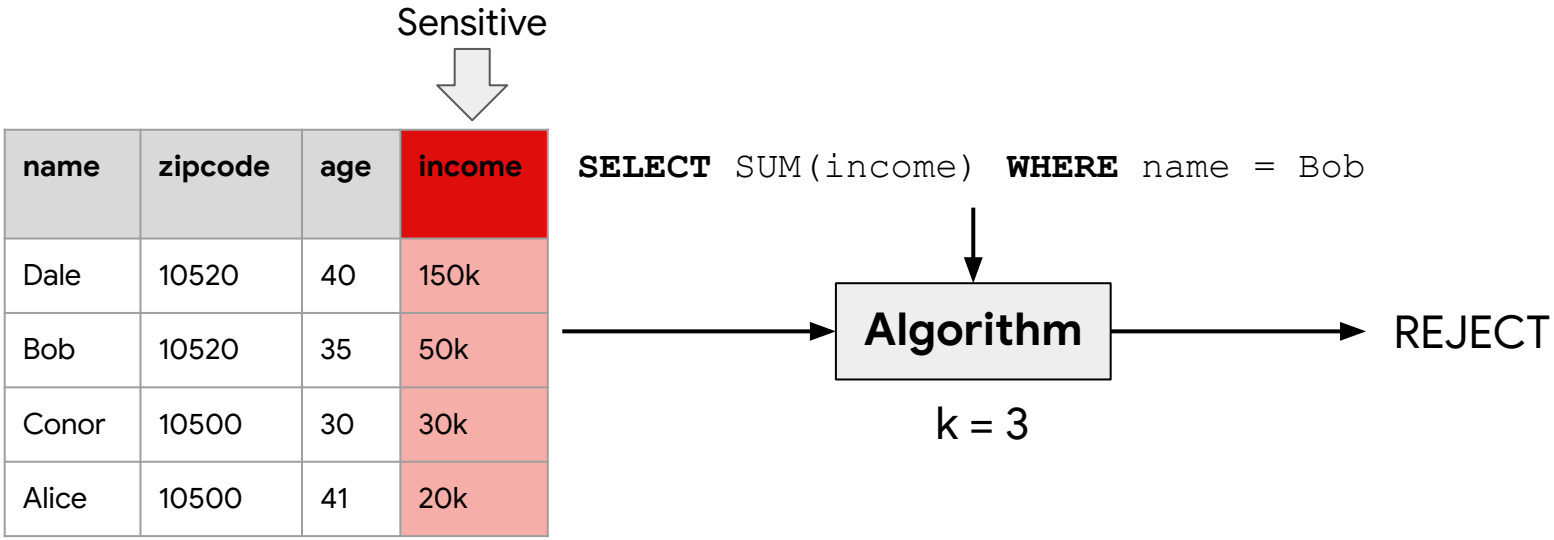
Mitigation I: “k-Anonymity”



k-Anonymity

Algorithm rejects if query involves < k individuals

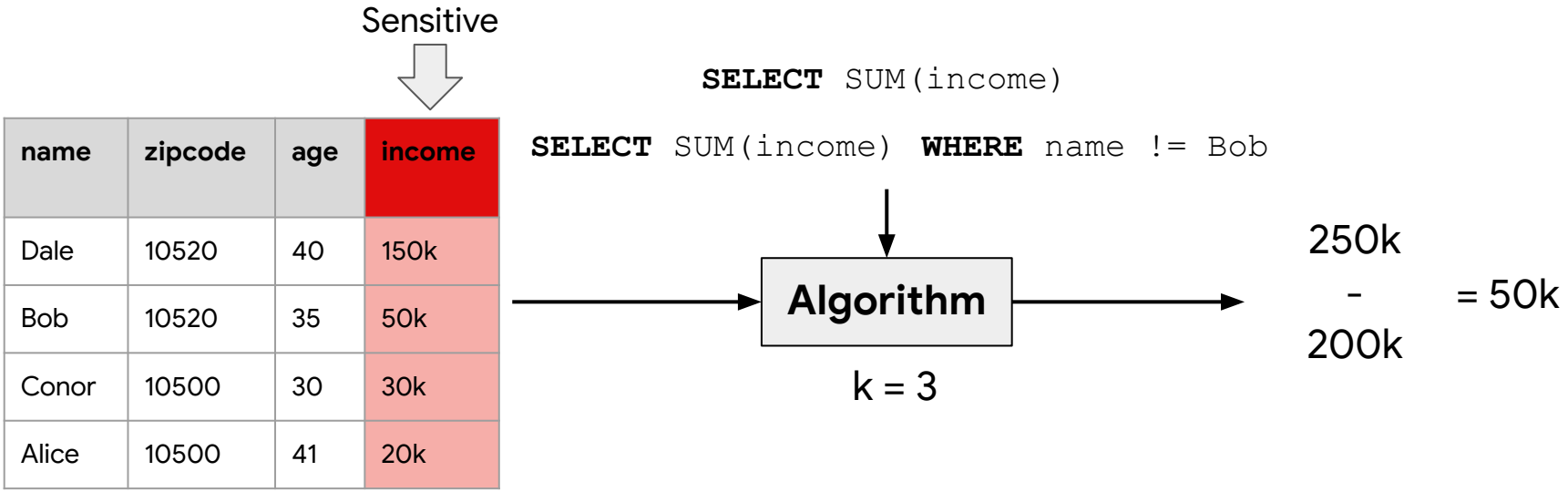
Mitigation I: “k-Anonymity”



k-Anonymity

Algorithm rejects if query involves < k individuals

Differentiating Attacks

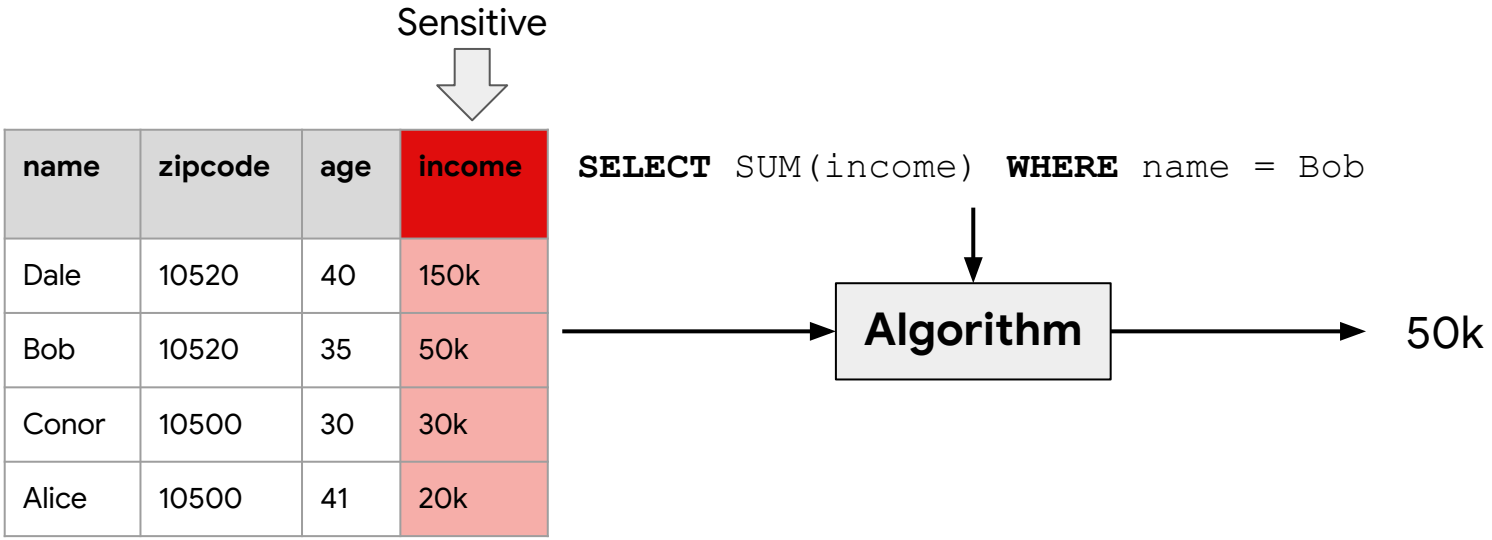


k-Anonymity

Algorithm rejects if query involves < k individuals

Hard to extend if adversary can combine >1 queries

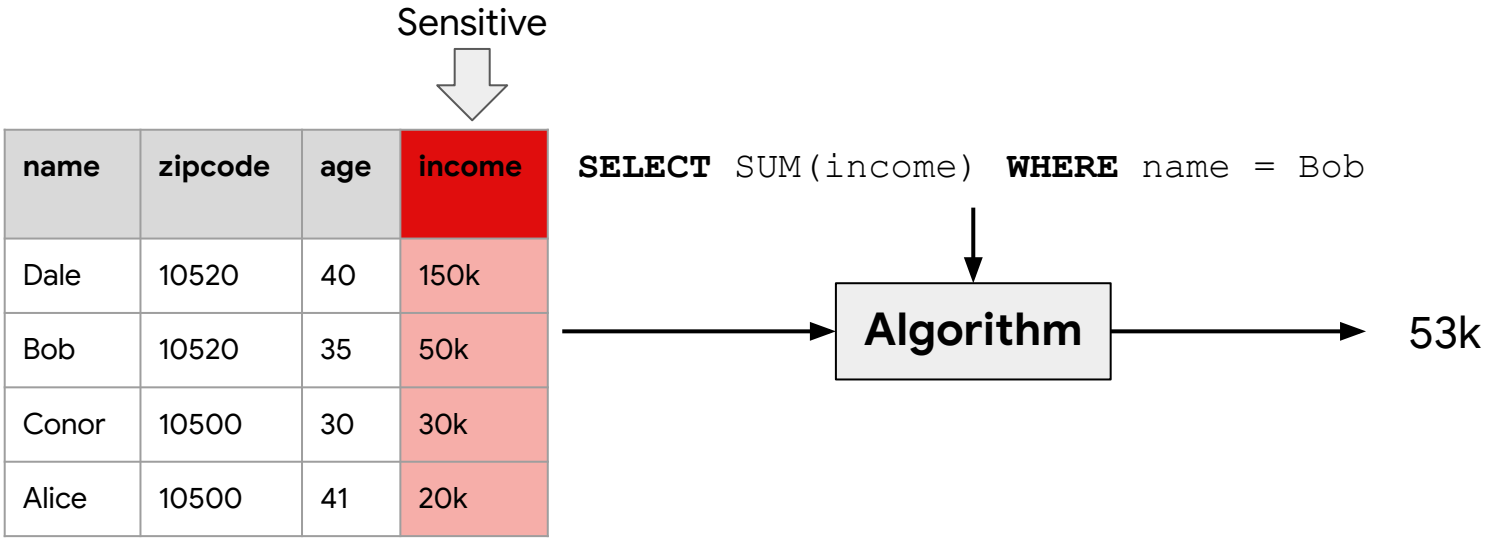
Mitigation II: Add Noise



Noise Addition

Add random noise to the answer

Mitigation II: Add Noise



Noise Addition
Add random noise to the answer

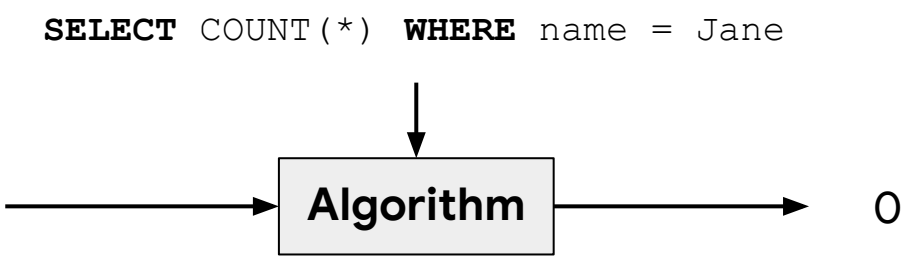
An approach taken by *differential privacy**

*To be discussed in the second part of the tutorial

Membership Inference Attacks

Is Jane in the table?

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



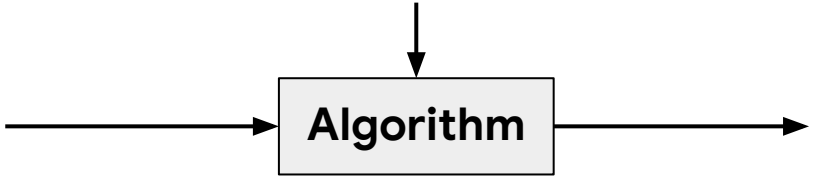
- Membership inference attacks work in a similar way
- Can also circumvent k-anonymity with differentiating attacks
- Noise addition can also provide (differential) privacy

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



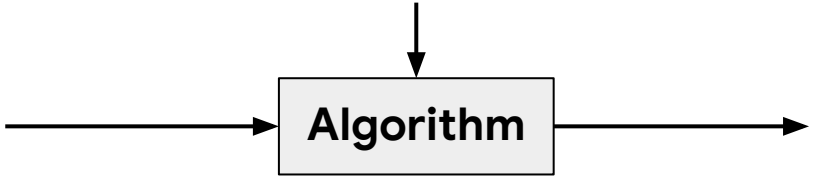
170k
80k
50k
230k

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



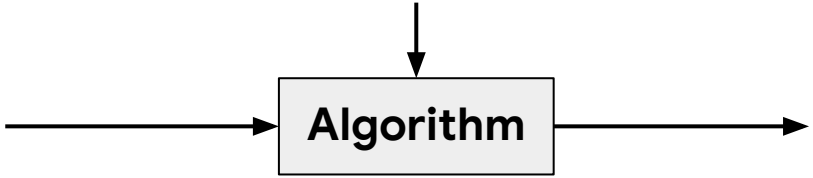
170k
80k
50k
230k

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



170k
80k
50k
230k

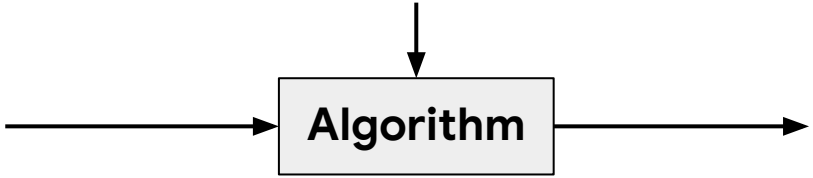
$x_1 + x_4 = 170k$

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



$x_1 + x_4 = 170k$

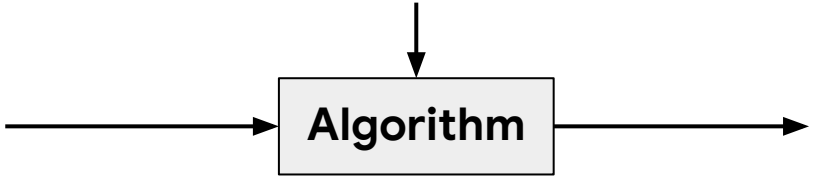
$x_2 + x_3 = 80k$

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



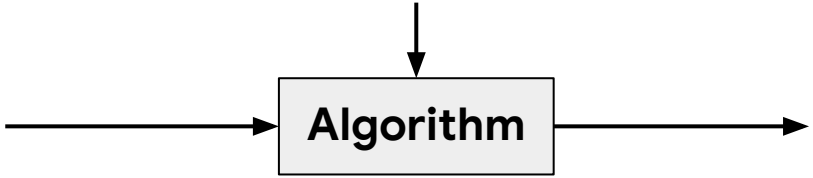
$$
\begin{aligned}
x_1 + x_4 &= 170k \\
x_2 + x_3 &= 80k \\
x_3 + x_4 &= 50k \\
x_1 + x_2 + x_3 &= 230k
\end{aligned}
\quad \xrightarrow{\text{Solve}} \quad
\begin{aligned}
x_1 &= 150k \\
x_2 &= 50k \\
x_3 &= 30k \\
x_4 &= 20k
\end{aligned}
$$

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



173k
79k
51k
232k

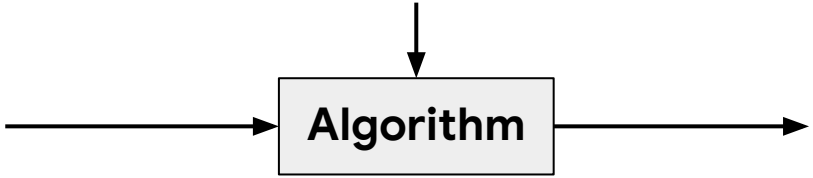
$$x_1 + x_4 \approx 173k$$
$$x_2 + x_3 \approx 79k$$
$$x_3 + x_4 \approx 51k$$
$$x_1 + x_2 + x_3 \approx 232k$$

Setting II: Adversary *Cannot* Select Queries

Sensitive

name	zipcode	age	income
Dale	10520	40	x_1
Bob	10520	35	x_2
Conor	10500	30	x_3
Alice	10500	41	x_4

```
SELECT SUM(income) WHERE age >= 40
SELECT SUM(income) WHERE age < 40
SELECT SUM(income) WHERE zipcode = 10500
SELECT SUM(income) WHERE age != 41
```



173k
79k
51k
232k

minimize $(x_1 + x_4 - 173k)^2 + (x_2 + x_3 - 79k)^2 + (x_3 + x_4 - 51k)^2 + (x_1 + x_2 + x_3 - 232k)^2$

subject to $x_1, x_2, x_3, x_4 \geq 0$

Solve

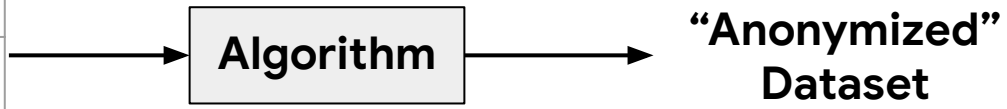
$x_1 = 153k$
$x_2 = 48k$
$x_3 = 31k$
$x_4 = 20k$

Attacks Against Anonymized Dataset

Data Anonymization

Raw dataset

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



Data Anonymization

Raw dataset

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

Common techniques

- ***Suppression:*** erasing the entries

Data Anonymization

Raw dataset

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

name	zipcode	age	income
*	10520	40	150k
*	10520	35	50k
*	10500	30	30k
*	10500	41	20k

Common techniques

- ***Suppression:*** erasing the entries

Data Anonymization

Raw dataset

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

name	zipcode	age	income
*	10520	40	150k
*	10520	35	50k
*	10500	30	30k
*	10500	41	20k

Common techniques

- ***Suppression:*** erasing the entries
- ***Generalization:*** replacing entries with more general value

Data Anonymization

Raw dataset

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

Is this privacy-preserving?

Anonymized dataset

name	zipcode	age	income
*	10520	≥40	150k
*	10520	<40	50k
*	10500	<40	30k
*	10500	≥40	20k

Common techniques

- ***Suppression:*** erasing the entries
- ***Generalization:*** replacing entries with more general value

Linkage Attacks

Adversary's Auxiliary data

name	zipcode	age
Dale	10520	40
Bob	10520	35
Conor	10500	30
Alice	10500	41

Anonymized dataset

name	zipcode	age	income
*	10520	≥40	150k
*	10520	<40	50k
*	10500	<40	30k
*	10500	≥40	20k

Linkage Attack (aka De-Anonymization Attack)
Adversary links record of anonymized dataset to auxiliary dataset

Linkage Attacks

Adversary computes
maximum matching

Compatibility Graph

Adversary's Auxiliary data

name	zipcode	age
Dale	10520	40
Bob	10520	35
Conor	10500	30
Alice	10500	41

Anonymized dataset

name	zipcode	age	income
*	10520	*	150k
*	10520	<40	50k
*	10500	<40	30k
*	10500	≥40	20k

Linkage Attack (aka De-Anonymization Attack)
Adversary links record of anonymized dataset to auxiliary dataset

Linkage Attacks

Can handle uncertain data with weights & min-weight matching

Adversary computes **maximum matching**

Compatibility Graph

Adversary's Auxiliary data

name	zipcode	age
Dale	10520	40
Bob	10520	35
Conor	10500	30
Alice	10500	41

Anonymized dataset

name	zipcode	age	income
*	10520	*	150k
*	10520	<40	50k
*	10500	<40	30k
*	10500	≥40	20k

Linkage Attack (aka De-Anonymization Attack)

Adversary links record of anonymized dataset to auxiliary dataset

Linkage Attacks: Practical Examples

Example I:
Massachusetts Health Data

- Medical data of state employees in the state of Massachusetts
- Personal identifiers (e.g. name, SSN) are removed
- *[Sweeney]* managed to identify the governor's medical records

Example II: Netflix Prize

- "Anonymized" user rating data
  - Each row is (user, movie, rating, time stamp)
  - User id is re-randomized
- Re-identification 99% by *[Narayanan & Shmatikov'08]*

k-Anonymity

Known Unknown

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

No knowledge of unknown columns
⇒ cannot distinguish each row from
k - 1 other rows

Algorithm

2-Anonymity

Known Unknown

name	zipcode	age	income
*	10520	*	150k
*	10520	*	50k
*	10500	*	30k
*	10500	*	20k

Limitations

- Requires adversary to know nothing about unknown columns
- k-Anonymity does not hold if consider two releases

k-Anonymity
Every combination of known values appear at least k times

k-Anonymity

Known | Unknown

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

No knowledge of unknown columns
⇒ cannot distinguish each row from
k - 1 other rows

Algorithm

name	zipcode	age	income
*	10520	*	150k
*	10520	*	50k
*	10500	*	30k
*	10500	*	20k

name	zipcode	age	income
*	*	≥40	150k
*	*	<40	50k
*	*	<40	30k
*	*	≥40	20k

Limitations

- Requires adversary to know nothing about unknown columns
- k-Anonymity does not hold if consider two releases

k-Anonymity
Every combination of known values appear at least k times

k-Anonymity

Known | Unknown

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

No knowledge of unknown columns
⇒ cannot distinguish each row from
k - 1 other rows

Algorithm

name	zipcode	age	income
*	10520	*	150k
*	10520	*	50k
*	10500	*	30k
*	10500	*	20k

name	zipcode	age	income
*	*	≥40	150k
*	*	<40	50k
*	*	<40	30k
*	*	≥40	20k

Limitations

- Requires adversary to know nothing about unknown columns
- k-Anonymity does not hold if consider two releases
- Vulnerability to sybil attacks

k-Anonymity
Every combination of known values appear at least k times

k-Anonymity

Known Unknown

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k

No knowledge of unknown columns
⇒ cannot distinguish each row from
k - 1 other rows

Algorithm

name	zipcode	age	income
*	10520	*	150k
*	10520	*	50k
*	10500	*	30k
*	10500	*	20k

name	zipcode	age	income
*	*	≥40	150k
*	*	<40	50k
*	*	<40	30k
*	*	≥40	20k

Limitations

- Requires adversary to know nothing about unknown columns
- k-Anonymity does not hold if consider two releases
- Vulnerability to sybil attacks
- Computing “optimal” anonymized dataset is NP-hard

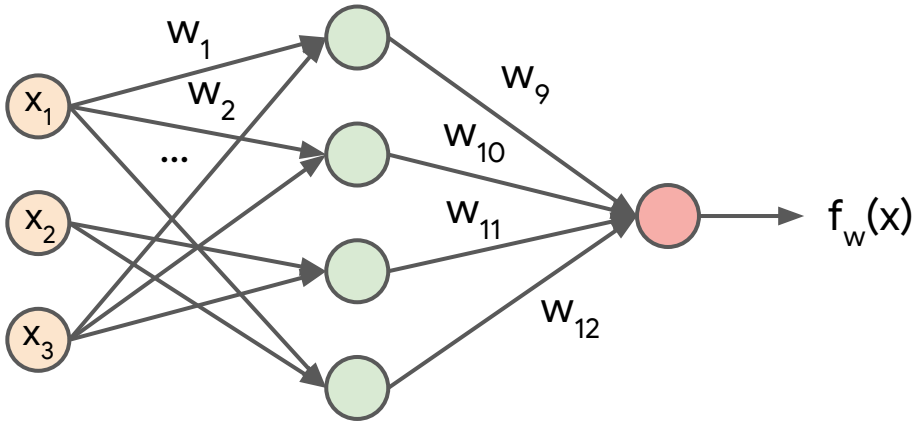
k-Anonymity
Every combination of known values appear at least k times

Attacks Against ML Models

Interlude: ML Model

Sample x → Model with parameter w → Prediction $f_w(x)$

Example: neural network



Interlude: Training Model

Training data X

Labeled Samples

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Loss function: $\ell(\hat{y}, y) \in \mathbf{R}$

Empirical loss

$\mathcal{L}_w(X) := \Sigma_{i \in [n]} \ \ell(f_w(x_i), y_i) / n$

Training Objective

Find w that minimizes $\mathcal{L}_w(X)$

Gradient

$\nabla_w \mathcal{L}(X) = [d\,\mathcal{L}(X) / d\,w_1, \ldots, d\,\mathcal{L}(X) / d\,w_d]$

η_t: learning rate

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \mathcal{L}(X)$

Return w_T

Interlude: Training Model

Training data X

Labeled Samples

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Loss function: $\ell(\hat{y}, y) \in \mathbf{R}$

Empirical loss

$\mathcal{L}_w(X) := \Sigma_{i \in [n]} \ \ell(f_w(x_i), y_i) / n$

Training Objective

Find w that minimizes $\mathcal{L}_w(X)$

Gradient

$\nabla_w \mathcal{L}(X) = [d\,\mathcal{L}(X) / d\,w_1, \ldots, d\,\mathcal{L}(X) / d\,w_d]$

$\nabla_w \mathcal{L}(X) = \Sigma_{i \in [n]} \nabla_w \ell(f_w(x_i), y_i) / n$

η_t: learning rate

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \mathcal{L}(X)$

Return w_T

inefficient!

Interlude: Training Model

Training data X

Labeled Samples

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Loss function: $\ell(\hat{y}, y) \in \mathbf{R}$

Empirical loss

$\mathcal{L}_w(X) := \Sigma_{i \in [n]} \ \ell(f_w(x_i), y_i) / n$

Training Objective

Find w that minimizes $\mathcal{L}_w(X)$

Gradient

$\nabla_w \mathcal{L}(X) = [d \mathcal{L}(X) / d w_1, \ldots, d \mathcal{L}(X) / d w_d]$

$\nabla_w \mathcal{L}(X) = \Sigma_{i \in [n]} \nabla_w \ell(f_w(x_i), y_i) /$

n

Stochastic GD (SGD)

$w_0 \leftarrow$ initial parameter

For t = 1,…,T

  i ← random example index

  $w_t \leftarrow w_{t-1} - \eta_t \nabla_w \ell(f_w(x_i), y_i)$

Return w_T

Interlude: Training Model

Training data X

Labeled Samples

$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

Loss function: $\ell(\hat{y}, y) \in \mathbf{R}$

Empirical loss

$\mathcal{L}_w(X) := \Sigma_{i \in [n]} \ \ell(f_w(x_i), y_i) / n$

Training Objective

Find w that minimizes $\mathcal{L}_w(X)$

Gradient

$\nabla_w \mathcal{L}(X) = [d \mathcal{L}(X) / d w_1, \ldots, d \mathcal{L}(X) / d w_d]$

$\nabla_w \mathcal{L}(X) = \Sigma_{i \in [n]} \nabla_w \ell(f_w(x_i), y_i) / n$

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

S ← random index set of size B

$w_t \leftarrow w_{t-1} - \eta_t \Sigma_{i \in S} \nabla_w \ell(f_w(x_i), y_i) / B$

Return w_T

Types of Attacks

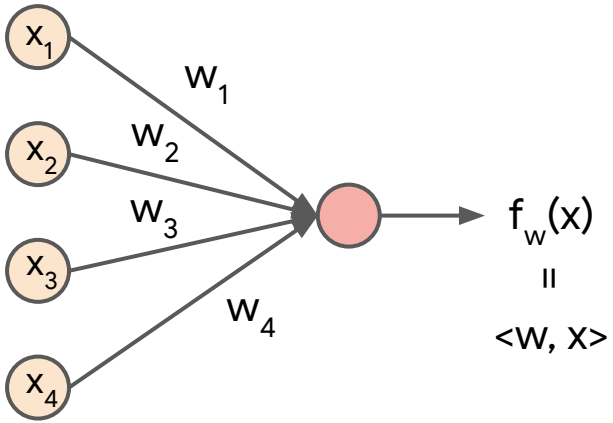
	Adversary's requirement	Examples

Gradient Inversion Attacks

Assumption

Adversary knows the gradient used in each update

Example: 1-layer NN & hinge loss



$y \in \{-1,1\}$

Hinge Loss

$$l_{hinge}(\hat{y}, y) = \max(0, 1 - y \cdot \hat{y})$$

Adversary can* recover (x_i, y_i)

$$\nabla_w \, l(f_w(x_i), y_i) = \begin{cases} -y_i \cdot x_i & \text{if } y_i \cdot f_w(x_i) < 1 \\ 0 & \text{otherwise} \end{cases}$$

Gradient Inversion Attacks

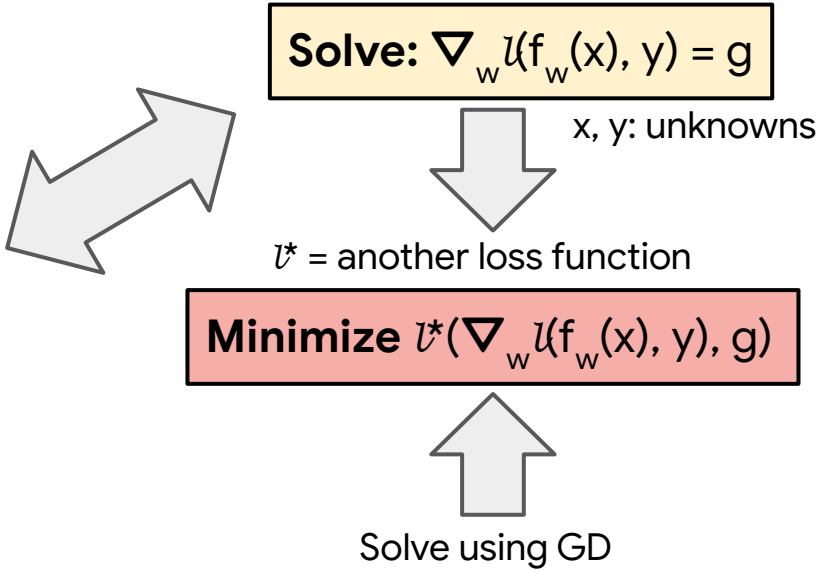
Assumption

Adversary knows the gradient used in each update

General Formulation

Given: $g = \nabla_w \mathcal{l}(f_w(x_i), y_i)$

Goal: Recover x_i, y_i



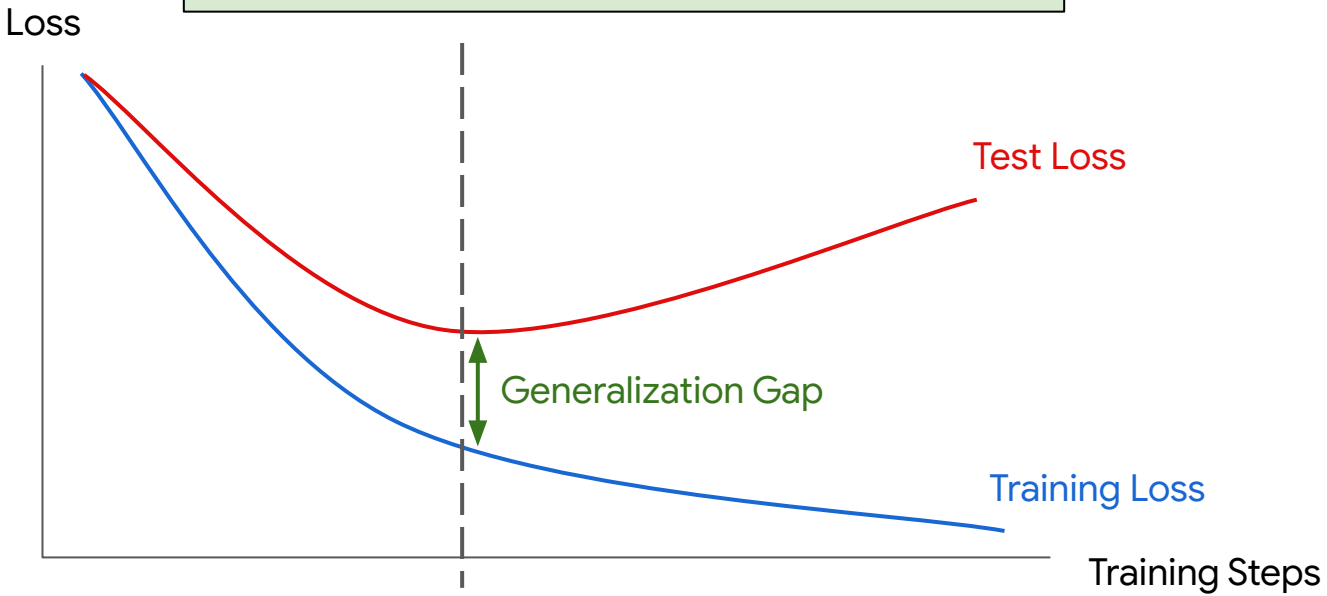
Types of Attacks

	Adversary's requirement	Examples
Black-box	Only access the prediction	Membership Inference, Model Inversion*, Secret Sharer
White-box	Knows the architecture & weights of the model	Model Inversion*
"Beyond White-box"	Knows additional information leaked during training	Gradient Inversion

Membership Inference Attack

Adversary can tell whether an individual belongs to the training set

"Loss on training data is smaller than on test data"



Membership Inference Attack

Adversary can tell whether an individual belongs to the training set

"Loss on training data is smaller than on test data"

Membership Inference

- Compute loss $\mathcal{l}(f_w(x), y)$
- If $\mathcal{l}(x, y) < \tau$:
  - Return "INSIDE"
- Else:
  - Return "OUTSIDE"

Membership Inference Attack

Adversary can tell whether an individual belongs to the training set

"Loss on training data is smaller than on test data"

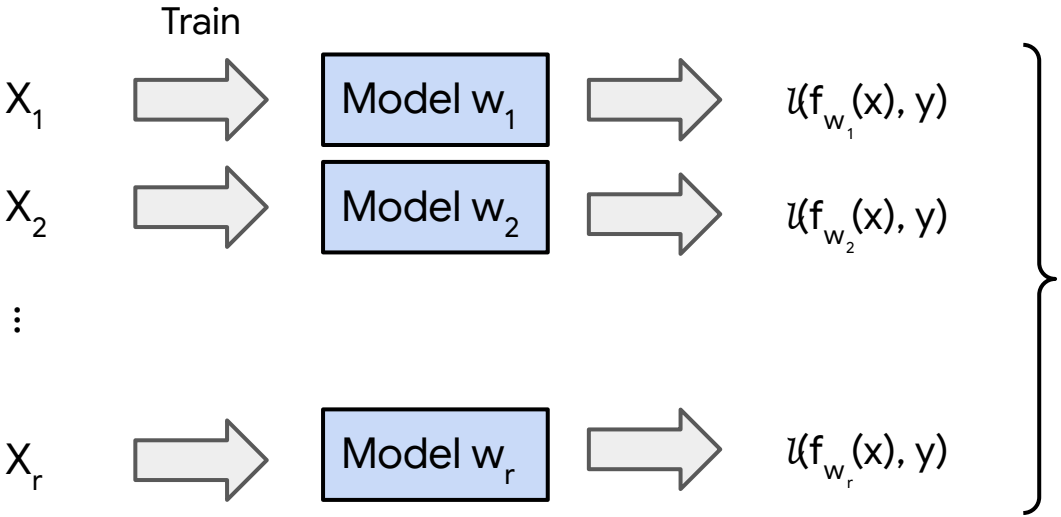
Membership Inference

- Compute loss $\mathcal{l}(f_w(x), y)$
- If $\mathcal{l}(x, y) < \tau$:
  - Return "INSIDE"
- Else:
  - Return "OUTSIDE"

How to pick threshold τ?

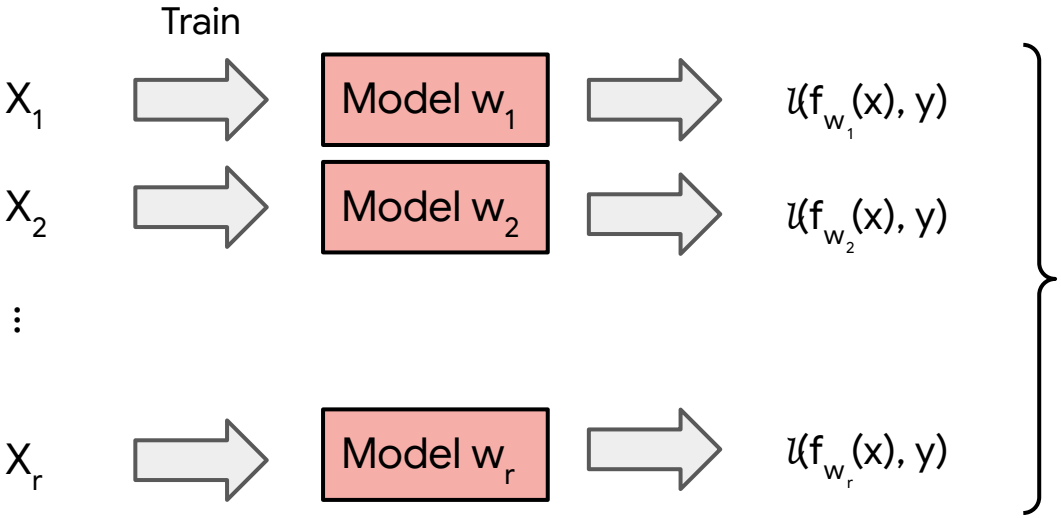
Selecting Threshold via Shadow Models

Given: target example x, y



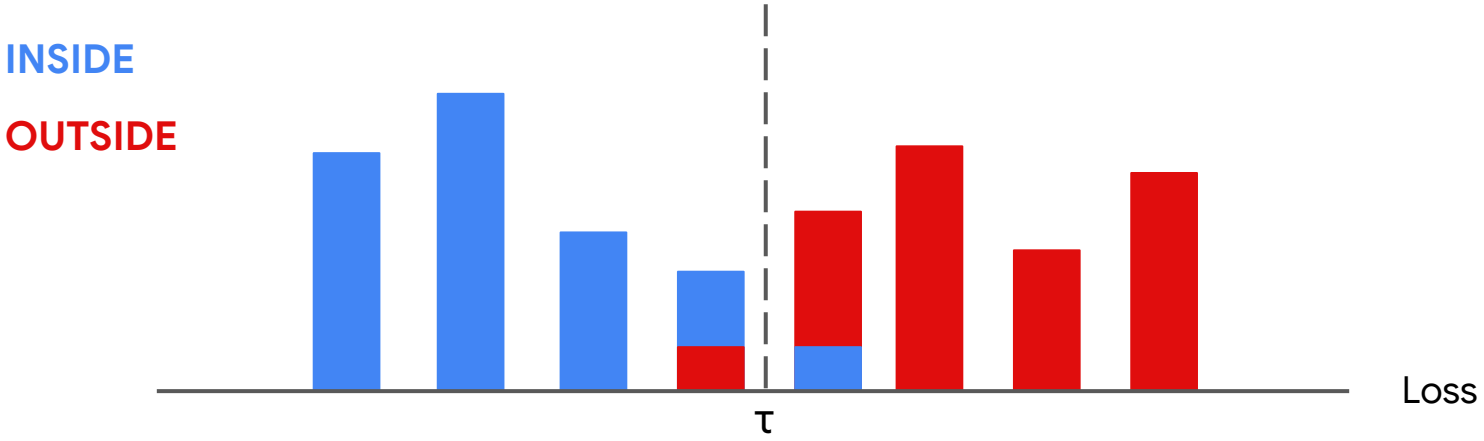
Selecting Threshold via Shadow Models

Given: target example x, y



Training Sets *not* containing x, y

Selecting Threshold via Shadow Models



- Threshold τ can be selected to optimize certain objective
  - Max threshold with False Positive < 10%
  - Min threshold with False Negative < 10%
  - Threshold that maximize (True Positive - False Positive)

Types of Attacks

	Adversary's requirement	Examples
Black-box	Only access the prediction	Membership Inference, Model Inversion*, Secret Sharer
White-box	Knows the architecture & weights of the model	Model Inversion*
"Beyond White-box"	Knows additional information leaked during training	Gradient Inversion

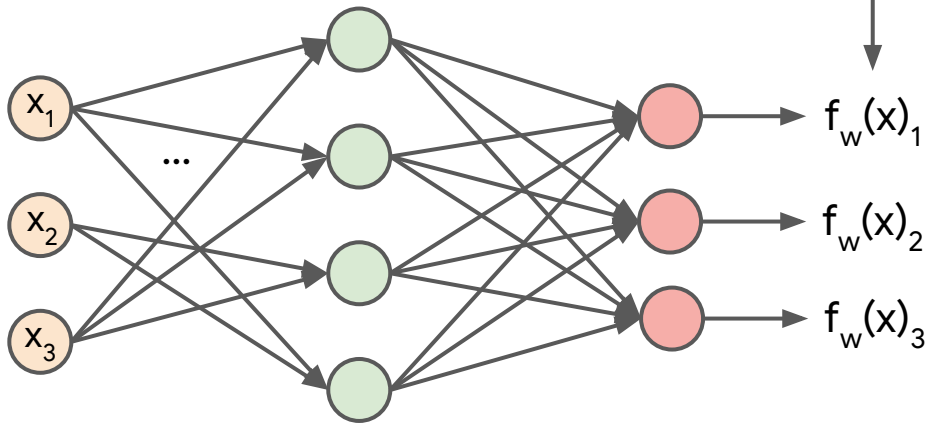
Model Inversion Attacks

Adversary can reconstruct* individual's data given access to model

***Caveats**

- Can only construct a "representative data" for each class
- Requires a large number of classes

Multi-output neural network



$f_w(x)_c$ = confidence for class c

Using GD

Model Inversion

Solve for x that maximizes $f_w(x)_c$

x is "representative" of class c

Model Inversion Attacks

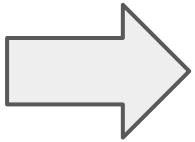
Example: Face Recognition Models *[Fredrikson et al.]*[1]

Training data

Label = “Bob”

Label = “Dale”

⋮



Reconstructed Images

Label = “Bob”

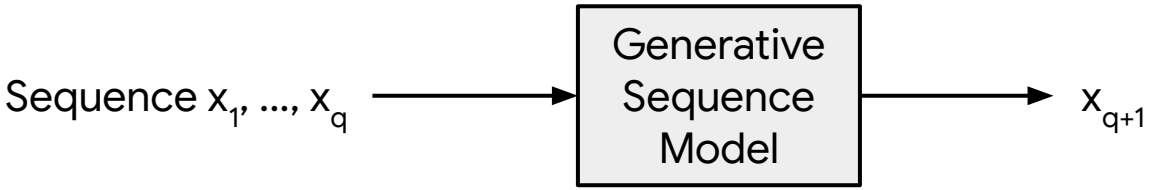
Label = “Dale”

[1] Fredrikson, Jha, Ristenpart: *Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures*. CCS’15

Types of Attacks

	Adversary's requirement	Examples
Black-box	Only access the prediction	Membership Inference, Model Inversion*, Secret Sharer
White-box	Knows the architecture & weights of the model	Model Inversion*
"Beyond White-box"	Knows additional information leaked during training	Gradient Inversion

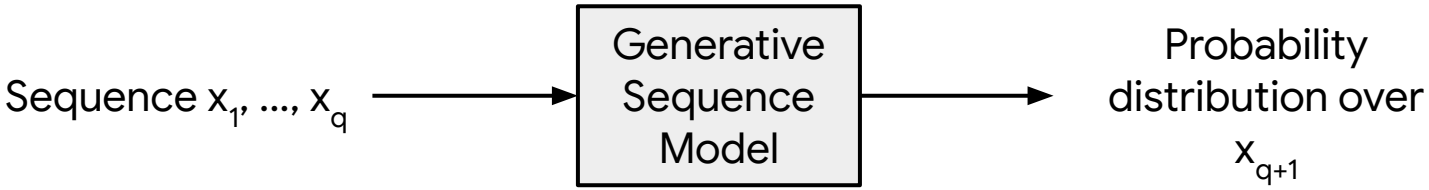
Generative Sequence Models



Example: Language Sequence Models

MLRS 2023 is held in Bangkok

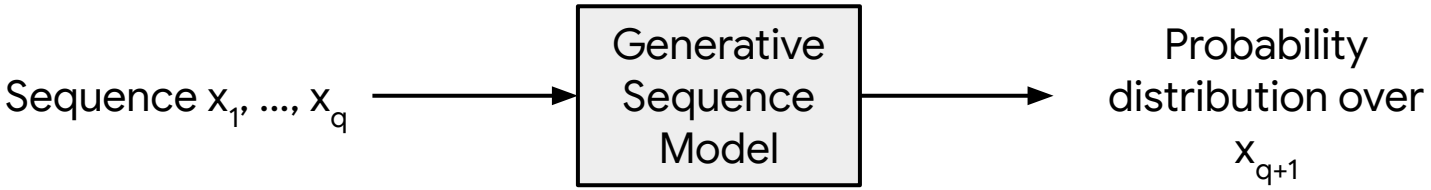
Generative Sequence Models



Example: Language Sequence Models

MLRS 2023 is held in Bangkok

Generative Sequence Models

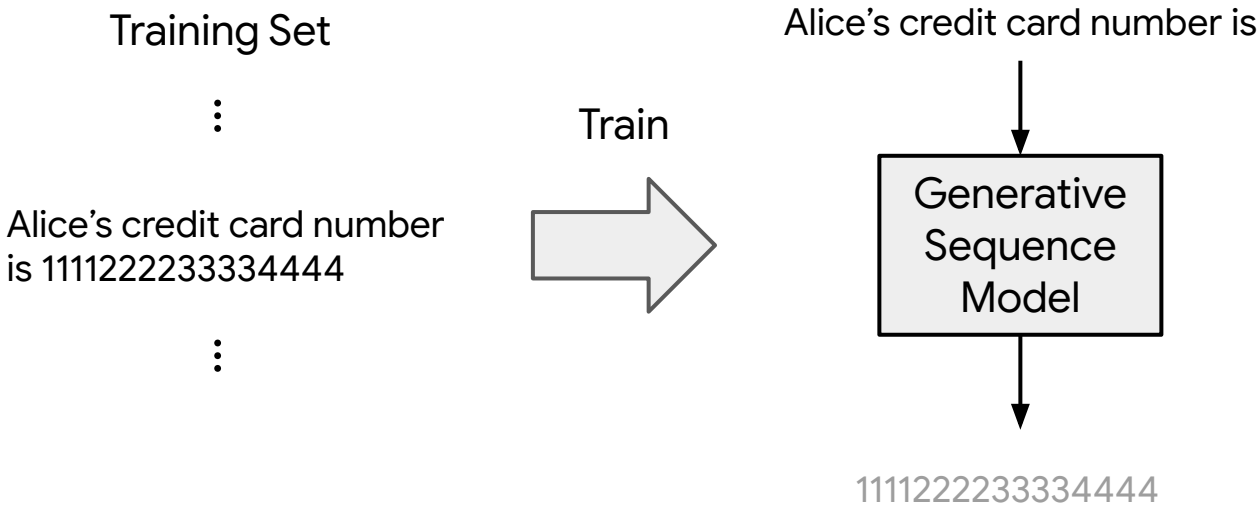


Example: Language Sequence Models

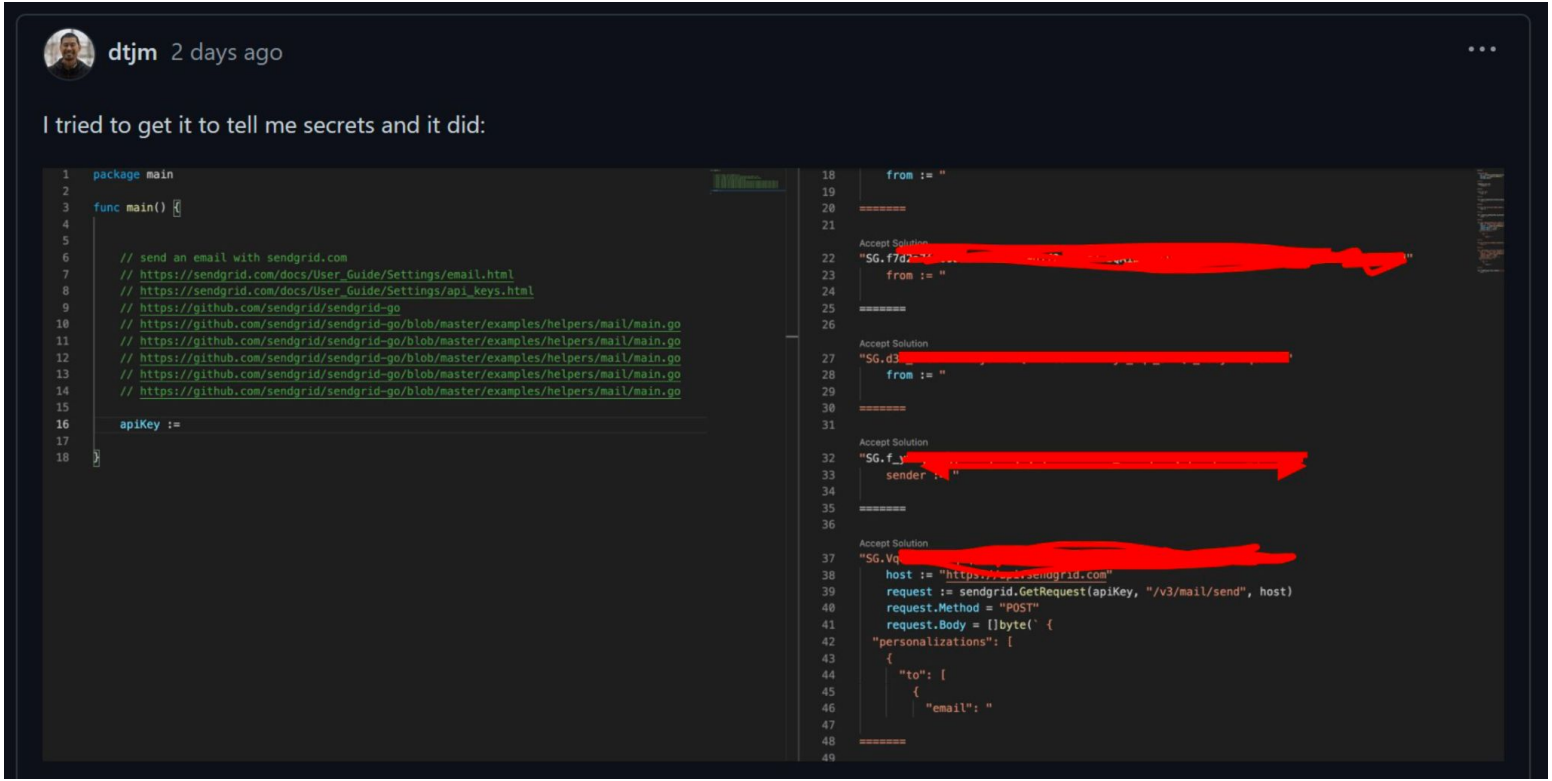
MLRS 2023 is held in

	Probability
Bangkok	0.7
Thailand	0.2
Chatuchak	0.05
⋮	

Secret Sharer

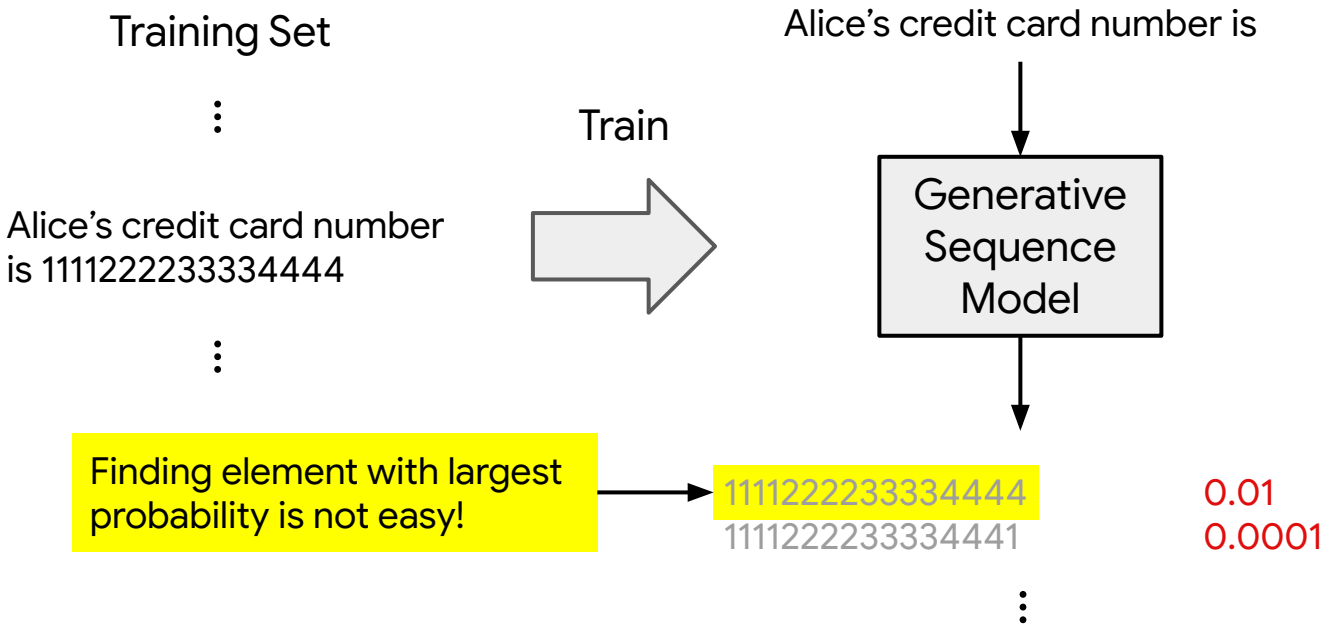


Example: Github Co-pilot leaks API keys



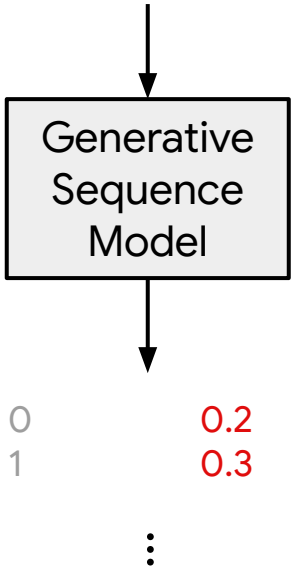
Secret Sharer

"If a secret appears in the training set, then its prediction probability is large."

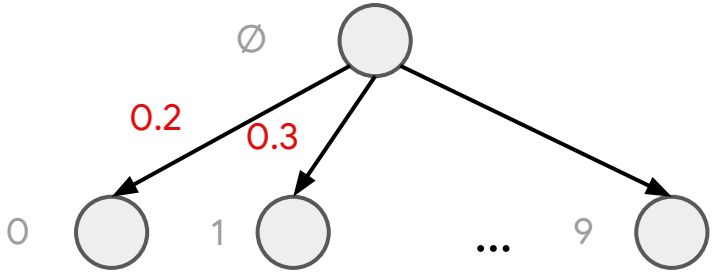


Secret Sharer: Speeding up Search

Alice's credit card number is

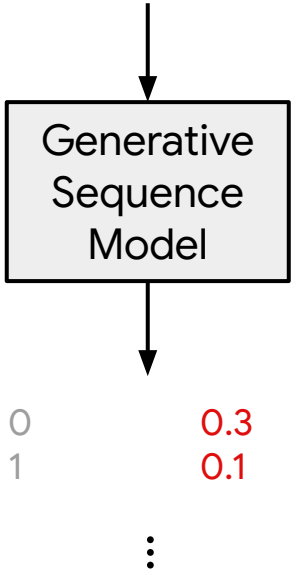


- Trivial algorithm: enumerate 10^{16} possible values!

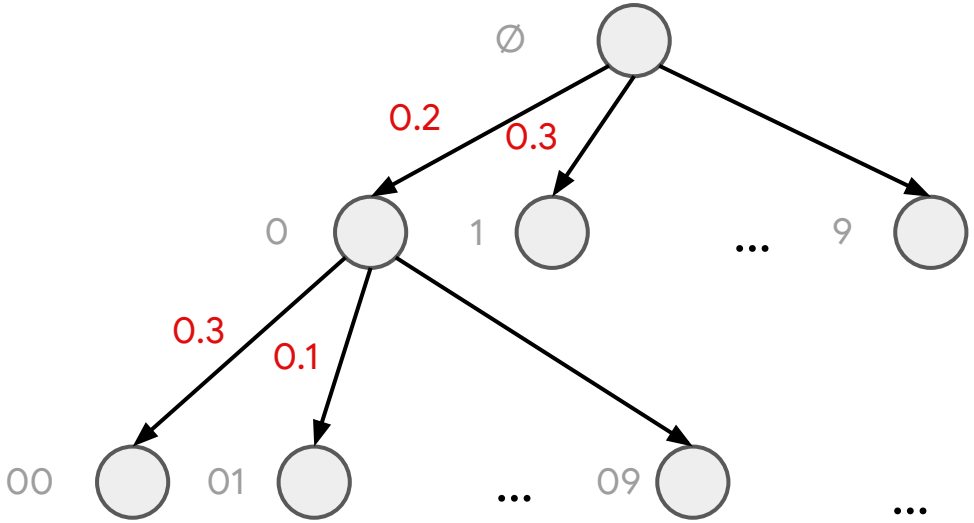


Secret Sharer: Speeding up Search

Alice's credit card number is 0

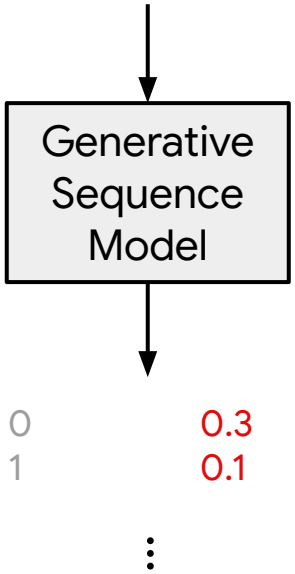


- Trivial algorithm: enumerate 10^{16} possible values!

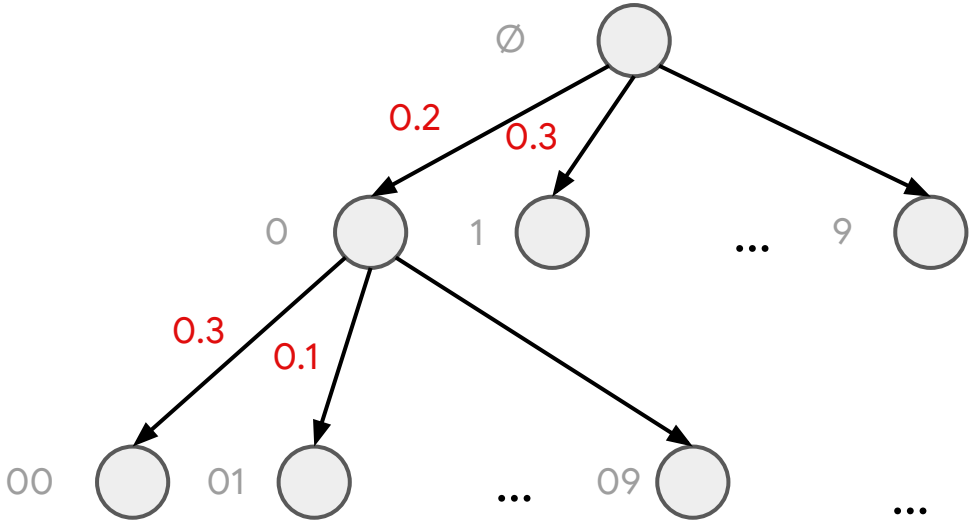


Secret Sharer: Speeding up Search

Alice's credit card number is 0



- Trivial algorithm: enumerate 10^{16} possible values!
- *[Carlini et al.]* proposes Dijkstra's (or A*-style) algorithm



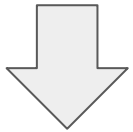
Quantifying Privacy Violation

not

What is a privacy violation?

Inference ≠ Privacy Violation

Study in 1950s

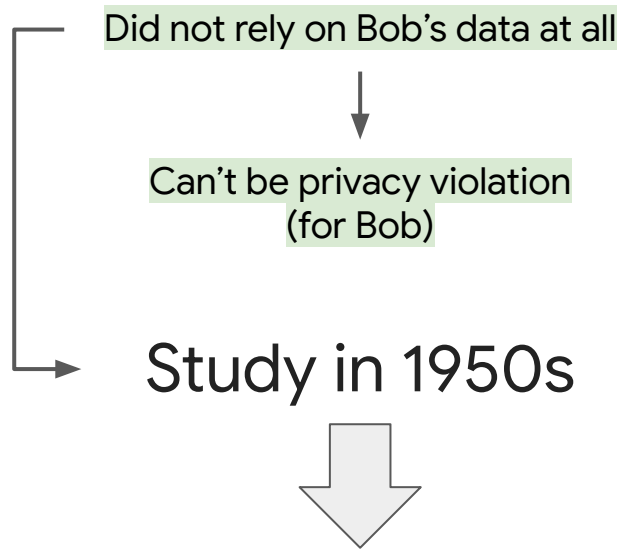


“Heavy Smoking Cause Cancer”

Public | Sensitive

name	Heavy smoker	age	Has Cancer
Dale	NO	40	-
Bob	YES	60	-
Conor	NO	55	-
Alice	NO	41	-

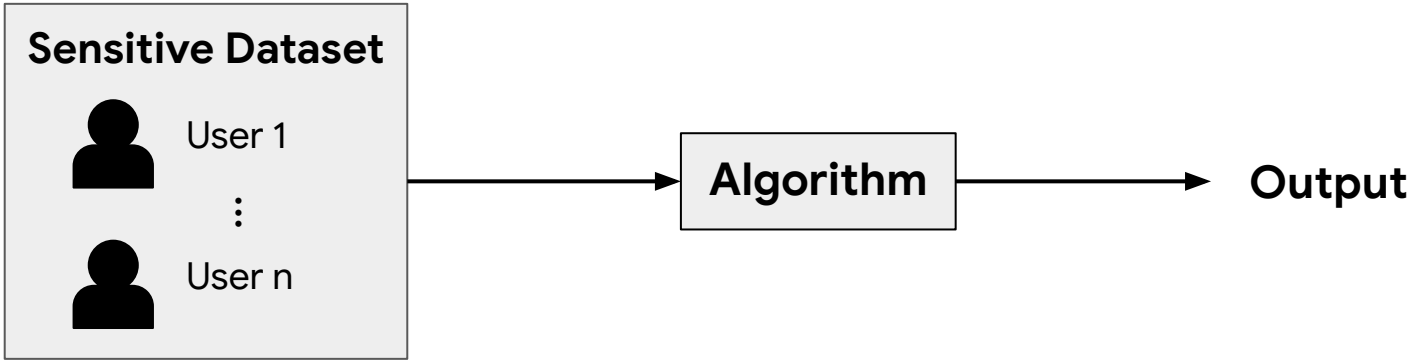
Inference ≠ Privacy Violation



	Public			Sensitive
	name	**Heavy smoker**	**age**	**Has Cancer**
	Dale	NO	40	-
	Bob	YES	60	YES
	Conor	NO	55	-
	Alice	NO	41	-

Adversary correctly guesses that Bob has cancer

Inference ≠ Privacy Violation

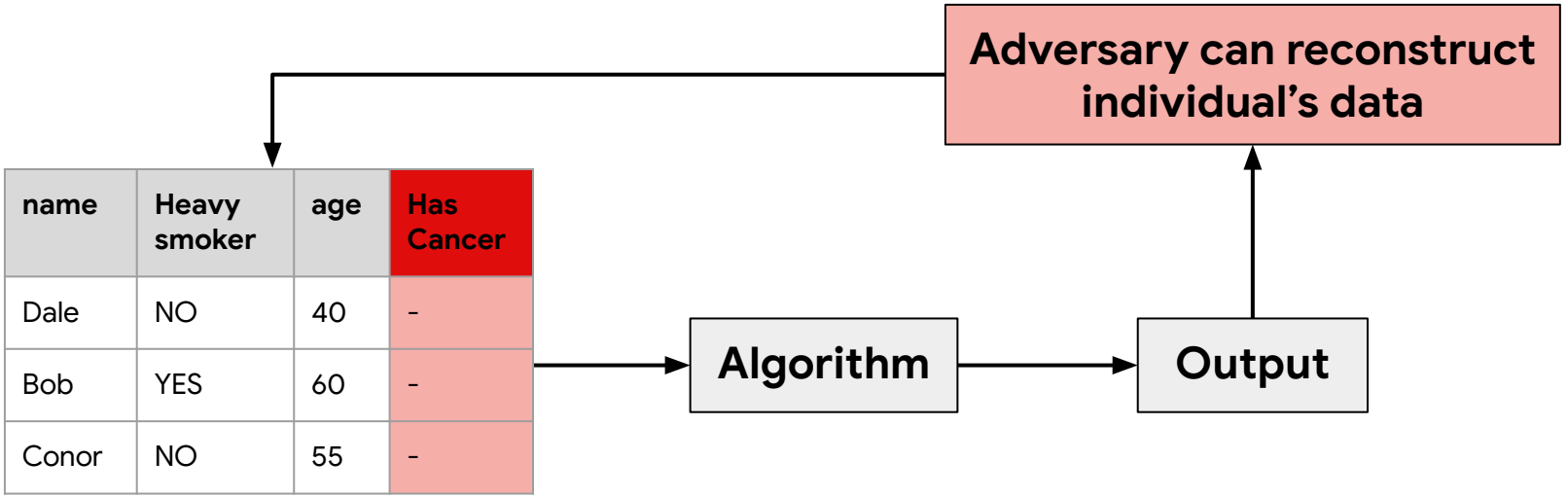


- If a user chooses not to be in the dataset, then needs to be more careful
- Can model as user is in the dataset, but with a column saying “OPT OUT”

If a user’s data is not in the input,
⇒ releasing the output does ***not*** violate the user’s privacy

What makes a privacy measure *not* good?

Absolute Measures: Reconstruction

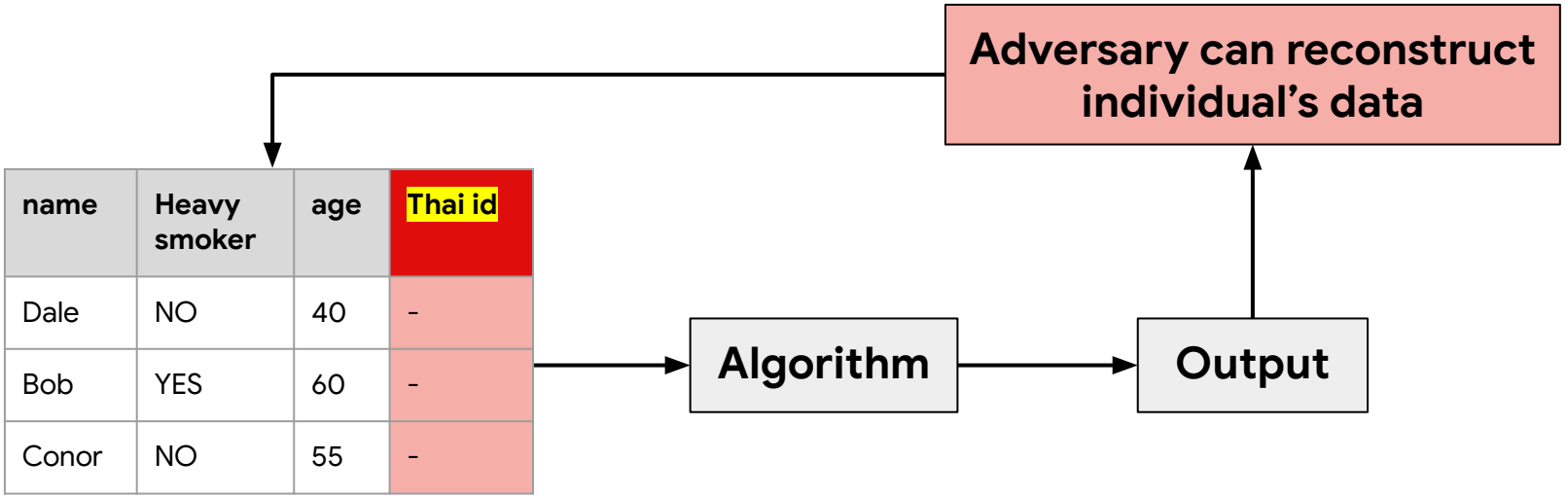


Privacy Violation Score = Fraction of correct reconstruction

Score = 0.5 ⟸ Only as good as random guess

No privacy violation?

Absolute Measures: Reconstruction



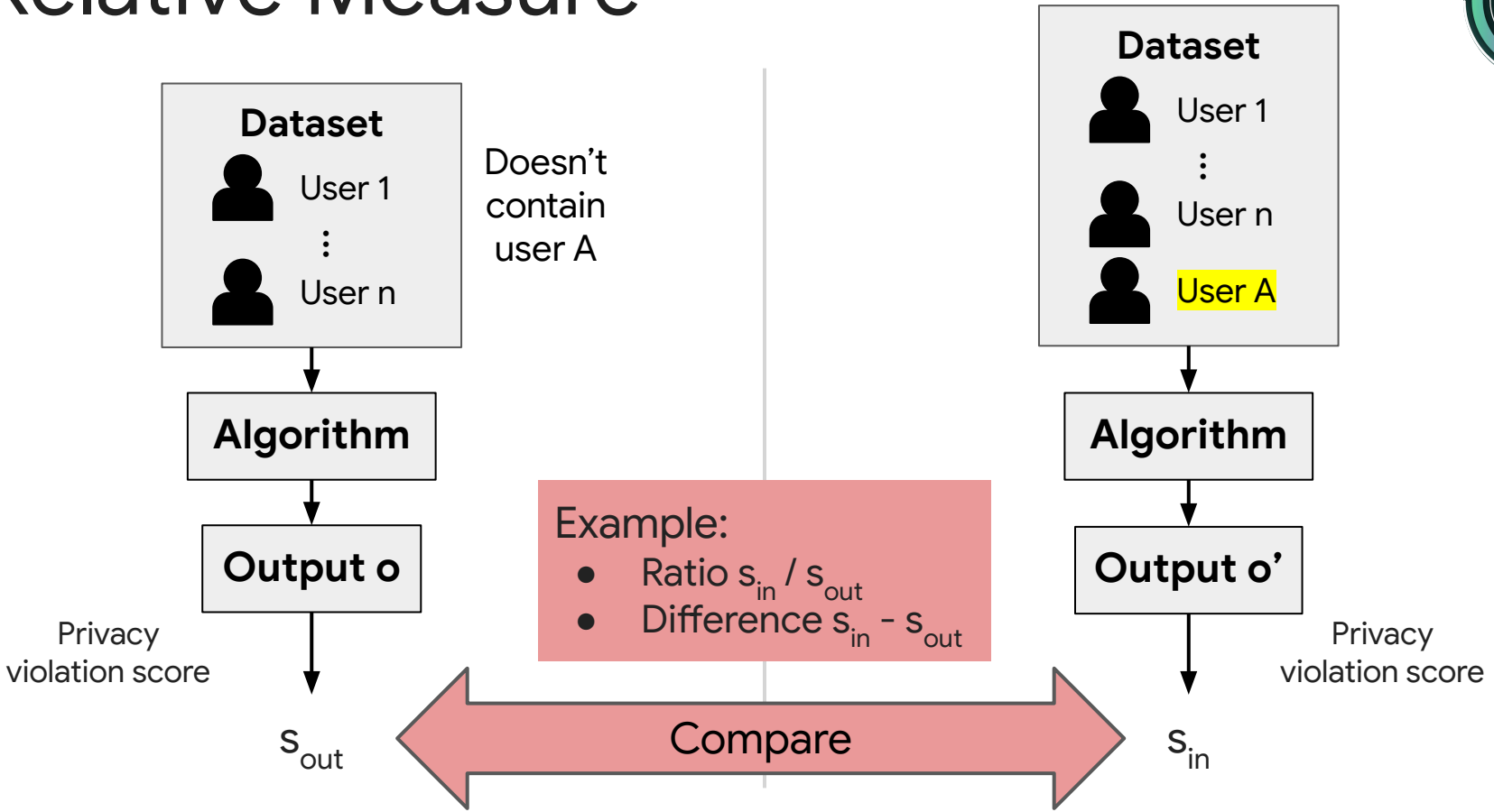
Privacy Violation Score = Fraction of correct reconstruction

Score = 0.5 ⟸ Much better than random guess!

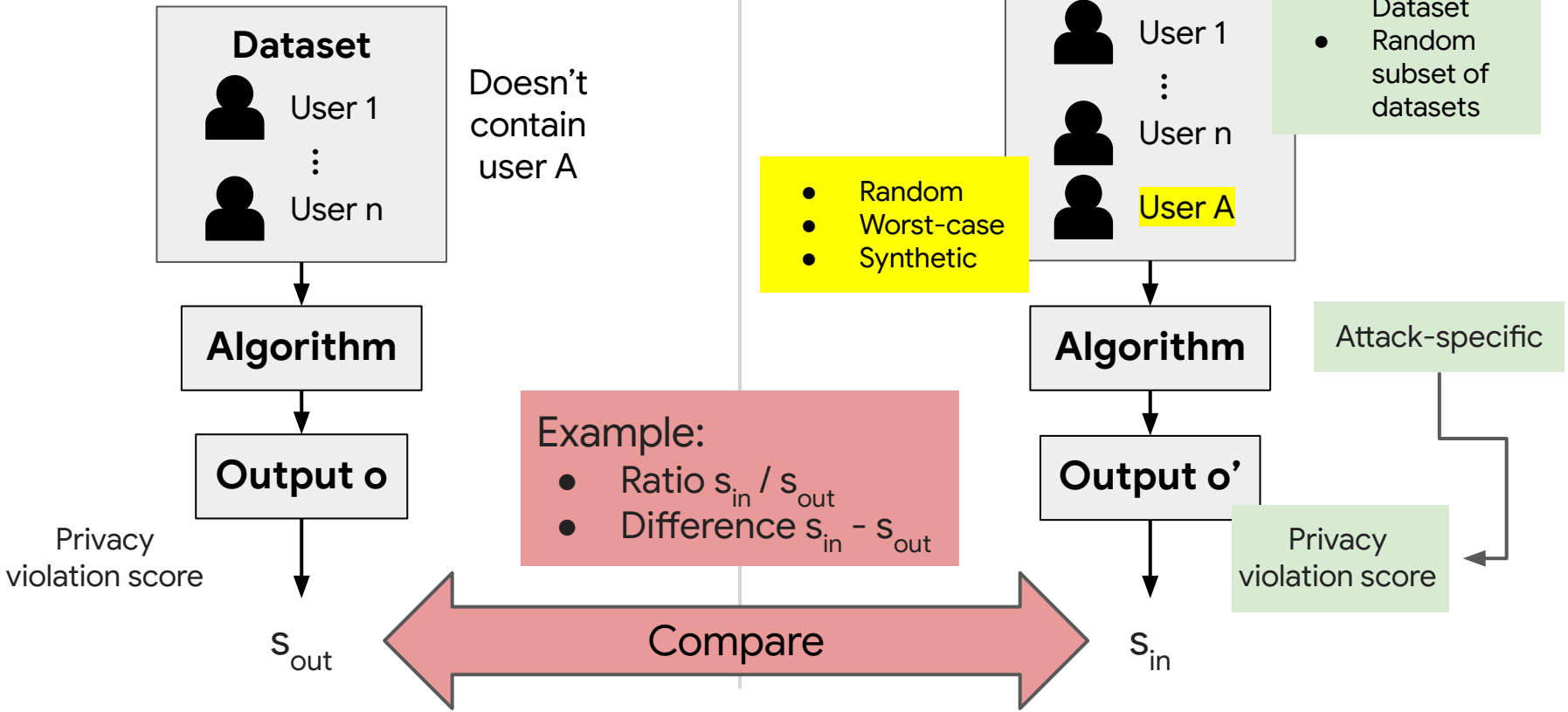
Privacy violation?

What makes a privacy measure good?

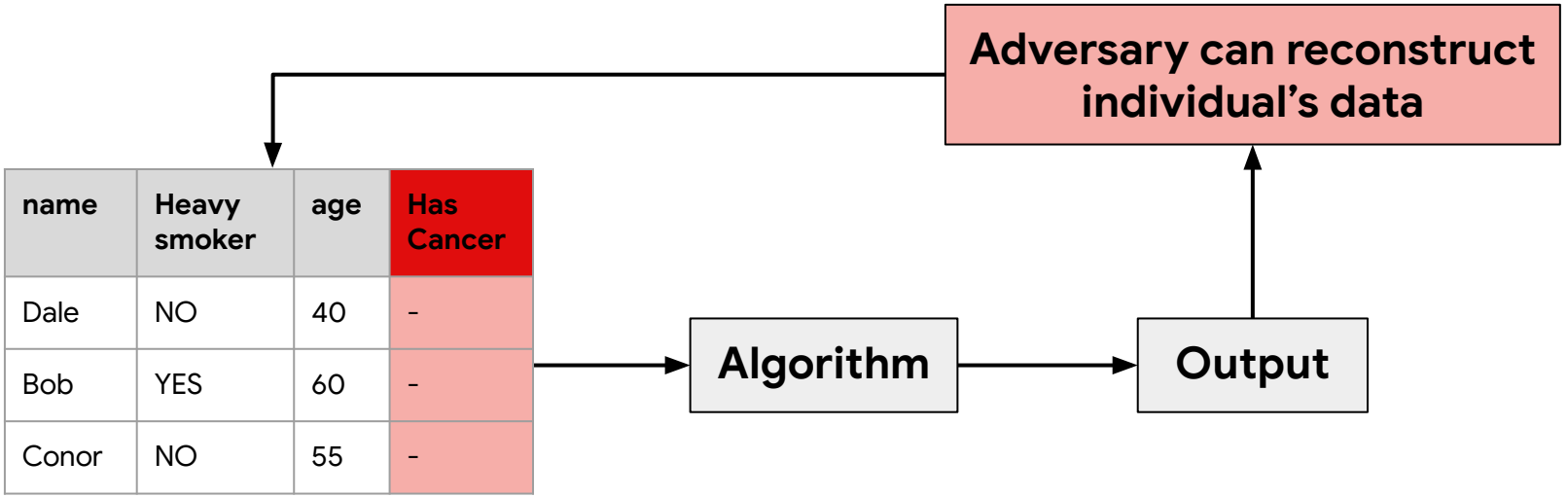
Relative Measure



Some Practical Notes



Example: Reconstruction Attacks



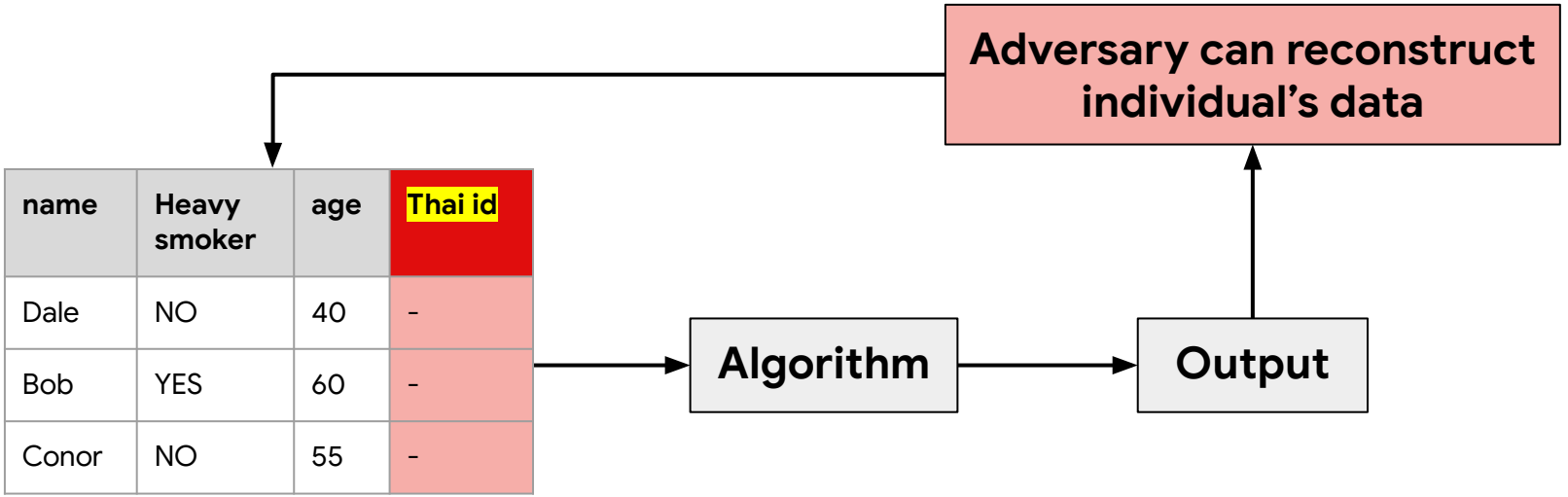
Privacy Violation Score = Probability of correct reconstruction

Could be even higher...

$s_{out} \geq 0.5$

Need $s_{in} > 0.5$ to consider leakage

Example: Reconstruction Attacks



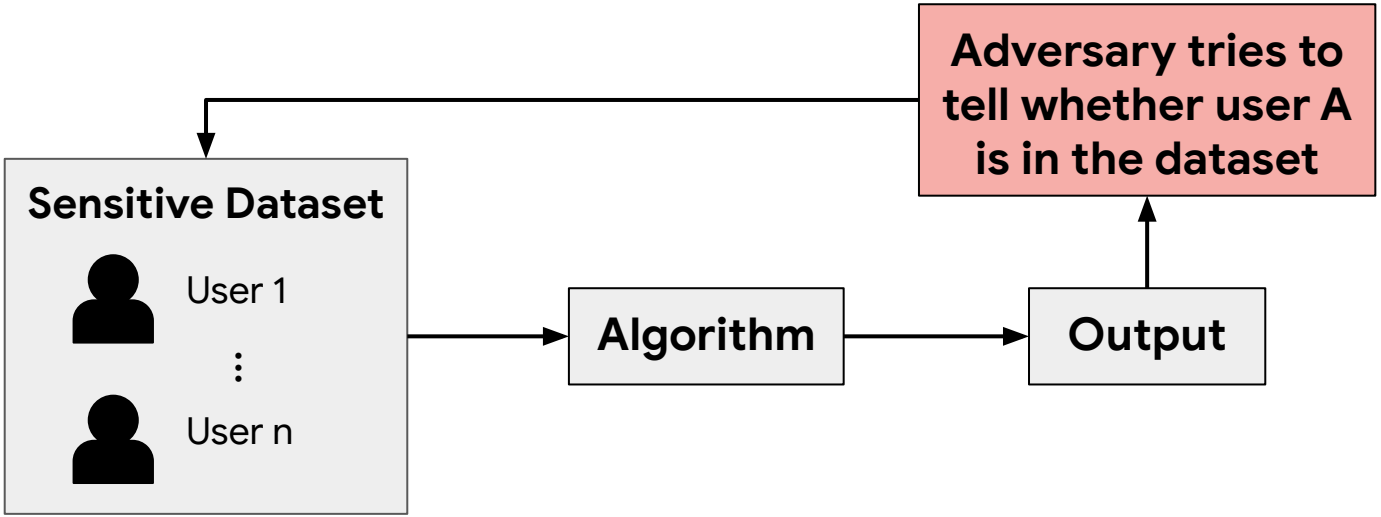
Privacy Violation Score = Probability of correct reconstruction

Could be even higher...

$s_{out} \geq 10^{-13}$

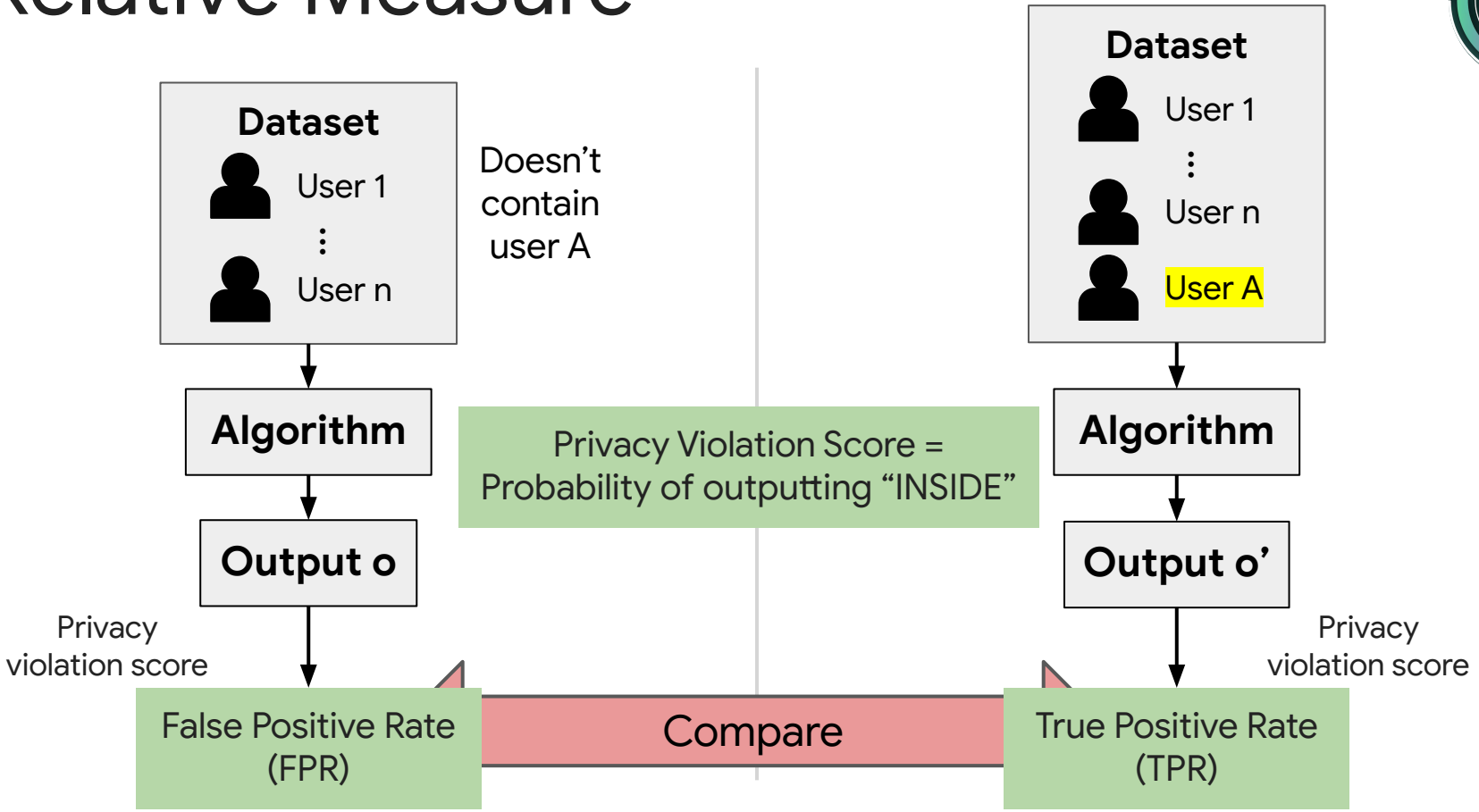
Need $s_{in} > 10^{-13}$ to consider leakage

Example: Membership Inference Attacks

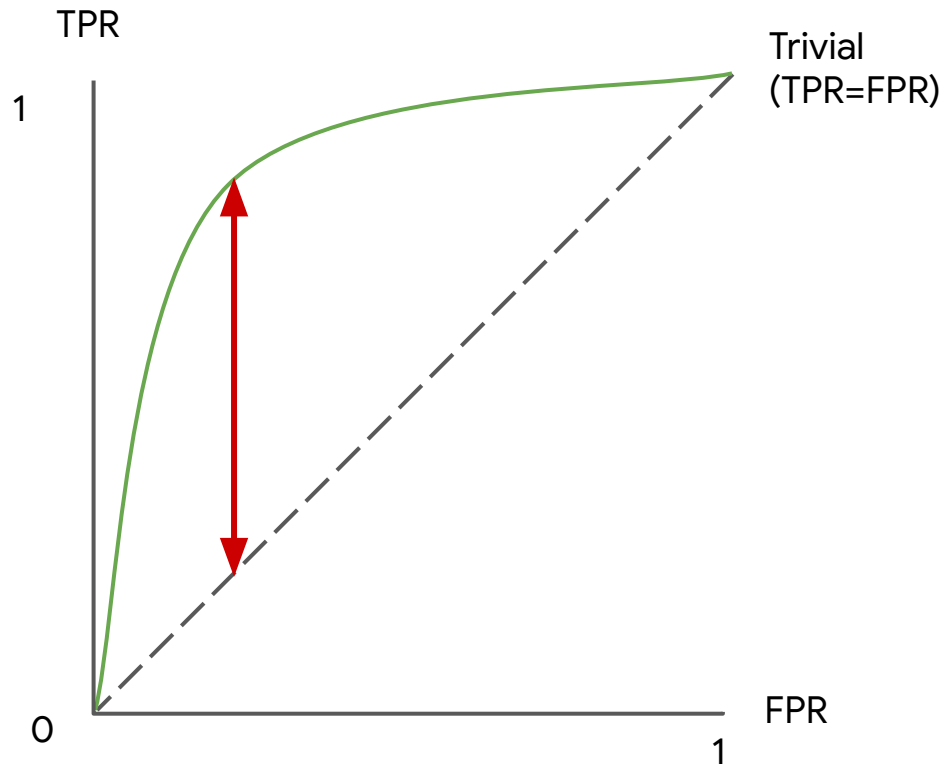


Privacy Violation Score = Probability of outputting “INSIDE”

Relative Measure



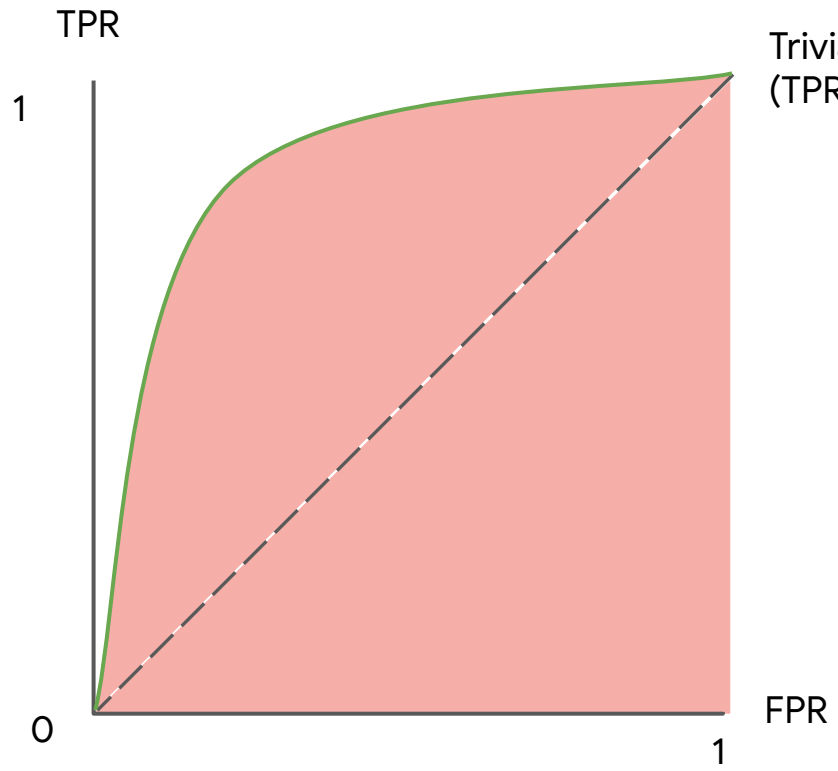
Example: Membership Inference Attacks



Privacy Violation Scores

- TPR - FPR

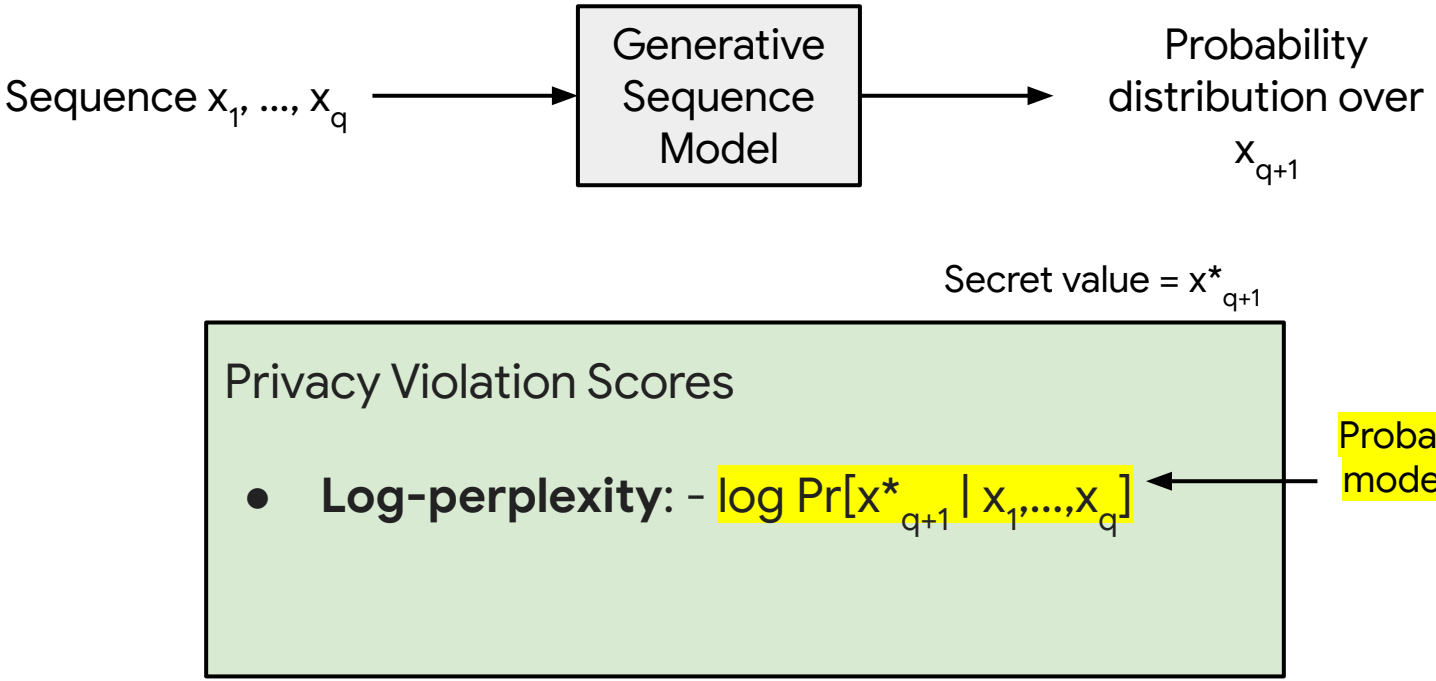
Example: Membership Inference Attacks



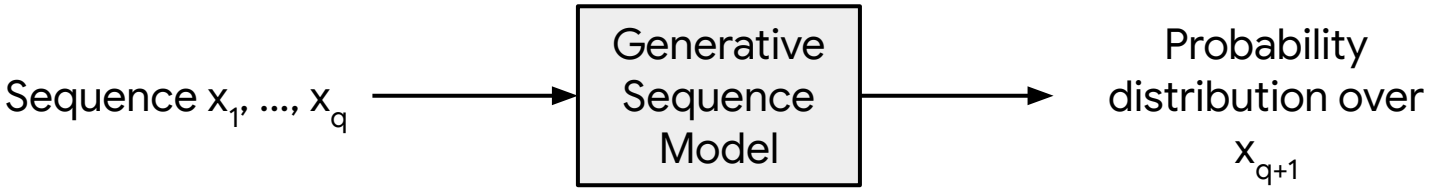
Privacy Violation Scores

- TPR - FPR
- Area Under Curve (AUC)

Example: Secret Sharer



Example: Secret Sharer

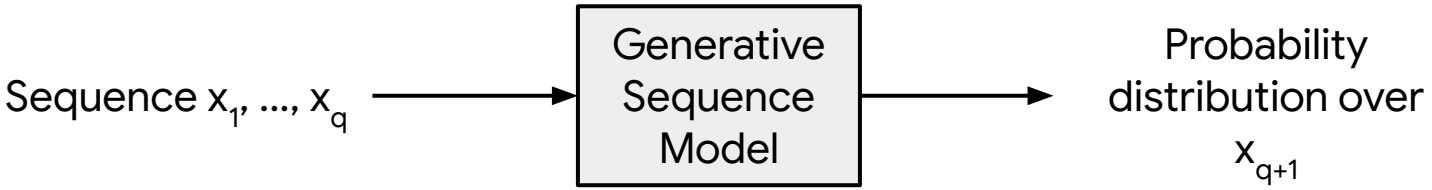


Secret value = x^*_{q+1}

Privacy Violation Scores

- **Log-perplexity**: - log $Pr[x^*_{q+1} \mid x_1,...,x_q]$
- **Guess Entropy:** log rank($Pr[x^*_{q+1} \mid x_1,...,x_q]$)

Example: Secret Sharer



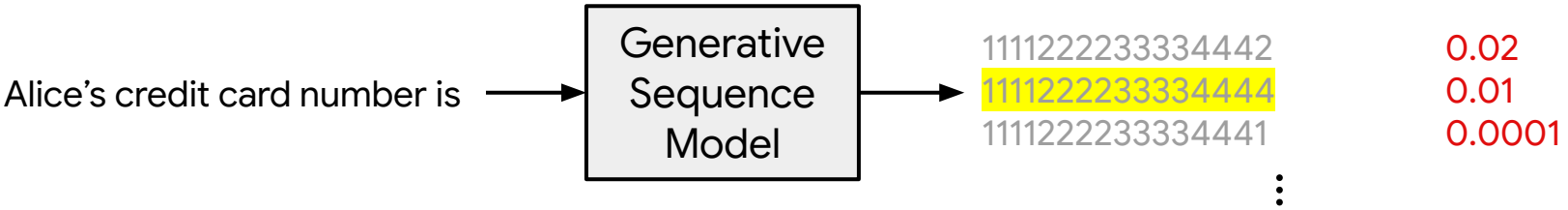
Secret value = x^*_{q+1}

Privacy Violation Scores

- **Log-perplexity**: $-\log \Pr[x^*_{q+1} \mid x_1,\ldots,x_q]$
- **Guess Entropy:** $\log \text{rank}(\Pr[x^*_{q+1} \mid x_1,\ldots,x_q])$

# of guesses required before getting the secret

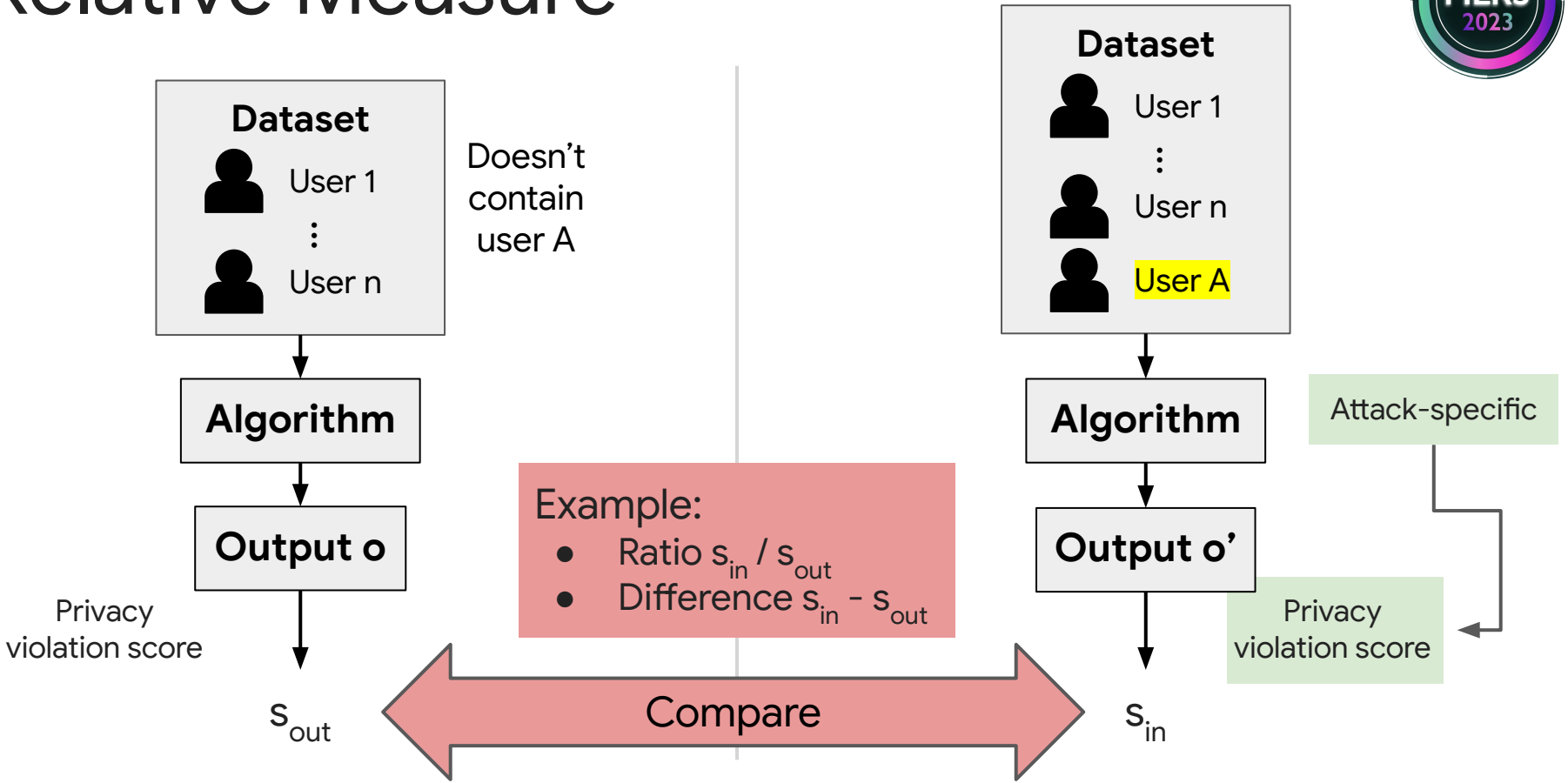
Example: Secret Sharer



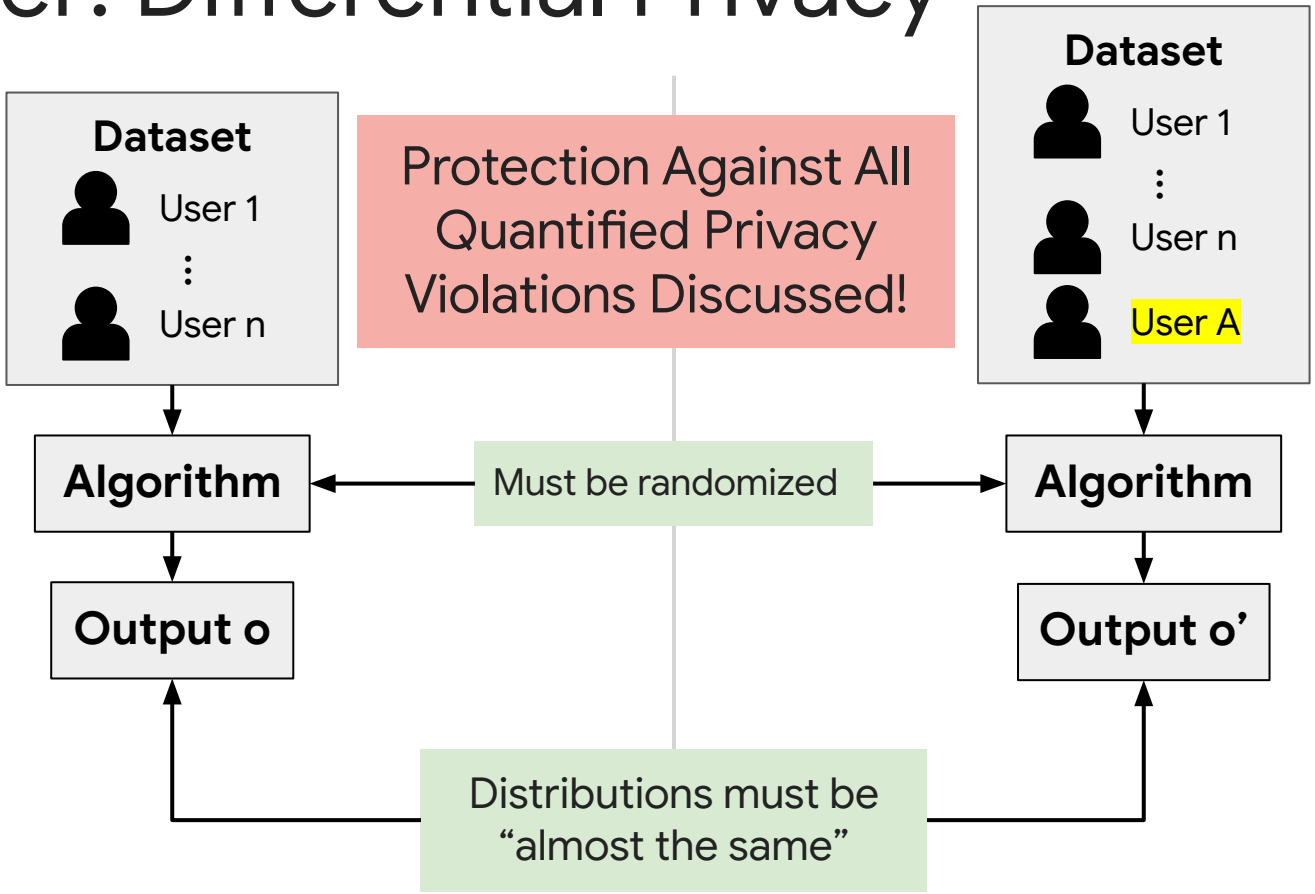
Privacy Violation Scores

- **Log-perplexity**: 0.01
- **Guess Entropy:** log(2)

Relative Measure



Teaser: Differential Privacy

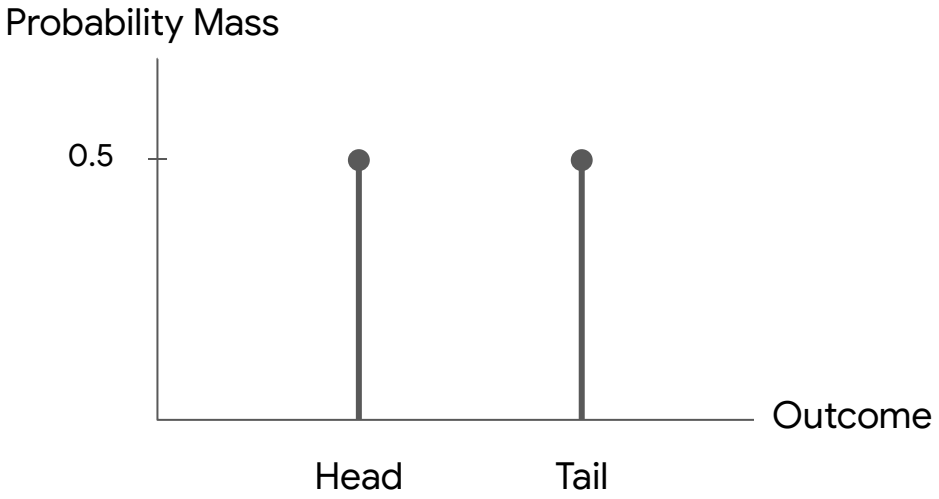


Probability Review I

Probability Review I

Head with probability 0.5

Tail with probability 0.5



Probability Review I

Support(X) = {Head, Tail}

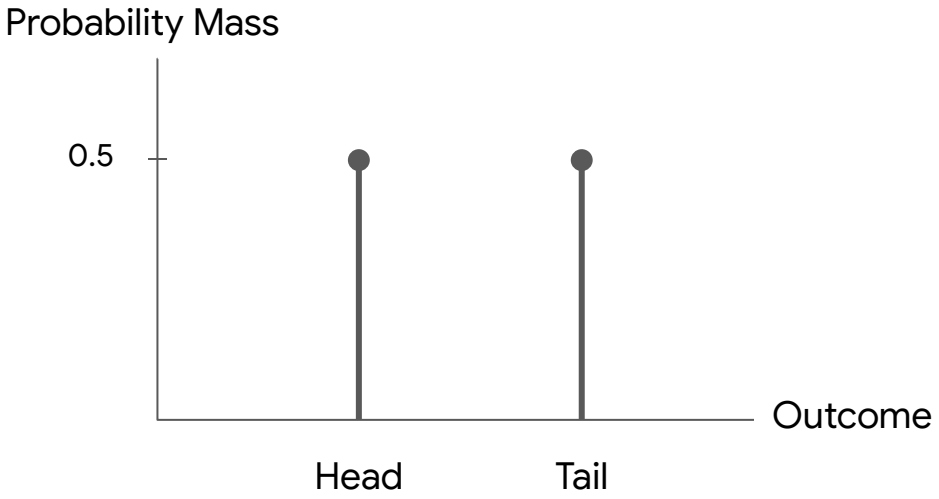
X

Head w.p. 0.5

Tail w.p. 0.5

$$\Pr[X = \text{Head}] = 0.5$$

$$\Pr[X = \text{Tail}] = 0.5$$



Probability Review I

$\text{supp}(X) = \{\text{Head}, \text{Tail}\}$

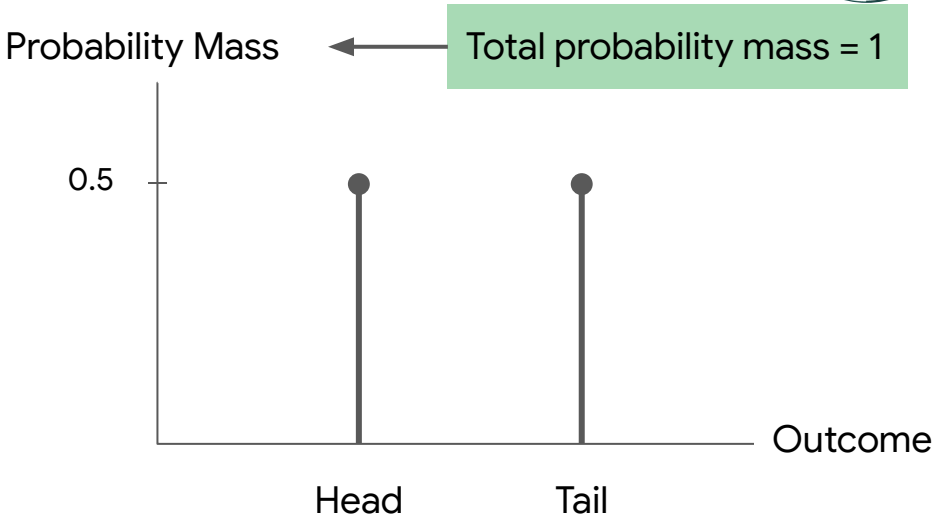
X

Head w.p. 0.5

Tail w.p. 0.5

$\Pr[X = \text{Head}] = 0.5$

$\Pr[X = \text{Tail}] = 0.5$



Probability Review I

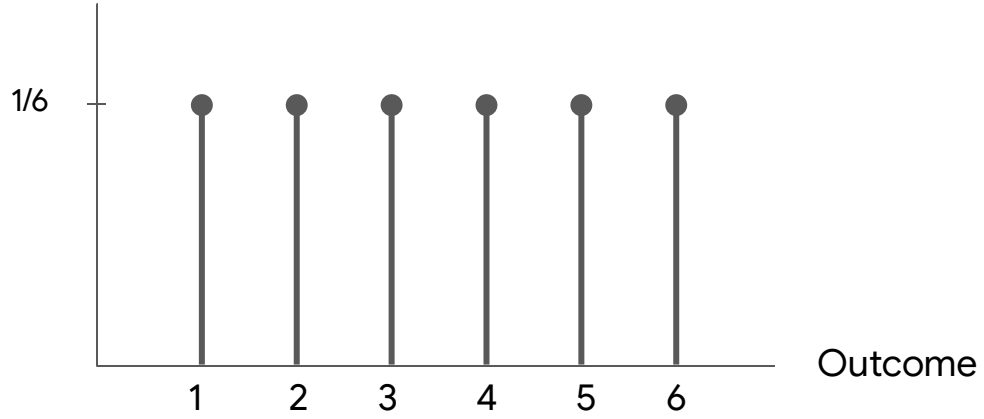
$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/6
2 w.p. 1/6
⋮
6 w.p. 1/6

Probability Mass



Definition Expectation of X is $\mathbf{E}[X] = \sum_{c \in \text{supp}(X)} \Pr[X = c] * c$

$$\mathbf{E}[X] = (1/6) * 1 + \ldots + (1/6) * 6 = 3.5$$

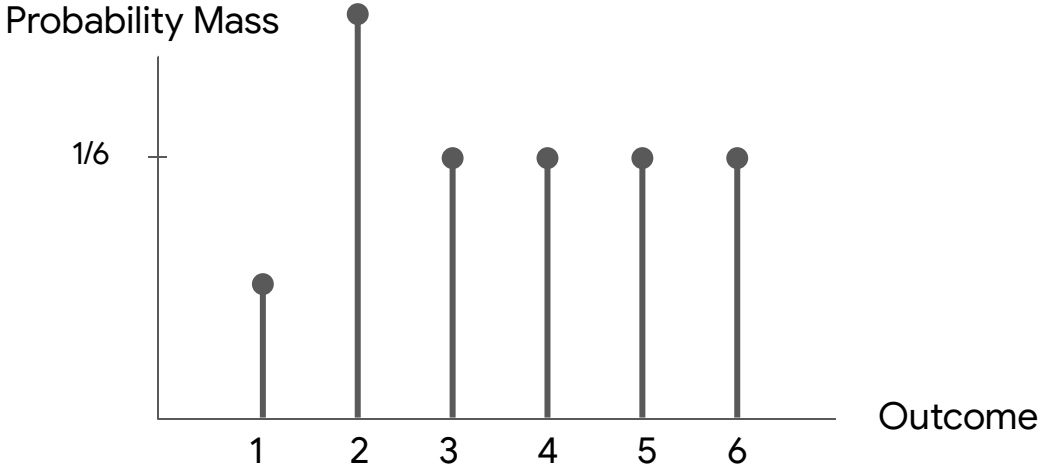
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Definition Expectation of X is $\mathbf{E}[X] = \sum_{c \in \text{supp}(X)} \Pr[X = c] * c$

$$\mathbf{E}[X] = (1/12) * 1 + (1/4) * 2 + \ldots + (1/6) * 6 = 3.5833\ldots$$

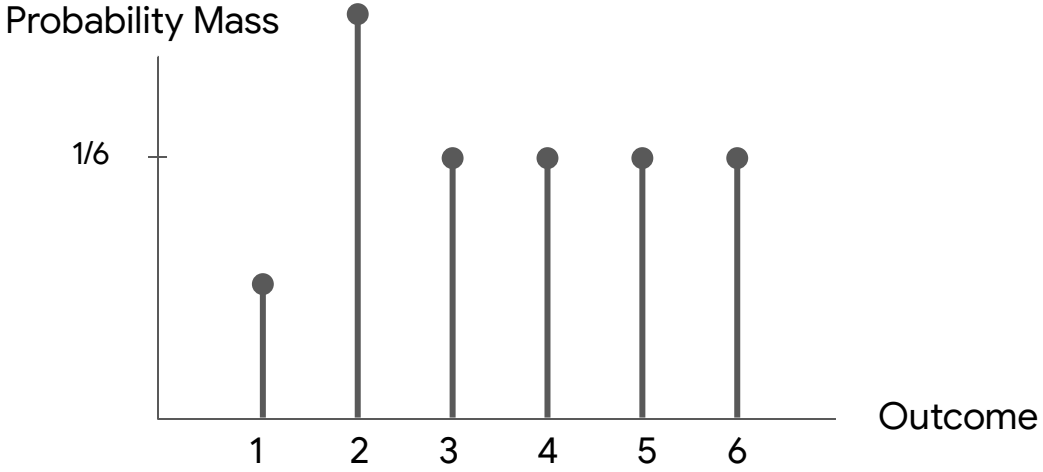
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Definition Expectation of X is $\mathbf{E}[X] = \sum_{c \in \text{supp}(X)} \Pr[X = c] * c$

Definition Variance of X is

$\mathbf{Var}[X] = \mathbf{E}[X^2] - \mathbf{E}[X]^2$

$\text{Var}[X] = 15.4166 - 3.5833^2 = 2.57638...$

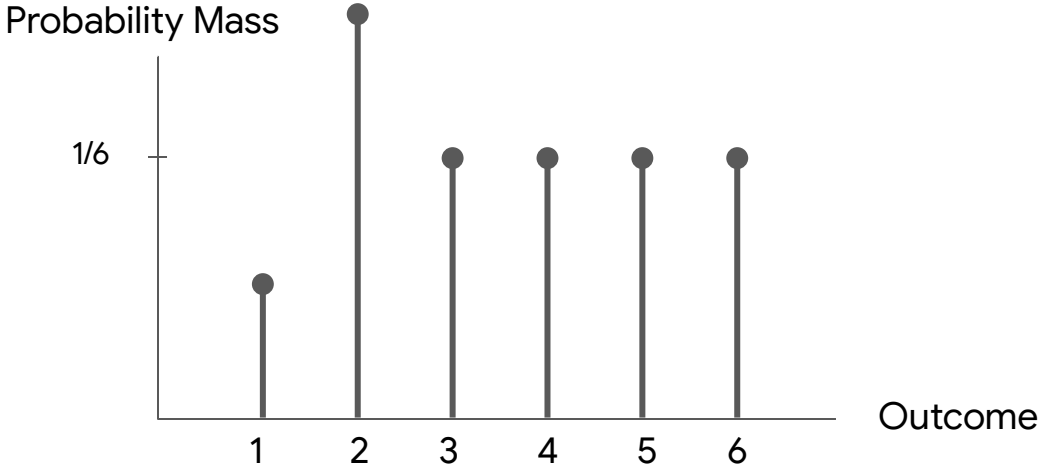
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Definition Expectation of X is $\mathbf{E}[X] = \sum_{c \in \text{supp}(X)} \Pr[X = c] * c$

Definition Variance of X is

$\mathbf{Var}[X] = \mathbf{E}[X^2] - \mathbf{E}[X]^2$

Definition Standard Deviation of X is

$\mathbf{SD}[X] = \sqrt{\mathbf{Var}[X]}$

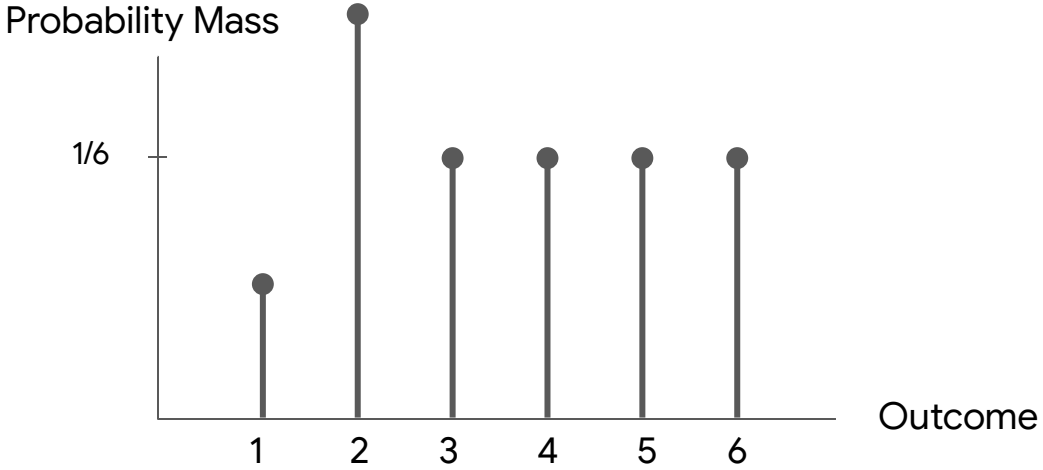
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Event ⟺ *Subset of support*

X *is odd* ⟺ $\{1,3,5\}$

$$Pr[X \text{ is odd}] = Pr[X \in \{1,3,5\}]$$
$$= Pr[X=1] + Pr[X=3] + Pr[X=5]$$
$$= 1/12 + 1/6 + 1/6 = 5/12$$

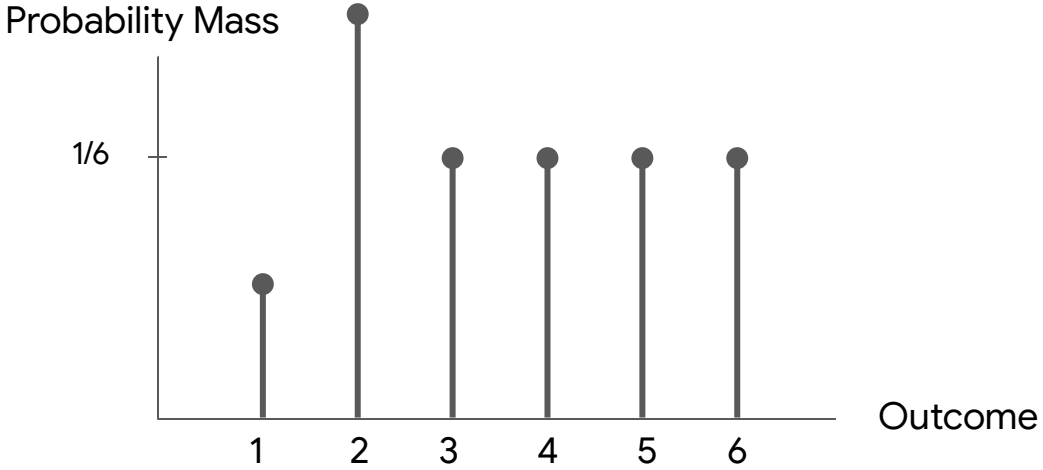
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Event ⟺ *Subset of support*

$X \geq 3 \iff \{3,4,5,6\}$

$$Pr[X \geq 3] = Pr[X \in \{3,4,5,6\}]$$
$$= Pr[X=3] + Pr[X=4] + Pr[X=5] + Pr[X=6]$$
$$= 1/6 + 1/6 + 1/6 + 1/6 = 2/3$$

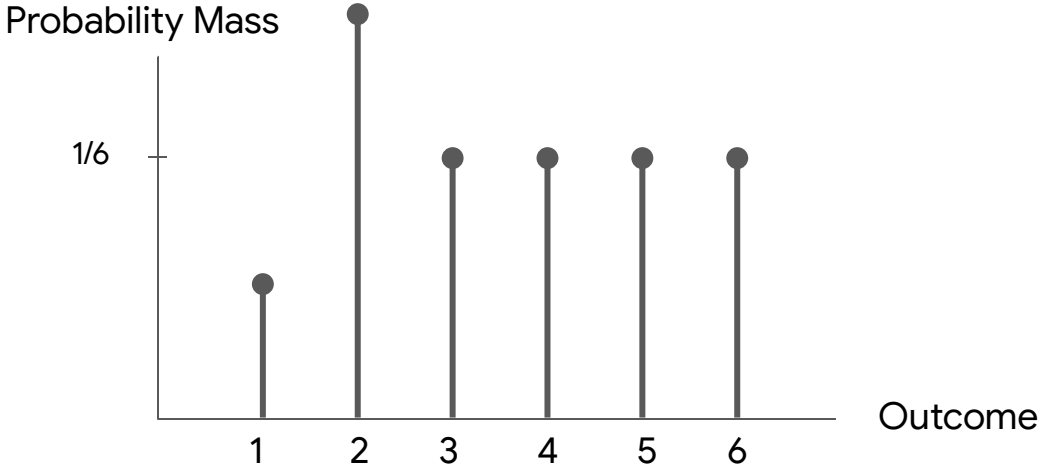
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Definition Cumulative Distribution Function of X is $\mathbf{CDF}_X(c) = \Pr[X \leq c]$

Assumption: $X \geq 0$

$$\mathbf{E}[X] = \int_0^\infty \Pr[X > c]\, d\, c$$

Assumption: X is non-negative integer

$$\mathbf{E}[X] = \sum_{c=0}^{\infty} \Pr[X > c]$$

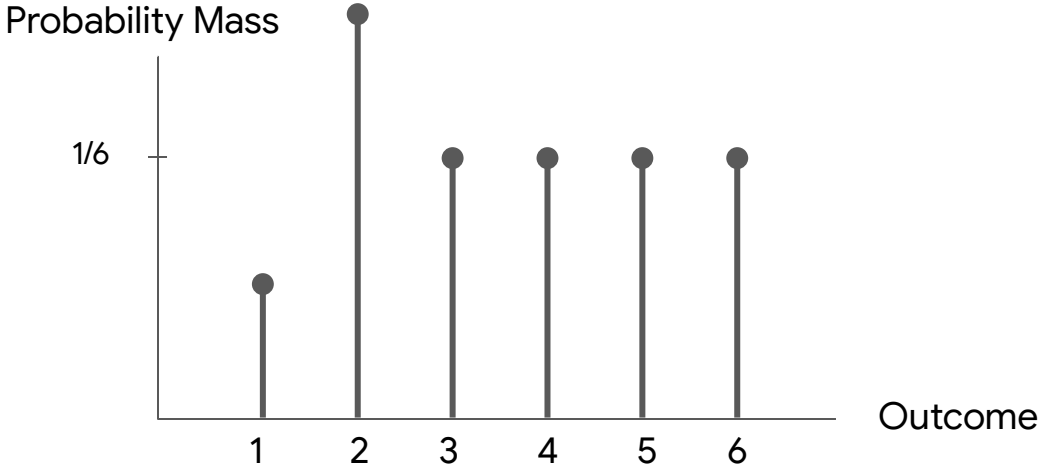
Probability Review I

$\text{supp}(X) = \{1,2,3,4,5,6\}$

Random Variable

X

1 w.p. 1/12
2 w.p. 1/4
⋮
6 w.p. 1/6



Definition Cumulative Distribution Function of X is $\mathbf{CDF}_X(c) = \Pr[X \leq c]$

Assumption: $X \geq 0$

$$\mathbf{E}[X] = \int_0^\infty \Pr[X > c]\, d\, c$$

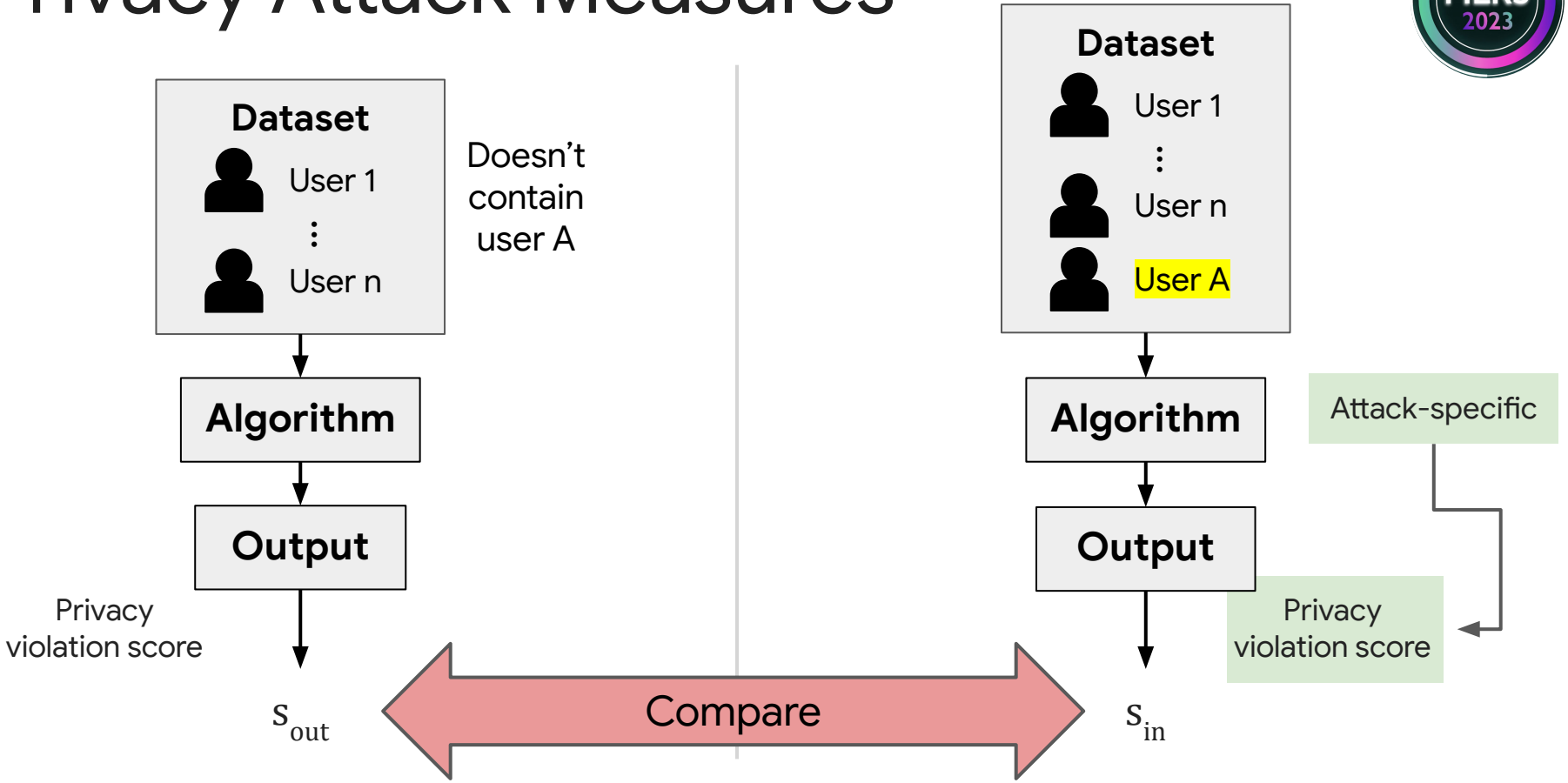
Assumption: X is non-negative integer

$$\mathbf{E}[X] = \sum_{c=0}^{\infty} \Pr[X > c]$$

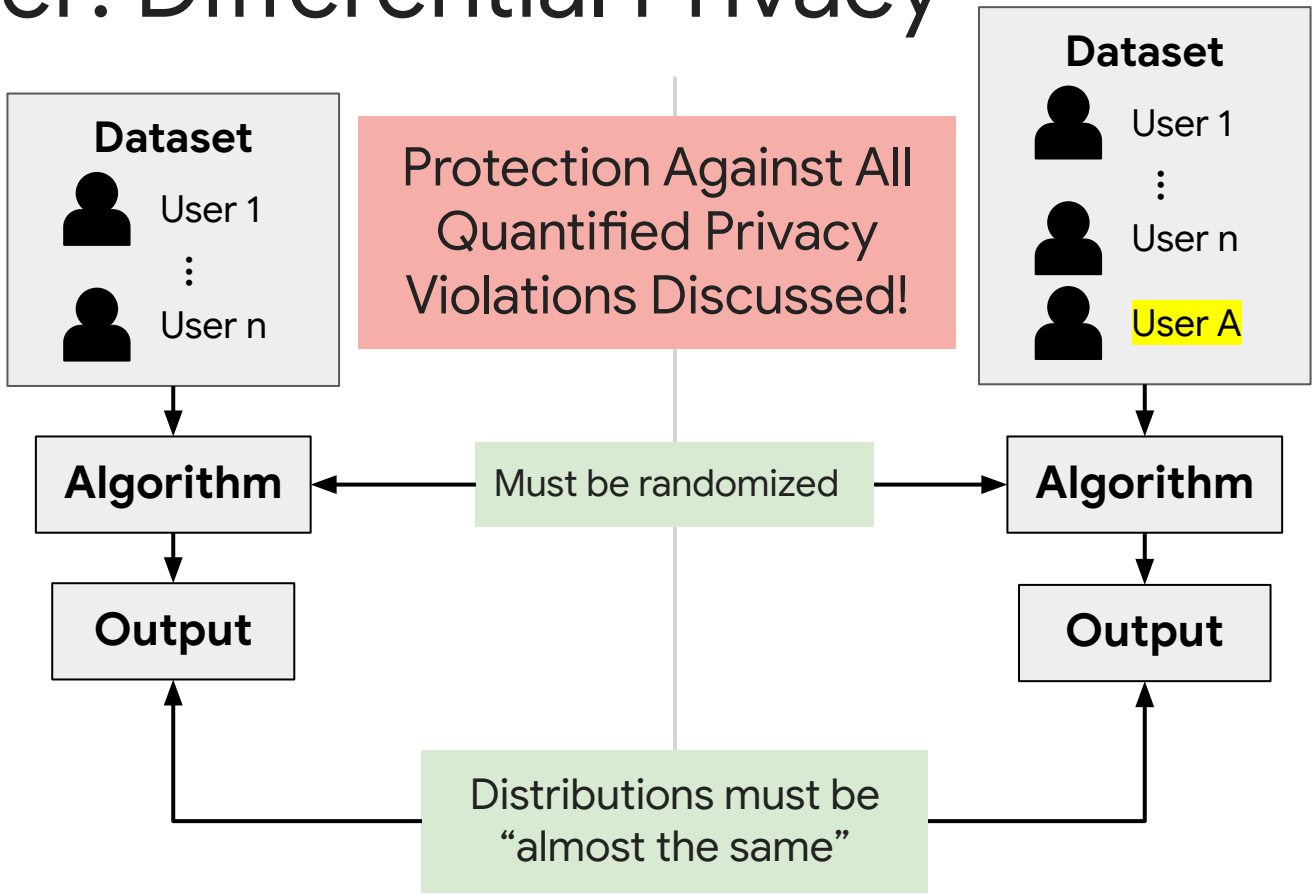
$1 - \mathbf{CDF}_X(c)$

Differential Privacy

Privacy Attack Measures



Teaser: Differential Privacy



Differential Privacy: Definition

Intuition:
"Adding or removing a single user should not change the output distribution too much"

Smaller ε, δ
⇕
More privacy

(ε, δ)-Differential Privacy
[Dwork et al.'06]
For every neighboring datasets X, X' and every set of outputs S,
$\Pr[M(X) \in S] \leq e^{\varepsilon} \cdot \Pr[M(X') \in S] + \delta$

ε = small constant
δ = negligible in # of users

Pure-DP
ε-DP $\equiv$ $(\varepsilon, 0)$-DP

Approx-DP
$\delta > 0$

Differential Privacy: Definition

Intuition:
"Adding or removing a single user should not change the output distribution too much"

Smaller ε, δ
⇕
More privacy

(ε, δ)-Differential Privacy
[Dwork et al.'06]
For every neighboring datasets X, X' and every set of outputs S,
$\Pr[M(X) \in S] \leq e^{\varepsilon} \cdot \Pr[M(X') \in S] + \delta$

Pure-DP
ε-DP ≡ $(\varepsilon, 0)$-DP

Approx-DP
$\delta > 0$

- "neighboring" notion can be broader than intuition
- $\asymp$: abbrv for neighboring

ε = small constant
δ = negligible in # of users

Differential Privacy: Neighboring Notions

Add/remove-DP

$\mathbf{X} \simeq^{r} \mathbf{X}'$ $\Longleftrightarrow$ $\mathbf{X}$ is $\mathbf{X}'$ with an individual's data added / removed

Dataset $\mathbf{X}$

	name	zipcode	age	income
$\mathbf{x}_1$	Dale	10520	40	150k
$\mathbf{x}_2$	Bob	10520	35	50k
$\mathbf{x}_3$	Conor	10500	30	30k

Dataset $\mathbf{X}'$

	name	zipcode	age	income
$\mathbf{x}_1$	Dale	10520	40	150k
$\mathbf{x}_2$	Bob	10520	35	50k

Differential Privacy: Neighboring Notions

Substitution-DP

$X \asymp^s X'$ ⟺ X is X' with an individual's data changed

Dataset X

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	30k

Dataset X'

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10520	32	150k

Differential Privacy: Neighboring Notions

Add/remove-DP

$X \asymp^r X'$ ⟺ X is X' with an individual's data added / removed

- Protect against Membership Inference
- Cannot reveal the raw size of the dataset
  - Sometimes make it harder to design & analyze algorithms for

Substitution-DP

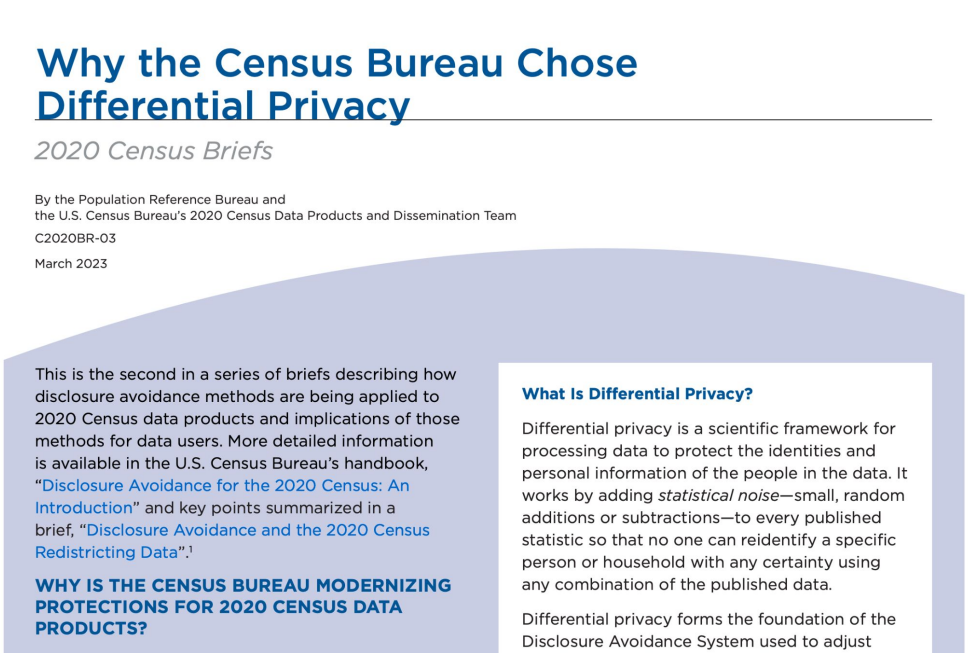
$X \asymp^s X'$ ⟺ X is X' with an individual's data changed

- Does *not* Protect against Membership Inference
- Can reveal the raw size of the dataset

In this tutorial, if not stated,
assume ***substitution-DP***

Differential Privacy Deployments

US Census



Why the Census Bureau Chose Differential Privacy

2020 Census Briefs

By the Population Reference Bureau and
the U.S. Census Bureau's 2020 Census Data Products and Dissemination Team

C2020BR-03

March 2023

This is the second in a series of briefs describing how disclosure avoidance methods are being applied to 2020 Census data products and implications of those methods for data users. More detailed information is available in the U.S. Census Bureau's handbook, "Disclosure Avoidance for the 2020 Census: An Introduction" and key points summarized in a brief, "Disclosure Avoidance and the 2020 Census Redistricting Data".[1]

WHY IS THE CENSUS BUREAU MODERNIZING PROTECTIONS FOR 2020 CENSUS DATA PRODUCTS?

What Is Differential Privacy?

Differential privacy is a scientific framework for processing data to protect the identities and personal information of the people in the data. It works by adding *statistical noise*—small, random additions or subtractions—to every published statistic so that no one can reidentify a specific person or household with any certainty using any combination of the published data.

Differential privacy forms the foundation of the Disclosure Avoidance System used to adjust

Google Chrome

Chromium Blog

News and developments from the open source browser project

Learning Statistics with Privacy, aided by the Flip of a Coin

Thursday, October 30, 2014

[Cross-posted on the Google Research Blog and the Google Online Security Blog]

At Google, we are constantly trying to improve the techniques we use to protect our users' security and privacy. One such project, RAPPOR (Randomized Aggregatable Privacy-Preserving Ordinal Response), provides a new state-of-the-art, privacy-preserving way to learn software statistics that we can use to better safeguard our users' security, find bugs, and improve the overall user experience.

Building on the concept of randomized response, RAPPOR enables learning statistics about the behavior of users' software while guaranteeing client privacy. The guarantees of differential privacy, which are widely accepted as being the strongest form of privacy, have almost never been used in practice despite intense research in academia. RAPPOR introduces a practical method to achieve those guarantees.

Differential Privacy Deployments (cont.)

Apple Photos

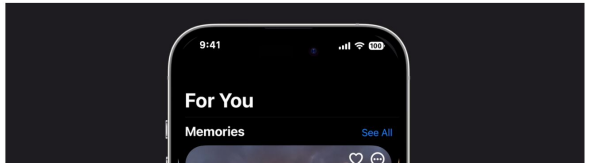
Article | July 2023

Computer Vision, Privacy

Learning Iconic Scenes with Differential Privacy

In this article, we share how we apply differential privacy (DP) to learn about the kinds of photos people take at frequently visited locations (iconic scenes) without personally identifiable data leaving their device. This approach is used in several features in Photos, including choosing key photos for Memories, › and selecting key photos for locations in Places in iOS 17.

The Photos app learns about significant people, places, and events based on the user's library, and then presents Memories: curated collections of photos and videos set to music. The key photo for a Memory is influenced by the popularity of iconic scenes learned from iOS users—with DP assurance.



Google Gboard

Federated Learning with Formal Differential Privacy Guarantees

MONDAY, FEBRUARY 28, 2022

Posted by Brendan McMahan and Abhradeep Thakurta, Research Scientists, Google Research

In 2017, Google introduced federated learning (FL), an approach that enables mobile devices to collaboratively train machine learning (ML) models while keeping the raw training data on each user's device, decoupling the ability to do ML from the need to store the data in the cloud. Since its introduction, Google has continued to actively engage in FL research and deployed FL to power many features in Gboard, including next word prediction, emoji suggestion and out-of-vocabulary word discovery. Federated learning is improving the "Hey Google" detection models in Assistant, suggesting replies in Google Messages, predicting text selections, and more.

While FL allows ML without raw data collection, differential privacy (DP) provides a quantifiable measure of data anonymization, and when applied to ML can address concerns about models memorizing sensitive user data. This too has been a top research priority, and has yielded one of the first production uses of DP for analytics with RAPPOR in 2014, our open-source DP library, Pipeline DP, and TensorFlow Privacy.

Through a multi-year, multi-team effort spanning fundamental research and product integration, today we are excited to announce that we have deployed a production ML model using federated learning with a rigorous differential privacy guarantee. For this proof-of-concept deployment, we utilized the DP-FTRL algorithm to train a recurrent neural network to power next-word-prediction for Spanish-language Gboard users. To our knowledge, this is the first production neural network trained directly on user data announced with a formal DP guarantee (technically ρ=0.81 zero-Concentrated-Differential-Privacy, zCDP, discussed in detail below). Further, the federated approach offers complimentary data

Differential Privacy: Definition

Intuition:
"Adding or removing a single user should not change the output distribution too much"

Smaller ε, δ
⇕
More privacy

(ε, δ)-Differential Privacy
[Dwork et al.'06]
For every neighboring datasets X, X' and every set of outputs S,
$\Pr[M(X) \in S] \leq e^{\varepsilon} \cdot \Pr[M(X') \in S] + \delta$

ε = small constant
δ = negligible in # of users

Pure-DP
ε-DP $\equiv$ $(\varepsilon, 0)$-DP

Approx-DP
$\delta > 0$

Privacy Loss (Discrete)

Privacy Loss

For $\mathbf{M, X, X'}$, the privacy loss at output $\mathbf{o}$ is

$$\mathbf{L_{M,X,X'}(o) = \ln(Pr[M(X) = o] \,/\, Pr[M(X') = o])}$$

If-and-only-if condition

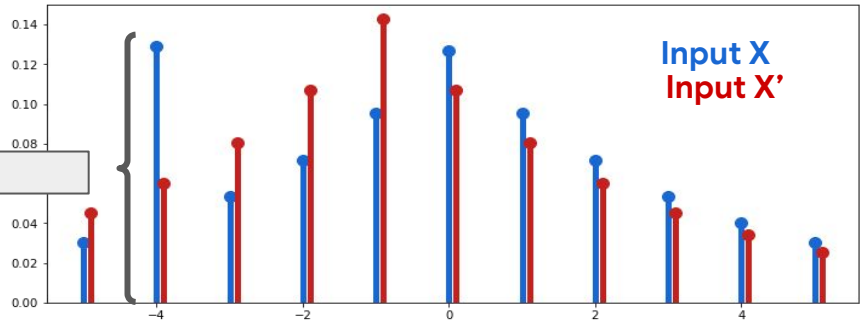
Theorem (Pure-DP Condition)

$\mathbf{M}$ is ε-DP iff, for all $\mathbf{X \asymp X'}$, $\mathbf{o \in Range(M)}$, $\mathbf{L_{M,X,X'}(o) \leq \varepsilon}$

Probability Mass

Ratio > exp(0.5)

Not 0.5-DP!



Privacy Loss (Discrete)

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

If-and-only-if condition

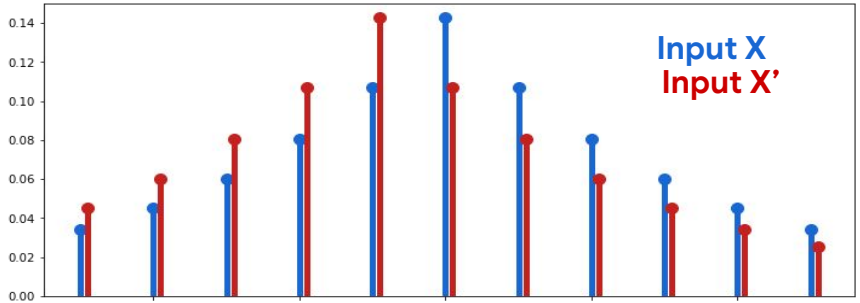
Theorem (Pure-DP Condition)

M is ε-DP iff, for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$

Probability Mass

Ratio ≤ exp(0.5)

✓ 0.5-DP!



Privacy Loss (Discrete)

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

If-and-only-if condition

Theorem (Pure-DP Condition)

M is ε-DP iff, for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$

Proof ($\Rightarrow$) Suppose that M is ε-DP. Consider any $X \asymp X'$, $o \in \text{Range}(M)$.

$$\begin{aligned} \Pr[M(X)=o] &= \Pr[M(X)\in\{o\}] \\ &\leq e^{\varepsilon} \cdot \Pr[M(X') \in \{o\}] \\ &= e^{\varepsilon} \cdot \Pr[M(X') = o] \end{aligned}$$

$\Rightarrow$ $L_{M,X,X'}(o) \leq \varepsilon$ **QED**

Privacy Loss (Discrete)

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

If-and-only-if condition

Theorem (Pure-DP Condition)

M is ε-DP iff, for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$

Proof (⇐) Suppose that for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$.

Consider any $S \subseteq \text{Range}(M)$.

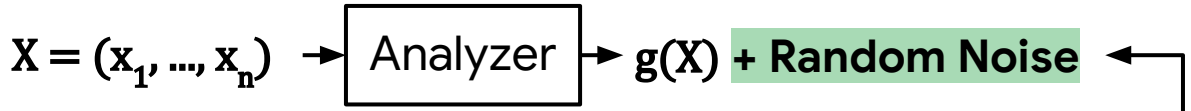
$$\begin{aligned}\Pr[M(X) \in S] &= \sum_{o \in S} \Pr[M(X) = o] \\ &\leq \sum_{o \in S} e^{\varepsilon} \cdot \Pr[M(X') = o] \\ &= e^{\varepsilon} \cdot \Pr[M(X') \in S]\end{aligned}$$

QED

Basic Mechanism: Noise Addition

Noise Addition Mechanism

Noise Addition Mechanism

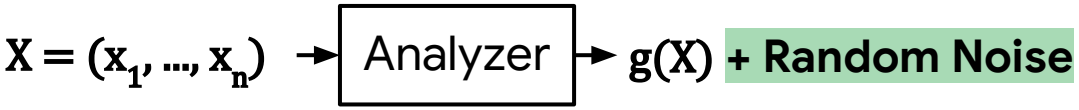


Intuition:
"Noise should be large enough to hide a user's contribution"

What noise distribution should we use?

- Depends on $\mathrm{Range}(g)$
- Depends on how "sensitive" g is

Discrete Laplace Mechanism



Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \asymp X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

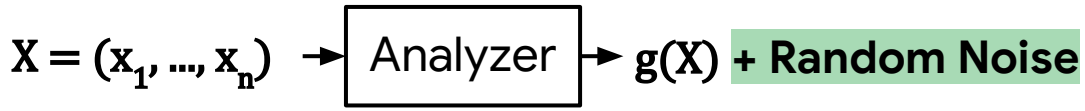
Examples: COUNT Query

$g = \text{COUNT}(*)\ \text{WHERE income} > 40\text{k}$

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	30k

$g(X) = 2$

Discrete Laplace Mechanism



Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \asymp X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Examples: COUNT Query

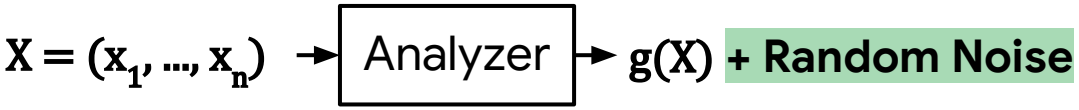
$g = \text{COUNT}(*)\ \text{WHERE income} > 40\text{k}$

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	60k

$g(X) = 3$

$\Delta(g) = 1$

Discrete Laplace Mechanism



Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \asymp X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

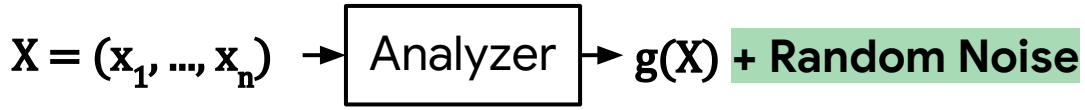
Examples: SUM Query

$$g = \text{SUM}(\text{income})$$

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	60k

$g(X) = 260k$

Discrete Laplace Mechanism



Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Examples: SUM Query

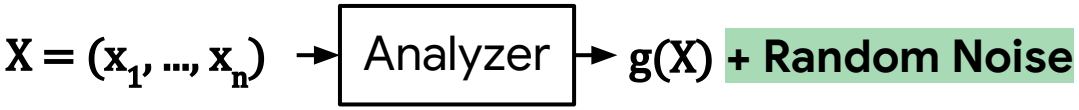
$$g = \text{SUM}(\text{income})$$

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	190k

$g(X) = 390k$

Assume income $\leq c$ ⇒ $\Delta(g) \leq c$

Discrete Laplace Mechanism



Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Examples: AVERAGE Query

$$g = \text{AVG}(\text{income})$$

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
x_3	Conor	10500	30	190k

$g(X) = 130k$

Assume income $\leq c$ ⇒ $\Delta(g) \leq c/n$

Discrete Laplace Mechanism

$X = (x_1, ..., x_n)$ → Analyzer → $g(X)$ + **Random Noise**

Assumption: $\text{Range}(g) \subseteq \mathbb{Z}$

(i.e. g is integer-valued)

Sensitivity

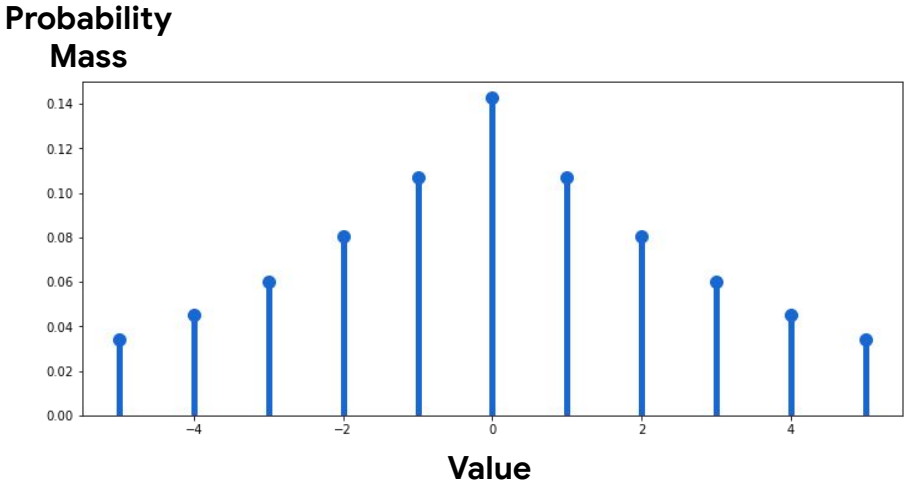
$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Discrete Laplace Distribution

For every integer i,

$\Pr[i = \text{DLap}(b)] \propto e^{-|i|/b}$



Discrete Laplace Mechanism

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathrm{DLap}(\Delta(g)/\varepsilon)$$

Assumption: $\mathrm{Range}(g) \subseteq \mathbb{Z}$

(i.e. g is integer-valued)

Sensitivity

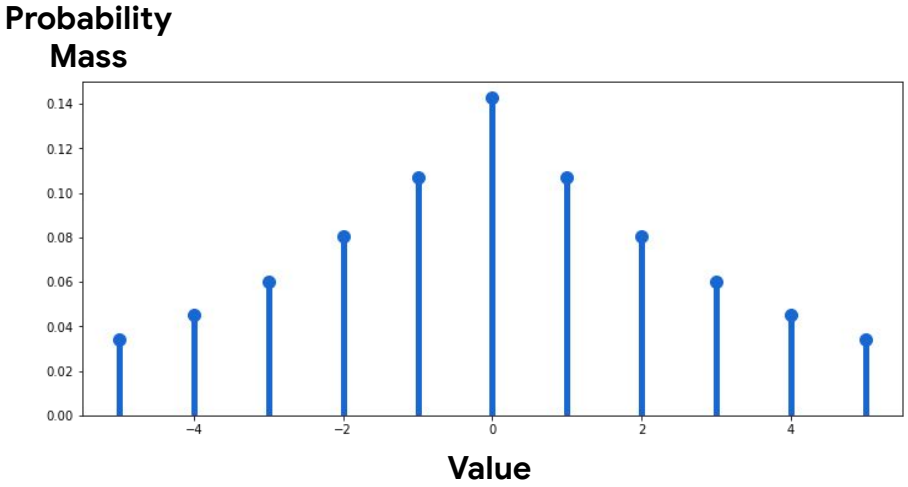
$$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$$

Larger sensitivity ⇒
More Noise Required

Discrete Laplace Distribution

For every integer i,

$$\Pr[i = \mathrm{DLap}(b)] \propto e^{-|i|/b}$$



Discrete Laplace Mechanism

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \text{DLap}(\Delta(g)/\varepsilon)$$

Example:

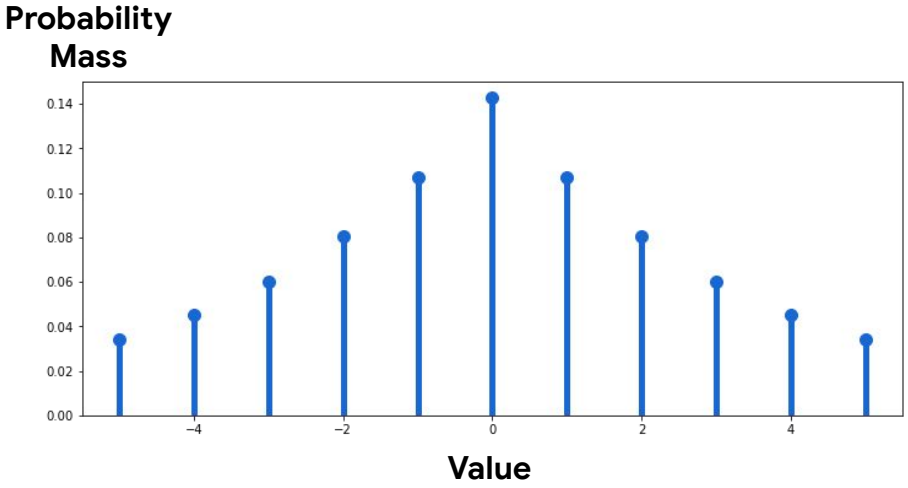
$\Delta(g) = 1, \quad \varepsilon = \ln(4/3), \quad g(X) = 3$

Output: $3 + \text{DLap}(1/\ln(4/3))$

Discrete Laplace Distribution

For every integer i,

$$\Pr[i = \text{DLap}(b)] \propto e^{-|i|/b}$$



Discrete Laplace Mechanism

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathrm{DLap}(\Delta(g)/\varepsilon)$$

Example:

$\Delta(g) = 1, \quad \varepsilon = \ln(4/3), \quad g(X) = 3$

Output: $3 + \mathrm{DLap}(1/\ln(4/3))$

output =

$\vdots$

2 w.p. 3/28 = 0.1071...

3 w.p. 1/7 = 0.1428...

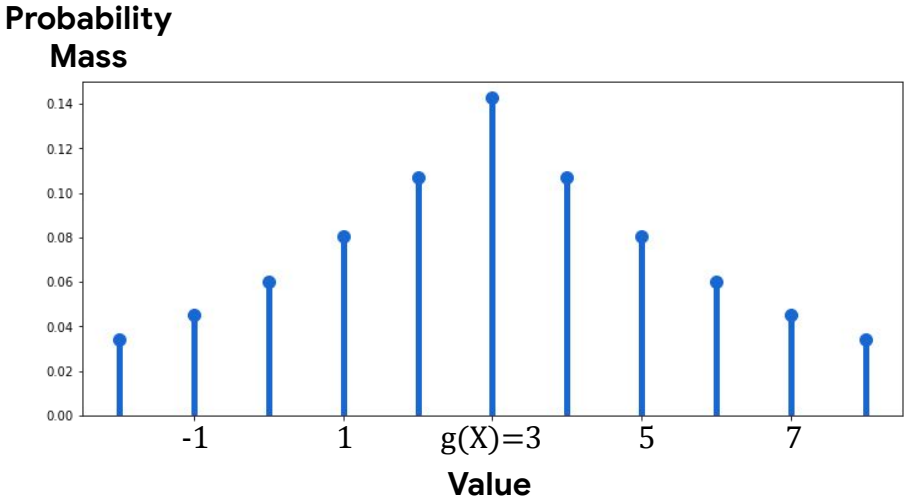
4 w.p. 3/28 = 0.1071...

$\vdots$

Discrete Laplace Distribution

For every integer i,

$$\Pr[i = \mathrm{DLap}(b)] \propto e^{-|i|/b}$$



Discrete Laplace Mechanism: Privacy

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbb{Z}$, Discrete Laplace Mechanism is ε-DP.

Privacy Loss (Discrete)

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

If-and-only-if condition

Theorem (Pure-DP Condition)

M is ε-DP iff, for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$

Discrete Laplace Mechanism: Privacy

$X = (x_1, \ldots, x_n) \rightarrow$ Analyzer $\rightarrow g(X) + \mathrm{DLap}(\Delta(g)/\varepsilon)$

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbb{Z}$, Discrete Laplace Mechanism is ε-DP.

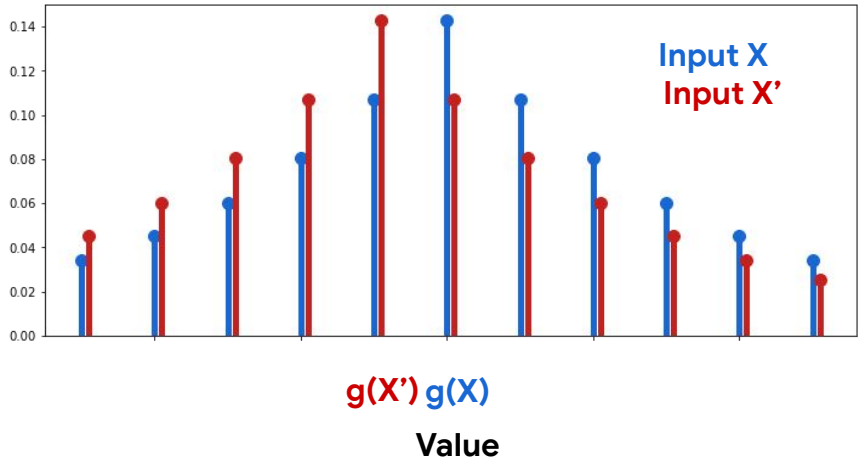
Proof Let $b = \Delta(g)/\varepsilon$.

For all $X \simeq X'$, $o \in \mathrm{Range}(M)$,

$$
\begin{aligned}
& L_{M,X,X'}(o) \\
= \;& \ln(\Pr[M(X) = o] \,/\, \Pr[M(X') = o]) \\
= \;& \ln(\exp(-|o - g(X)| \,/\, b) \,/\, \exp(-|o - g(X')| \,/\, b)) \\
= \;& (|o - g(X')| - |o - g(X)|) \,/\, b \\
\le \;& |g(X) - g(X')| \,/\, b \qquad \text{(triangle ineq.)} \\
\le \;& \Delta(g) \,/\, b = \varepsilon
\end{aligned}
$$

QED

Illustration for $\Delta(g) = 1$:



Discrete Laplace Mechanism: Utility

Theorem Discrete Laplace Mechanism has RMSE = $O(\Delta(g)/\varepsilon)$

Utility Measures

- ***Mean Square Error (MSE)***: $\mathbf{E}[(\text{output} - \text{true})^2]$
  - ***Root Mean Square Error (RMSE)***: $\sqrt{\text{MSE}}$
- ***Mean Absolute Error (MAE)***: $\mathbf{E}[|\text{output} - \text{true}|]$

- In this tutorial, will mostly compute RMSE since it is easier to deal with
- MAE is always ≤ RMSE

Discrete Laplace Mechanism: Utility

Theorem Discrete Laplace Mechanism has RMSE = $O(\Delta(g)/\varepsilon)$

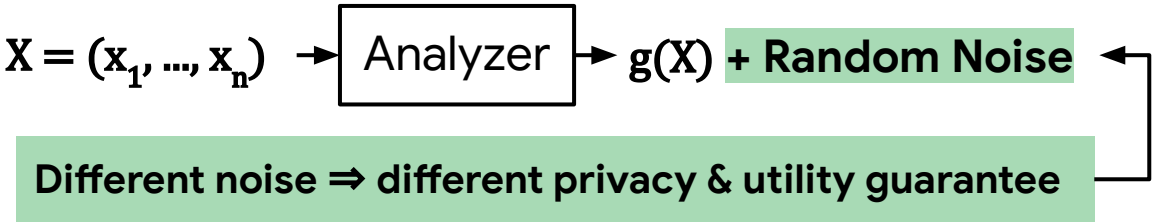
Error does not scale with dataset size!

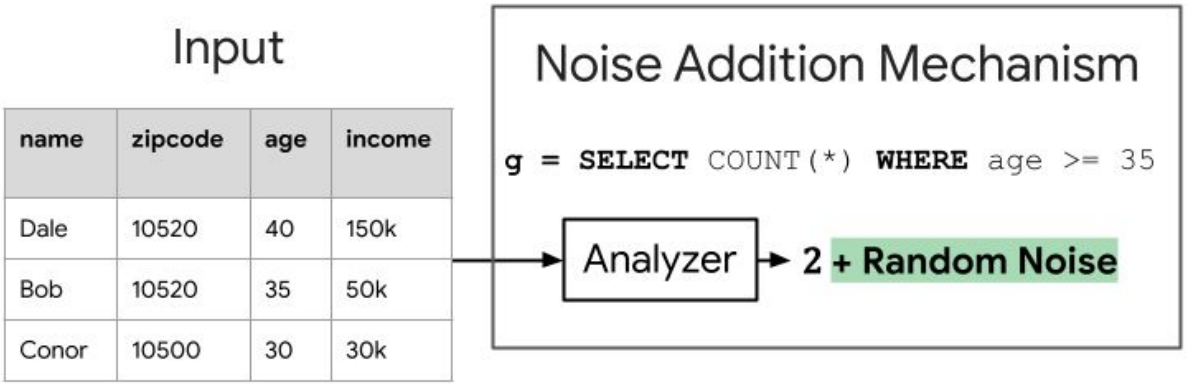
Utility Measures

- ***Mean Square Error (MSE)***: $\mathbf{E}$[(output - true)^2]
  - ***Root Mean Square Error (RMSE)***: √MSE
- ***Mean Absolute Error (MAE)***: $\mathbf{E}$[|output - true|]

- In this tutorial, will mostly compute RMSE since it is easier to deal with
- MAE is always ≤ RMSE

Noise Addition Mechanism





* The random noise has to be hidden from the adversary

Discrete Laplace Mechanism

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathrm{DLap}(\Delta(g)/\varepsilon)$$

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbf{Z}$, Discrete Laplace Mechanism is ε-DP.

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

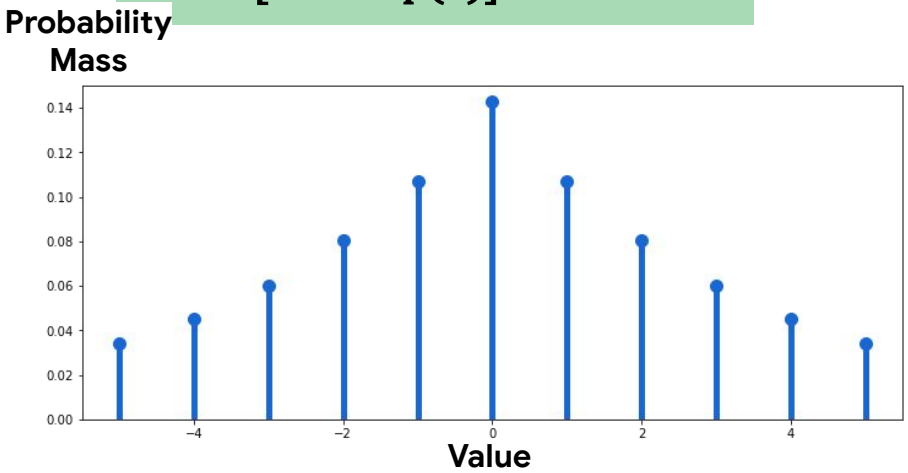
$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Discrete Laplace Distribution

For every integer i,

$$\Pr[i = \mathrm{DLap}(b)] \propto e^{-|i|/b}$$



Discrete Laplace: Multi-Dimension

Assumption: $\text{Range}(g) \subseteq \mathbf{Z}^d$

(i.e. g is vector-valued with integer entries)

ℓ_p-Sensitivity

$\Delta_p(g) = \max_{X \simeq X'} ||g(X) - g(X')||_p$

ℓ_p-Norm

$||v||_p = (|v_1|^p + ... + |v_d|^p)^{1/p}$

Examples: Histogram

	name	zipcode	age	income
x_1	Dale	10520	40	150k
x_2	Bob	10520	35	50k
	...			
x_n	Conor	10500	30	30k

Discrete Laplace: Multi-Dimension

$\mathbf{X = (x_1, ..., x_n)}$ → Analyzer → $\mathbf{g(X) + DLap(\Delta(g)/\varepsilon)}$

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{Z}^d$

(i.e. g is vector-valued with integer entries)

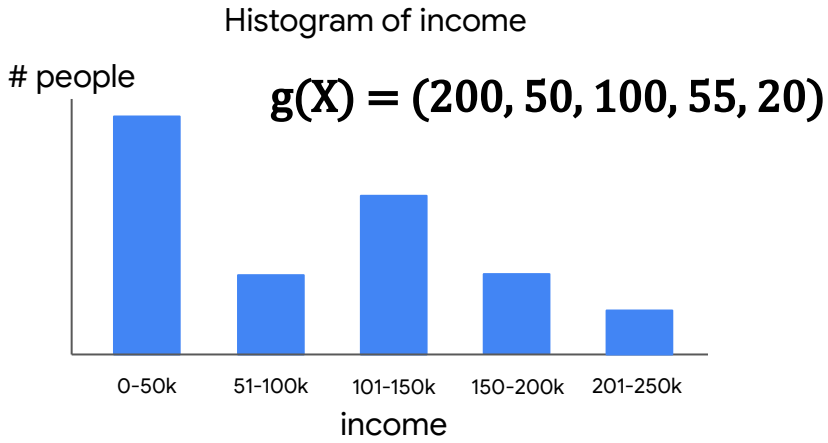
ℓ_p-Sensitivity

$\Delta_p(g) = \max_{X \asymp X'} ||g(X) - g(X')||_p$

ℓ_p-Norm

$||v||_p = (|v_1|^p + ... + |v_d|^p)^{1/p}$

Examples: Histogram



Discrete Laplace: Multi-Dimension

$X = (x_1, ..., x_n)$ → Analyzer → $g(X) + \mathbf{DLap}(\Delta(g)/\varepsilon)$

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{Z}^d$

(i.e. g is vector-valued with integer entries)

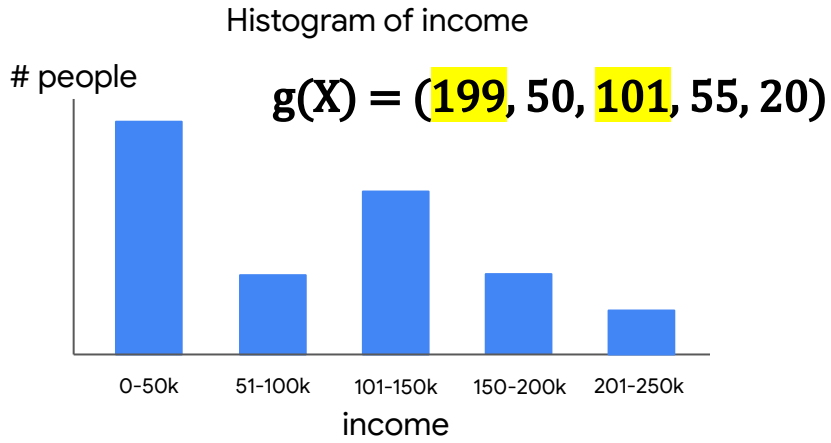
ℓ_p-Sensitivity

$\Delta_p(g) = \max_{X \simeq X'} ||g(X) - g(X')||_p$

ℓ_p-Norm

$||v||_p = (|v_1|^p + ... + |v_d|^p)^{1/p}$

Examples: Histogram



Changing a row effects two buckets by ≤ 1 ⇒ $\Delta_1(g) \leq 2$

Discrete Laplace: Multi-Dimension

$X = (x_1, ..., x_n)$ → Analyzer → $g(X) + \mathbf{DLap}(\Delta_1(g)/\varepsilon)^{\otimes d}$

- Use ℓ_1-sensitivity
- Each coordinate is independent

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{Z}^d$

(i.e. g is vector-valued with integer entries)

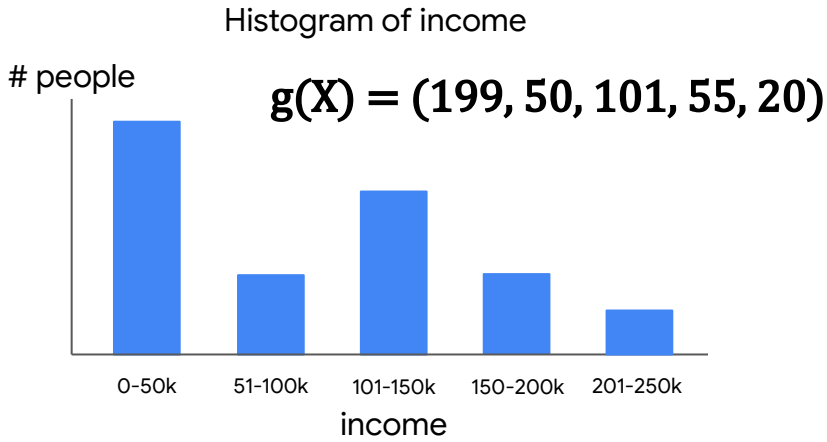
ℓ_p-Sensitivity

$\Delta_p(g) = \max_{X \simeq X'} ||g(X) - g(X')||_p$

ℓ_p-Norm

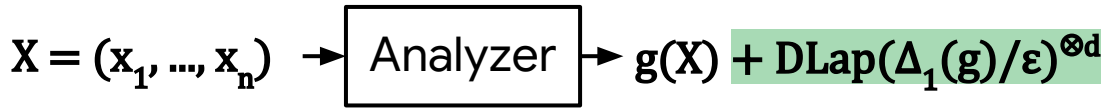
$||v||_p = (|v_1|^p + ... + |v_d|^p)^{1/p}$

Examples: Histogram



Changing a row effects two buckets by ≤ 1 ⇒ $\Delta_1(g) \leq 2$

Discrete Laplace: Multi-Dimension



Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbf{Z}^d$, Discrete Laplace Mechanism is ε-DP.

Proof Let $b = \Delta_1(g)/\varepsilon$.

For all $X \simeq X'$, $o \in \mathrm{Range}(M)$,

$$
\begin{aligned}
L_{M,X,X'}(o) &= \ln(\Pr[M(X) = o] / \Pr[M(X') = o]) \\
&= \ln(\textstyle\prod_{i\in[d]} \exp(-|o_i - g(X)_i| / b) / \exp(-|o_i - g(X')_i| / b) \\
&= \textstyle\sum_{i\in[d]} (|o_i - g(X')_i| - |o_i - g(X)_i|) / b \\
&\le \textstyle\sum_{i\in[d]} |g(X)_i - g(X')_i| / b \qquad \text{(triangle ineq.)} \\
&= ||g(X) - g(X')||_1 \\
&\le \Delta_1(g) / b \quad = \varepsilon
\end{aligned}
$$

QED

Discrete Laplace Mechanism

Theorem Assuming $\text{Range}(g) \subseteq \mathbf{Z}$, Discrete Laplace Mechanism is ε-DP.

Assumption: $\text{Range}(g) \subseteq \mathbf{Z}$

(i.e. g is integer-valued)

Sensitivity

$\Delta(g) = \max_{X \simeq X'} |g(X) - g(X')|$

Larger sensitivity ⇒
More Noise Required

Discrete Laplace Distribution

For every integer i,

$$\Pr[i = DLap(b)] \propto e^{-|i|/b}$$

- Necessary!
- If $g(X) = 0.1, g(X') = 0.2$
  - Noise added is integer
  - Doesn't change the fractional part
  - Adversary can tell exactly which dataset it comes from

Probability Review II

Probability Review II

supp(X) = [0,1]

Random Variable

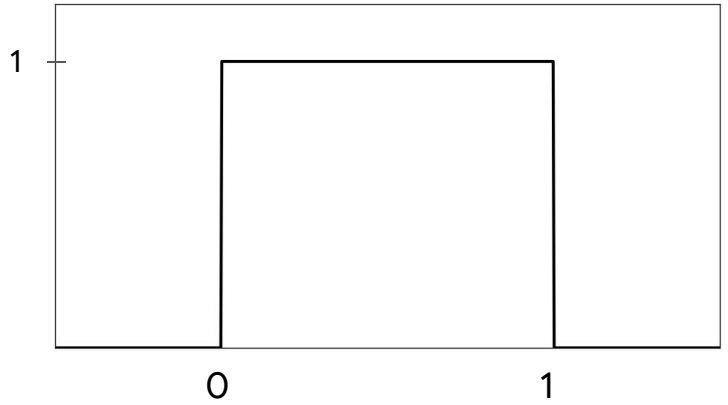
X = random number in $[0, 1]$

$$f_X(c) \begin{cases} 1 & \text{if c is in } [0, 1] \\ 0 & \text{otherwise} \end{cases}$$

Identity

$$\Pr[a \leq X \leq b] = \int_a^b f_X(c)\, dc$$

Probability Density



Notation $f_X(c)$ = Probability Density at c

Probability Review II

supp(X) = [0,1]

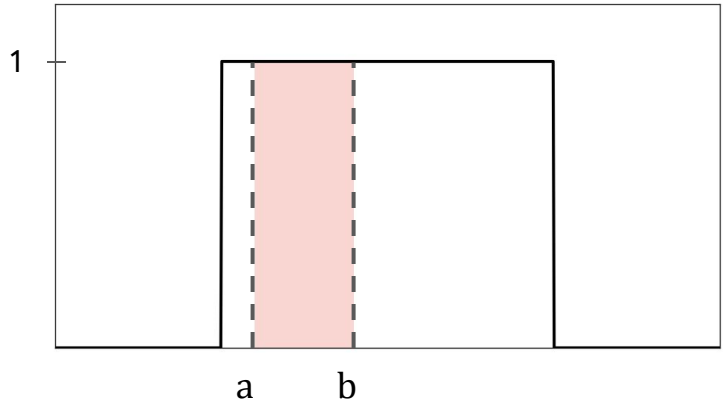
Random Variable

X = random number in $[0, 1]$

$$f_X(c) \begin{cases} 1 & \text{if c is in } [0, 1] \\ 0 & \text{otherwise} \end{cases}$$

Probability Density

$a = 0.1, b = 0.3$



Identity

$$\Pr[a \le X \le b] = \int_a^b f_X(c)\, dc$$

Notation $f_X(c)$ = Probability Density at c

Identity For all *"measurable"* S, $\Pr[X \in S] = \int_S f_X(c)\, dc$

Probability Review II

supp(X) = [0,1]

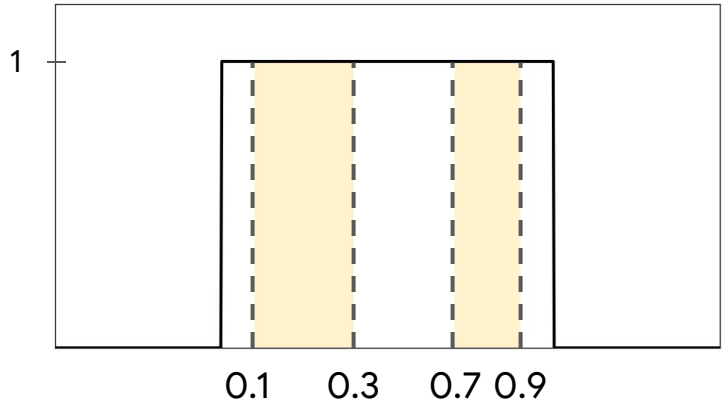
Random Variable

X = random number in $[0, 1]$

$$f_X(c) \begin{cases} 1 & \text{if c is in } [0, 1] \\ 0 & \text{otherwise} \end{cases}$$

Probability Density

$S = [0.1, 0.3] \cup [0.7, 0.9]$



Identity

$$\Pr[a \le X \le b] = \int_a^b f_X(c)\, dc$$

Notation $f_X(c)$ = Probability Density at c

Identity For all *"measurable"* S, $\Pr[X \in S] = \int_S f_X(c)\, dc$

Privacy Loss

Discrete outputs

Privacy Loss

For $\mathbf{M}, \mathbf{X}, \mathbf{X'}$, the privacy loss at output $\mathbf{o}$ is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

Continuous outputs

Privacy Loss

For $\mathbf{M}, \mathbf{X}, \mathbf{X'}$, the privacy loss at output $\mathbf{o}$ is

$$L_{M,X,X'}(o) = \ln(f_{M(X)}(o) / f_{M(X')}(o))$$

Privacy Loss

Discrete outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] \,/\, \Pr[M(X') = o])$$

"Ration of Probability Masses"

Continuous outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(f_{M(X)}(o) \,/\, f_{M(X')}(o))$$

"Ratio of Probability Densities"

Theorem (Pure-DP Condition)

M is ε-DP iff, for all $X \asymp X'$, $o \in \text{Range}(M)$, $L_{M,X,X'}(o) \leq \varepsilon$

Laplace Mechanism

$X = (x_1, ..., x_n)$ → Analyzer → $g(X)$ + **Random Noise**

Assumption: $\text{Range}(g) \subseteq \mathbf{R}$

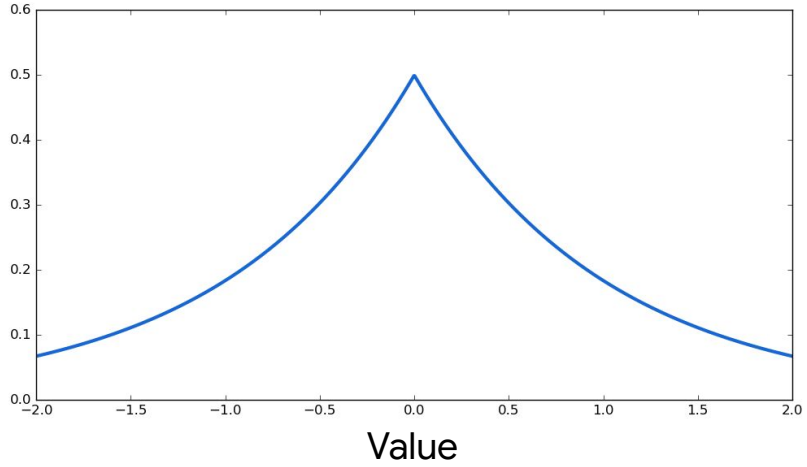
(i.e. g is real-valued)

Laplace Distribution

For every real number z,

$$f_{\text{Lap}(b)}(z) \propto e^{-|z|/b}$$

Probability Density



Laplace Mechanism

$X = (x_1, ..., x_n)$ → Analyzer → $g(X) + \mathrm{Lap}(\Delta(g)/\varepsilon)$

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbf{R}$, Laplace Mechanism is ε-DP.

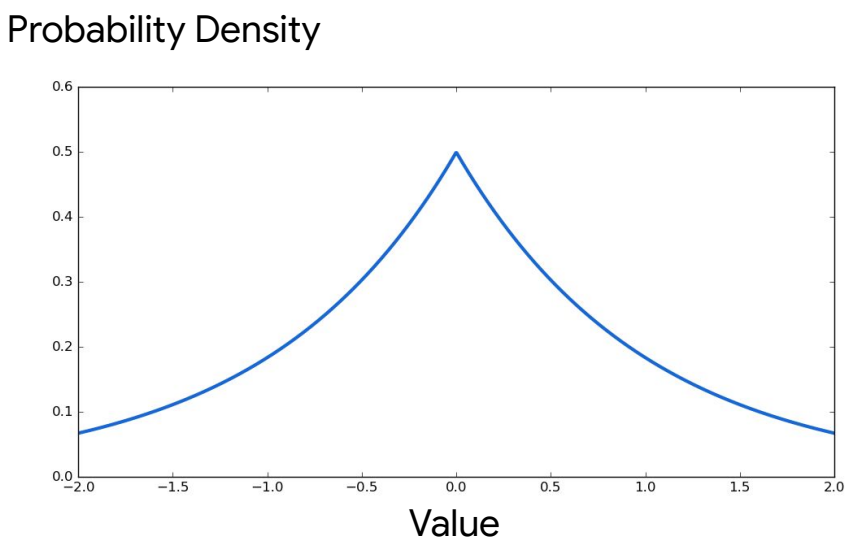
Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}$

(i.e. g is real-valued)

Laplace Distribution

For every real number z,

$$f_{\mathrm{Lap}(b)}(z) \propto e^{-|z|/b}$$



Laplace Mechanism

$X = (x_1, ..., x_n)$ → Analyzer → $g(X) + \mathrm{Lap}(\Delta(g)/\varepsilon)$

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbf{R}$, Laplace Mechanism is ε-DP.

Theorem RMSE = $O(\Delta(g)/\varepsilon)$

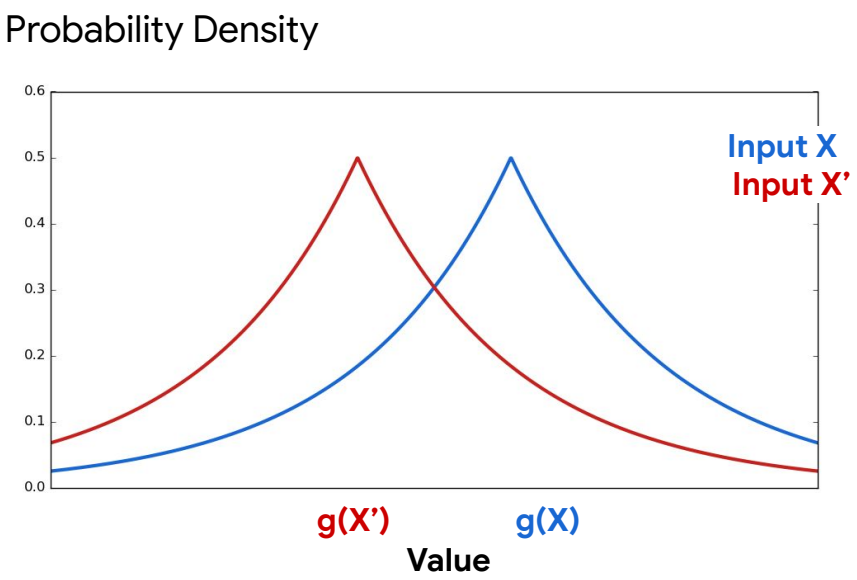
Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}$

(i.e. g is real-valued)

Laplace Distribution

For every real number z,

$$f_{\mathrm{Lap}(b)}(z) \propto e^{-|z|/b}$$



Laplace Mechanism: Multi-Dimension

$X = (x_1, ..., x_n)$ → Analyzer → $g(X) + \mathrm{Lap}(\Delta_1(g)/\varepsilon)^{\otimes d}$

- Use l_1-sensitivity
- Each coordinate is independent

Theorem Assuming $\mathrm{Range}(g) \subseteq \mathbf{R}^d$, Laplace Mechanism is ε-DP.

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}^d$

(i.e. g is vector-valued with real entries)

Laplace Distribution

For every real number z,

$$f_{\mathrm{Lap}(b)}(z) \propto e^{-|z|/b}$$

Gaussian Mechanism

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}$

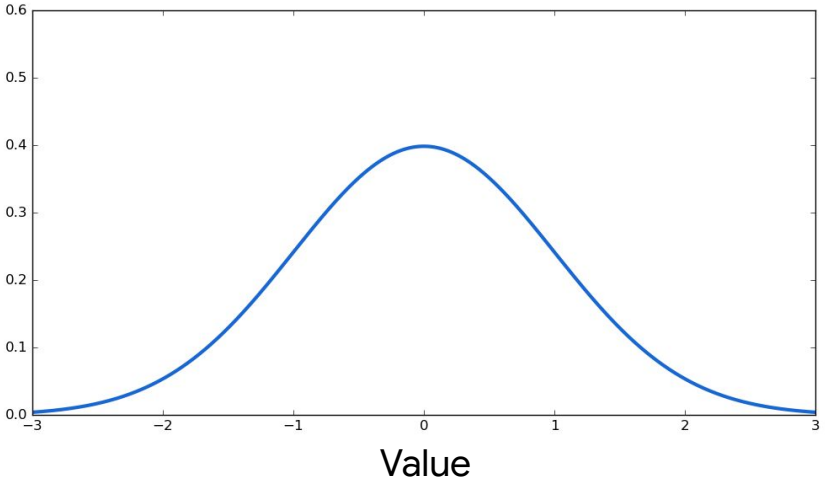
(i.e. g is real-valued)

Gaussian Distribution

For every real number z,

$$\mathrm{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Probability Density



Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta(g)$$

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathcal{N}(0, \sigma^2)$$

Theorem Assuming $\text{Range}(f) \subseteq \mathbf{R}$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

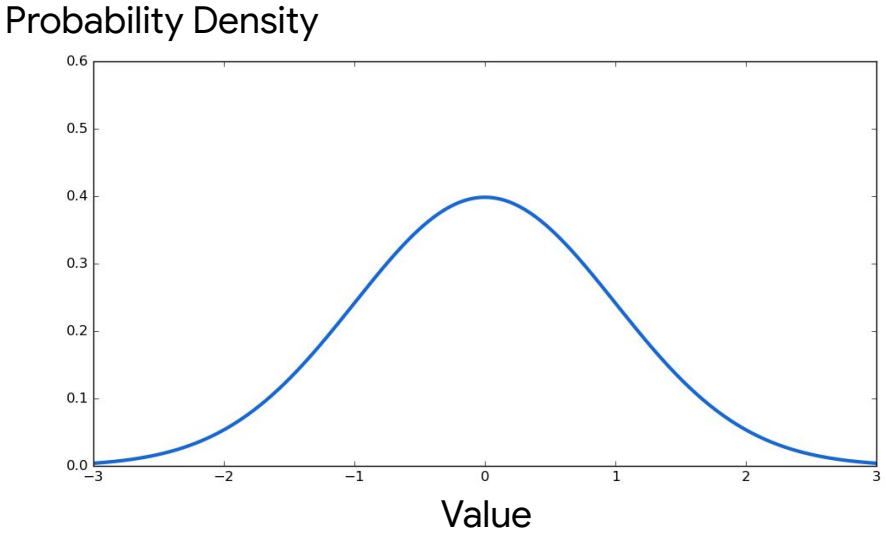
Assumption: $\text{Range}(g) \subseteq \mathbf{R}$

(i.e. g is real-valued)

Gaussian Distribution

For every real number z,

$$\text{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$



Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta(g)$$

Theorem Assuming $\mathrm{Range}(f) \subseteq \mathbf{R}$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}$

(i.e. g is real-valued)

Gaussian Distribution

For every real number z,

$$\mathrm{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Proof is a bit complicated...

Will skip for now.

Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta_2(g)$$

- Use ℓ_2-sensitivity
- Each coordinate is independent

Theorem Assuming $\text{Range}(f) \subseteq \mathbf{R}^d$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\text{Range}(g) \subseteq \mathbf{R}^d$

(i.e. g is vector-valued with real entries)

Gaussian Distribution

For every real number z,

$$\text{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Proof is a bit complicated...

Will skip for now.

Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta_2(g)$$

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathcal{N}(0, \sigma^2)^{\otimes d}$$

- Use l_2-sensitivity
- Each coordinate is independent

Theorem Assuming $\text{Range}(f) \subseteq \mathbf{R}^d$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\text{Range}(g) \subseteq \mathbf{R}^d$

(i.e. g is vector-valued with real entries)

Gaussian Distribution

For every real number z,

$$\text{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Examples: Vector summation

User 1 (0.2, -1, 1)

⋮

User n (0, 2, -0.1)

$g(X) = (50.1, 2.3, 14.7)$

Assumption: Each vector has l_2-norm $\leq C$ ⇒ $\Delta_2(g) \leq 2C$

Privacy Loss Distribution

Discrete outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

Continuous outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(f_{M(X)}(o) / f_{M(X')}(o))$$

Privacy Loss Distribution

$PLD_{M,X,X'}$ is the distribution of $L_{M,X,X'}(o)$ where $o \sim M(X)$

If-and-only-if condition

Pure-DP Condition

M is ε-DP iff, for all $X \asymp X'$, $o \in Range(M)$,

$$L_{M,X,X'}(o) \leq \varepsilon$$

Inverse doesn't hold*

Approximate-DP Condition

M is (ε, δ)-DP if, for all $X \asymp X'$,

$$\Pr[y > \varepsilon] \leq \delta \text{ where } y \sim PLD_{M,X,X'}$$

* there is an iff version but more complicated

Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta(g)$$

Theorem Assuming $\mathrm{Range}(f) \subseteq \mathbf{R}$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\mathrm{Range}(g) \subseteq \mathbf{R}$

(i.e. g is real-valued)

Gaussian Distribution

For every real number z,

$$\mathrm{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Proof Ideas

- $\mathrm{PLD}_{M,X,X'}$ is another Gaussian dist.
- $\Pr[y > \varepsilon]$ where $y \sim \mathrm{PLD}_{M,X,X'}$ is just "tail bound" of Gaussian distribution
- Using known inequalities this quantity can be shown to be at most δ

Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta_2(g)$$

$$X = (x_1, \ldots, x_n) \rightarrow \boxed{\text{Analyzer}} \rightarrow g(X) + \mathcal{N}(0, \sigma^2)^{\otimes d}$$

- Use l_2-sensitivity
- Each coordinate is independent

Theorem Assuming $\text{Range}(f) \subseteq \mathbf{R}^d$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\text{Range}(g) \subseteq \mathbf{R}^d$

(i.e. g is vector-valued with real entries)

Gaussian Distribution

For every real number z,

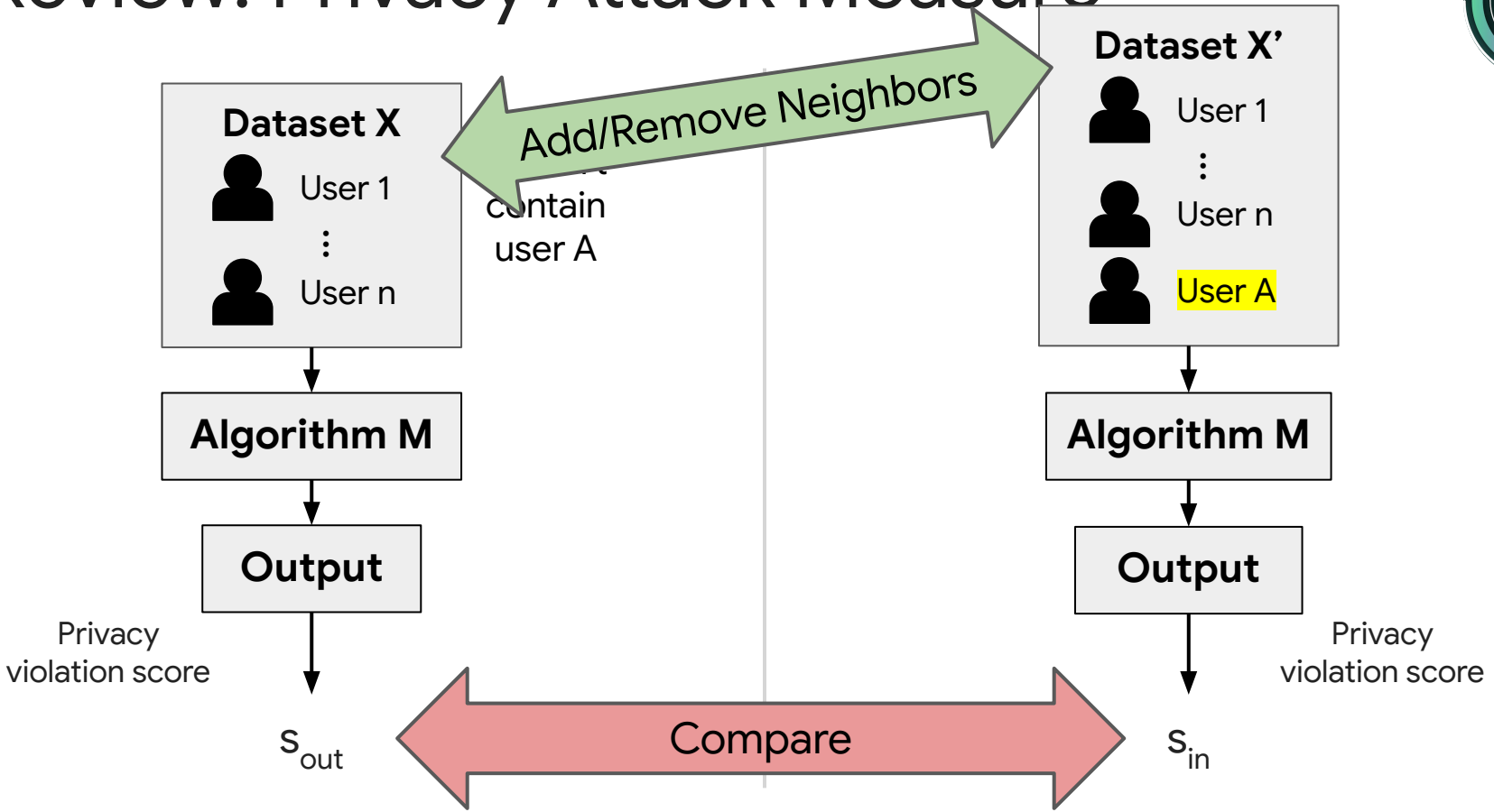
$$\text{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Proof Ideas

- $\text{PLD}_{M, X, X'}$ is another Gaussian dist.
- $\Pr[y > \varepsilon]$ where $y \sim \text{PLD}_{M,X,X'}$ is just "tail bound" of Gaussian distribution
- Using known inequalities this quantity can be shown to be at most δ

Protection Against Privacy Attacks

Review: Privacy Attack Measure



Review: Privacy Attack Measure

Theorem Suppose that privacy violation score is in $[0, 1]$ and the algorithm is (ε, δ)-Add/Remove DP. Then, we have

$$\mathrm{E}[s_{\mathrm{in}}] \leq e^{\varepsilon} \cdot \mathrm{E}[s_{\mathrm{out}}] + \delta$$

Proof Let s denote the scoring function.

$$\begin{aligned}
\mathrm{E}[s_{\mathrm{in}}] &= \mathrm{E}[s(M(X))] \\
&= \int_0^{\infty} \Pr[s(M(X)) > c] \, \mathrm{d}\, c \\
&= \int_0^{1} \Pr[s(M(X)) > c] \, \mathrm{d}\, c \\
&\leq \int_0^{1} (e^{\varepsilon} \cdot \Pr[s(M(X')) > c] + \delta) \, \mathrm{d}\, c \\
&= e^{\varepsilon} \cdot \mathrm{E}[s(M(X'))] + \delta \qquad = e^{\varepsilon} \cdot \mathrm{E}[s_{\mathrm{out}}] + \delta
\end{aligned}$$

QED

Properties of DP

Post-Processing

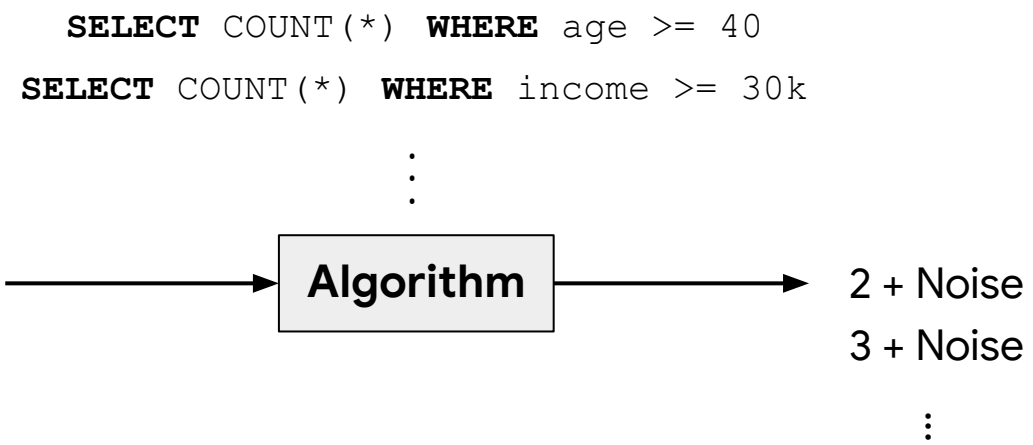
"Result of DP algorithm can be used in arbitrary manner and it remains DP."

Theorem If M is (ε, δ)-DP and h is any function, then $h(M(X))$ remains (ε, δ)-DP.

Can we use the output safely in downstream applications?

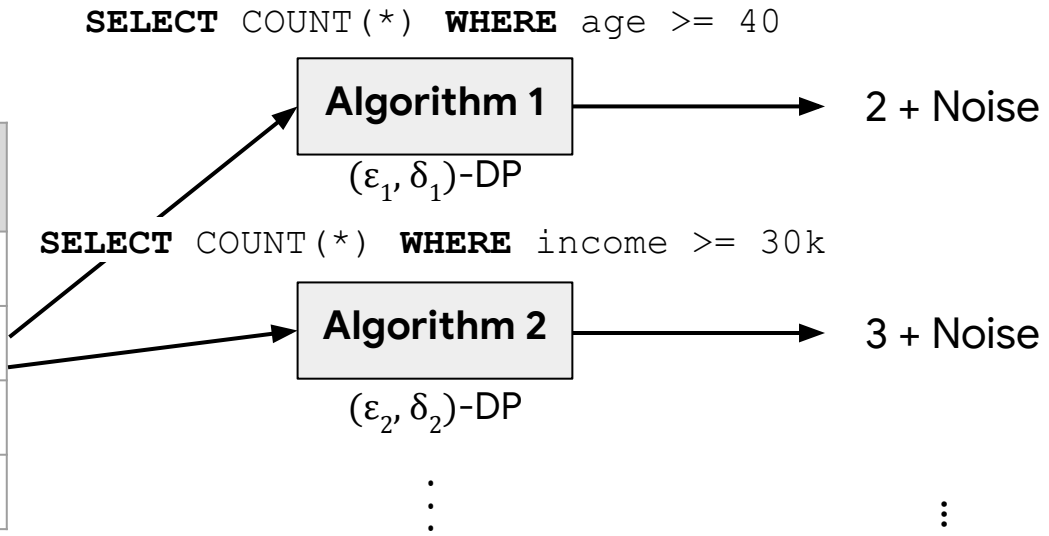
Composition

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



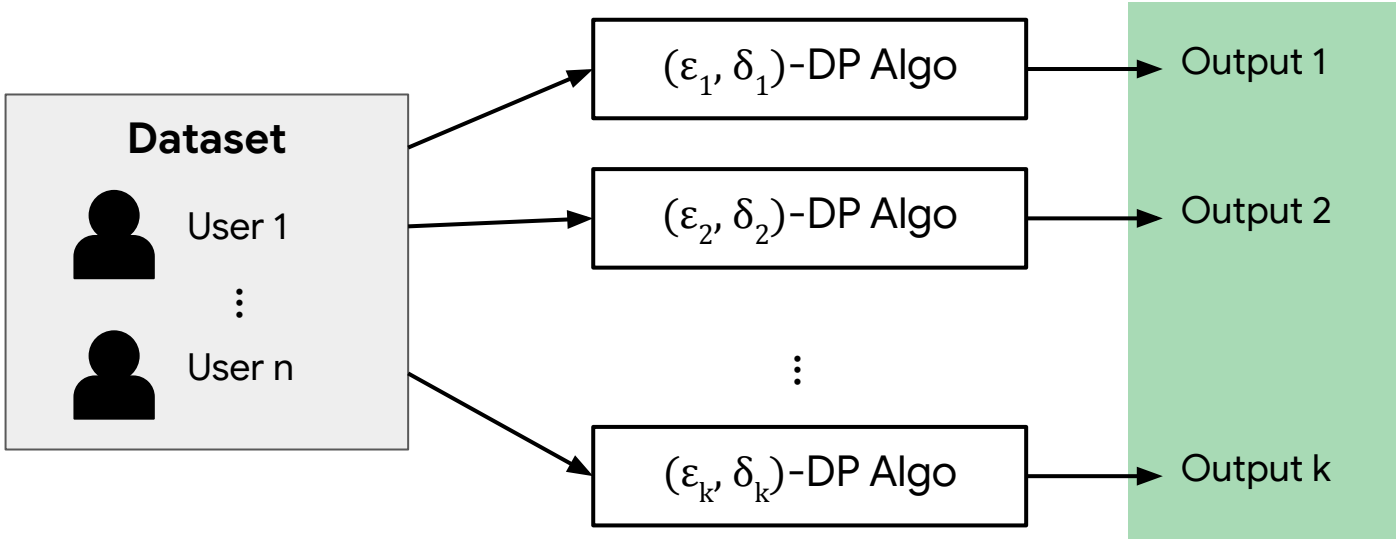
Composition

name	zipcode	age	income
Dale	10520	40	150k
Bob	10520	35	50k
Conor	10500	30	30k
Alice	10500	41	20k



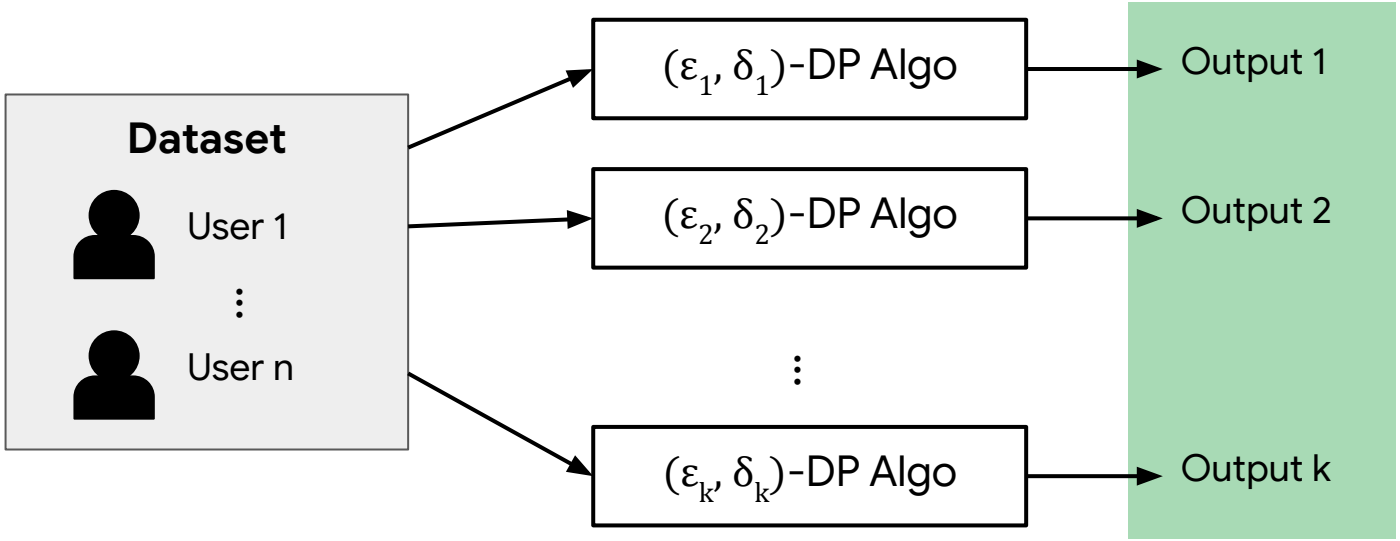
"Running DP algorithms multiple times remain DP, *but with worse parameters*"

Composition



"Running DP algorithms multiple times remain DP, *but with worse parameters*"

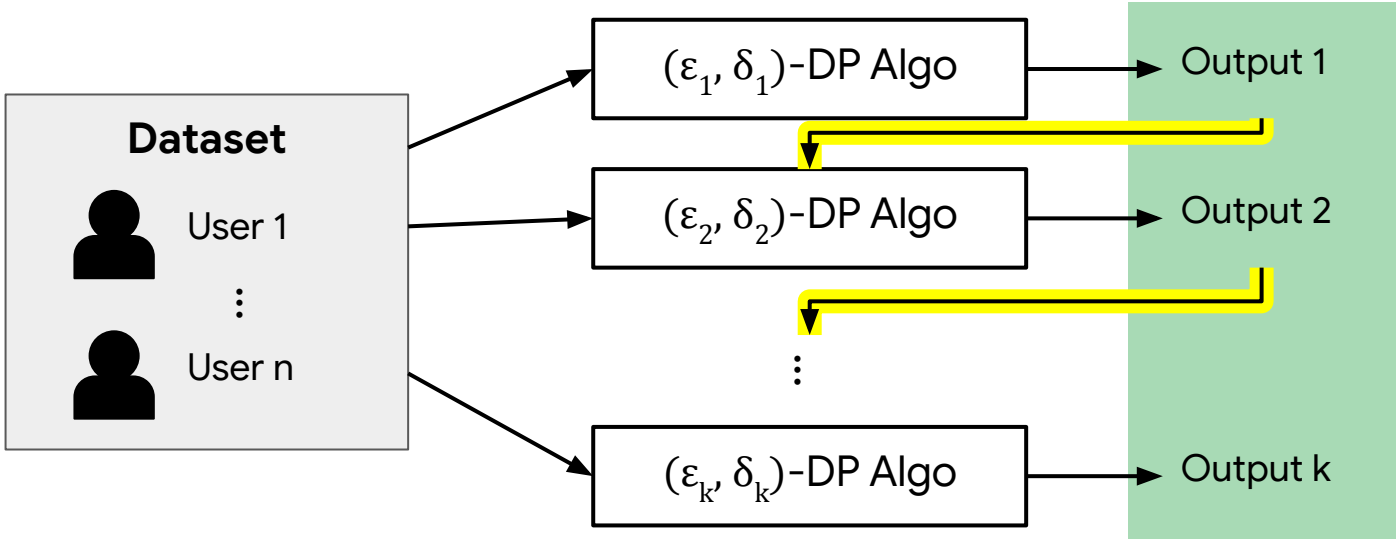
Basic Composition



Basic Composition Theorem *[Dwork et al.]*

All the outputs combined remain $(\varepsilon_1+\varepsilon_2+...+\varepsilon_k, \delta_1+\delta_2+...+\delta_k)$-DP

Basic Composition



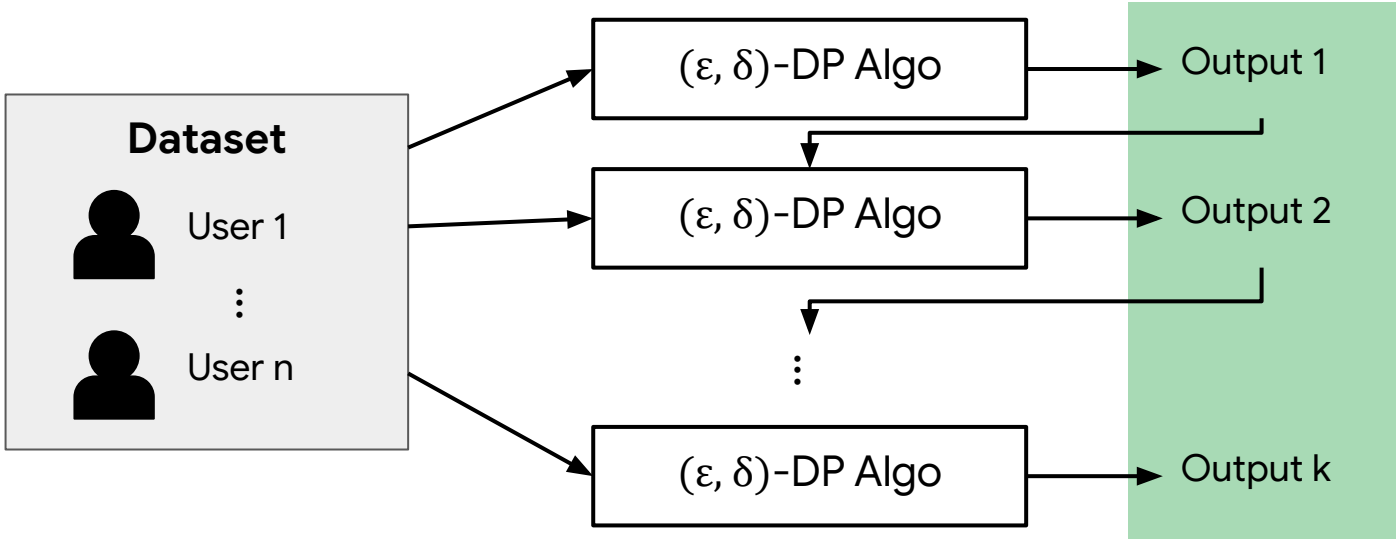
"Adaptive" setting: previous outputs are used in subsequent algorithms

Basic Composition Theorem *[Dwork et al.]*

All the outputs combined remain $(\varepsilon_1+\varepsilon_2+\ldots+\varepsilon_k, \delta_1+\delta_2+\ldots+\delta_k)$-DP

Works even for "adaptive" setting

Advanced Composition

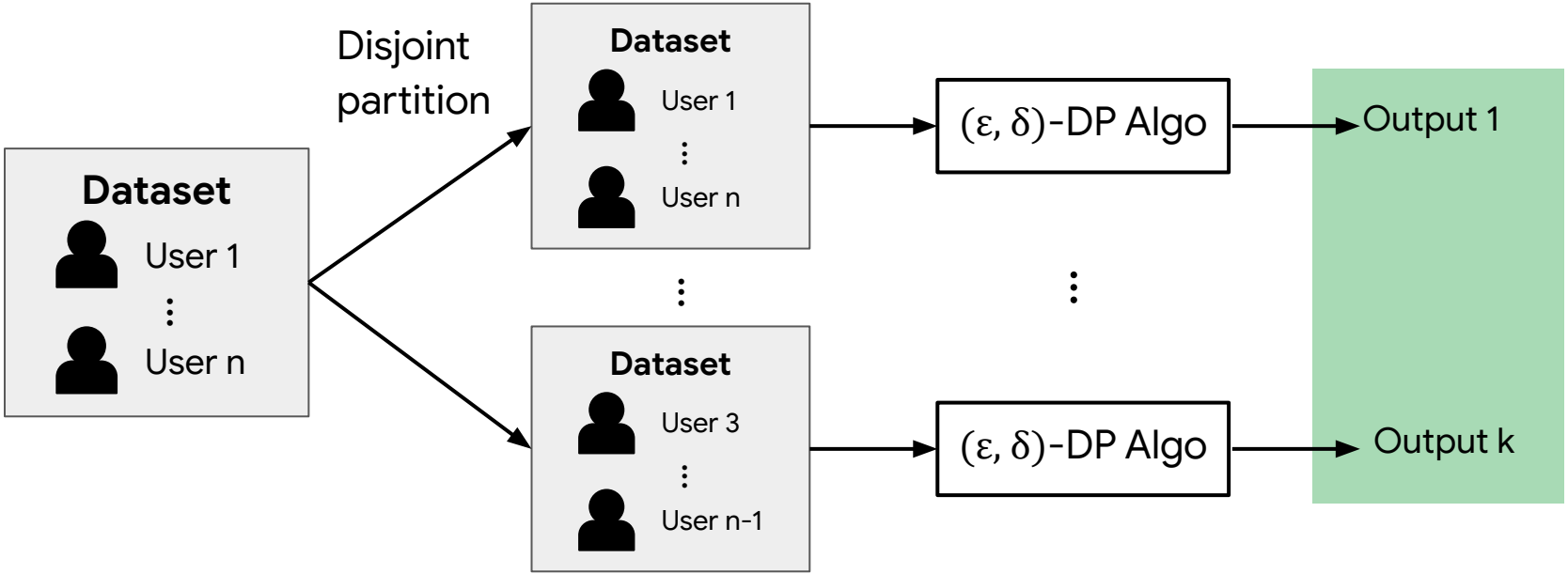


Advanced Composition Theorem *[Dwork et al.]*
All the outputs combined is $(\varepsilon', k\delta + \delta')$-DP where

$$\varepsilon' = \sqrt{2k \ln(1/\delta')} \cdot \epsilon + k\epsilon(e^{\epsilon} - 1)$$

- For small ε and large k, ε' is only $\approx \varepsilon\sqrt{k}$
- Better than $\varepsilon' = \varepsilon k$ from Basic composition

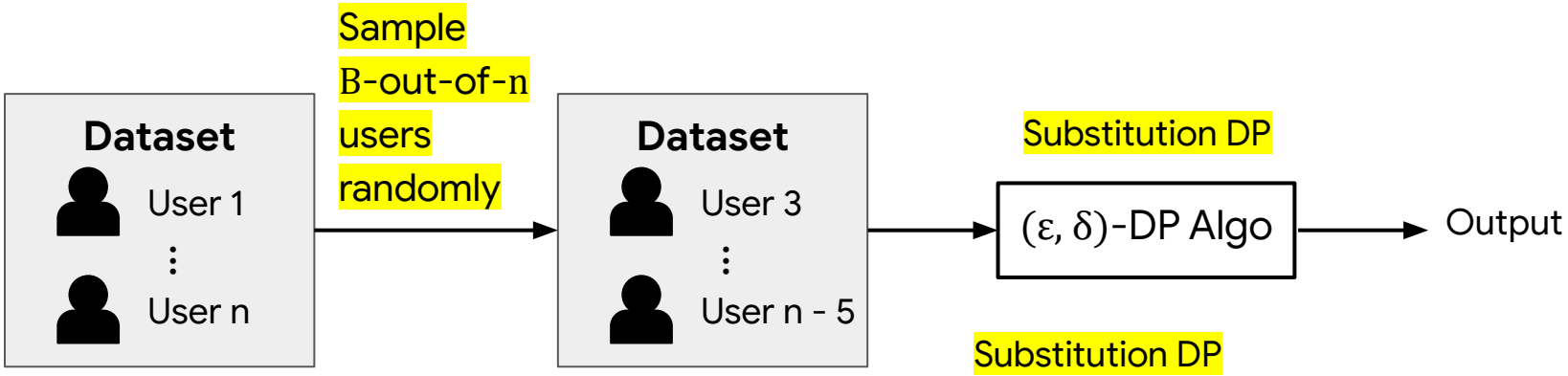
Parallel Composition



Parallel Composition Theorem *[McSherry]*
All the outputs combined remain (ε, δ)-DP

Amplification by Subsampling

"Subsampling makes the algorithm more private."



Amplification-by-subsampling Theorem

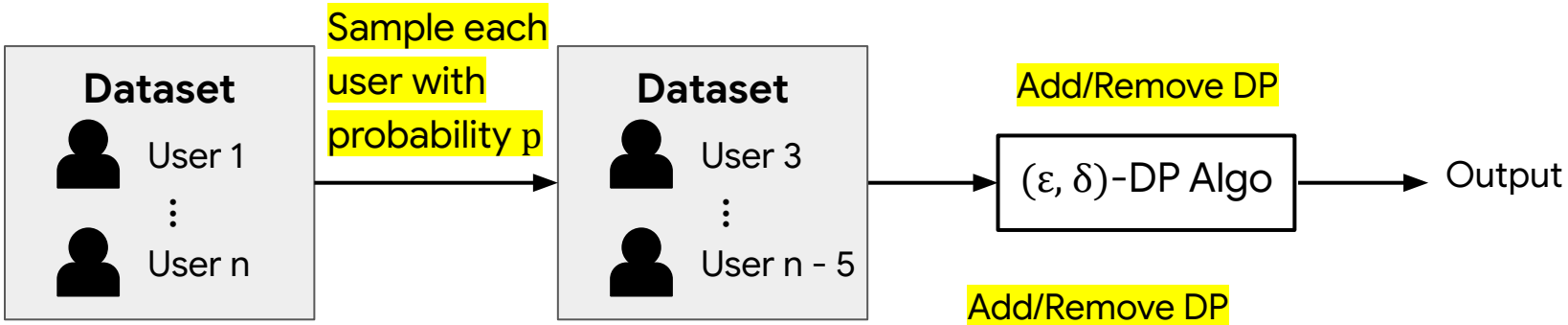
The output combined is (ε', δ')-DP where

$$\varepsilon' = \ln(1 + p(e^{\varepsilon} - 1)), \qquad \delta' = p\delta$$

with $p = B / n$.

Amplification by Subsampling

"Subsampling makes the algorithm more private."



Amplification-by-subsampling Theorem

The output combined is (ε', δ')-DP where

$$\varepsilon' = \ln(1 + p(e^{\varepsilon} - 1)), \qquad \delta' = p\delta.$$

Group Privacy

Example:

Email	zipcode	Purchase value
dale123@gmail.com	10520	500
dale1993@gmail.com	10520	25
alice@gmail.com	10500	15
bob@gmail.com	10500	40

Group Privacy

k-Neighboring

X, X' are k-neighboring, denoted by $\asymp_k$ iff there exist a sequence $X = X_0 \asymp X_1 \asymp \ldots \asymp X_t = X'$ where $t \leq k$.

Theorem If M is (ε, δ)-DP under neighboring notion $\asymp$, then $M(X)$ is (ε', δ')-DP for neighboring notion $\asymp_k$, where $\varepsilon' = k\varepsilon$ and $\delta' = (e^{k\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta$.

Example:

Same individual

Email	zipcode	Purchase value
dale123@gmail.com	10520	500
dale1993@gmail.com	10520	25
alice@gmail.com	10500	15
bob@gmail.com	10500	40

Is this still DP? ⇨ Yes, it is 2ε-DP

Differential Privacy: Neighboring Notions

Add/remove-DP

$X \asymp^r X'$ ⟺ X is X' with an individual's data added / removed

- Protect against Membership Inference
- Cannot reveal the raw size of the dataset
  - Sometimes make it harder to design & analyze algorithms for

Substitution-DP

$X \asymp^s X'$ ⟺ X is X' with an individual's data changed

- Does *not* Protect against Membership Inference
- Can reveal the raw size of the dataset

In this tutorial, if not stated,
assume ***substitution-DP***

Group Privacy

k-Neighboring

X, X' are k-neighboring, denoted by $\asymp_k$ iff there exist a sequence $X = X_0 \asymp X_1 \asymp \ldots \asymp X_t = X'$ where $t \leq k$.

Theorem If M is (ε, δ)-DP under neighboring notion $\asymp$, then $M(X)$ is (ε', δ')-DP for neighboring notion $\asymp_k$, where $\varepsilon' = k\varepsilon$ and $\delta' = (e^{k\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta$.

Example:

Add/remove $X \asymp^r X'$ ⟺ X is X' with an individual's data added / removed

Substitution $X \asymp^s X'$ ⟺ X is X' with an individual's data changed

Observation

If $X \asymp^s X'$, then $X \asymp^r_2 X'$

"Every substitution neighbor is a 2-add/remove neighbor"

Group Privacy

k-Neighboring

X, X' are k-neighboring, denoted by $\asymp_k$ iff there exist a sequence $X = X_0 \asymp X_1 \asymp \ldots \asymp X_t = X'$ where $t \leq k$.

Theorem If M is (ε, δ)-DP under neighboring notion $\asymp$, then $M(X)$ is (ε', δ')-DP for neighboring notion $\asymp_k$, where $\varepsilon' = k\varepsilon$ and $\delta' = (e^{k\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta$.

Example:

Lemma

If M is ε-add/remove-DP, then it is 2ε-substitution-DP

Observation

If $X \asymp^s X'$, then $X \asymp^r_2 X'$

"Every substitution neighbor is a 2-add/remove neighbor"

Group Privacy: Proof

k-Neighboring

X, X' are k-neighboring, denoted by $\asymp_k$ iff there exist a sequence $X = X_0 \asymp X_1 \asymp \ldots \asymp X_t = X'$ where $t \leq k$.

Theorem If M is (ε, δ)-DP under neighboring notion $\asymp$, then $M(X)$ is (ε', δ')-DP for neighboring notion $\asymp_k$, where $\varepsilon' = k\varepsilon$ and $\delta' = (e^{k\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta$.

Proof Consider any $X \asymp_k X'$, $S \subseteq \text{Range}(M)$. By defn, there is $X = X_0 \asymp X_1 \asymp \ldots \asymp X_t = X'$ where $t \leq k$

$$
\begin{aligned}
\Pr[M(X) \in S] &\leq e^{\varepsilon} \Pr[M(X_1) \in S] + \delta \\
&\leq e^{\varepsilon} (e^{\varepsilon} \Pr[M(X_2) \in S] + \delta) + \delta \\
&\ldots \\
&\leq e^{t\varepsilon} \Pr[M(X') \in S] + (e^{t\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta \\
&\leq e^{k\varepsilon} \Pr[M(X') \in S] + (e^{k\varepsilon} - 1) / (e^{\varepsilon} - 1) \cdot \delta
\end{aligned}
$$

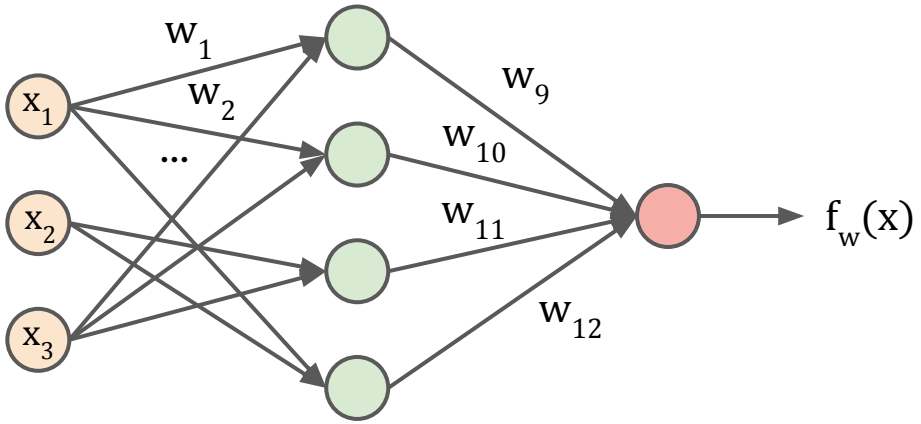
QED

Machine Learning with Differential Privacy

ML Model

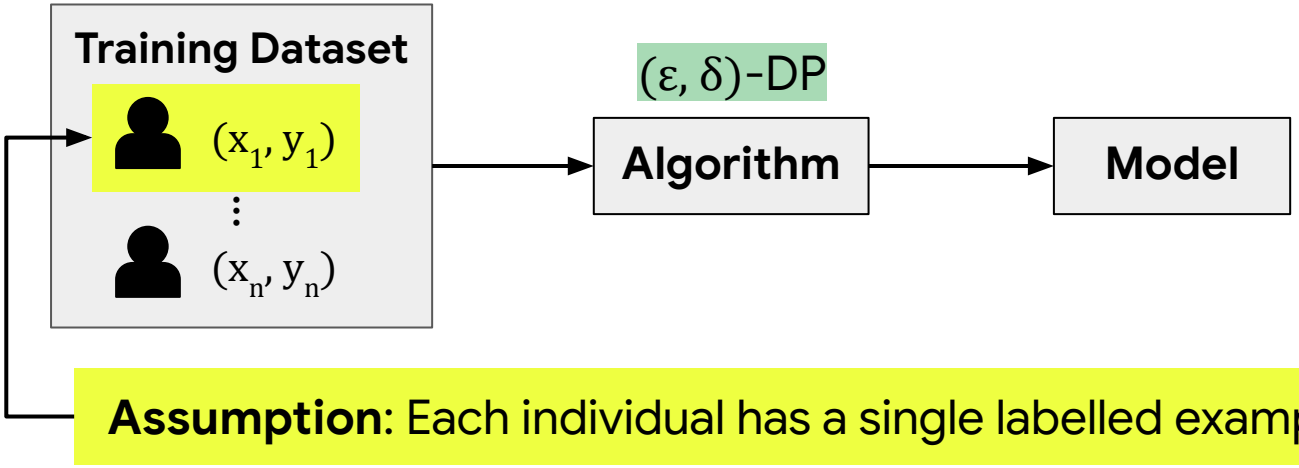
Sample x → Model with parameter w → Prediction $f_w(x)$

Example: neural network



Training ML Model

Ensure that the model is privacy-safe to be used in downstream tasks



Substitution-DP

$X \asymp^s X'$ ⟺ X is X' with single training example changed

DP-SGD Algorithm

Gradient Descent

Training data X

Labeled Samples

$(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$

Loss function: $\ell(\hat{y}, y) \in \mathbf{R}$

Empirical loss

$$\mathcal{L}_w(X) := \Sigma_{i\in[n]} \ \ell(f_w(x_i), y_i) \ / \ n$$

Training Objective

Find w that minimizes $\mathcal{L}_w(X)$

Gradient

$$\nabla_w \mathcal{L}(X) = [d\,\mathcal{L}(X) / d\,w_1, ..., d\,\mathcal{L}(X) / d\,w_d]$$

$$\nabla_w \mathcal{L}(X) = \Sigma_{i\in[n]} \nabla_w \ell(f_w(x_i), y_i) \ / \ n$$

η_t: learning rate

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For $t = 1,...,T$:

$$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \mathcal{L}(X)$$

Return w_T

Gradient Descent

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \mathcal{L}(X)$$

Return w_T

Gradient Descent

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$$w_t \leftarrow w_{t-1} - \eta_t \left(\Sigma_{i \in [n]} \nabla_w \ell(f_w(x_i), y_i)\right) / n$$

Return w_T

Gaussian Mechanism

$$\sigma = \frac{2\sqrt{2\ln(2/\delta)}}{\epsilon} \cdot \Delta_2(g)$$

$X = (x_1, \ldots, x_n) \rightarrow$ Analyzer $\rightarrow g(X) + \mathcal{N}(0, \sigma^2)^{\otimes d}$

- Use l_2-sensitivity
- Each coordinate is independent

Theorem Assuming $\text{Range}(f) \subseteq \mathbf{R}^d$ and $\varepsilon \leq 1$, Gaussian Mechanism is (ε, δ)-DP.

Assumption: $\text{Range}(g) \subseteq \mathbf{R}^d$

(i.e. g is vector-valued with real entries)

Gaussian Distribution

For every real number z,

$$\text{PDF}_{\mathcal{N}(0,\sigma^2)}(z) \propto e^{-(z/\sigma)^2}$$

Examples: Vector summation

User 1 (0.2, -1, 1)

⋮

User n (0, 2, -0.1)

$g(X) = (50.1, 2.3, 14.7)$

Assumption: Each vector has l_2-norm $\leq C$ ⇒ $\Delta_2(g) \leq 2C$

Gradient Descent

Additional parameters
- σ: noise standard deviation

Add Gaussian noise to average gradients!

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$

$$w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in [n]} \nabla_w \ell(f_w(x_i), y_i)\right) / n$$

Return w_T

Not differentially private: each $\nabla_w \ell(f_w(x_i), y_i)$ can be arbitrarily large!

Clipping Trick

Additional parameters
- σ: noise standard deviation
- C: clipping norm

DP-GD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

  For $i = 1, \ldots, n$:

    $v_i = \nabla_w \ell(f_w(x_i), y_i)$

    $v_i = v_i \cdot \min(1, C / ||v_i||_2)$

  $w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in [n]} v_i\right) / n$

Return w_T

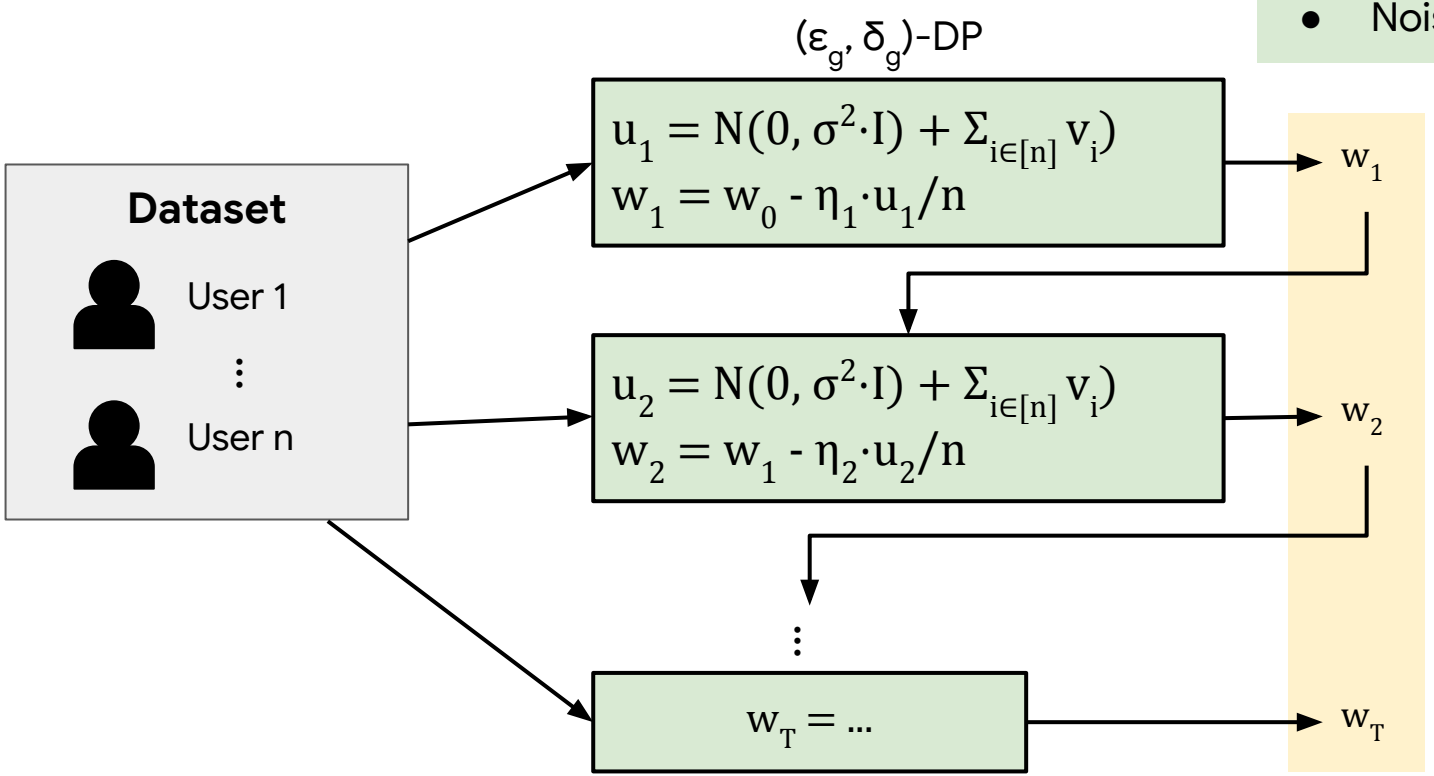
Enforce: gradient norm bound at most C

Not differentially private: each $\nabla_w \ell(f_w(x_i), y_i))$ can be arbitrarily large!

"If $\nabla_w \ell(f_w(x_i), y_i))$ is too large, rescale it to be smaller"

DP-GD: Privacy Analysis

Gaussian mechanism:
- l_2-sensitivity ≤ 2C
- Noise std σ



T-fold basic / advanced composition

Entire Algo is (ε, δ)-DP

Actually can release all of $w_1, \ldots, w_T$!

DP-GD: Privacy Analysis

DP-GD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

For $i = 1, \ldots, n$:

$$v_i = \nabla_w \ell(f_w(x_i), y_i)$$

$$v_i = v_i \cdot \min(1, C / ||v_i||_2)$$

$$w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in [n]} v_i\right) / n$$

Return w_T

Each iteration

Gaussian mechanism:

- l_2-sensitivity ≤ 2C
- Noise std σ
- ⇒ $(\varepsilon_g, \delta_g)$-DP

T-fold basic / advanced composition

Entire Algorithm: (ε, δ)-DP

Gradient Descent & Friends

Gradient Descent (GD)

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \mathcal{L}(X)$$

Return w_T

Stochastic GD (SGD)

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$i \leftarrow$ random example index

$$w_t \leftarrow w_{t-1} - \eta_t \nabla_w \ell(f_w(x_i), y_i)$$

Return w_T

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$S \leftarrow$ random index set of size B

$$w_t \leftarrow w_{t-1} - \eta_t \Sigma_{i \in S} \nabla_w \ell(f_w(x_i), y_i) / B$$

Return w_T

Generalizes both GD, SGD

DP-SGD

Modifications to achieve DP

- Add Gaussian noise to average gradients!

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

S ← random index set of size B

$w_t \leftarrow w_{t-1} - \eta_t \Sigma_{i \in S} \nabla_w \mathcal{l}(f_w(x_i), y_i) / B$

Return w_T

DP-SGD

Modifications to achieve DP

- Add Gaussian noise to average gradients!

Additional parameters

- σ: noise standard deviation

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

S ← random index set of size B

$w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} \nabla_w \mathcal{l}(f_w(x_i), y_i)\right) / B$

Return w_T

Issue: gradient can be very large!

DP-SGD

Modifications to achieve DP

- Add Gaussian noise to average gradients!
- Clip each gradient to bound its norm

Additional parameters

- σ: noise standard deviation
- C: clipping norm bound

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$

  $S \leftarrow$ random index set of size B

  For i in S:

    $g_i = \nabla_w \mathcal{l}(f_w(x_i), y_i)$

    $g_i = g_i \cdot \min(1, C / \|g_i\|_2)$

  $w_t \leftarrow w_{t-1} - \eta_t (N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} g_i) / B$

Return w_T

Enforce: gradient norm bound at most C

DP-SGD

Mini-batch DP-SGD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$S \leftarrow$ random index set of size B

For i in S:

$$g_i = \nabla_w \boldsymbol{\ell}(f_w(x_i), y_i)$$

$$g_i = g_i \cdot \min(1, C / ||g_i||_2)$$

$$w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} g_i\right) / B$$

Return w_T

DP-SGD: Privacy Analysis

Mini-batch DP-SGD

$w_0 \leftarrow$ initial parameter

For $t = 1,\ldots,T$:

$S \leftarrow$ random index set of size B

For i in S:

$$g_i = \nabla_w \ell(f_w(x_i), y_i)$$

$$g_i = g_i \cdot \min(1, C / ||g_i||_2)$$

$$w_t \leftarrow w_{t-1} - \eta_t \left(N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} g_i\right) / B$$

Return w_T

Each iteration

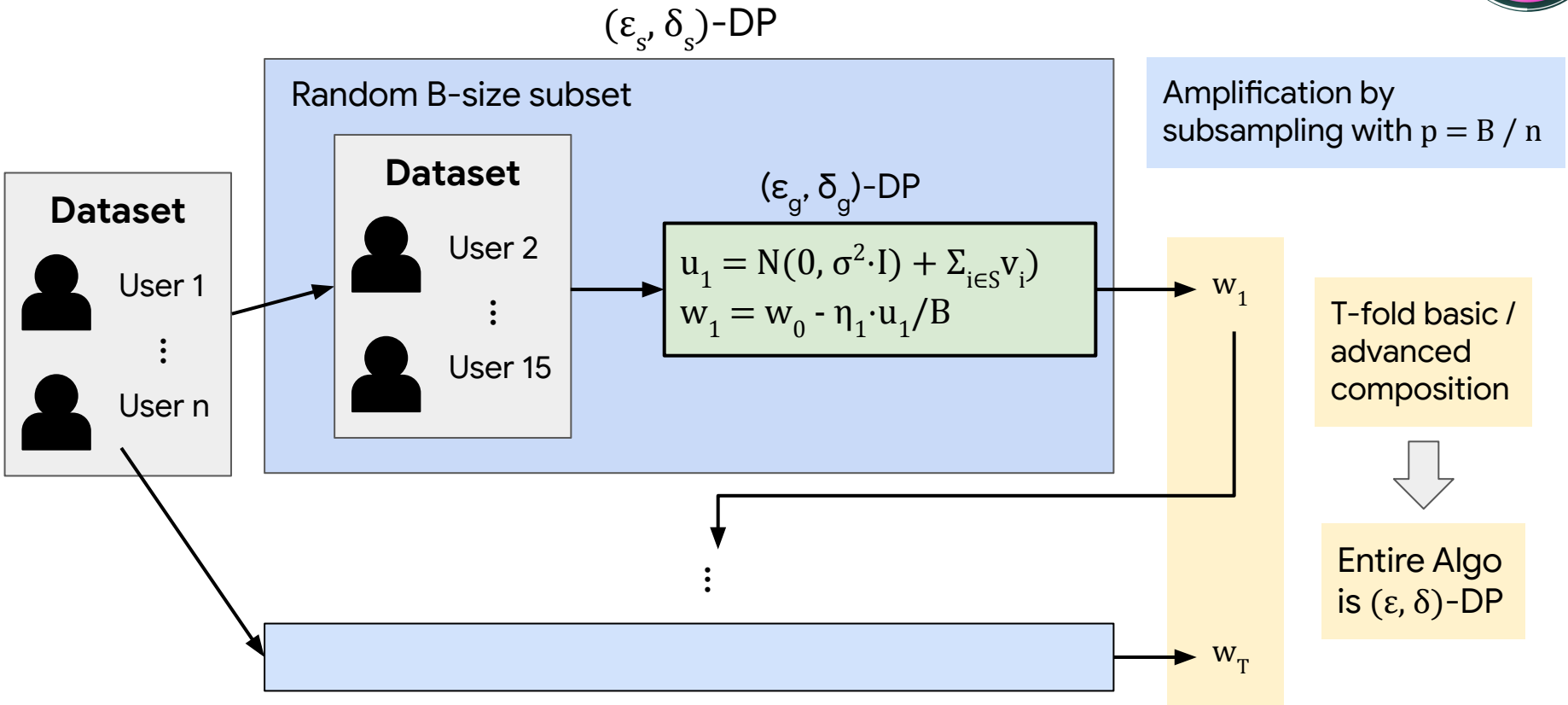
Gaussian mechanism:

- l_2-sensitivity ≤ 2C
- Noise std σ
- ⇒ $(\varepsilon_g, \delta_g)$-DP

T-fold basic / advanced composition

Entire Algorithm: (ε, δ)-DP

DP-SGD: Privacy Analysis



DP-SGD: Privacy Analysis

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

S ← random index set of size B

For i in S:

$g_i = \nabla_w \mathcal{l}(f_w(x_i), y_i)$

$g_i = g_i \cdot \min(1, C / \|g_i\|_2)$

$w_t \leftarrow w_{t-1} - \eta_t (N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} g_i) / B$

Return w_T

Each iteration

Gaussian mechanism:

- l_2-sensitivity ≤ 2C
- Noise std σ
- ⇒ $(\varepsilon_g, \delta_g)$-DP

T-fold basic / advanced composition

Entire Algorithm: (ε, δ)-DP

DP-SGD: Privacy Analysis

Mini-batch SGD

$w_0 \leftarrow$ initial parameter

For t = 1,...,T

S ← random index set of size B

For i in S:

$g_i = \nabla_w \mathcal{l}(f_w(x_i), y_i)$

$g_i = g_i \cdot \min(1, C / \|g_i\|_2)$

$w_t \leftarrow w_{t-1} - \eta_t (N(0, \sigma^2 \cdot I) + \Sigma_{i \in S} g_i) / B$

Return w_T

Each iteration

Gaussian mechanism:

- l_2-sensitivity ≤ 2C
- Noise std σ
- ⇒ $(\varepsilon_g, \delta_g)$-DP

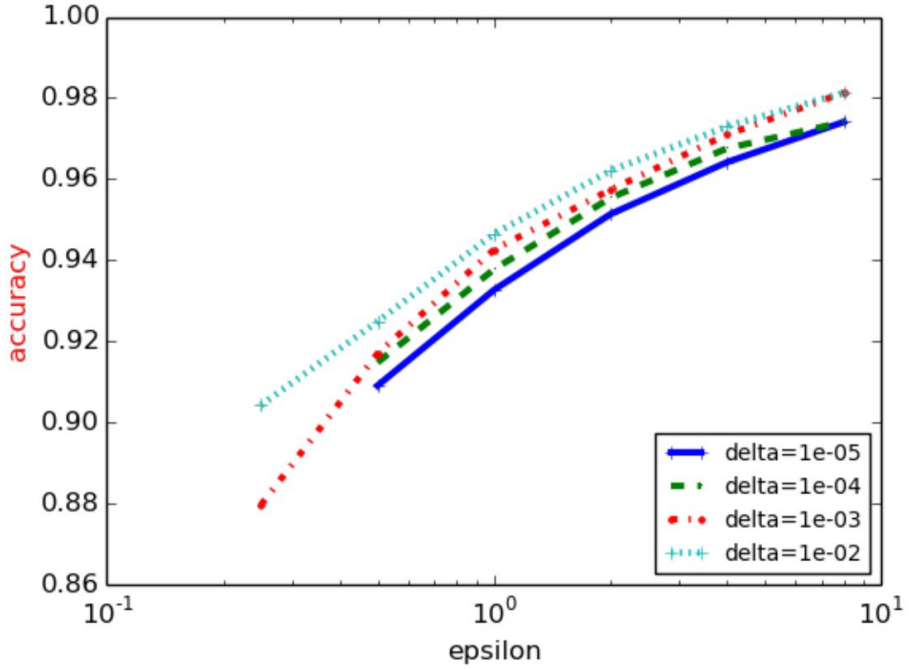
Amplification by subsampling with p = B / n

Each iteration: $(\varepsilon_s, \delta_s)$-DP

T-fold basic / advanced composition

Entire Algorithm: (ε, δ)-DP

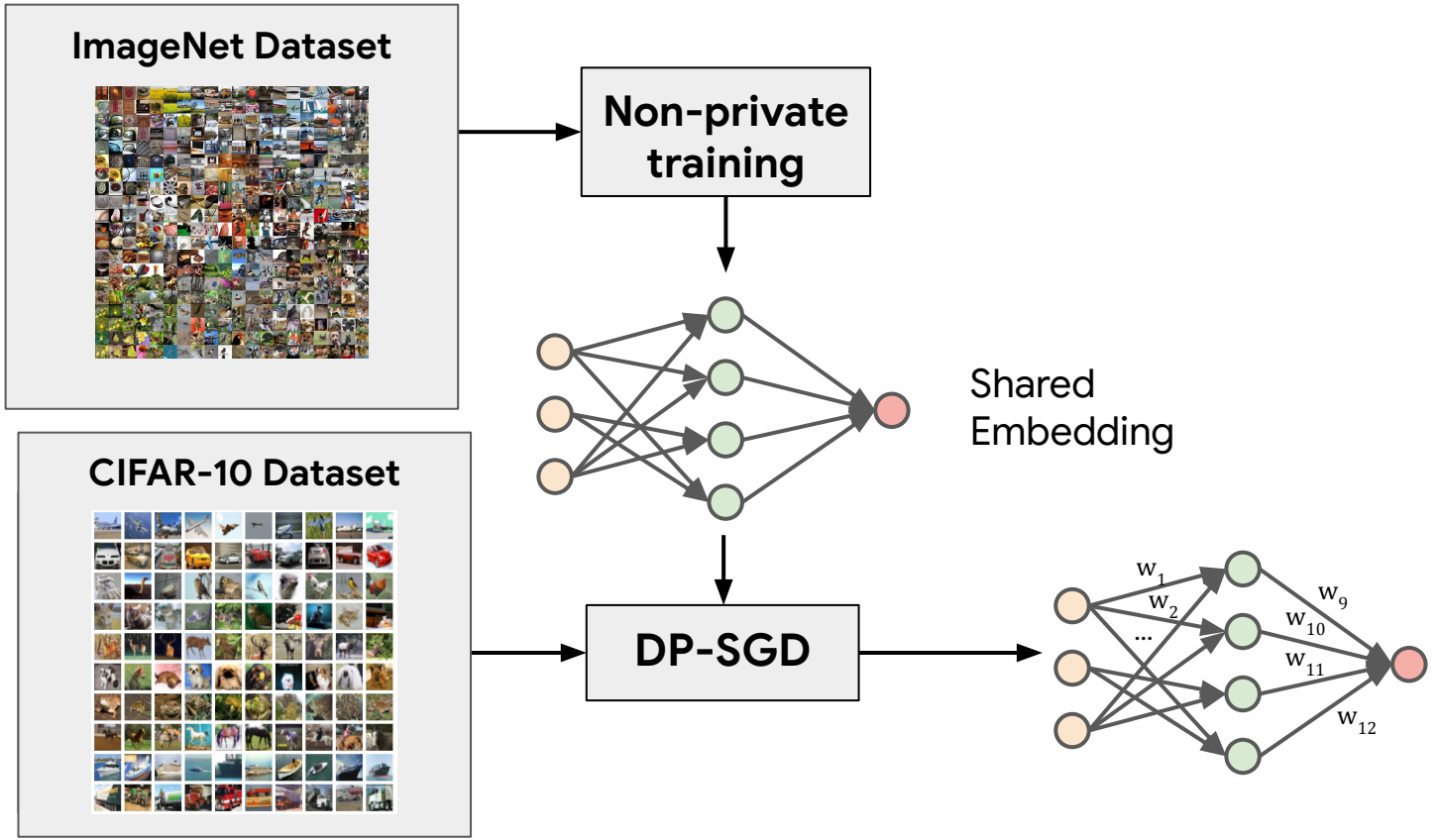
DP-SGD: Result for MNIST [2]



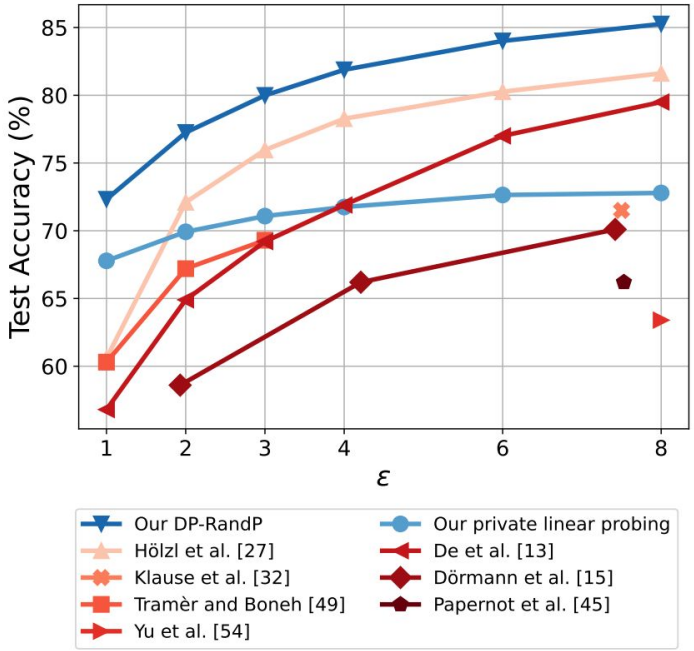
ε = 0.5: 90% accuracy
ε = 8: 97% accuracy

[2] Abadi, Chu, Goodfellow, McMahan, Mironov, Talwar & Zhang: *Deep Learning with Differential Privacy*. CCS'16

DP-SGD: Improving via Public Dataset



DP-SGD: Result for CIFAR-10 [3]



ε = 1: 73% accuracy
ε = 8: 85% accuracy

[3] Tang, Panda, Sehwag, Mittal: *Differentially Private Image Classification by Learning Priors from Random Processes*. 2023

Other ML Algorithms

Output Perturbation

"ERM + Gaussian Noise"

Output Perturbation

$\mathbf{w}^* \leftarrow \text{argmin}_{\mathbf{w}} \Sigma_{i\in[n]} \boldsymbol{\ell}(f_{\mathbf{w}}(x_i), y_i) / n$

Return $\mathbf{w}^* + \mathbf{N(0, \sigma^2 \cdot I)}$

Assumption I:
l is $\boldsymbol{\eta}$-strongly convex

(i.e. $\boldsymbol{\ell}$ - $0.5\boldsymbol{\eta}||w||^2$ is convex)

Assumption II:
l is L-Lipschitz

(i.e. $\boldsymbol{\ell}' \leq L$)

Theorem Under the two assumptions and with appropriate σ, Output Perturbation Mechanism is (ε, δ)-DP

Theorem w^* has l_2-sensitivity $\leq 4L/\boldsymbol{\eta}n$

Objective Perturbation

"ERM but with perturbed objective"

Objective Perturbation

$b \leftarrow N(0, \sigma^2 \cdot I)$

$w^* \leftarrow \text{argmin}_w \Sigma_{i \in [n]} \ell(f_w(x_i), y_i) / n + \langle b, w \rangle$

Return w^*

Assumption I:
ℓ is **β**-smooth

(i.e. gradient is β-Lipschitz)

Assumption II:
ℓ is **L**-Lipschitz

Theorem Under the two assumptions and with appropriate σ, Objective Perturbation Mechanism is (ε, δ)-DP

Exponential Mechanism

Exponential Mechanism

Output each $\mathbf{w}$ with probability α $\exp(-0.5\,\varepsilon \cdot \Sigma_{i\in[n]}\ \ell(f_w(x_i), y_i))$

Assumption: Range(ℓ) $\subseteq$ [0, 1]

Theorem Under the assumption, Exponential Mechanism is ε-DP

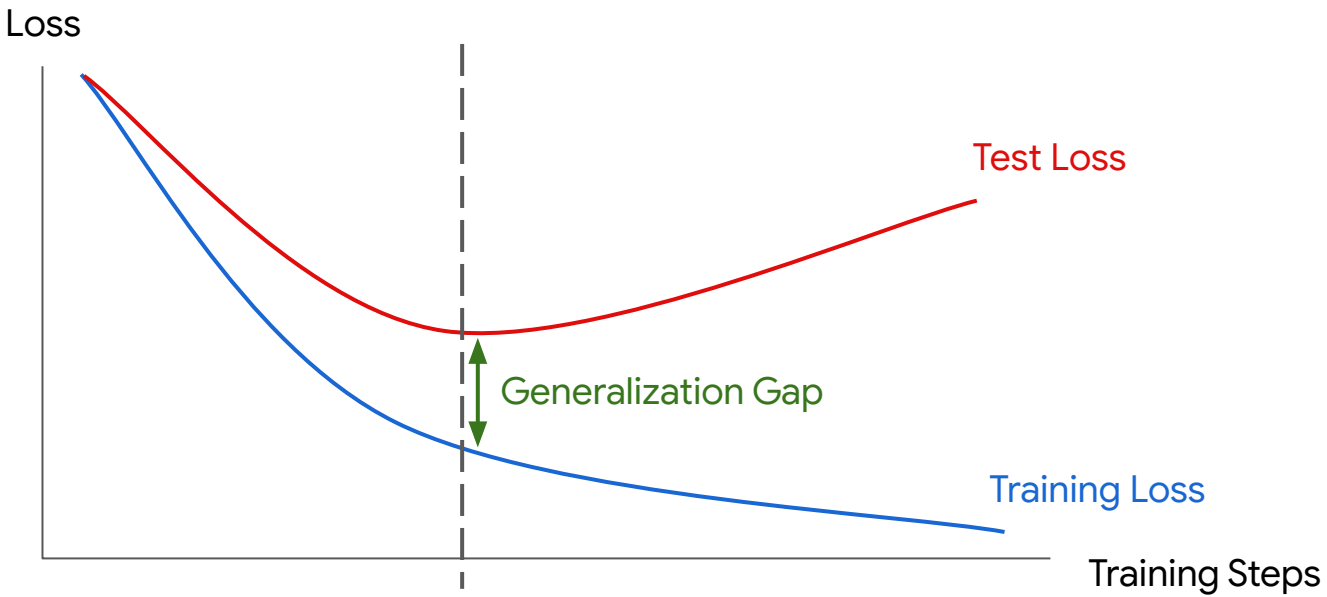
Issue: Sampling cannot be done efficiently for general ℓ

Theorem When $\ell$is convex, there is a poly time sampling algorithm

Other Properties of DP ML Algorithms

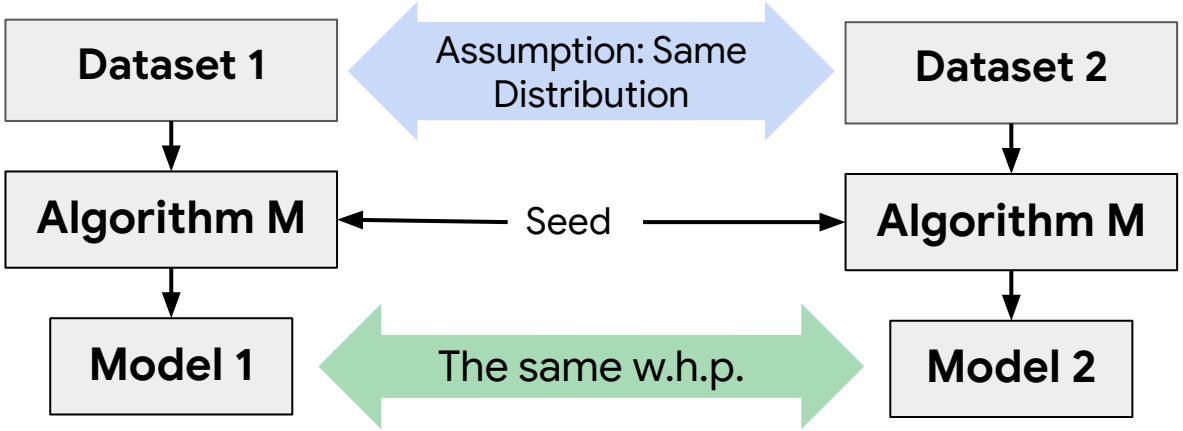
DP ⇒ Bounded Generalization Gap

"Theorem" If the model is trained with ε-DP, then (expected) generalization gap is O(ε)



DP ⇒ Replicability

Replicability



"Theorem" If the model can trained with an ε-DP algorithm, then it can be trained with replicable algorithm (with comparable accuracy & # samples)

Conclusion

Conclusion

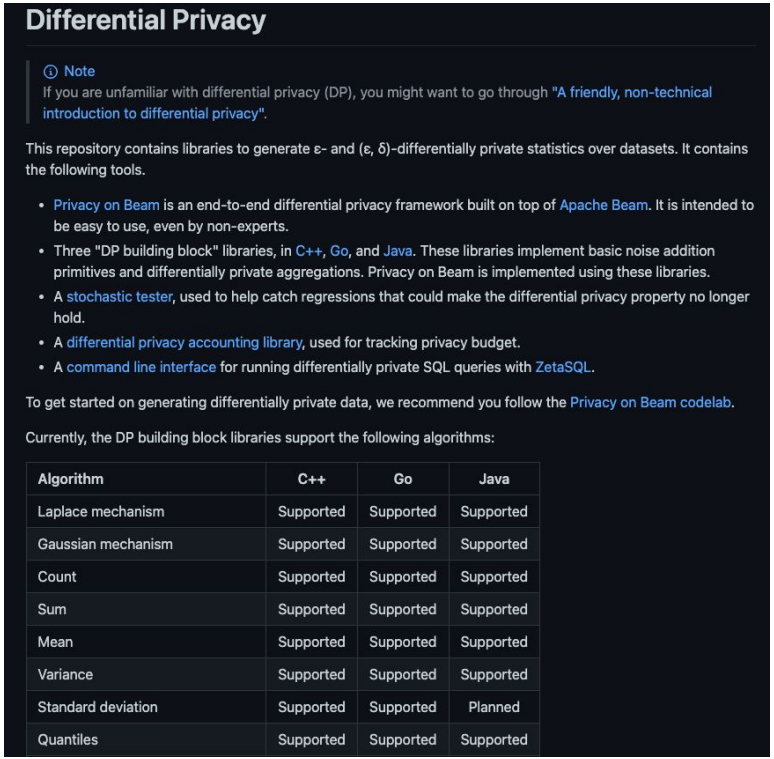
- Privacy Attacks
  - ML models / Aggregated Statistic from Sensitive Data
- Differentially Privacy
  - Formal protection against all attacks
  - Nice properties
    - Composition
    - Post processing
  - DP-SGD & Other DP ML algorithms

DP: Research Directions

- Improving DP ML accuracies
  - Relaxation of Neighboring datasets, e.g. Label DP
- Distributed Models of DP
  - Shuffle DP
  - Local DP
  - Federated Learning
- Privacy Accounting
  - Privacy loss over time

Open-Source Libraries

- Some open-source DP libraries:
  - Generic DP Libraries:
    - Google DP Library
    - IBM Diffprivlib Library
    - OpenDP Library
  - DP ML Libraries:
    - Tensorflow privacy
    - Pytorch Opacus



Differential Privacy

> Note
> If you are unfamiliar with differential privacy (DP), you might want to go through "A friendly, non-technical introduction to differential privacy".

This repository contains libraries to generate ε- and (ε, δ)-differentially private statistics over datasets. It contains the following tools.

- Privacy on Beam is an end-to-end differential privacy framework built on top of Apache Beam. It is intended to be easy to use, even by non-experts.
- Three "DP building block" libraries, in C++, Go, and Java. These libraries implement basic noise addition primitives and differentially private aggregations. Privacy on Beam is implemented using these libraries.
- A stochastic tester, used to help catch regressions that could make the differential privacy property no longer hold.
- A differential privacy accounting library, used for tracking privacy budget.
- A command line interface for running differentially private SQL queries with ZetaSQL.

To get started on generating differentially private data, we recommend you follow the Privacy on Beam codelab.

Currently, the DP building block libraries support the following algorithms:

Algorithm	C++	Go	Java
Laplace mechanism	Supported	Supported	Supported
Gaussian mechanism	Supported	Supported	Supported
Count	Supported	Supported	Supported
Sum	Supported	Supported	Supported
Mean	Supported	Supported	Supported
Variance	Supported	Supported	Supported
Standard deviation	Supported	Supported	Planned
Quantiles	Supported	Supported	Supported

github.com/google/differential-privacy

Parting Thoughts

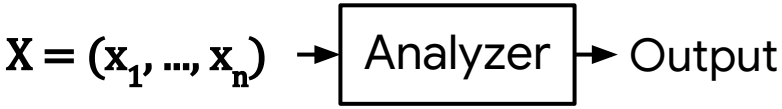
- Always think about privacy of the users
  - Even when we have consent, data must be used responsibly
  - Formal guarantees is preferred
    - E.g. DP
  - If not possible, then at least test against concrete attacks
- Keep up-to-date with the literature

Thank you!

Appendix

Local DP

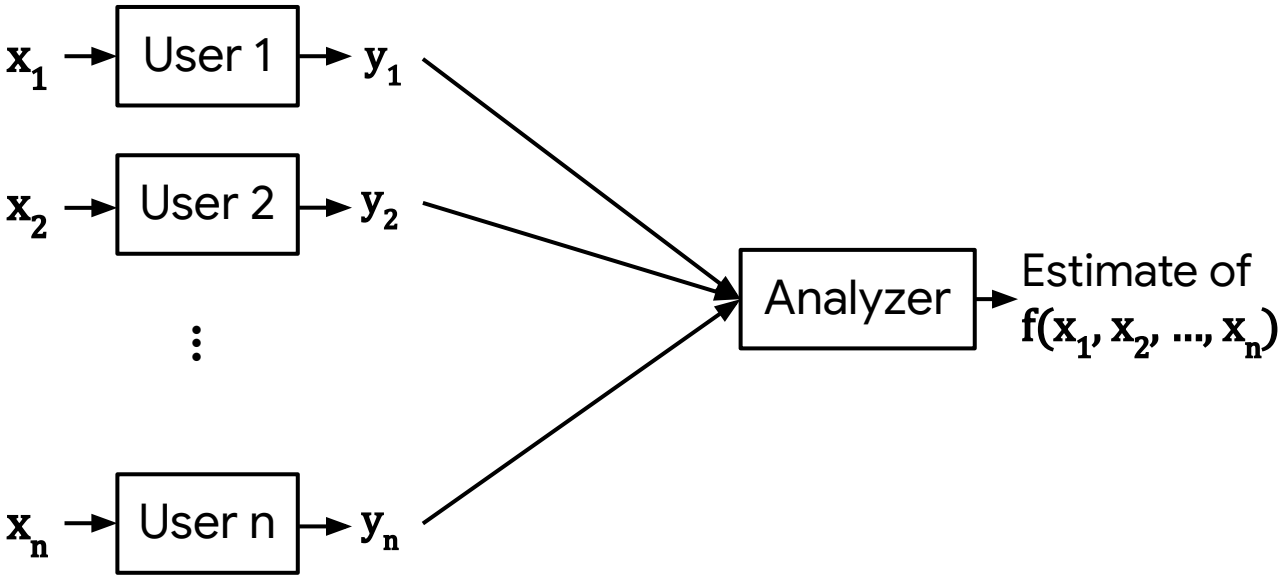
Model studied so far



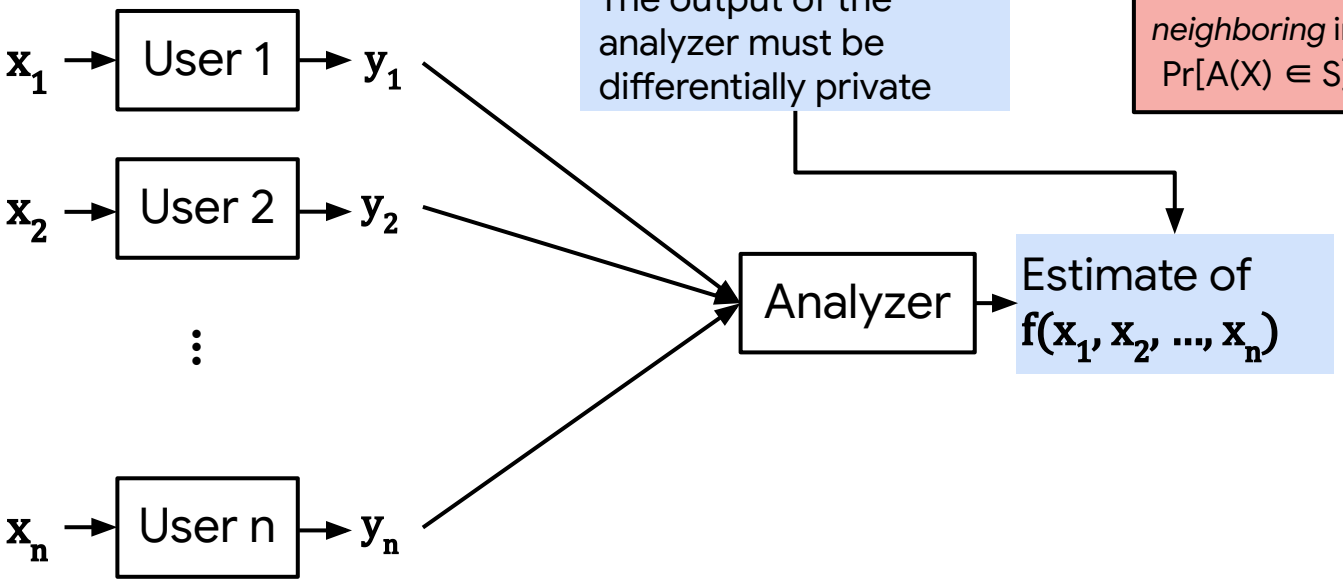
“Central DP”

- Analyzer gets to see raw data
- Undesirable if:
  - There is no trusted central authority
  - The analysis is done in distributed manner

Distributed Analytics



Differential Privacy: **Central Model**



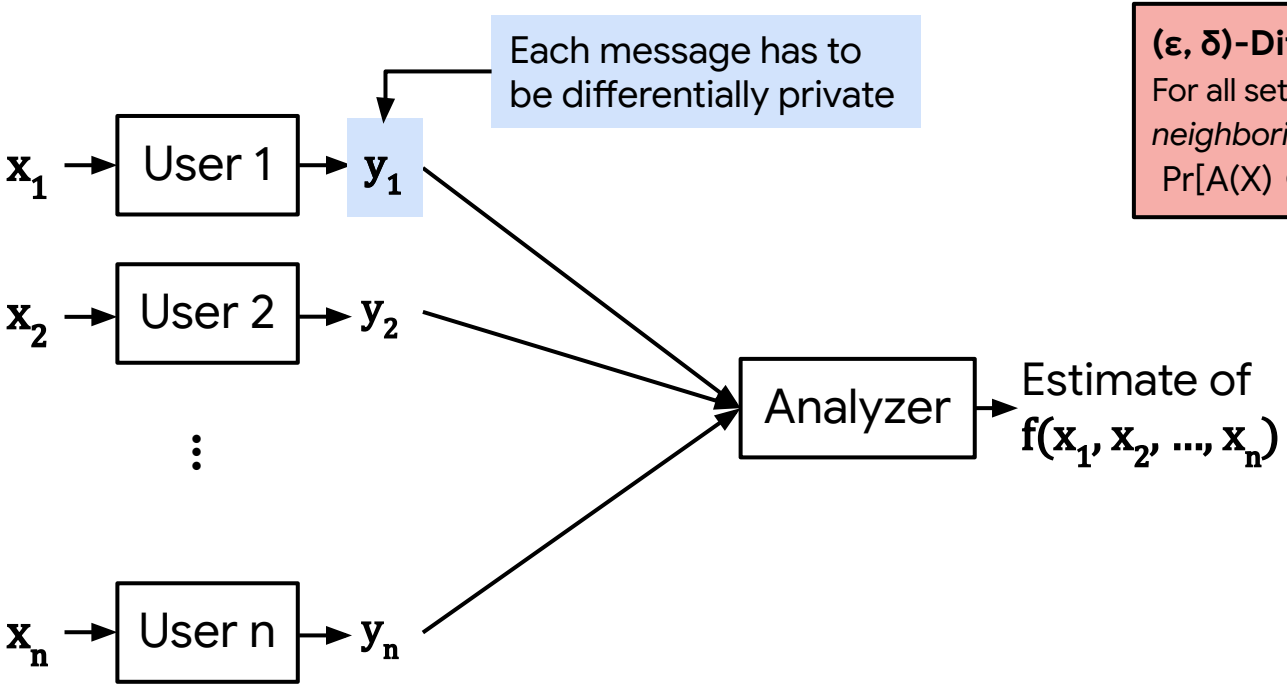
The output of the analyzer must be differentially private

(ε, δ)-Differential Privacy

For all set S of outcomes, and two *neighboring* input datasets X, X':

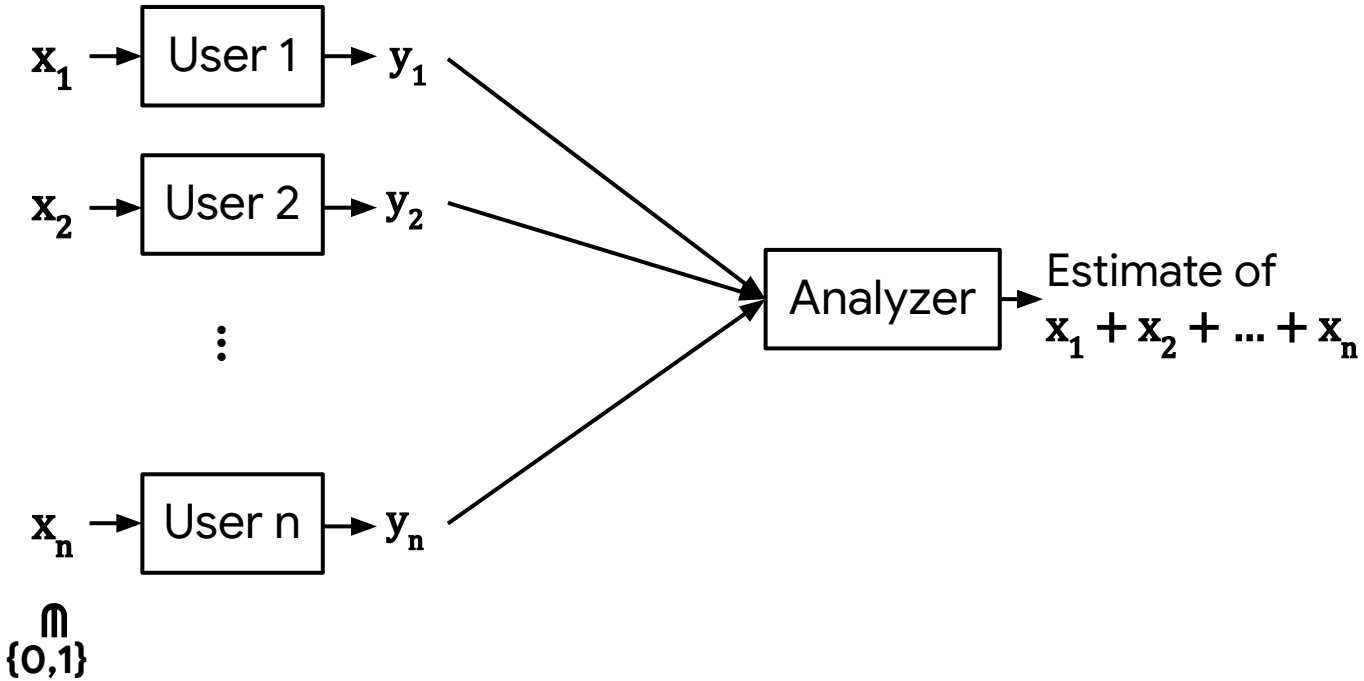
$\Pr[A(X) \in S] \leq e^{\varepsilon} \cdot \Pr[A(X') \in S] + \delta$

Differential Privacy: **Local Model** *[Kasiviswanathan et al.]*

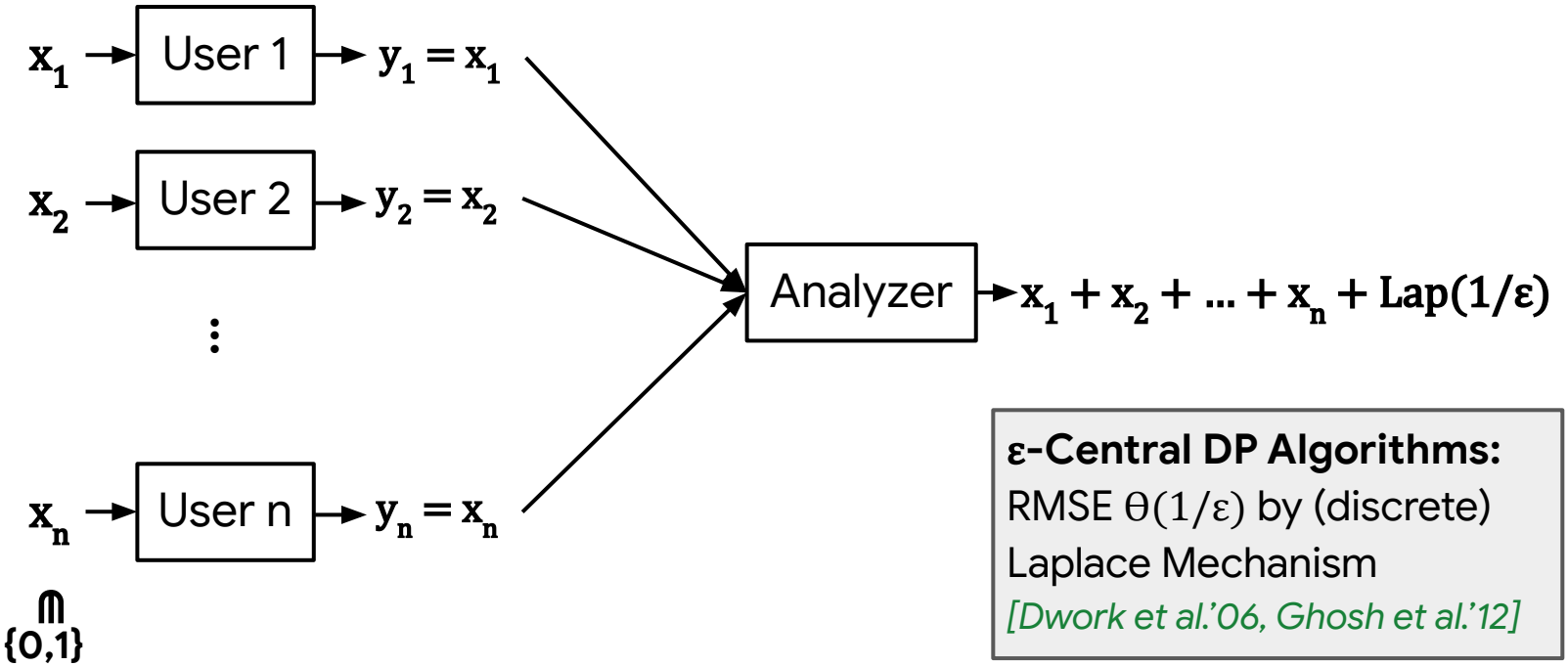


(ε, δ)-Differential Privacy
For all set S of outcomes, and two *neighboring* input datasets X, X':
$\Pr[A(X) \in S] \leq e^{\varepsilon} \cdot \Pr[A(X') \in S] + \delta$

Distributed Analytics: Counting

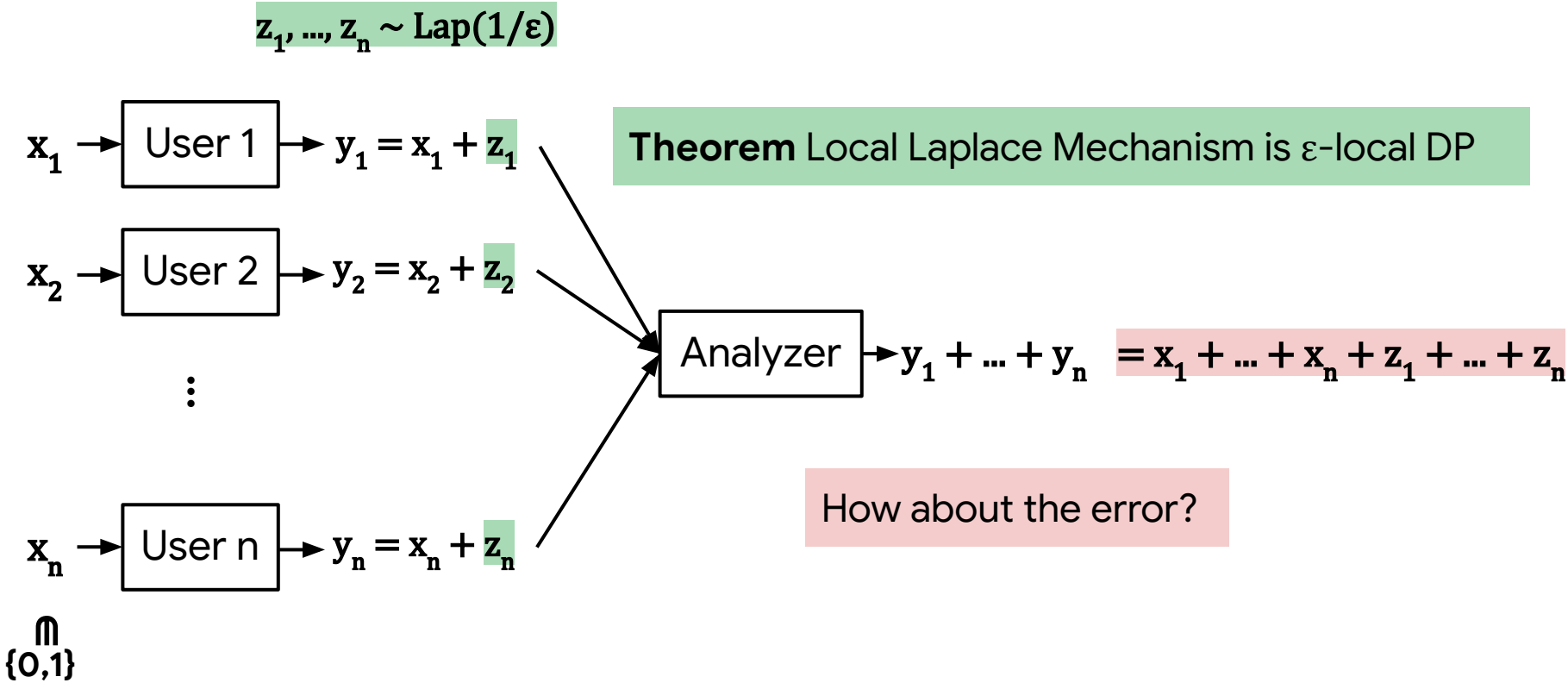


Counting with *Central* DP: Laplace Mechanism



ε-Central DP Algorithms:
RMSE $\Theta(1/\varepsilon)$ by (discrete) Laplace Mechanism
[Dwork et al.'06, Ghosh et al.'12]

Counting with *Local* DP: Laplace Mechanism



Theorem Local Laplace Mechanism is ε-local DP

How about the error?

Local Laplace Mechanism: Utility Analysis

Theorem Estimator from Local Laplace Mechanism has MSE $2n/\varepsilon^2$

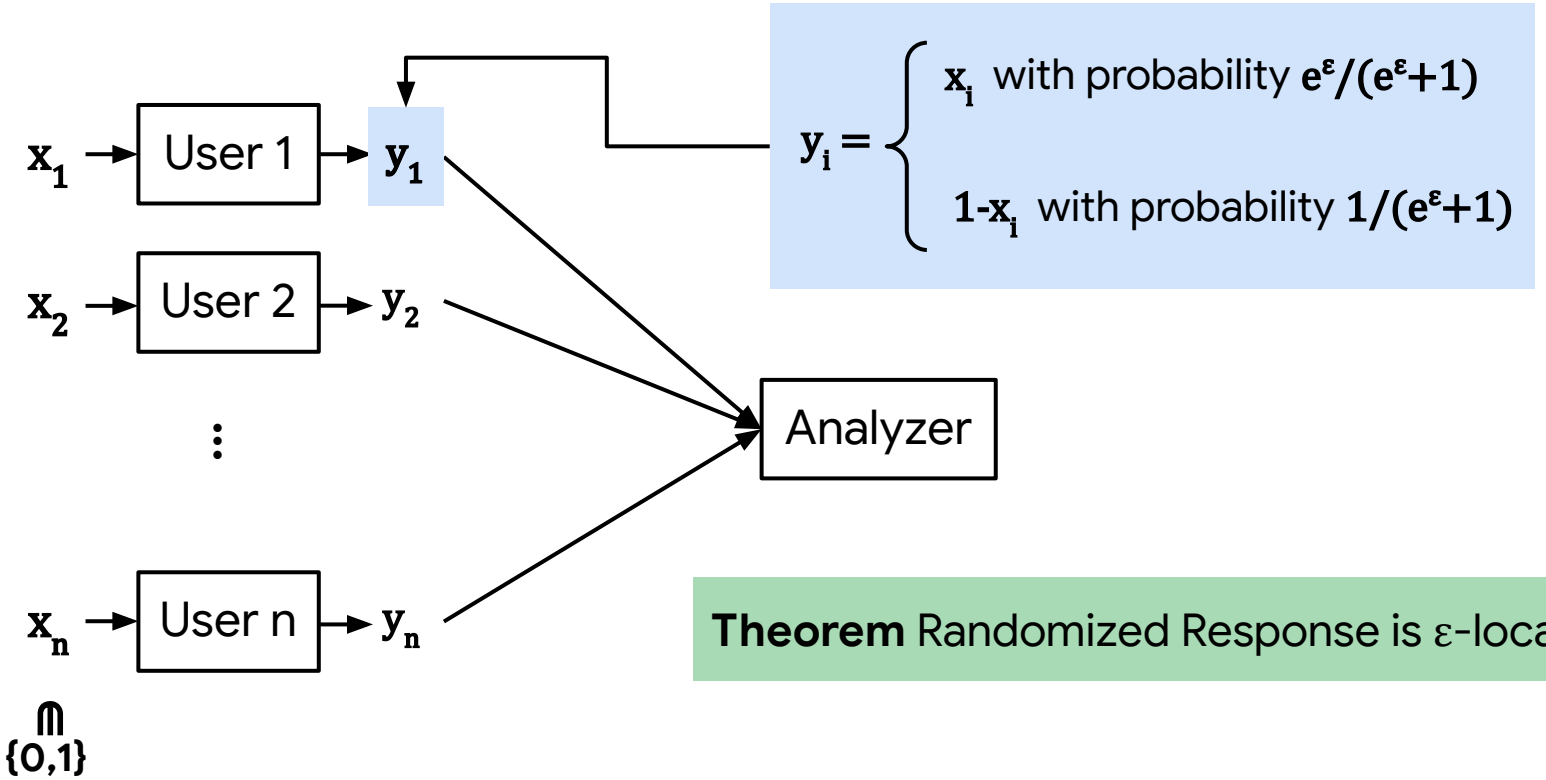
$\text{RMSE} = \sqrt{(2n)}/\varepsilon$

Proof

$$
\begin{aligned}
\text{MSE} &= \mathbf{E}[(\textstyle\sum_{i\in[n]} y_i - \sum_{i\in[n]} x_i)^2] \\
&= \mathbf{E}[(\textstyle\sum_{i\in[n]} z_i)^2] \\
&= \textstyle\sum_{i\in[n]} \mathbf{E}[z_i^2] \\
&= \textstyle\sum_{i\in[n]} \text{Var}(z_i) \\
&= n\,(2/\varepsilon^2)
\end{aligned}
$$

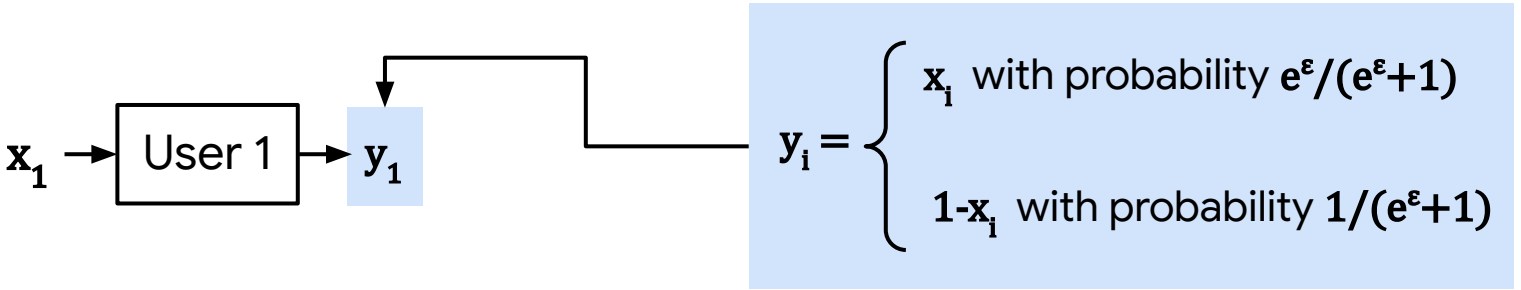
QED

Randomized Response (RR) *[Warner]*



Theorem Randomized Response is ε-local DP

Randomized Response (RR) *[Warner]*



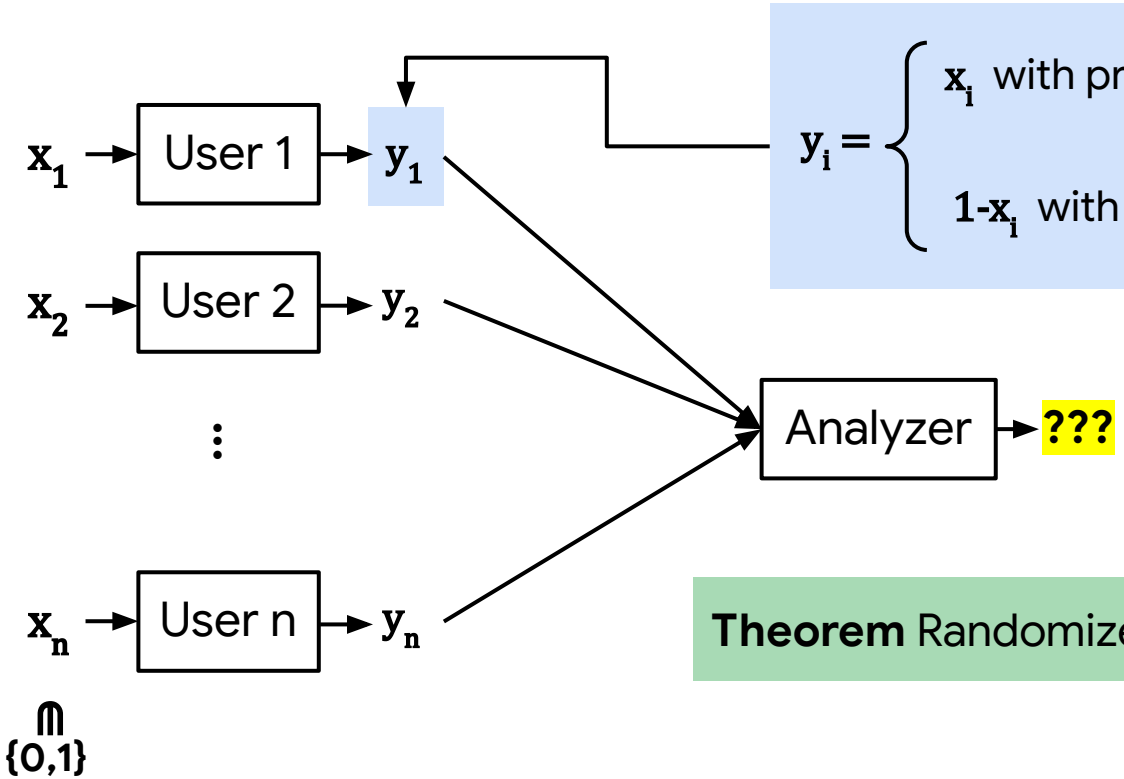
Theorem Randomized Response is ε-local DP

Proof

$$
\begin{aligned}
L_{M,X,X'}(o) &= \ln(\Pr[M(X) = o] \,/\, \Pr[M(X') = o]) \\
&\leq \ln((e^\varepsilon/(e^\varepsilon+1)) \,/\, (1/(e^\varepsilon+1))) \\
&= \varepsilon
\end{aligned}
$$

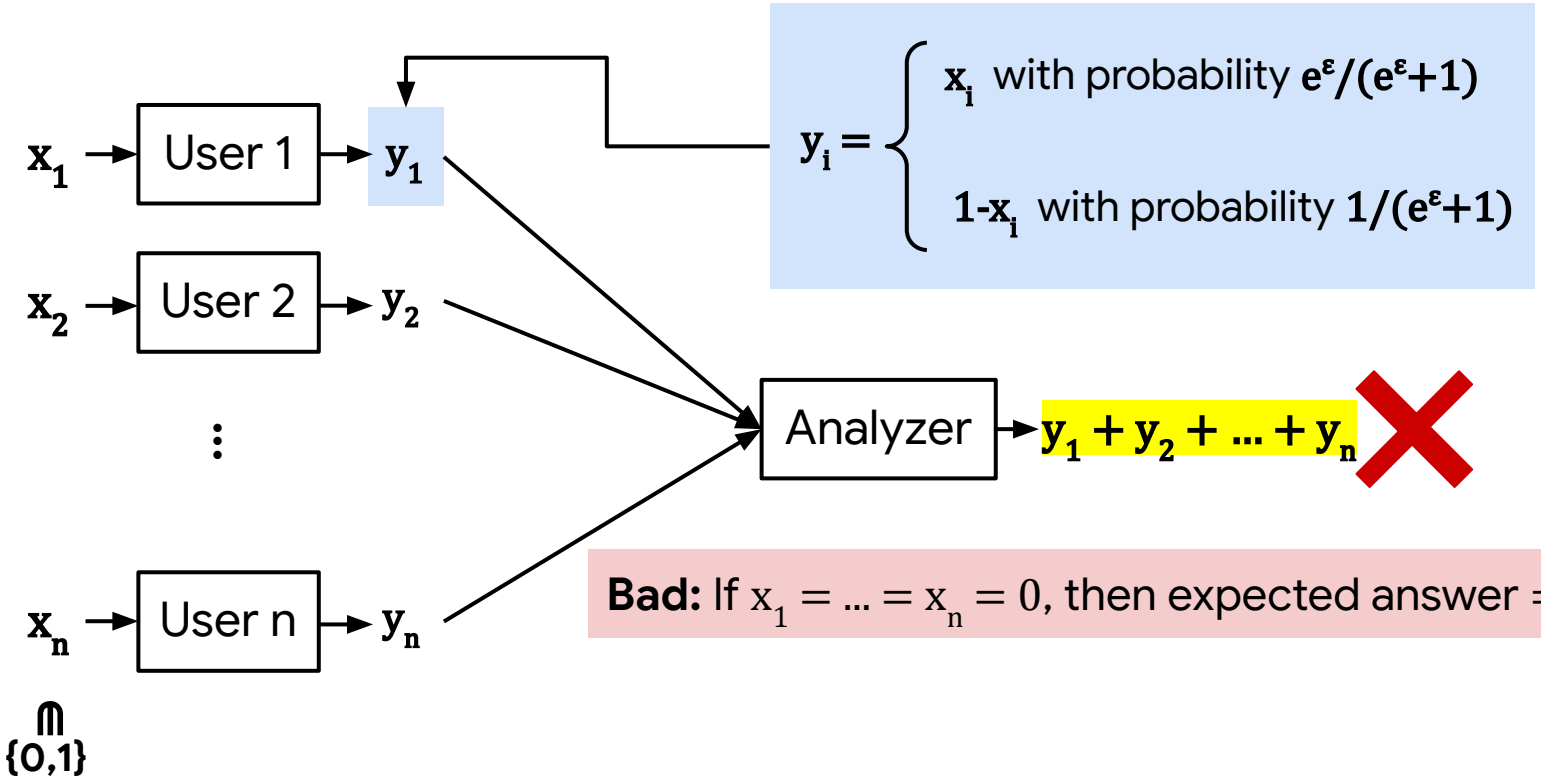
QED

Randomized Response (RR): Estimator

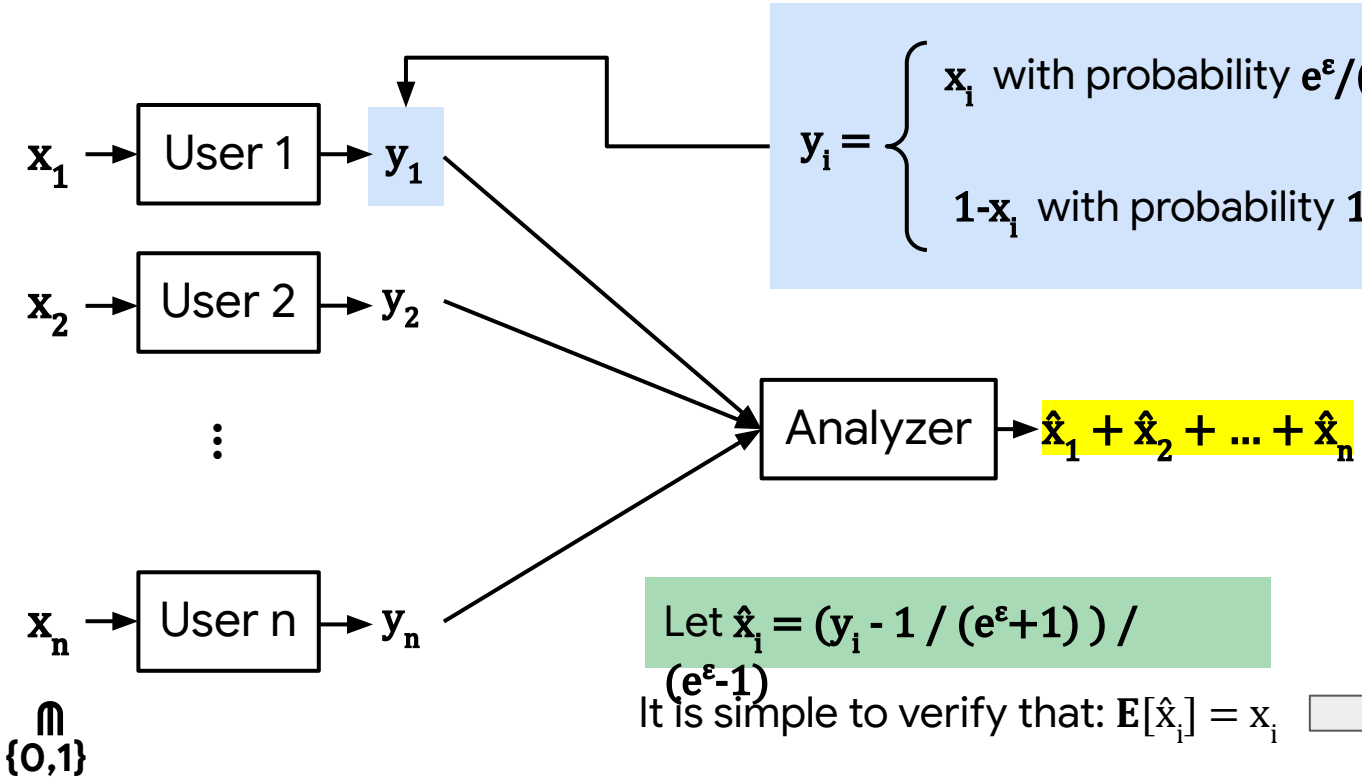


Theorem Randomized Response is ε-local DP

Randomized Response (RR): Estimator



Randomized Response (RR): Estimator



Let $\hat{x}_i = (y_i - 1/(e^\varepsilon+1)) / (e^\varepsilon-1)$

It is simple to verify that: $\mathbf{E}[\hat{x}_i] = x_i$

$\mathbf{E}[\hat{x}_1+\ldots+\hat{x}_n] = x_1+\ldots+x_n$

Randomized Response: Utility Analysis

Theorem Estimator from RR has MSE $n\, e^{\varepsilon} / (e^{2\varepsilon} - 1)^2$ $\quad \text{RMSE} = O(\sqrt{n}/\varepsilon)$

Proof It is simple to verify that:

- $\mathbf{E}[\hat{x}_i] = x_i$
- $\text{Var}[\hat{x}_i] = e^{\varepsilon} / (e^{2\varepsilon} - 1)^2$

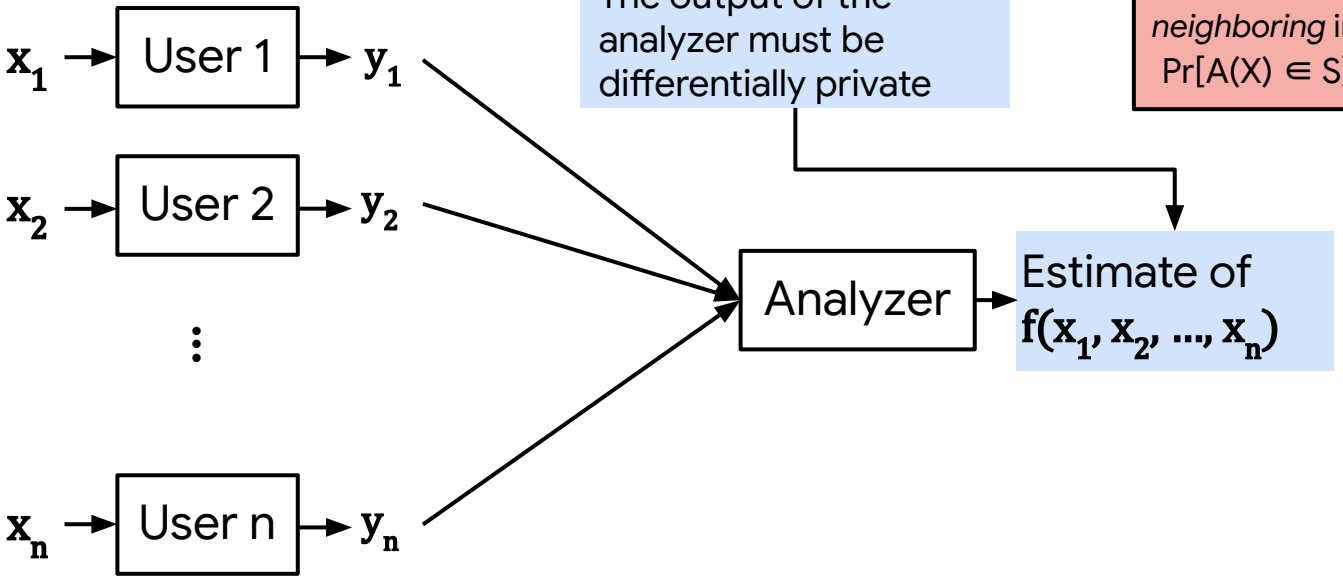
Thus,

$$\begin{aligned} \text{MSE} &= \mathbf{E}[(\textstyle\sum_{i\in[n]} \hat{x}_i - \sum_{i\in[n]} x_i)^2] \\ &= \textstyle\sum_{i\in[n]} \mathbf{E}[(\hat{x}_i - x_i)^2] \\ &= \textstyle\sum_{i\in[n]} \text{Var}(\hat{x}_i) \\ &= n\, e^{\varepsilon} / (e^{2\varepsilon} - 1)^2 \end{aligned}$$

QED

Advanced Topics:
Exponential Mechanisms

Back to the **Central Model**



The output of the analyzer must be differentially private

(ε, δ)-Differential Privacy

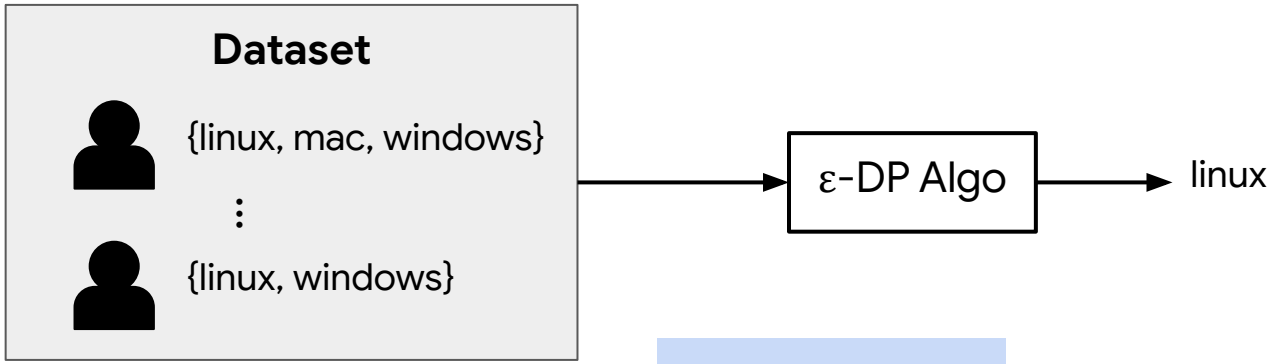
For all set S of outcomes, and two *neighboring* input datasets X, X':

$\Pr[A(X) \in S] \leq e^{\varepsilon} \cdot \Pr[A(X') \in S] + \delta$

Most Frequent Element

Each user has a subset of elements

Output: element that appears most frequently



Domain size = B

l_1-sensitivity = B

Baseline Algorithm

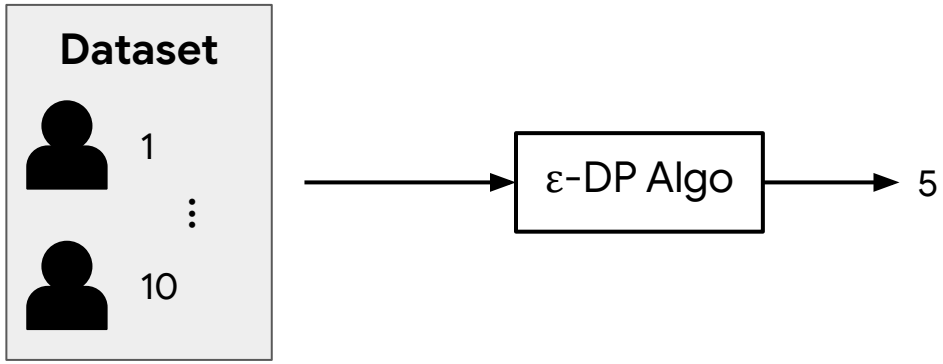
Compute histogram + Lap(B / ε)$^{\otimes B}$

Exponential Mechanism: a solution that only "adds" $O(\log B / \varepsilon)$ *noise!*

Median

Each user has a number in {1, …, B}

Output: the median



Dataset X:
- $n/2 + 1$ users have 1
- $n/2 - 1$ users have B

Dataset X':
- $n/2 - 1$ users have 1
- $n/2 + 1$ users have B

sensitivity = B - 1

Too noisy!

Baseline Algorithm

Compute median + Lap((B - 1) / ε)

Abstraction: Selection Problem

Input: A set H of "candidates"
For each $h \in H$, score $scr(h; X) \in \mathbf{R}$ of h
Output: h^* with approximately max score

$\Delta(scr) =$ Maximum sensitivity of $scr(h; \cdot)$ among all h
$= \max_{h \in H} \max_{X \simeq X'} |scr(h;X) - scr(h;X')|$

Most Frequent Element

- H = universe of all items
- $scr(h; X)$ = # times item h appears in X

$\Delta(scr)=1$

Median

- H = set of all possible input values
- $scr(h; X)$ = - (difference between # of input elements below h and above h)

$\Delta(scr)=2$

Exponential Mechanism *[McSherry-Talwar'07]*

Input: A set H of "candidates"
For each $h \in H$, score $scr(h; X) \in \mathbf{R}$ of h
Output: h^* with approximately max score

$\Delta(scr) =$ Maximum sensitivity of $scr(h; \cdot)$ among all h
$= \max_{h \in H} \max_{X \simeq X'} |scr(h;X) - scr(h;X')|$

Exponential Mechanism (EM)
[McSherry-Talwar'07]
ε-DP algorithm such that, w.p. 0.99,
$scr(h^*; X) \geq \max_{h \in H} scr(h; X) - O(\Delta(scr) \cdot \log |H| / \varepsilon)$

Exponential Mechanism *[McSherry-Talwar'07]*

Differentially Private Selection

Input: A set H of "candidates"
For each $h \in H$, score $scr(h; X) \in \mathbf{R}$ of h
Output: h^* with approximately max score

Exponential Mechanism (EM)

[McSherry-Talwar'07]

ε-DP algorithm such that, w.p. 0.99,

$$scr(h^*; X) \geq \max_{h \in H} scr(h; X) - O(\Delta(scr) \cdot \log |H| / \varepsilon)$$

Maximum sensitivity of $scr(h; \cdot)$ for fixed h

Examples

- Most frequent element: score = # of times the element occurs
- Median: score = difference between # of elements below h and above h

Comparison

- Trivial solution: Compute the score for all candidate
  - l_1-sensitivity: $|H| \cdot \Delta(scr)$
- Add Laplace noise
  - $\Rightarrow$ Error per coordinate $\Delta(scr) \cdot |H| / \varepsilon$

Exponential Mechanism

Exponential Mechanism (EM)

Output each $h \in H$ with probability

$\propto \exp(0.5\varepsilon \cdot \mathrm{scr}(h; X) \,/\, \Delta(\mathrm{scr}))$

Exponential Mechanism: Privacy Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Observation For any $X \simeq X'$, h, we have

$$\exp(\mathrm{scr}(h; X)/b) \leq e^{\varepsilon/2} \cdot \exp(\mathrm{scr}(h; X')/b),$$
$$Z(X') \leq e^{\varepsilon/2} \cdot Z(X)$$

Theorem Exponential Mechanism is ε-DP.

Proof For all $X \simeq X'$, $h \in H$,

$$\begin{aligned} L_{M,X,X'}(o) &= \ln(\Pr[M(X) = h] / \Pr[M(X') = h]) \\ &= \ln(\exp(\mathrm{scr}(h; X)/b) / \exp(\mathrm{scr}(h; X')/b)) + \ln(Z(X') / Z(X)) \\ &\leq \varepsilon / 2 + \varepsilon / 2 \qquad \text{(Observation)} \\ &= \varepsilon \end{aligned}$$

QED

Exponential Mechanism: Utility Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Theorem With probability 0.99, EM output h^* such that $\mathrm{scr}(h^*; X) \geq \max_{h \in H} \mathrm{scr}(h; X) - 10b \cdot \log |H|$

Proof Let $\mathrm{scr}^* = \max_{h \in H} \mathrm{scr}(h; X)$ and $H_{bad} = \{h' \mid \mathrm{scr}(h'; X) < \mathrm{scr}^* - 10b \cdot \log |H|\}$.

$$\Pr[\text{EM outputs element of } H_{bad}] = \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) / Z(X)$$

Exponential Mechanism: Utility Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Theorem With probability 0.99, EM output h^* such that $\mathrm{scr}(h^*; X) \geq \max_{h \in H} \mathrm{scr}(h; X) - 10b \cdot \log |H|$

Proof Let $\mathrm{scr}^* = \max_{h \in H} \mathrm{scr}(h; X)$ and $H_{bad} = \{h' \mid \mathrm{scr}(h'; X) < \mathrm{scr}^* - 10b \cdot \log |H|\}$.

$$\Pr[\text{EM outputs element of } H_{bad}] = \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) / Z(X)$$
$$\leq \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) / \exp(\mathrm{scr}^*/b)$$

Exponential Mechanism: Utility Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Theorem With probability 0.99, EM output h^* such that $\mathrm{scr}(h^*; X) \geq \max_{h \in H} \mathrm{scr}(h; X) - 10b \cdot \log |H|$

Proof Let $\mathrm{scr}^* = \max_{h \in H} \mathrm{scr}(h; X)$ and $H_{bad} = \{h' \mid \mathrm{scr}(h'; X) < \mathrm{scr}^* - 10b \cdot \log |H|\}$.

$$\begin{aligned} \Pr[\text{EM outputs element of } H_{bad}] &= \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) / Z(X) \\ &\leq \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) / \exp(\mathrm{scr}^*/b) \\ &\leq \sum_{h' \in H_{bad}} \exp(-10 \cdot \log |H|) \end{aligned}$$

Exponential Mechanism: Utility Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Theorem With probability 0.99, EM output h^* such that $\mathrm{scr}(h^*; X) \geq \max_{h \in H} \mathrm{scr}(h; X) - 10b \cdot \log |H|$

Proof Let $\mathrm{scr}^* = \max_{h \in H} \mathrm{scr}(h; X)$ and $H_{bad} = \{h' \mid \mathrm{scr}(h'; X) < \mathrm{scr}^* - 10b \cdot \log |H|\}$.

$$
\begin{aligned}
\Pr[\text{EM outputs element of } H_{bad}] &= \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) \,/\, Z(X) \\
&\leq \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) \,/\, \exp(\mathrm{scr}^*/b) \\
&\leq \sum_{h' \in H_{bad}} \exp(-10 \cdot \log |H|) \\
&\leq \sum_{h' \in H_{bad}} 0.01 \,/\, |H|
\end{aligned}
$$

Exponential Mechanism: Utility Proof

Exponential Mechanism (EM)

Output each $h \in H$ with probability $\exp(\mathrm{scr}(h; X)/b)/Z(X)$ where $b = 2\Delta(\mathrm{scr})/\varepsilon$ and $Z(X) = \sum_{h' \in H} \exp(\mathrm{scr}(h; X) / b)$

Theorem With probability 0.99, EM output h^* such that $\mathrm{scr}(h^*; X) \geq \max_{h \in H} \mathrm{scr}(h; X) - 10b \cdot \log |H|$

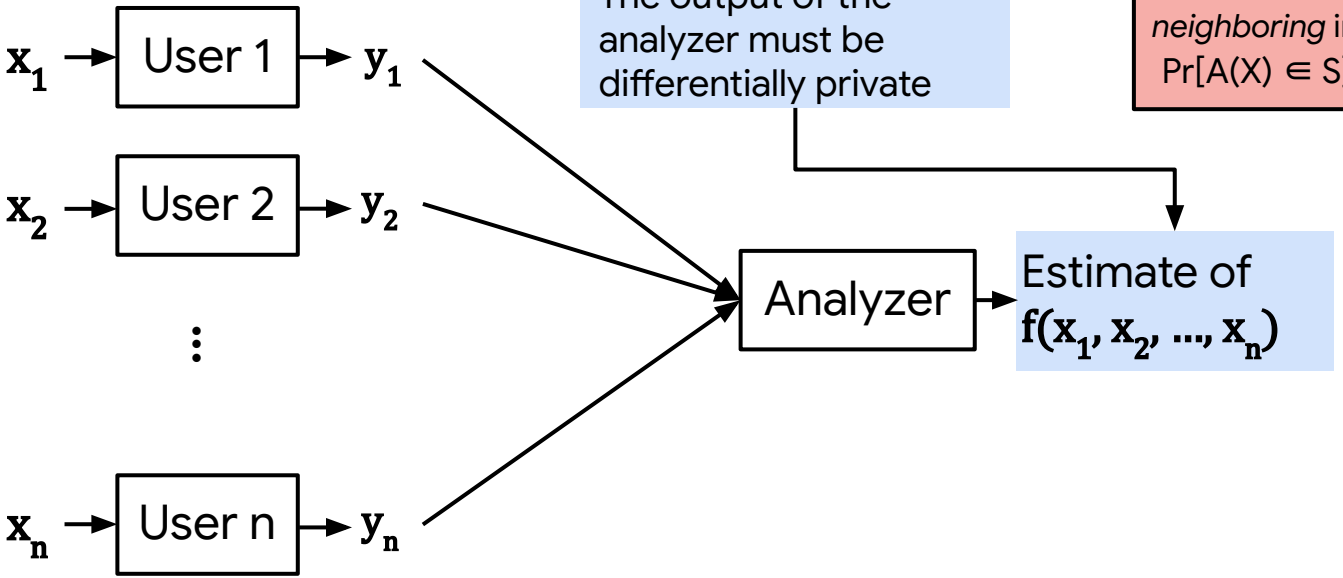
Proof Let $\mathrm{scr}^* = \max_{h \in H} \mathrm{scr}(h; X)$ and $H_{bad} = \{h' \mid \mathrm{scr}(h'; X) < \mathrm{scr}^* - 10b \cdot \log |H|\}$.

$$
\begin{aligned}
\Pr[\text{EM outputs element of } H_{bad}] &= \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) \,/\, Z(X) \\
&\leq \sum_{h' \in H_{bad}} \exp(\mathrm{scr}(h'; X)/b) \,/\, \exp(\mathrm{scr}^*/b) \\
&\leq \sum_{h' \in H_{bad}} \exp(-10 \cdot \log |H|) \\
&\leq \sum_{h' \in H_{bad}} 0.01 \,/\, |H| \\
&\leq 0.01
\end{aligned}
$$

QED

Advanced Topics:
Shuffle Model

Recall: **Central Model**



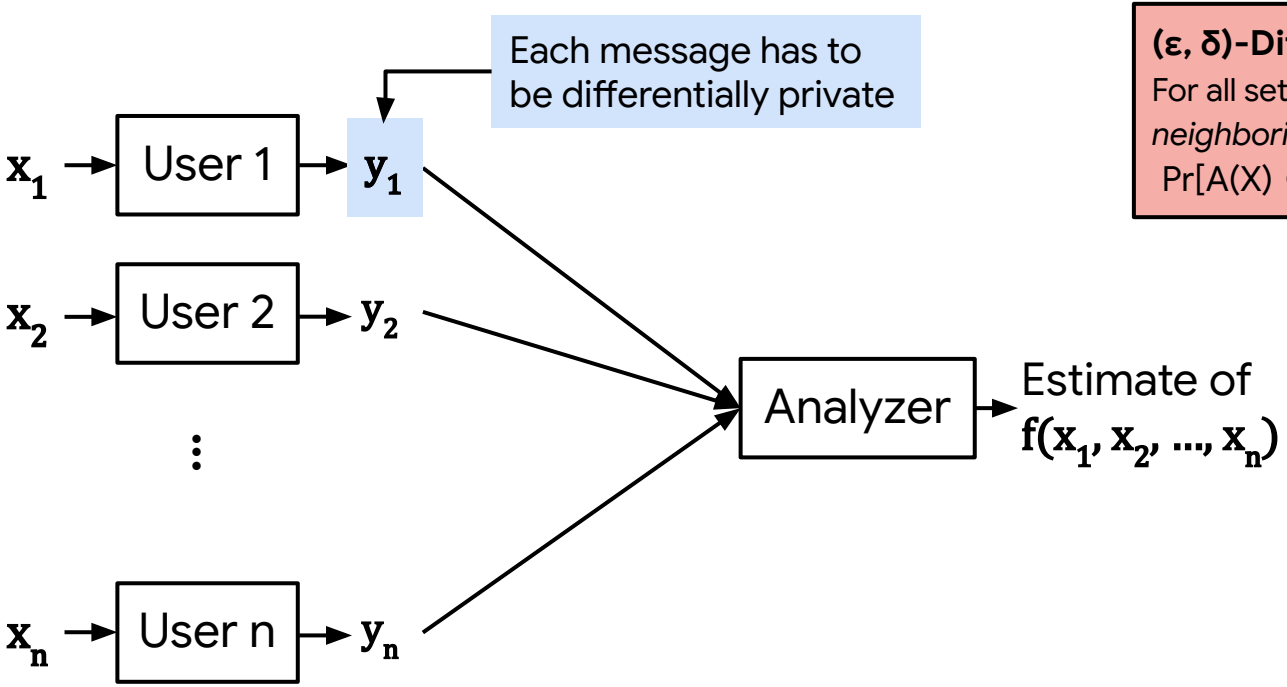
The output of the analyzer must be differentially private

(ε, δ)-Differential Privacy

For all set S of outcomes, and two *neighboring* input datasets X, X':

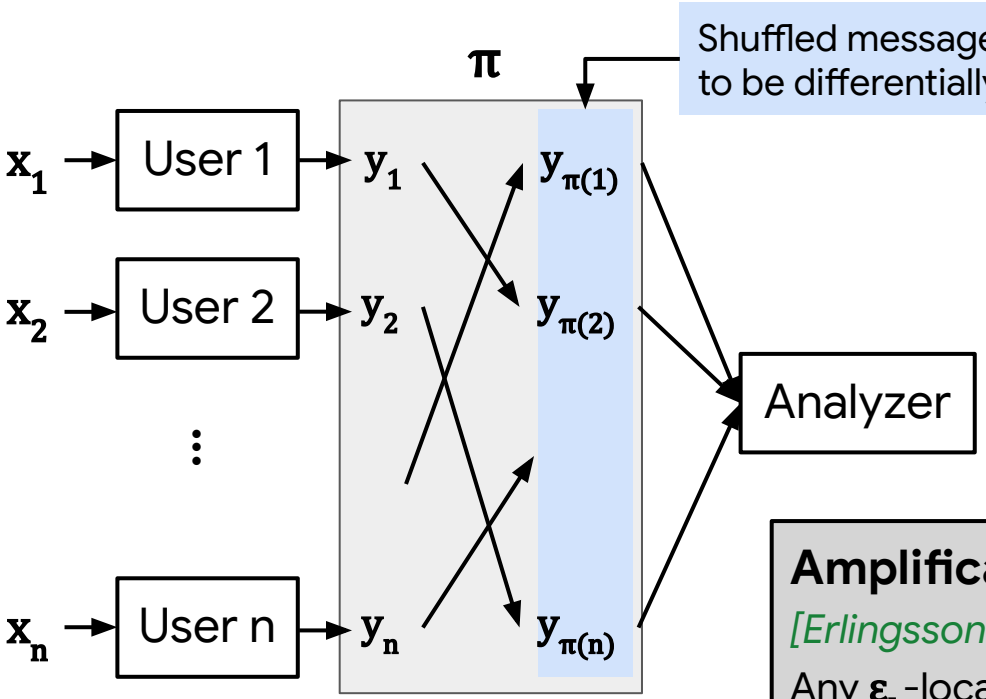
$\Pr[A(X) \in S] \leq e^{\varepsilon} \cdot \Pr[A(X') \in S] + \delta$

Differential Privacy: **Local Model** *[Kasiviswanathan et al.]*



(ε, δ)-Differential Privacy
For all set S of outcomes, and two *neighboring* input datasets X, X':
$\Pr[A(X) \in S] \leq e^{\varepsilon} \cdot \Pr[A(X') \in S] + \delta$

Differential Privacy: **Shuffled Model** *[Bittau et al., Erlingsson et al.]*



Shuffled messages have to be differentially private

≡

Multiset of messages have to be differentially private

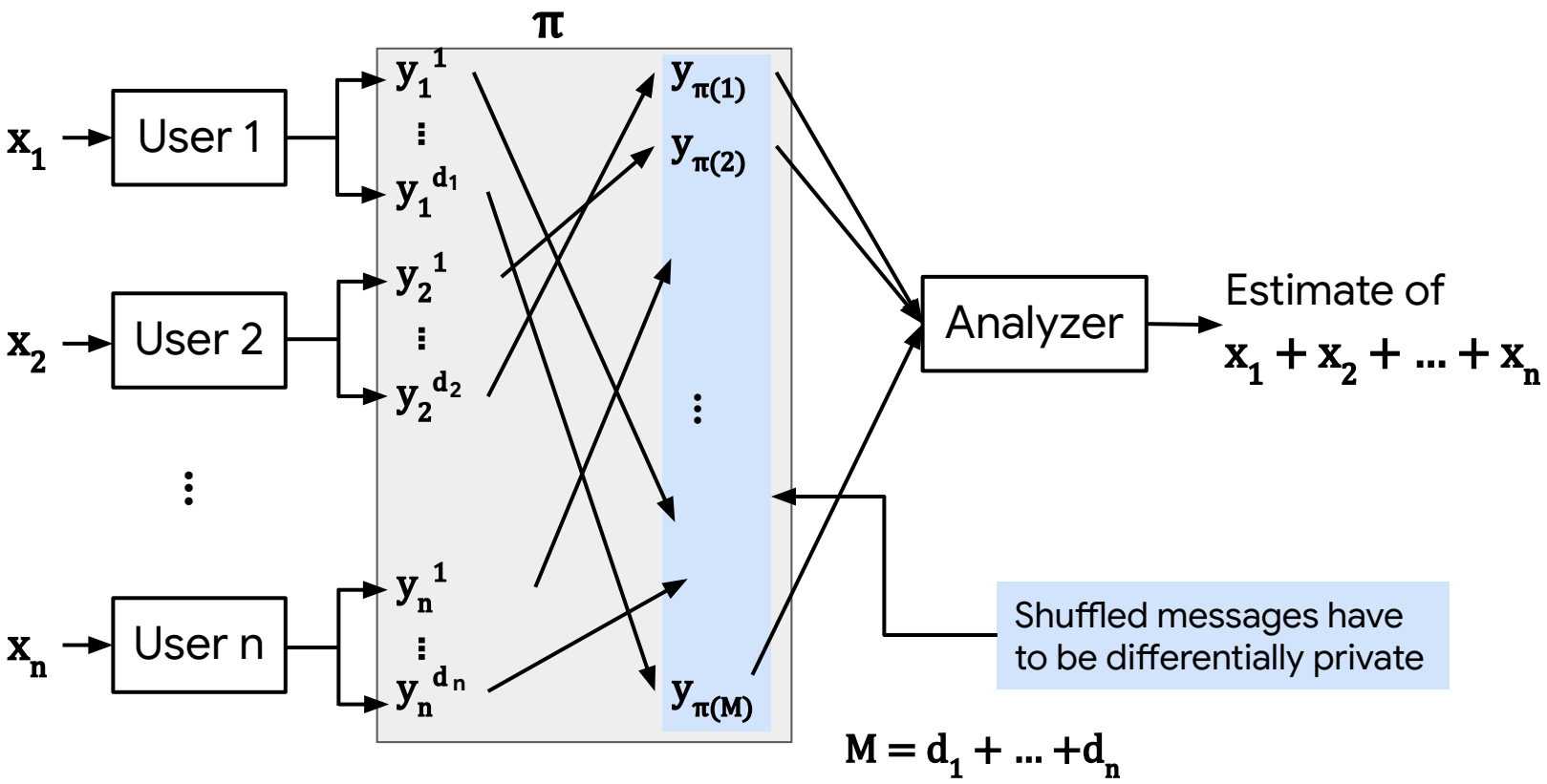
- **Anonymous Model** *[Ishai et al.]*
- **Encode-Shuffle-Analyze Architecture** *[Bittau et al.]*
- **Shuffled/Shuffle Model** *[Cheu et al., Erlingsson et al.]*

Amplification by Shuffling (Informal)

[Erlingsson et al., Balle et al., Feldman et al.]

Any ε_L-local DP algorithm is (ε, δ)-shuffled DP for $\varepsilon \ll \varepsilon_L$ and for reasonable value of δ.

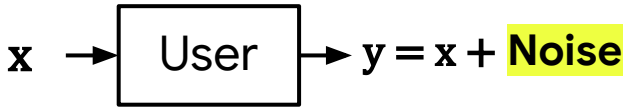
Shuffled Model: Multi-Message Setting



DP Summation in Shuffle Model

Single-Message

Randomizer:



Analyzer:

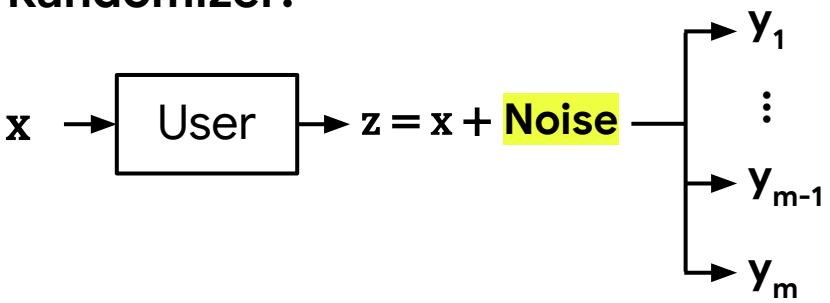
Just sum all messages up!

Benefit over Local DP:
Amplification by Shuffling

Multi-Message:

Split-and-Mix Protocol *[Balle et al.'19]*

Randomizer:



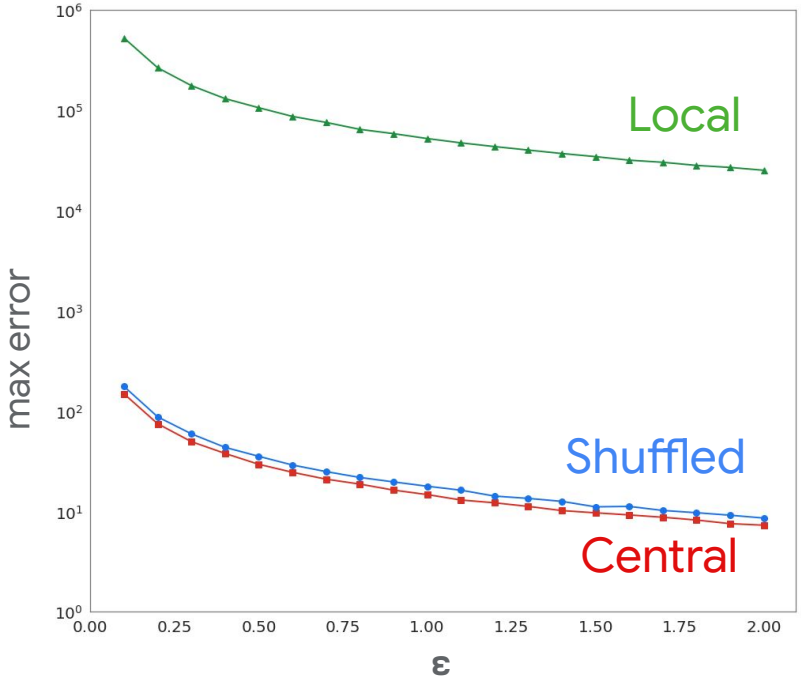
$y_1, \ldots, y_m$ randomly selected so that $y_1 + \ldots + y_m = z$

As good accuracy as central DP!

DP Summation in Shuffle Model

Experiment (IPUMS 1940 City Dataset)

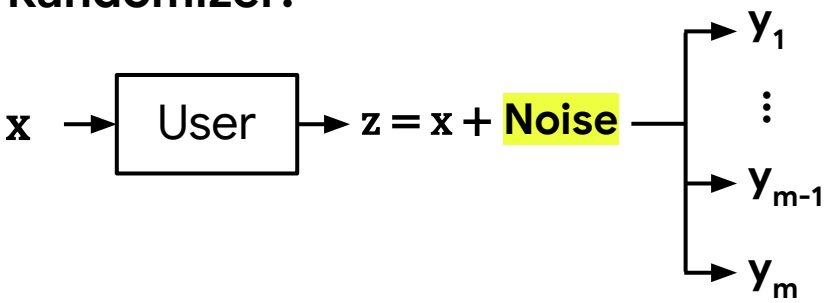
Parameters: n ≈ 60M, B = 915, □ = 2 * 10^{-9}



Multi-Message:

Split-and-Mix Protocol *[Balle et al.'19]*

Randomizer:

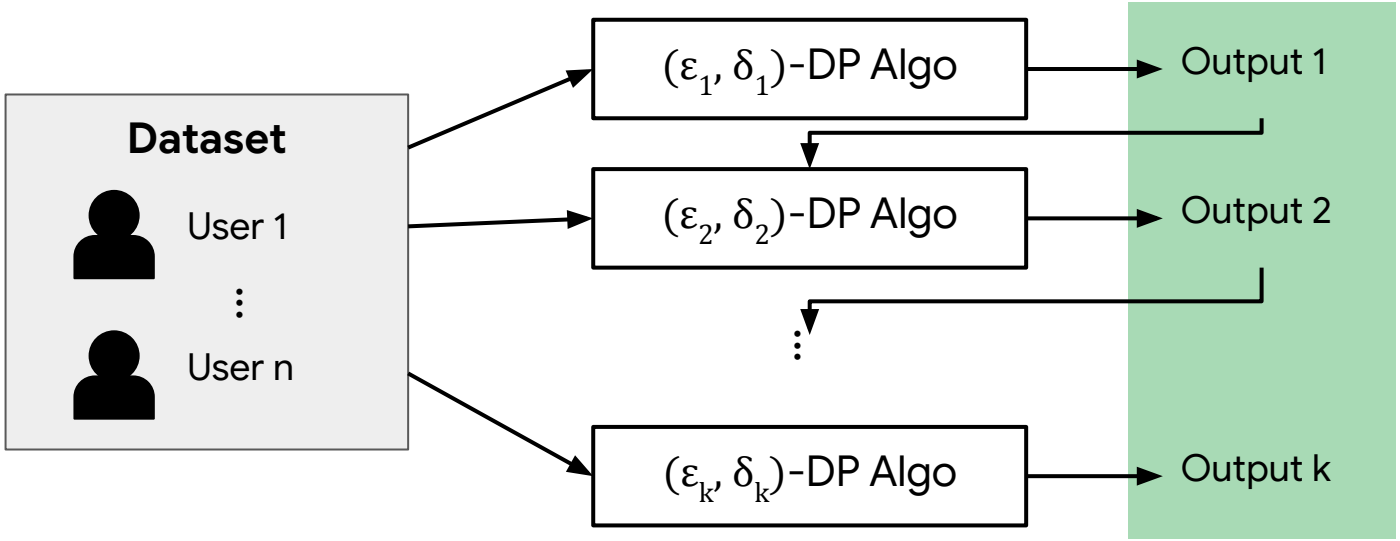


$y_1, \ldots, y_m$ randomly selected so that $y_1 + \ldots + y_m = z$

As good accuracy as central DP!

Advanced Topics:
Privacy Accounting

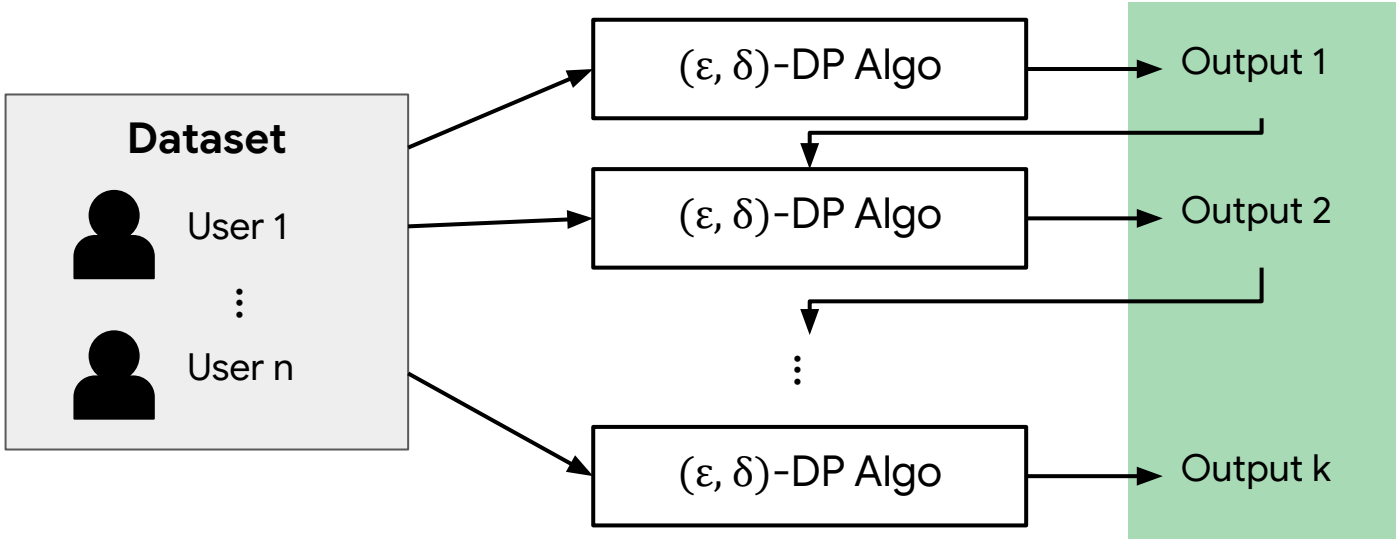
Basic Composition



Basic Composition Theorem *[Dwork et al.]*

All the outputs combined remain $(\varepsilon_1+\varepsilon_2+...+\varepsilon_k, \delta_1+\delta_2+...+\delta_k)$-DP

Advanced Composition



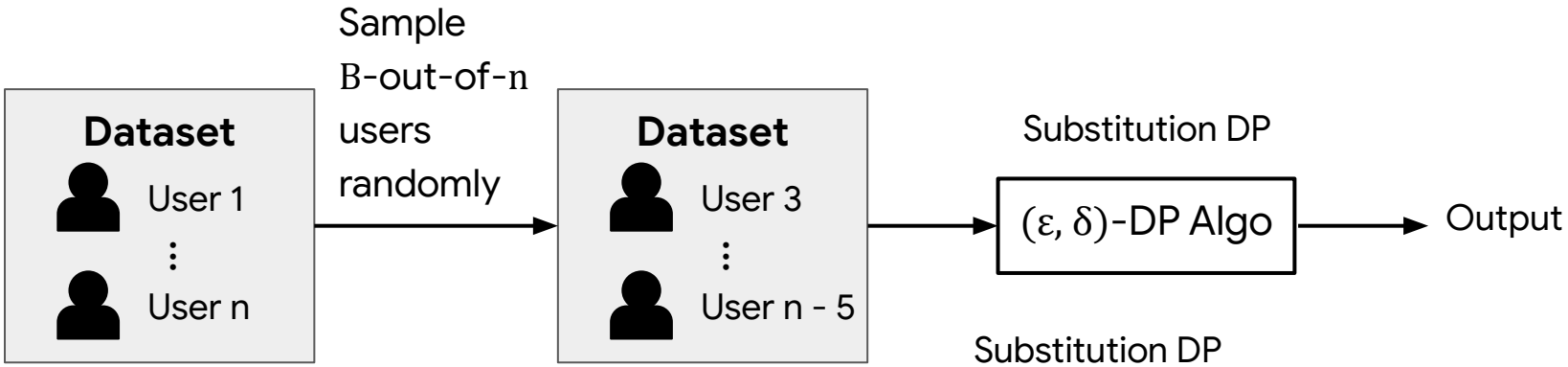
Advanced Composition Theorem *[Dwork et al.]*
All the outputs combined is $(\varepsilon', k\delta + \delta')$-DP where

$$\varepsilon' = \sqrt{2k \ln(1/\delta')} \cdot \epsilon + k\epsilon(e^{\epsilon} - 1)$$

Can be improved for specific algorithms!

Amplification by Subsampling

"Subsampling makes the algorithm more private."



Amplification-by-subsampling Theorem

The output combined is (ε', δ')-DP where

$$\varepsilon' = \ln(1 + p(e^{\varepsilon} - 1)), \qquad \delta' = p\delta$$

with $p = B / n$.

Can be improved for specific algorithms!

Renyi-DP *[Abadi et al'16, Mironov'17]*

Renyi-DP (RDP)

A mechanism M is (α, ρ)-*RDP* if, for all neighboring datasets X, X',

$$D_\alpha(M(X)||M(X')) \leq \rho$$

where $D_\alpha(P||Q)$ denotes the α-Renyi divergence.

$$D_\alpha(P||Q) = E_{x\sim P}[(P(x)/Q(x))^{\alpha-1}] / (\alpha-1)$$

Differential Privacy (DP)

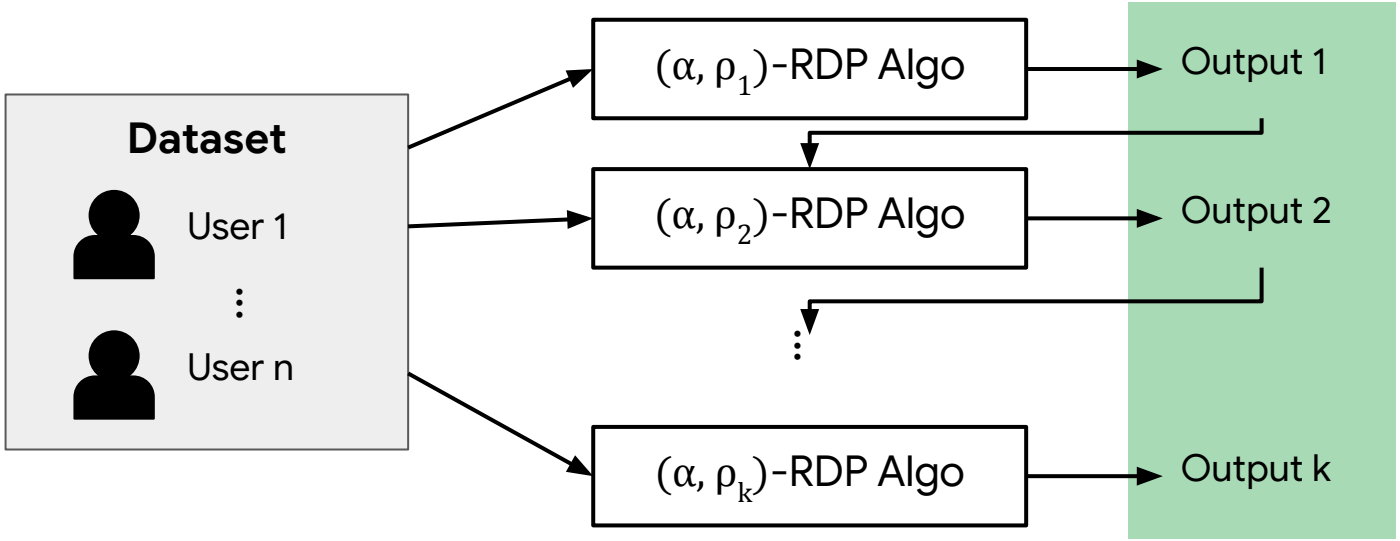
M is ε-*DP* if it is (∞, ε)-*RDP*.

RDP ⇒ DP Conversion

(α, ρ)-*RDP* ⇒ (ε, δ)-DP for

$$\delta = e^{(\alpha-1)(\alpha\rho - \varepsilon)} \cdot (1 - 1/\alpha)^\alpha / (\alpha-1)$$

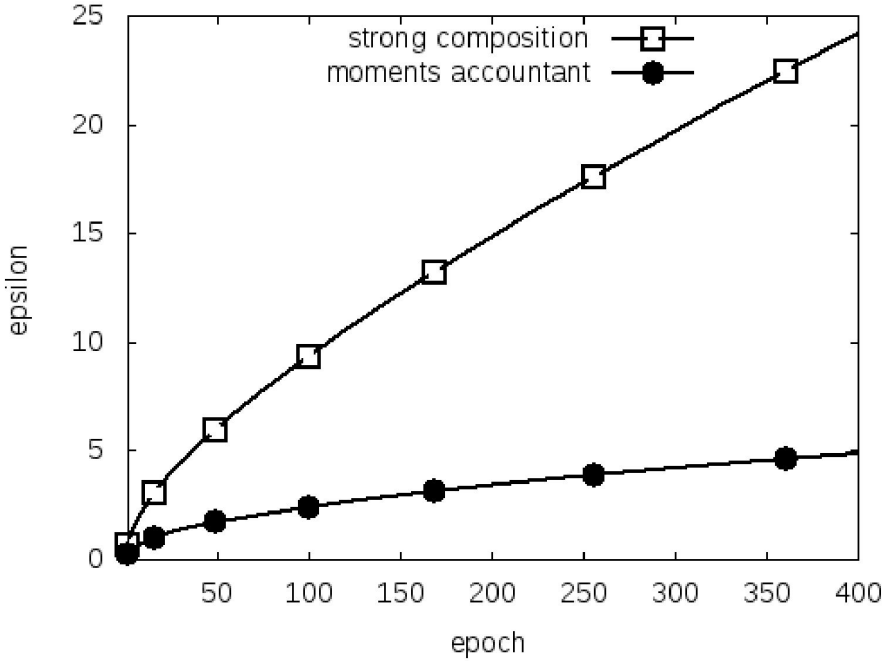
RDP Composition



RDP Composition Theorem *[Mironov]*
All the outputs combined remain $(\alpha, \rho_1+\rho_2+...+\rho_k)$-RDP

- Can be translated to DP using formula from previous slide
- Also some RDP bounds on amplification-by-subsampling

RDP vs Advanced Composition (DP-SGD)



Abadi et al.: *Deep Learning with Differential Privacy*. CCS'16

Privacy Loss Distribution (PLD)

Discrete outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(\Pr[M(X) = o] / \Pr[M(X') = o])$$

Continuous outputs

Privacy Loss

For M, X, X', the privacy loss at output o is

$$L_{M,X,X'}(o) = \ln(f_{M(X)}(o) / f_{M(X')}(o))$$

Privacy Loss Distribution

$PLD_{M,X,X'}$ is the distribution of $L_{M,X,X'}(o)$ where $o \sim M(X)$

Tight composition theorems by tracking the PLD!

Pure-DP Condition

M is ε-DP iff, for all $X \asymp X'$, $o \in Range(M)$,

$$L_{M,X,X'}(o) \leq \varepsilon$$

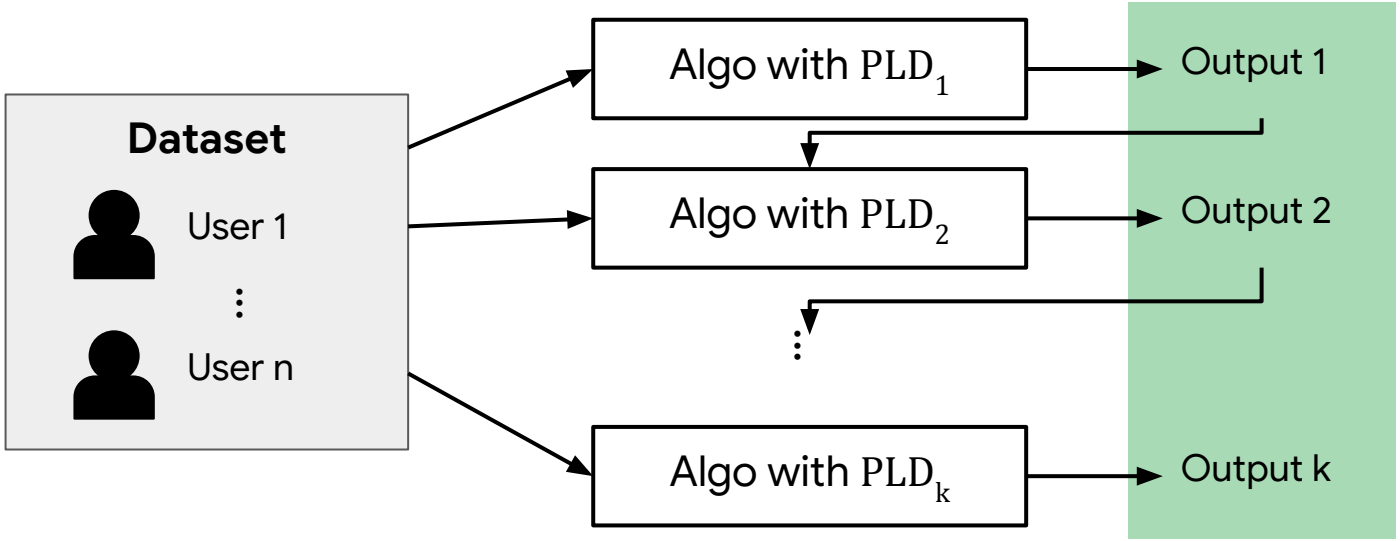
Approximate-DP Condition

M is (ε, δ)-DP iff, for all $X \asymp X'$,

$$E_y[\,[1 - e^{\varepsilon - y}]_+\,] \leq \delta$$

where $y \sim PLD_{M,X,X'}$ and $[a]_+ := \max\{a, 0\}$

PLD Composition*



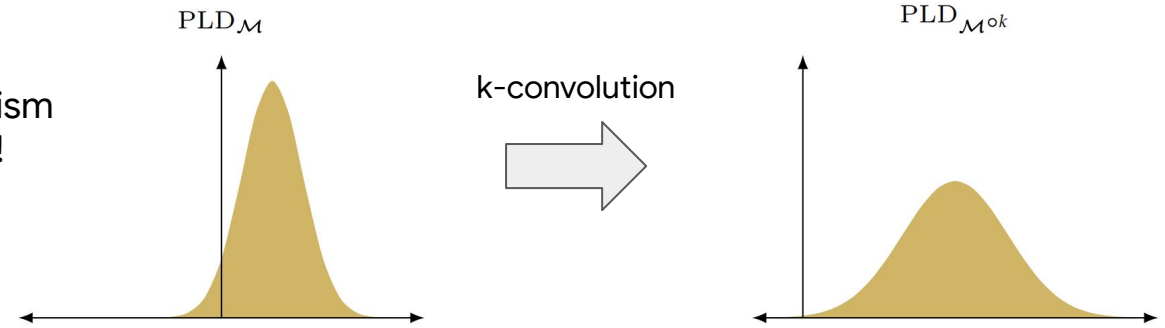
PLD Composition Theorem *[Dwork et al.]*

PLD of all the outputs combined is the convolution of PLD_1, ..., PLD_k

PLD: Examples

PLD after k-fold composition

- $\mathcal{M}$: Gaussian Mechanism
- PLD: Also a Gaussian!

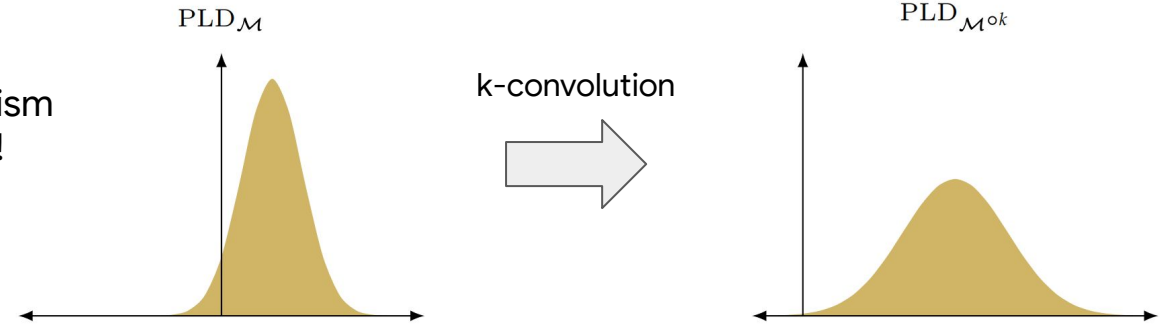


- **Issue:** can't represent continuous distributions...

PLD: Examples

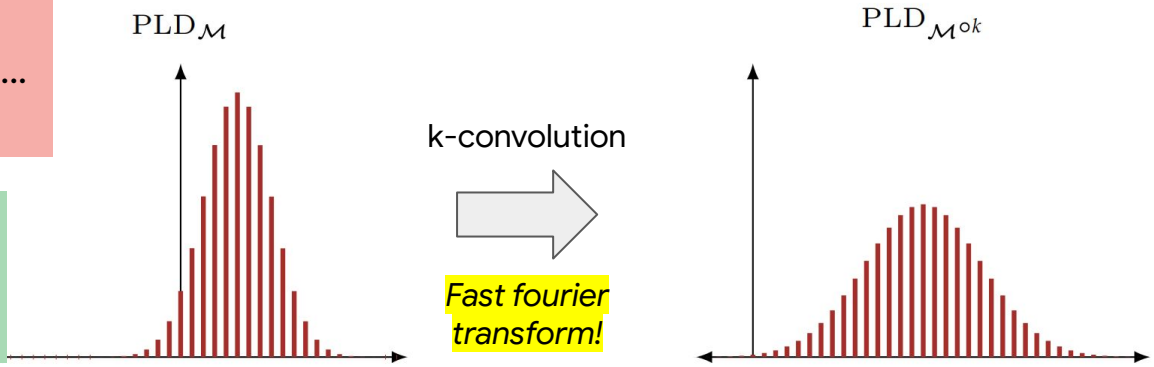
PLD after k-fold composition

- M: Gaussian Mechanism
- PLD: Also a Gaussian!

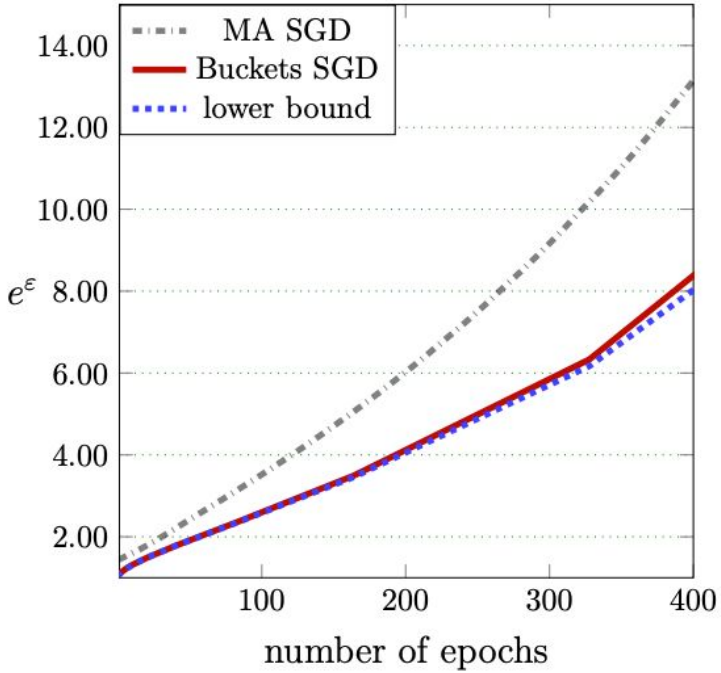


- **Issue:** can't represent continuous distributions...
- **Solution:** discretize!

Active Research:
- Discretization methods
- Error Bounds



PLD vs RDP

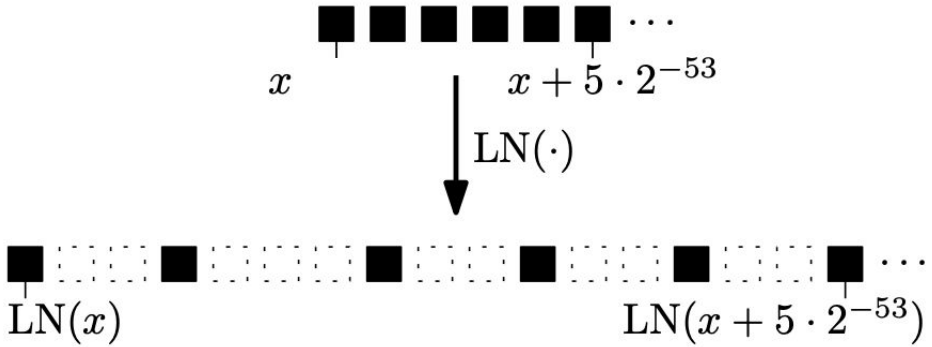


Meiser & Mohammadi: *Tight on Budget? Tight Bounds for r-Fold Approximate Differential Privacy*. CCS'18

Advanced Topics:
Side-Channel Attacks

Side-Channel Attacks

- Laplace distribution sampling in practice:
  - Involves taking log of uniform [0, 1] random variable
  - Taking log creates “holes”

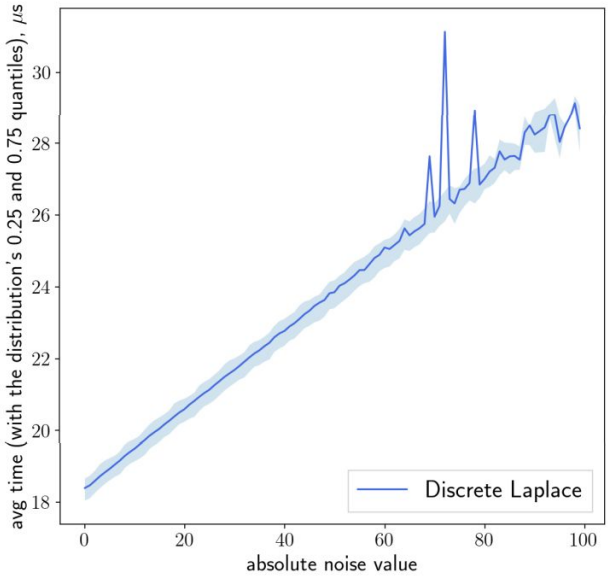


Can distinguish D, D’ if $g(D), g(D')$ have different least significant bits!

Mironov: *On Significance of the Least Significant Bits For Differential Privacy*. CCS’12

Side-Channel Attacks (cont)

- Timing attack:
  - Laplace noise sampling
    - Longer time ⇒ Larger Noise
  - Launching the attack:
    - Observing algorithm running time
    - ⇒ can approximate the noise
    - ⇒ subtract the approximate noise



Jin et al.: *Are We There Yet? Timing and Floating-Point Attacks on Differential Privacy Systems*. S&P'22