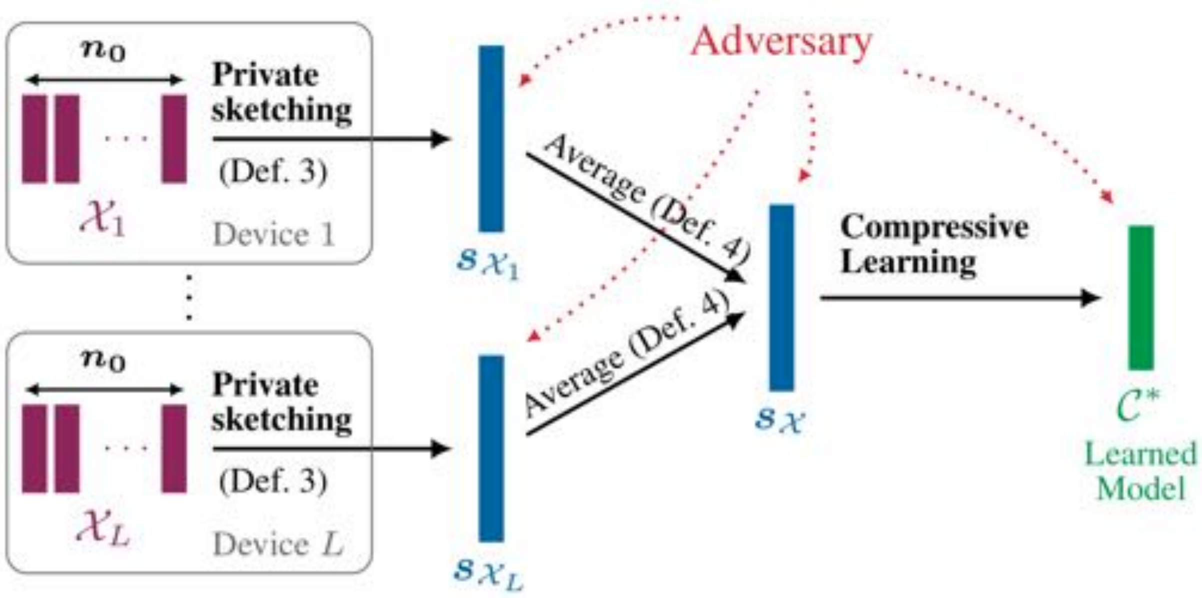
Compressive Learning meets privacy



Florimond Houssiau
Yves-Alexandre de Montjoye
Imperial College London

Vincent Schellekens
Laurent Jacques
UCLouvain

Antoine Chatalic
Rémi Gribonval
Inria Rennes

Machine Learning is ubiquous

Signal Processing

Statistics

Artificial Intelligence

Literally any scientific field

Optimization

Finance

Machine Learning is ubiquous

Machine Learning

Machine Learning

Machine Learning

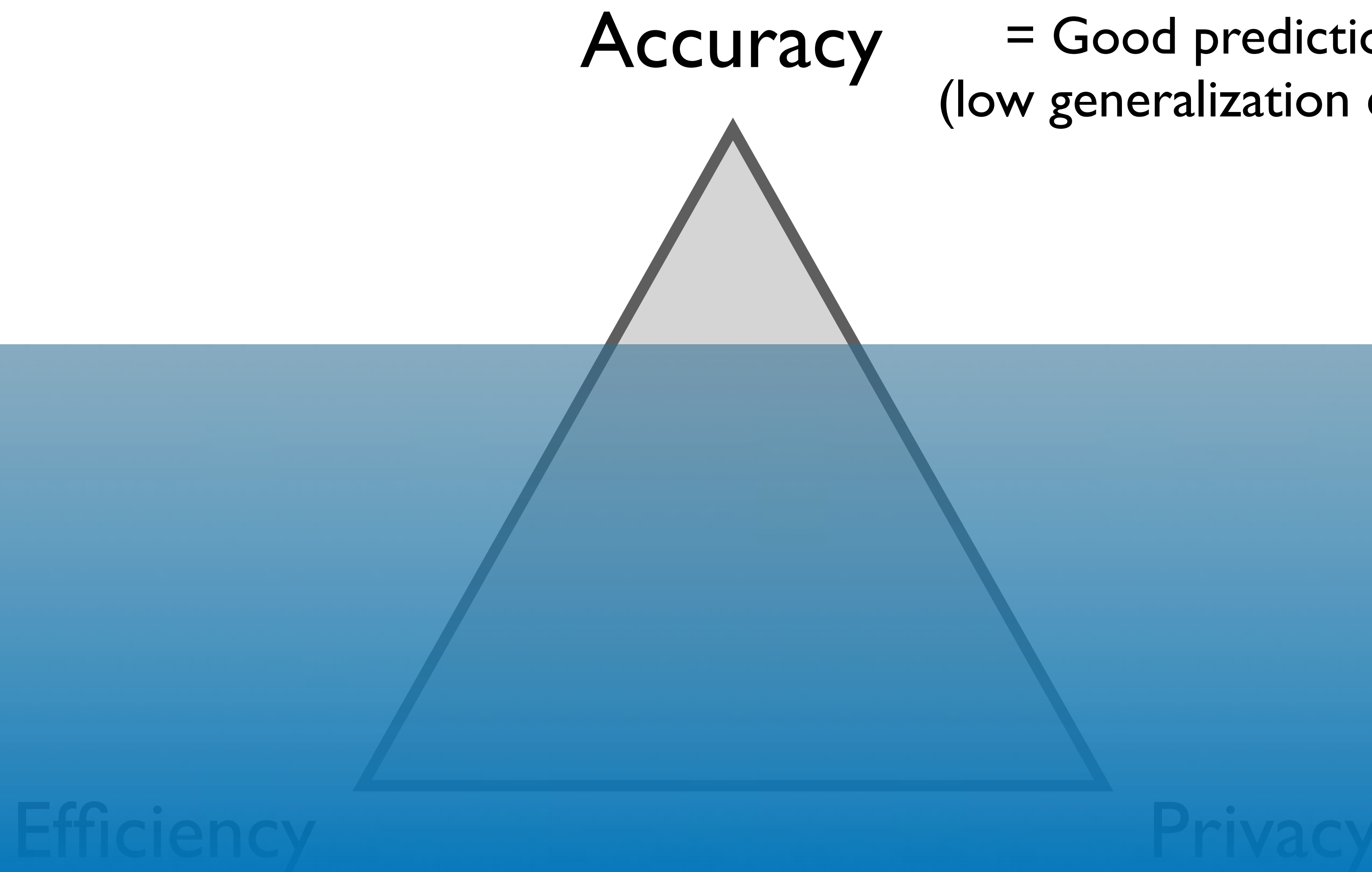
Machine Learning

Machine Learning

Machine Learning

Machine Learnings objective

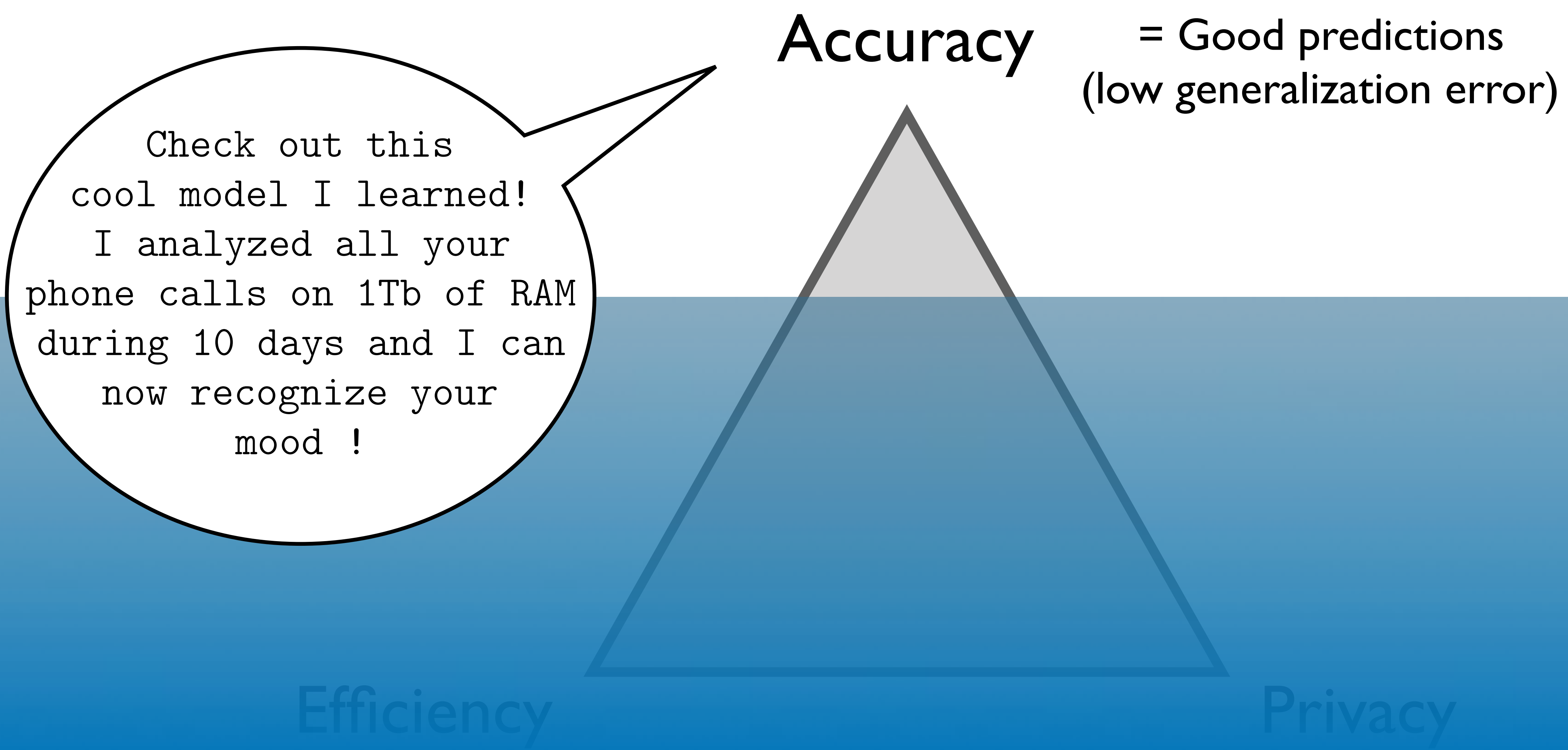
Machine Learning is popular because it works *very well*



= Good predictions
(low generalization error)

Machine Learnings objective

Machine Learning is popular because it works *very well*



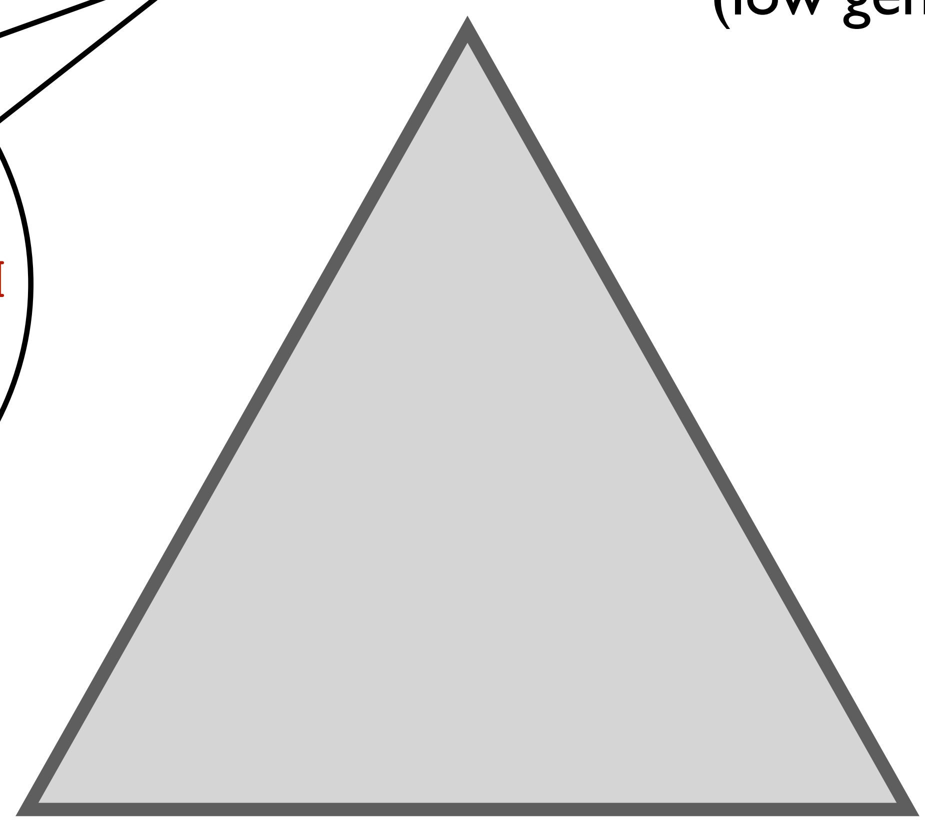
The top of the iceberg?

Machine Learnings objectiveS

Machine Learning is popular because it works *very well*… but…

Check out this cool model I learned! I analyzed all your phone calls on 1Tb of RAM during 10 days and I can now recognize your mood !

Accuracy
= Good predictions
(low generalization error)



Efficiency
= low memory/time requirements

Privacy
Protect sensitive data

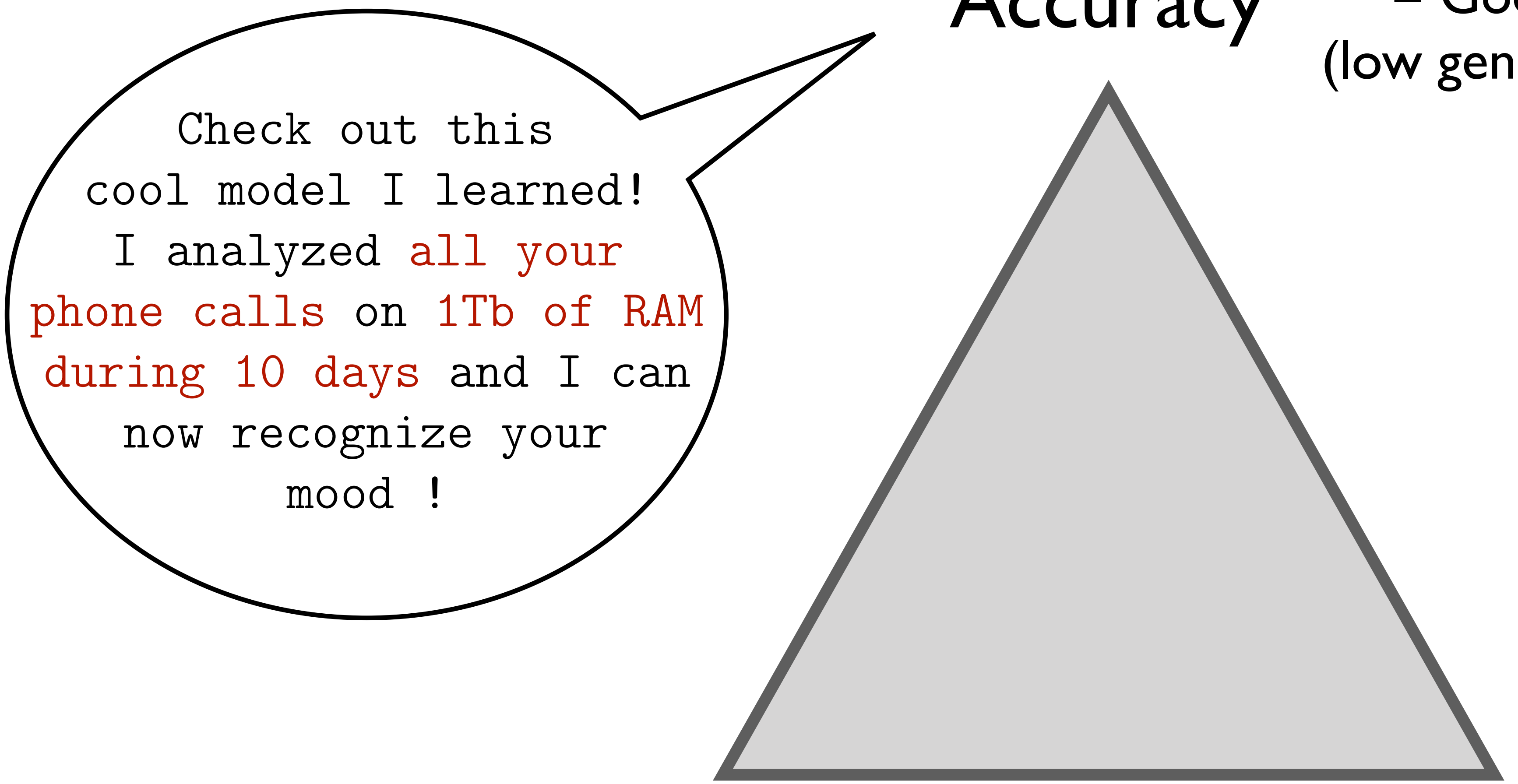
Several objectives that are **incompatible**!

E.g.,

- Reducing memory/time access lowers accuracy
- Ensuring privacy might require more computations…
- …or might require to “sabotage” the model (more later)

Machine Learnings objectiveS

Machine Learning is popular because it works *very well*… but…



Accuracy = Good predictions (low generalization error)

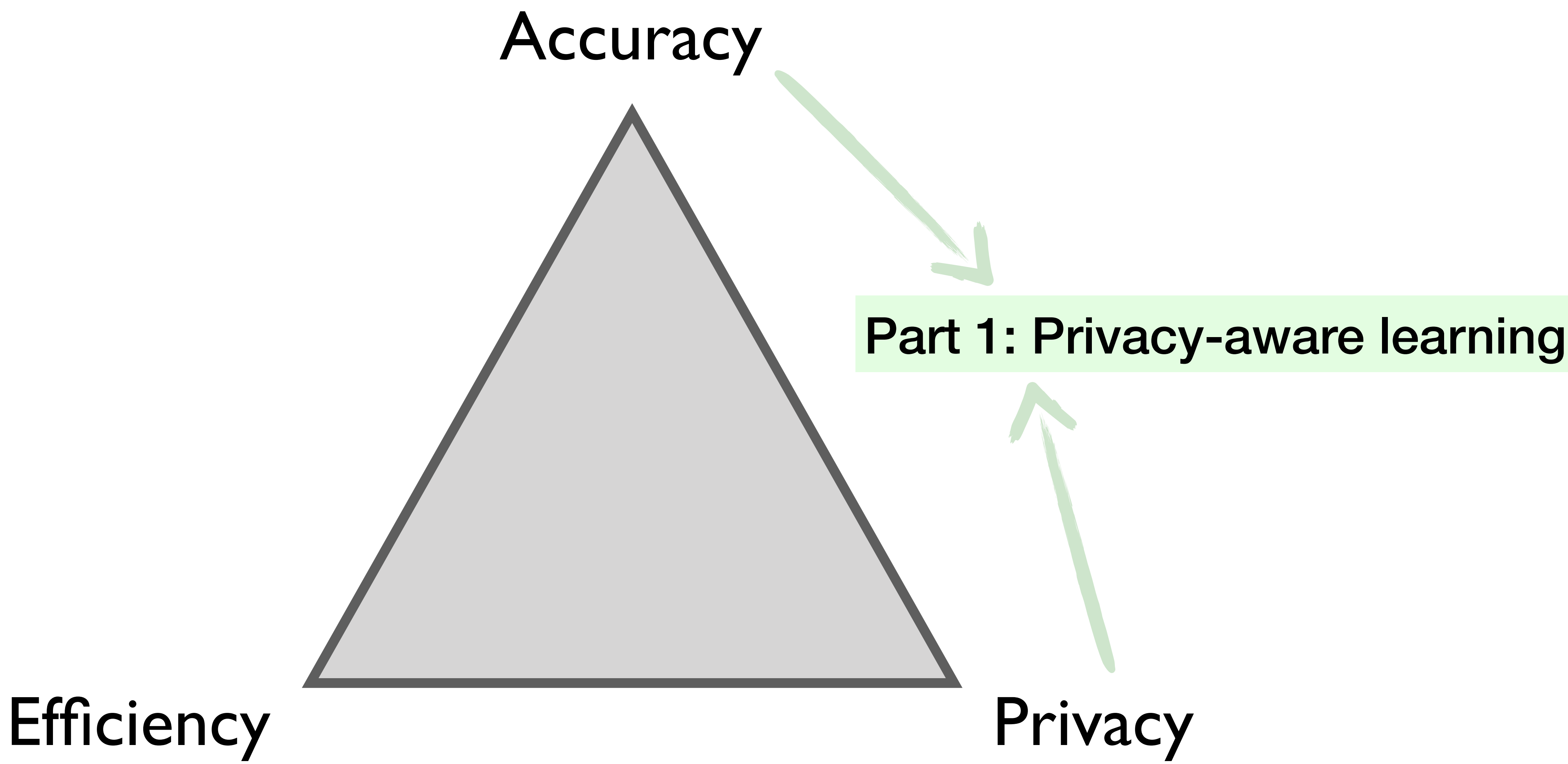
Efficiency = low memory/time requirements

Privacy Protect sensitive data

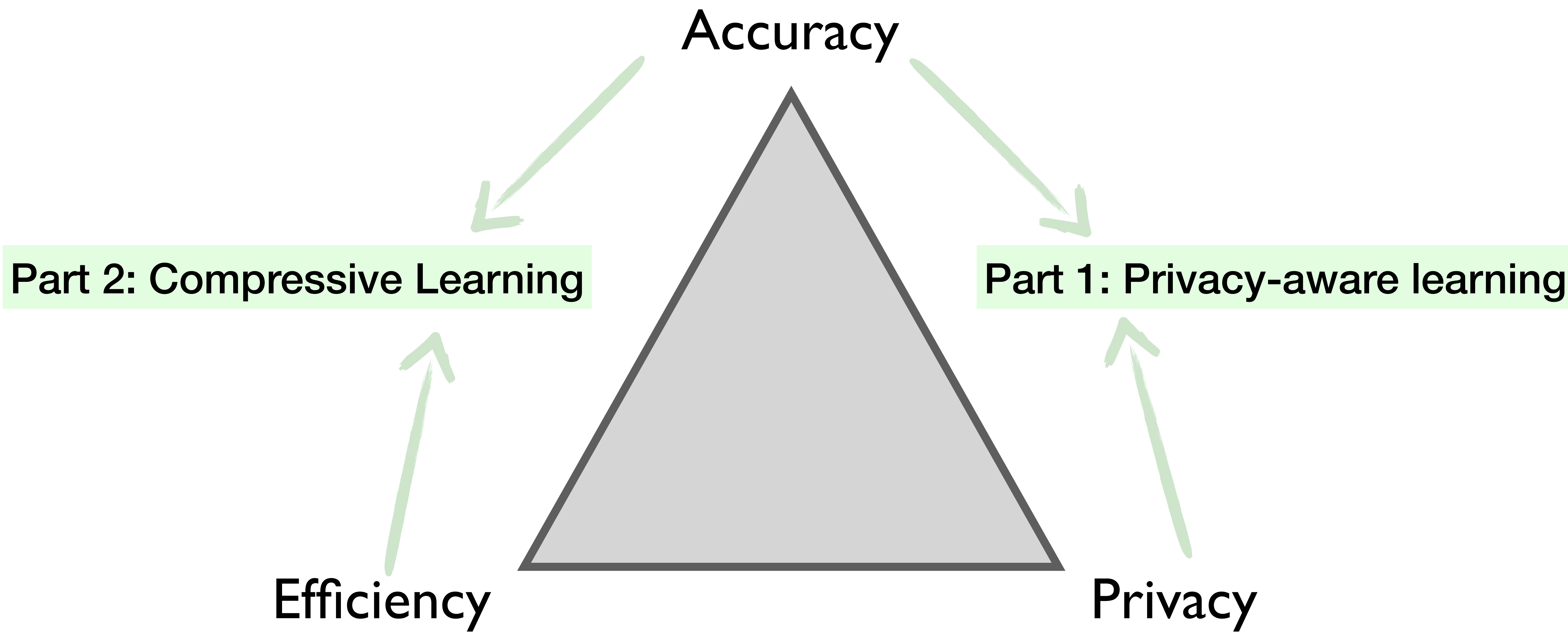
Several objectives that are **incompatible**!

There are probably others (e.g., robust ML, ethical ML), but we focus on these three

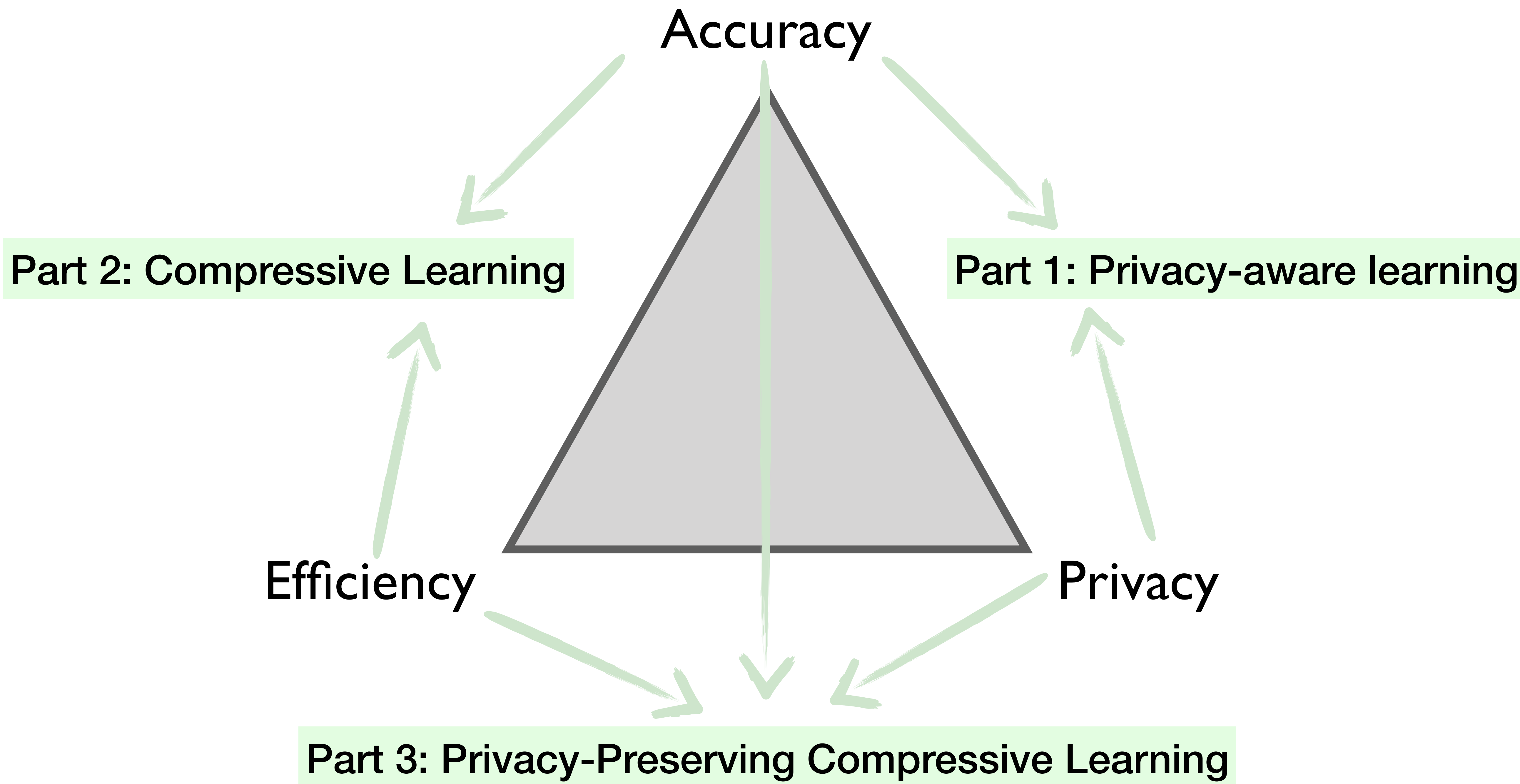
Outline


Accuracy
Part 1: Privacy-aware learning
Efficiency
Privacy

Outline


Accuracy
Part 2: Compressive Learning
Part 1: Privacy-aware learning
Efficiency
Privacy

Outline


Accuracy
Part 2: Compressive Learning
Part 1: Privacy-aware learning
Efficiency
Privacy
Part 3: Privacy-Preserving Compressive Learning

In this talk…

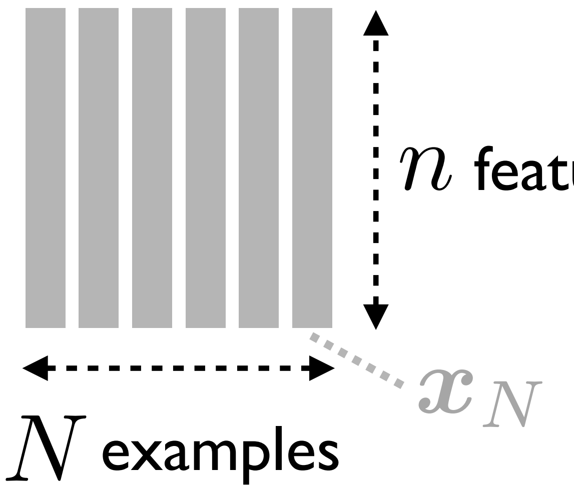
Part 1

Privacy-aware learning

Machine Learning recap'

(Unsupervised) Machine Learning

$X =$
dataset



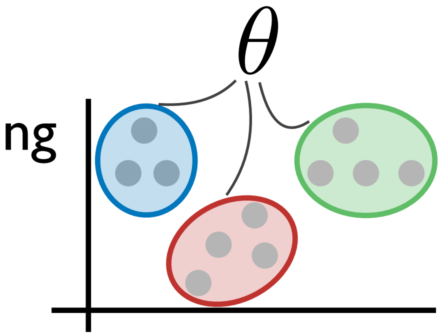
Learning $\longrightarrow$ θ

Mathematical model

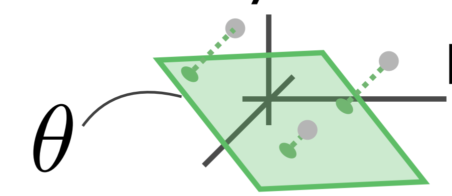
"explain" the data

E.g.,

- Clustering



- Dimensionality reduction

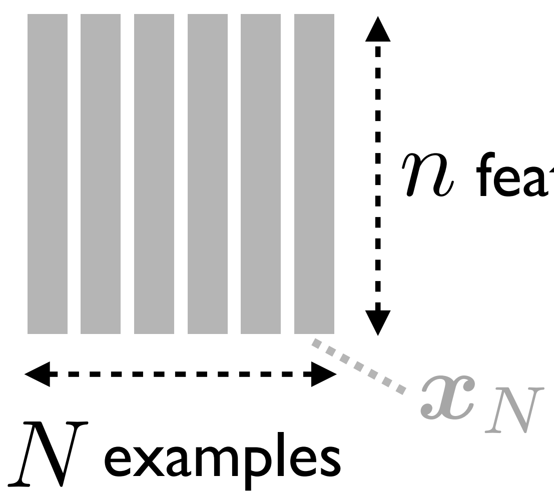


- Autoencoder, GAN, SOM…

Machine Learning recap'

(Unsupervised) Machine Learning

$X =$
dataset



Learning →

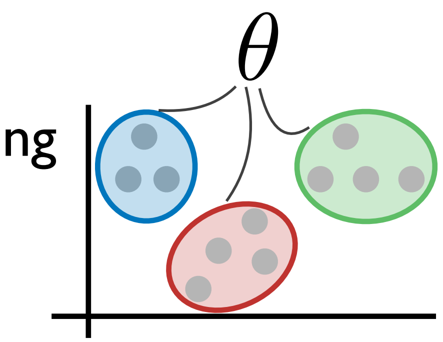
θ

Mathematical model

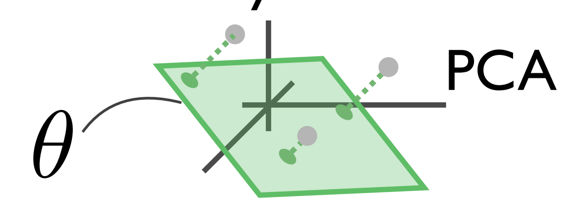
"explain" the data

E.g.,

- Clustering



- Dimensionality reduction



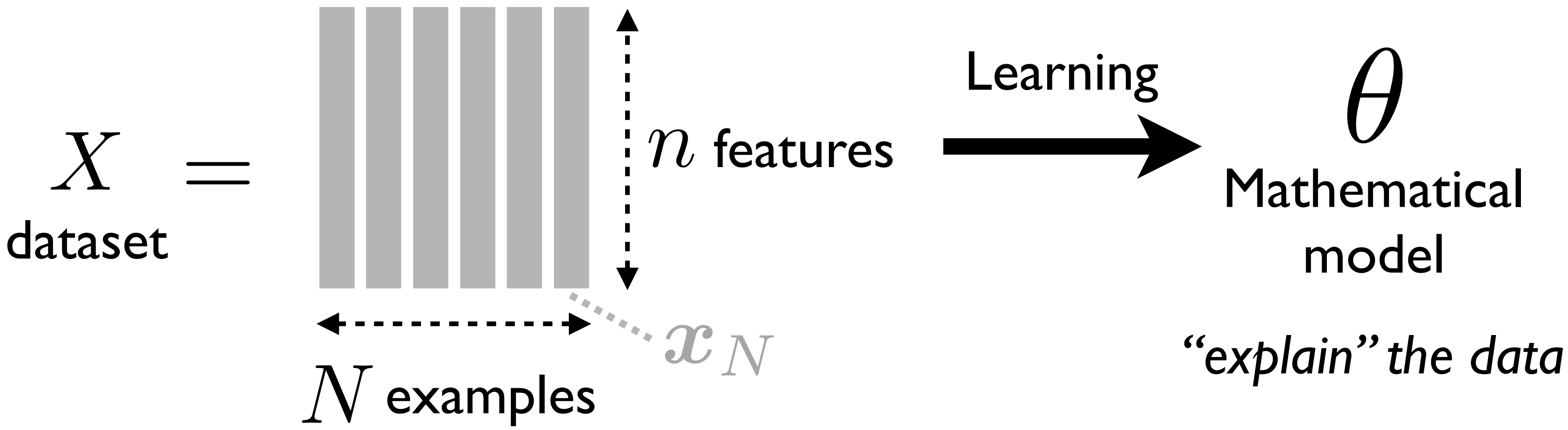
- Autoencoder, GAN, SOM…

But, what if the dataset contains sensitive information?

- DNA databases, medical records (results of HIV testing,…)
- Behavior on social media, web queries,…
- Touchy surveys (political opinions, drugs use, sexual preferences…)
- IoT devices
- …

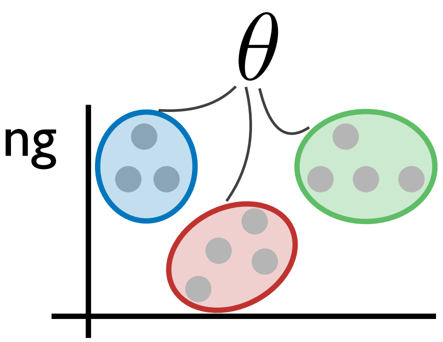
Machine Learning recap'

(Unsupervised) Machine Learning

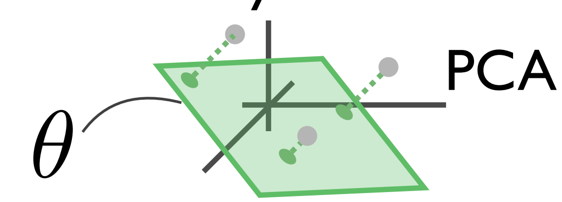


E.g.,

- Clustering



- Dimensionality reduction



- Autoencoder, GAN, SOM…

But, what if the dataset contains sensitive information?

- DNA databases, medical records (results of HIV testing,…)
- Behavior on social media, web queries,…
- Touchy surveys (political opinions, drugs use, sexual preferences…)
- IoT devices
- …

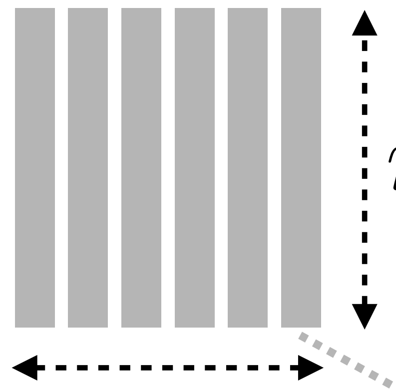
We want to learn (generalize) from the dataset while protecting its "privacy"!

Machine Learning recap'

(Unsupervised) Machine Learning

X $=$

dataset



n features

N examples

x_N

Learning →

θ

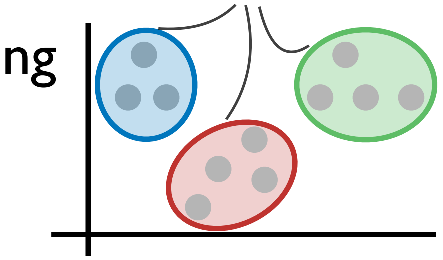
Mathematical model

"explain" the data

E.g.,

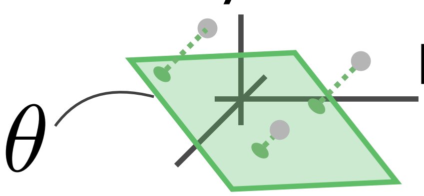
- Clustering

θ



- Dimensionality reduction

θ



PCA

- Autoencoder, GAN, SOM…

But, what if the dataset contains sensitive information?

- DNA databases, medical records (results of HIV testing,…)
- Behavior on social media, web queries,…
- Touchy surveys (political opinions, drugs use, sexual preferences…)
- IoT devices
- …

We want to learn (generalize) from the dataset while protecting its "privacy"!

What does it mean?

What IS privacy?

Privacy is very difficult to define!

Depends on the application (what do we want to protect), and the *attack model* (what do we want to protect against).

What IS privacy?

Privacy is very difficult to define!

Depends on the application (what do we want to protect), and the *attack model* (what do we want to protect against).

There exist a thousand* of privacy definitions, with different pro/cons

*citation needed

What IS privacy?

Privacy is very difficult to define!

Depends on the application (what do we want to protect), and the *attack model* (what do we want to protect against).

There exist a thousand* of privacy definitions, with different pro/cons

Mathematical privacy definitions:
- k-Anonymity
- Information-theoretic privacy definitions
- Differential Privacy
- …

But also to consider:
- Legal privacy definition
- Philosophical privacy definitions?

*citation needed

What IS privacy?

Privacy is very difficult to define!

Depends on the application (what do we want to protect), and the *attack model* (what do we want to protect against).

There exist a thousand* of privacy definitions, with different pro/cons

Mathematical privacy definitions:
- k-Anonymity
- Information-theoretic privacy definitions
- Differential Privacy
- …

In this work

But also to consider:
- Legal privacy definition
- Philosophical privacy definitions?

*citation needed

Towards DP: privacy by randomization

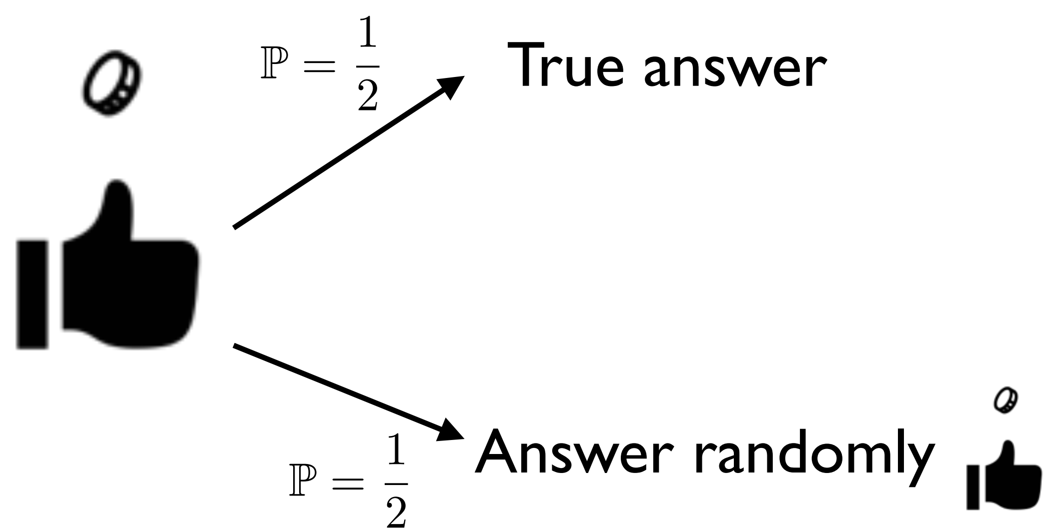
The predecessor to DP: randomized response (used for surveys)

Example: do you watch youtube videos at work?

Towards DP: privacy by randomization

The predecessor to DP: randomized response (used for surveys)

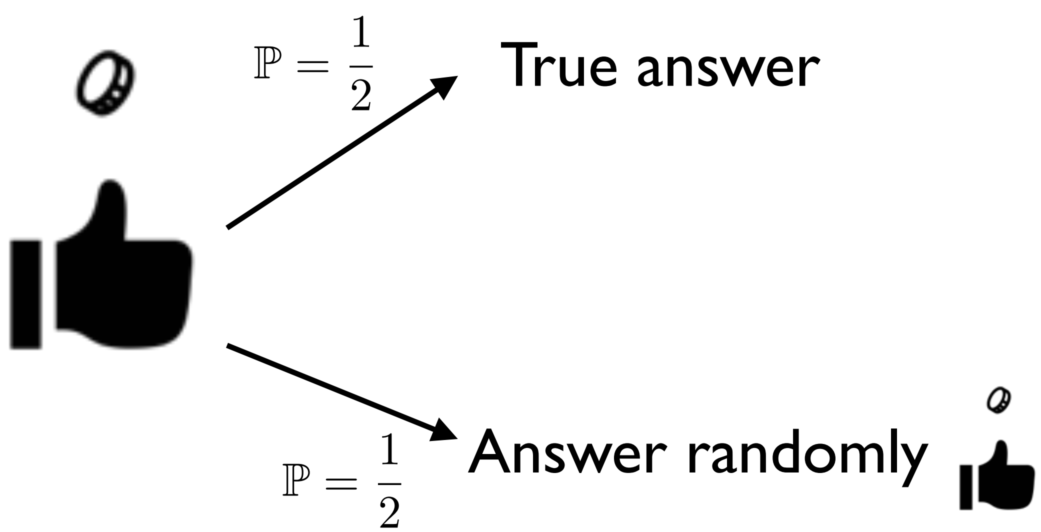
Example: do you watch youtube videos at work?



Towards DP: privacy by randomization

The predecessor to DP: randomized response (used for surveys)

Example: do you watch youtube videos at work?



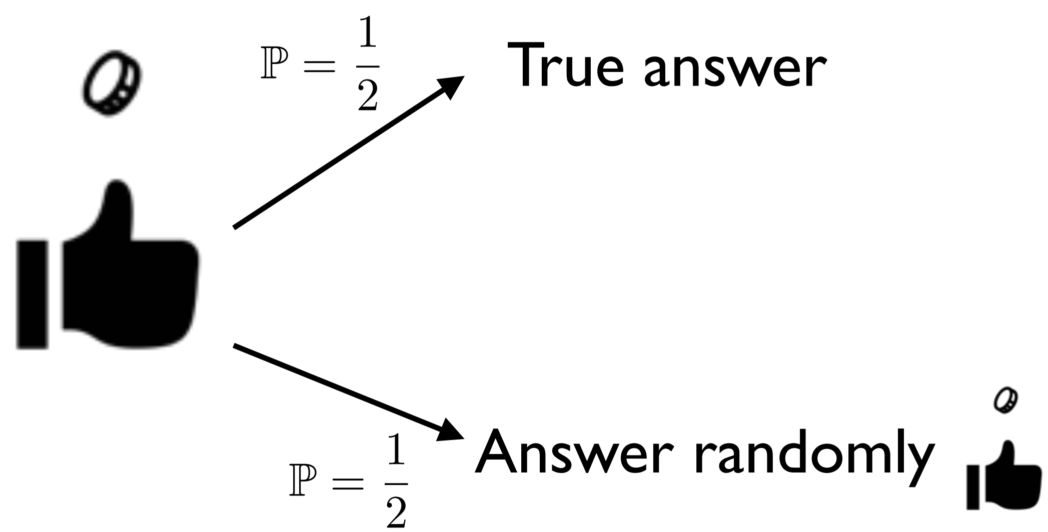
We obtain a fraction $\tilde{p}$ of "Yes"

$$\mathbb{E}\tilde{p} = \frac{p}{2} + \frac{1}{4}$$

Towards DP: privacy by randomization

The predecessor to DP: randomized response (used for surveys)

Example: do you watch youtube videos at work?



We obtain a fraction $\tilde{p}$ of "Yes"

$$\mathbb{E}\tilde{p} = \frac{p}{2} + \frac{1}{4}$$

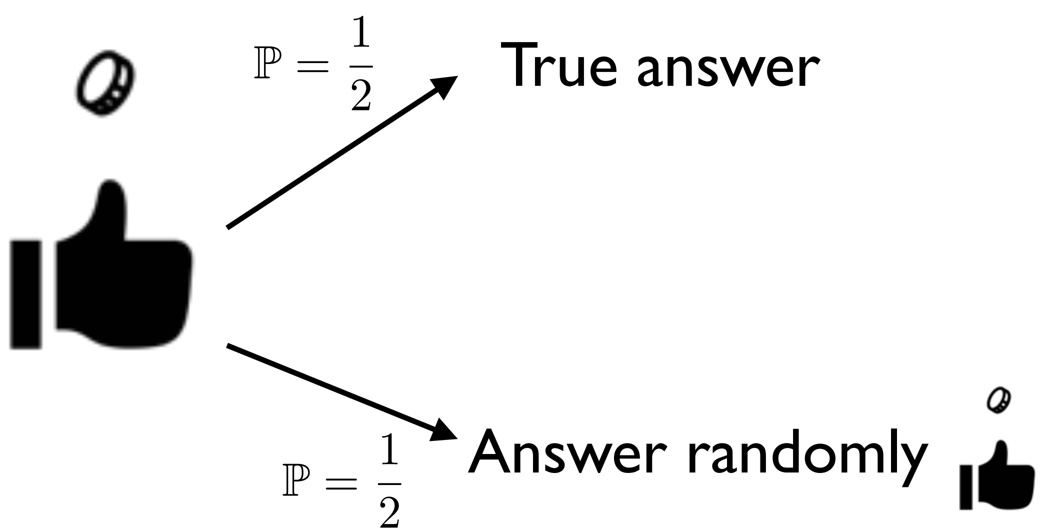
Estimation of the true proportion

$$\hat{p} = 2(\tilde{p} - \frac{1}{4}) \simeq p$$

Towards DP: privacy by randomization

The predecessor to DP: randomized response (used for surveys)

Example: do you watch youtube videos at work?



We obtain a fraction $\tilde{p}$ of "Yes"

$$\mathbb{E}\tilde{p} = \frac{p}{2} + \frac{1}{4}$$

Estimation of the true proportion

$$\hat{p} = 2(\tilde{p} - \frac{1}{4}) \simeq p$$

Randomness introduces *plausible deniability* (i.e., privacy comes from *uncertainty*)

S.L. Warner, "*Randomised response: a survey technique for eliminating evasive answer bias,*" Journal of the American Statistical Association, 1965

Differential Privacy: (a possible) definition

Intuitive definition:

> "An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

C. Dwork, "*Differential Privacy,*" 2006

Differential Privacy: (a possible) definition

Intuitive definition:

"An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

Formal definition: a *randomized* algorithm f is Differentially Private if

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

C. Dwork, "*Differential Privacy*," 2006

Differential Privacy: (a possible) definition

Intuitive definition:

"An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

Formal definition: a *randomized* algorithm f is Differentially Private if

Taken over the randomness in f

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

C. Dwork, "*Differential Privacy*," 2006

Differential Privacy: (a possible) definition

Intuitive definition:

> "An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

Formal definition: a *randomized* algorithm f is Differentially Private if

$$\epsilon - \mathrm{DP}$$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

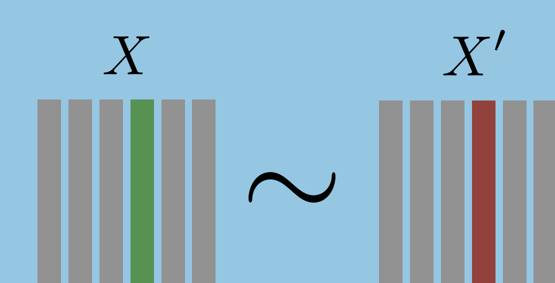
Taken over the randomness in f

For all "neighbour" DS

Neighbouring relation $\sim$

$X \sim X'$ if they differ by one entry

i.e. $|X| = |X'|$ and $|(X \cup X') \setminus (X \cap X')| \leq 2$



C. Dwork, "*Differential Privacy*," 2006

Differential Privacy: (a possible) definition

Intuitive definition:

> "An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

Formal definition: a *randomized* algorithm f is Differentially Private if

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S \quad \forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Taken over the randomness in f

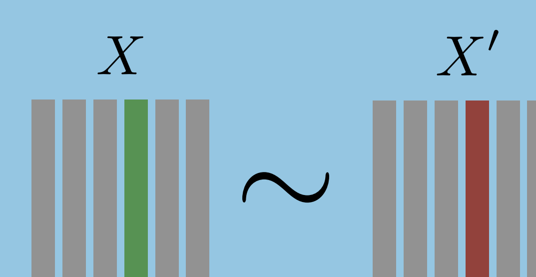
For all subsets of possible outcomes

For all "neighbour" DS

Neighbouring relation $\sim$

$X \sim X'$ if they differ by one entry

i.e. $|X| = |X'|$ and $|(X \cup X') \setminus (X \cap X')| \leq 2$



C. Dwork, "*Differential Privacy*," 2006

Differential Privacy: (a possible) definition

Intuitive definition:

> "An algorithm is Differentially Private if its output is not much influenced when one user of the dataset is changed"

"It is not possible to detect with high confidence whether I participated to the dataset or not"

Formal definition: a *randomized* algorithm f is Differentially Private if

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

For all subsets of possible outcomes

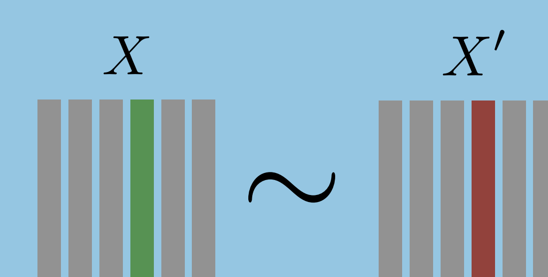
For all "neighbour" DS

Taken over the randomness in f

Privacy parameter/budget (should be small, see later)

Neighbouring relation $\sim$

$X \sim X'$ if they differ by one entry



i.e. $|X| = |X'|$ and $|(X \cup X') \setminus (X \cap X')| \leq 2$

C. Dwork, "*Differential Privacy*," 2006

Differential Privacy: interpretation

In practice, epsilon is small, so DP means

$$\mathbb{P}[f(X) \in S] \simeq \mathbb{P}[f(X') \in S] + \mathcal{O}(\epsilon)$$

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Differential Privacy: interpretation

In practice, epsilon is small, so DP means

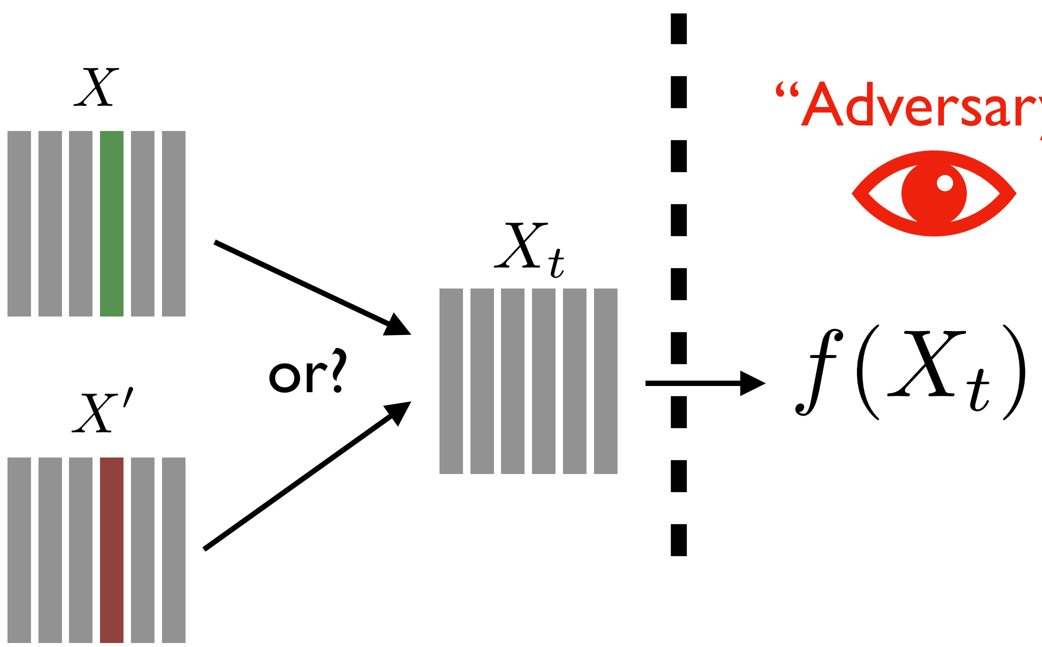
$$\mathbb{P}[f(X) \in S] \simeq \mathbb{P}[f(X') \in S] + \mathcal{O}(\epsilon)$$

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$, $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Interpretation of DP as plausible deniability: f almost doesn't decrease uncertainty



Assume the adversary has prior knowledge

$$\mathbb{P}[X_t = X] \quad \text{and} \quad \mathbb{P}[X_t = X']$$

Then $f(X_t)$ is publicly released!
What did the adversary "learn"?

Differential Privacy: interpretation

In practice, epsilon is small, so DP means

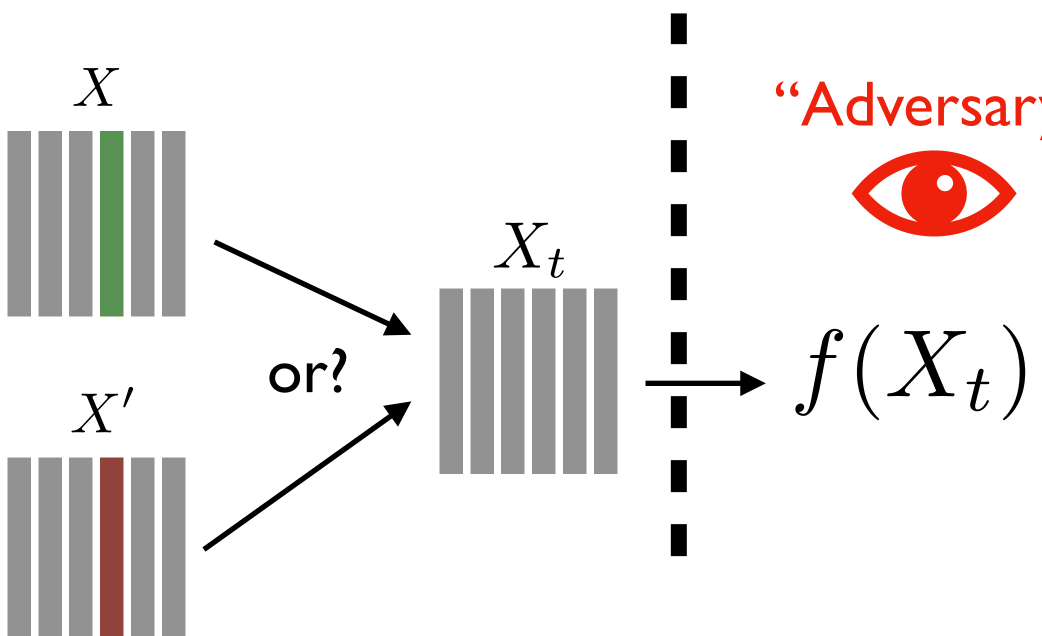
$$\mathbb{P}[f(X) \in S] \simeq \mathbb{P}[f(X') \in S] + \mathcal{O}(\epsilon)$$

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Interpretation of DP as plausible deniability: f almost doesn't decrease uncertainty



Assume the adversary has prior knowledge

$$\mathbb{P}[X_t = X] \quad \text{and} \quad \mathbb{P}[X_t = X']$$

Then $f(X_t)$ is publicly released!
What did the adversary "learn"?

$$\underbrace{\frac{\mathbb{P}[X_t = X | f(X_t) = s]}{\mathbb{P}[X_t = X' | f(X_t) = s]}}_{\text{Posterior "belief ratio"}} \overset{\text{Bayes}}{=} \frac{\mathbb{P}[f(X_t) = s | X_t = X]\mathbb{P}[X_t = X]}{\mathbb{P}[f(X_t) = s | X_t = X']\mathbb{P}[X_t = X']} \overset{\text{DP}}{\leq} \underbrace{e^{\epsilon}}_{\text{Small}} \underbrace{\frac{\mathbb{P}[X_t = X]}{\mathbb{P}[X_t = X']}}_{\text{Prior "belief ratio"}}$$

Differential Privacy: interpretation

In practice, epsilon is small, so DP means

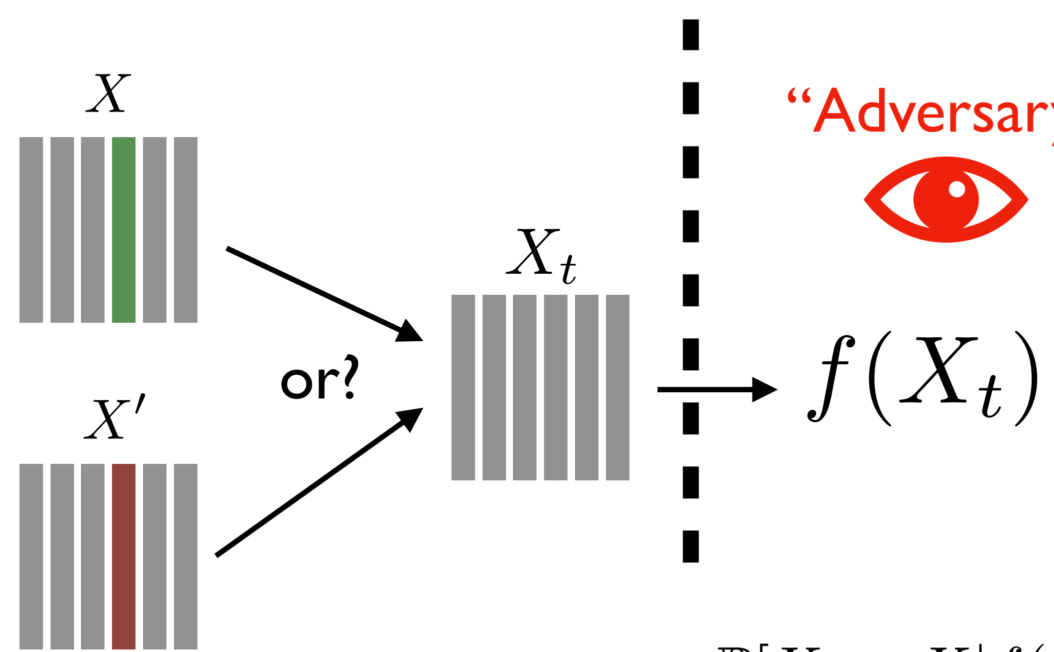
$$\mathbb{P}[f(X) \in S] \simeq \mathbb{P}[f(X') \in S] + \mathcal{O}(\epsilon)$$

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$, $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Interpretation of DP as plausible deniability: f almost doesn't decrease uncertainty



Assume the adversary has prior knowledge

$$\mathbb{P}[X_t = X] \quad \text{and} \quad \mathbb{P}[X_t = X']$$

Then $f(X_t)$ is publicly released!
What did the adversary "learn"?

$$\underbrace{\frac{\mathbb{P}[X_t = X | f(X_t) = s]}{\mathbb{P}[X_t = X' | f(X_t) = s]}}_{\text{Posterior "belief ratio"}} \overset{\text{Bayes}}{=} \frac{\mathbb{P}[f(X_t) = s | X_t = X]\mathbb{P}[X_t = X]}{\mathbb{P}[f(X_t) = s | X_t = X']\mathbb{P}[X_t = X']} \overset{\text{DP}}{\leq} e^{\epsilon} \underbrace{\frac{\mathbb{P}[X_t = X]}{\mathbb{P}[X_t = X']}}_{\text{Prior "belief ratio"}}$$

Small

Example: 2 possibilities >>>

90% / 10%

Differential Privacy: interpretation

In practice, epsilon is small, so DP means

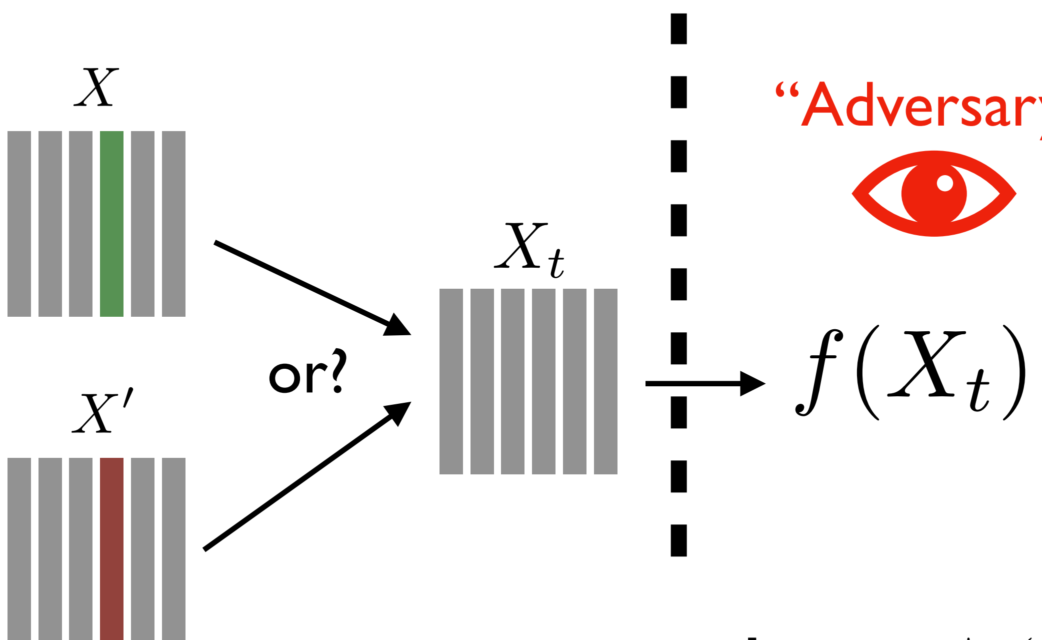
$$\mathbb{P}[f(X) \in S] \simeq \mathbb{P}[f(X') \in S] + \mathcal{O}(\epsilon)$$

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$, $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Interpretation of DP as plausible deniability: f almost doesn't decrease uncertainty



Assume the adversary has prior knowledge

$$\mathbb{P}[X_t = X] \quad \text{and} \quad \mathbb{P}[X_t = X']$$

Then $f(X_t)$ is publicly released!
What did the adversary "learn"?

$$\underbrace{\frac{\mathbb{P}[X_t = X | f(X_t) = s]}{\mathbb{P}[X_t = X' | f(X_t) = s]}}_{\text{Posterior "belief ratio"}} \overset{\text{Bayes}}{=} \frac{\mathbb{P}[f(X_t) = s | X_t = X]\mathbb{P}[X_t = X]}{\mathbb{P}[f(X_t) = s | X_t = X']\mathbb{P}[X_t = X']} \overset{\text{DP}}{\leq} \underbrace{e^{\epsilon}}_{\text{Small}} \underbrace{\frac{\mathbb{P}[X_t = X]}{\mathbb{P}[X_t = X']}}_{\text{Prior "belief ratio"}}$$

Example: 2 possibilities >>> 90.1% / 9.9% 1.01 90% / 10%

Differential Privacy: interpretation??

In practice, epsilon is small, so DP means

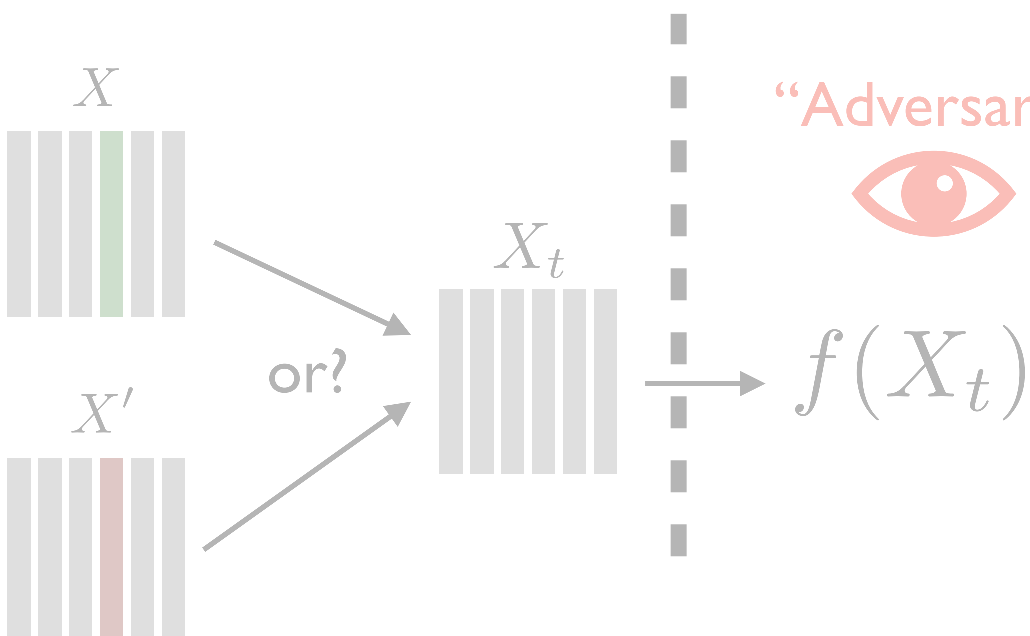
$\mathbb{P}[f(X) \in S] \simeq$ **How small exactly?!**

$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$ $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

Interpretation of DP as plausible deniability: f almost doesn't decrease uncertainty



Assume the adversary has prior knowledge

$\mathbb{P}[X_t = X]$ and $\mathbb{P}[X_t = X']$

Then $f(X_t)$ is publicly released!
What did the adversary "learn"?

$$\underbrace{\frac{\mathbb{P}[X_t = X | f(X_t) = s]}{\mathbb{P}[X_t = X' | f(X_t) = s]}}_{\text{Posterior "belief ratio"}} \overset{\text{Bayes}}{=} \frac{\mathbb{P}[f(X_t) = s | X_t = X]\mathbb{P}[X_t = X]}{\mathbb{P}[f(X_t) = s | X_t = X']\mathbb{P}[X_t = X']} \overset{\text{DP}}{\leq} \underbrace{e^{\epsilon}}_{\text{Small}} \underbrace{\frac{\mathbb{P}[X_t = X]}{\mathbb{P}[X_t = X']}}_{\text{Prior "belief ratio"}}$$

Example: 2 possibilities >>> 90.1% / 9.9% 1.01 90% / 10%

Differential Privacy: the epsilon problem

No satisfying rule to decide how small ϵ should be in practice :-(

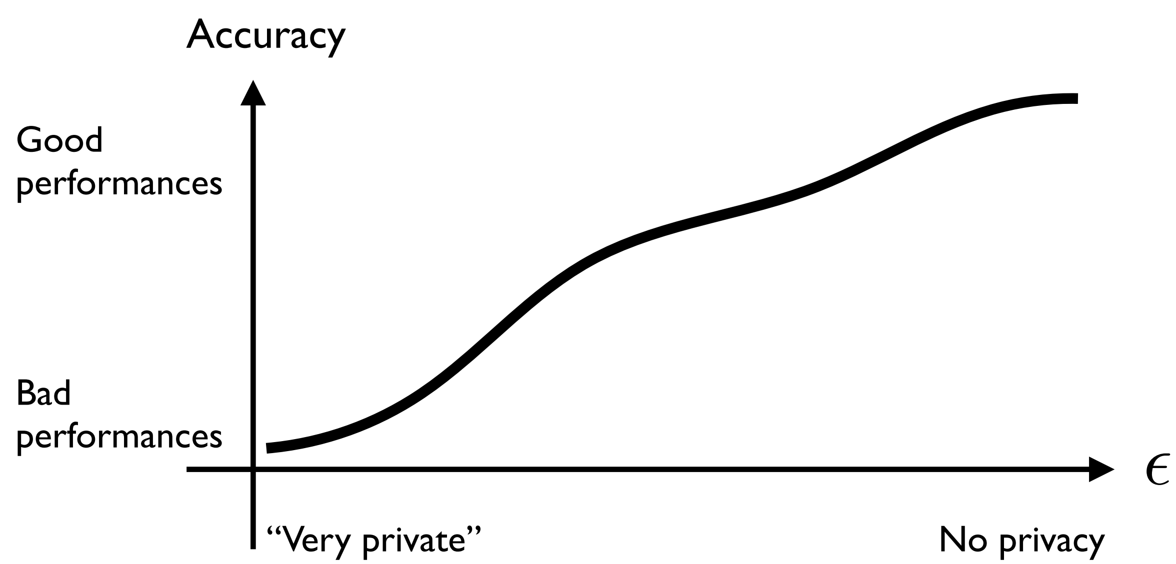
The least we can say is that it is heavily context-dependent and requires "expert knowledge"

Differential Privacy: the epsilon problem

No satisfying rule to decide how small ϵ should be in practice :-(

The least we can say is that it is heavily context-dependent and requires "expert knowledge"

In addition, there is a "privacy-utility" tradeoff (see more later)!



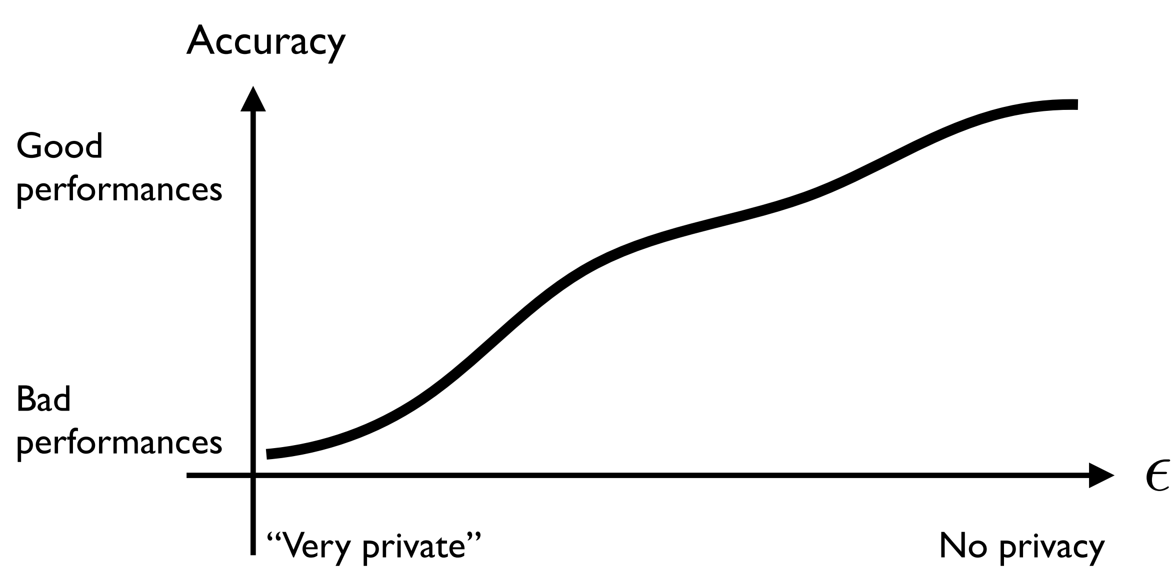
We should pick ϵ as large as possible to get the best accuracy… while not compromising privacy too much…

Differential Privacy: the epsilon problem

No satisfying rule to decide how small ϵ should be in practice :-(

The least we can say is that it is heavily context-dependent and requires "expert knowledge"

In addition, there is a "privacy-utility" tradeoff (see more later)!



We should pick ϵ as large as possible to get the best accuracy… while not compromising privacy too much…

The consensus seems to be that $\epsilon \simeq 10^{-2} \cdots 10^{-1}$ is "enough"…

…to take with a grain of salt!

DP how-to: Laplacian mechanism

(A) standard way to achieve DP: add randomness as additive Laplacian noise

The Laplacian mechanism

If $g(\cdot)$ is the target task, then

$f(X) = g(X) + n$ with $n \sim \mathrm{Lap}\left(\frac{\Delta g}{\epsilon}\right)$

is $\epsilon - \mathrm{DP}$

DP how-to: Laplacian mechanism

(A) standard way to achieve DP: add randomness as additive Laplacian noise

The Laplacian mechanism

If $g(\cdot)$ is the target task, then

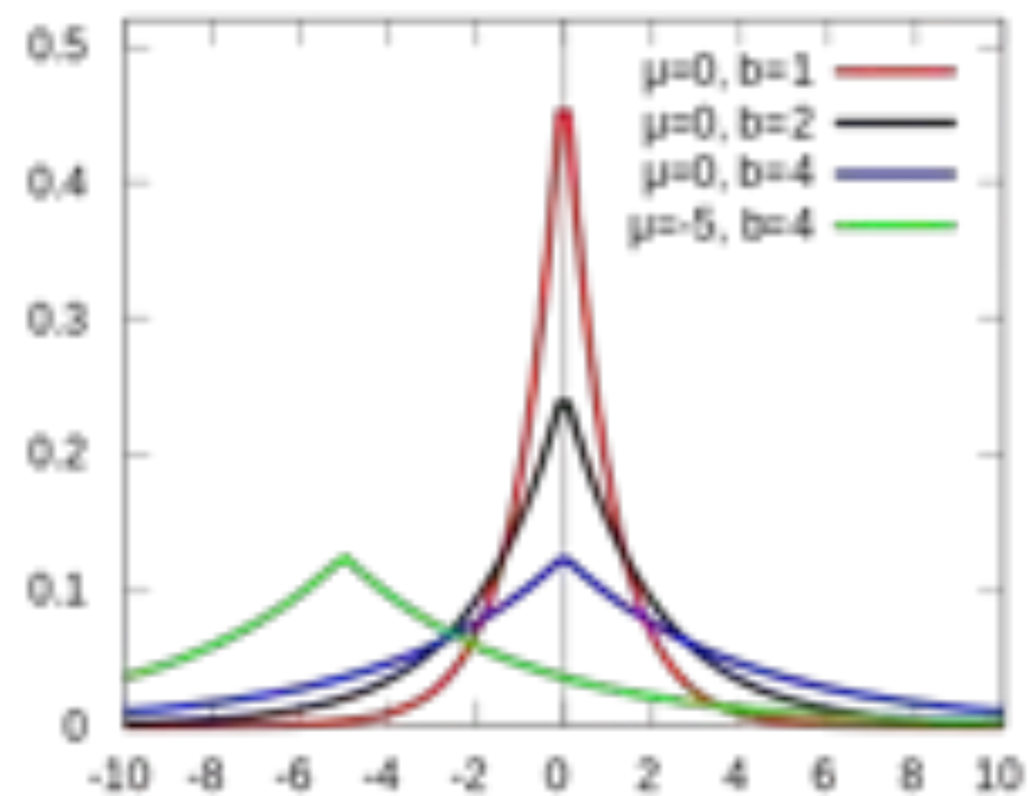
$$f(X) = g(X) + n \quad \text{with} \quad n \sim \mathrm{Lap}\left(\frac{\Delta g}{\epsilon}\right)$$

is $\epsilon - \mathrm{DP}$

Laplace random variable

$n \sim \mathrm{Lap}(b)$ has density $p_n(n) = \frac{1}{2b} e^{-\frac{|n|}{b}}$

Variance: $\sigma_n^2 = 2b^2$



DP how-to: Laplacian mechanism

(A) standard way to achieve DP: add randomness as additive Laplacian noise

The Laplacian mechanism

If $g(\cdot)$ is the target task, then

$$f(X) = g(X) + n \quad \text{with} \quad n \sim \mathrm{Lap}\left(\frac{\Delta g}{\epsilon}\right)$$

is $\epsilon - \mathrm{DP}$ Variance: $\sigma_n^2 = 2(\Delta g/\epsilon)^2$

Laplace random variable

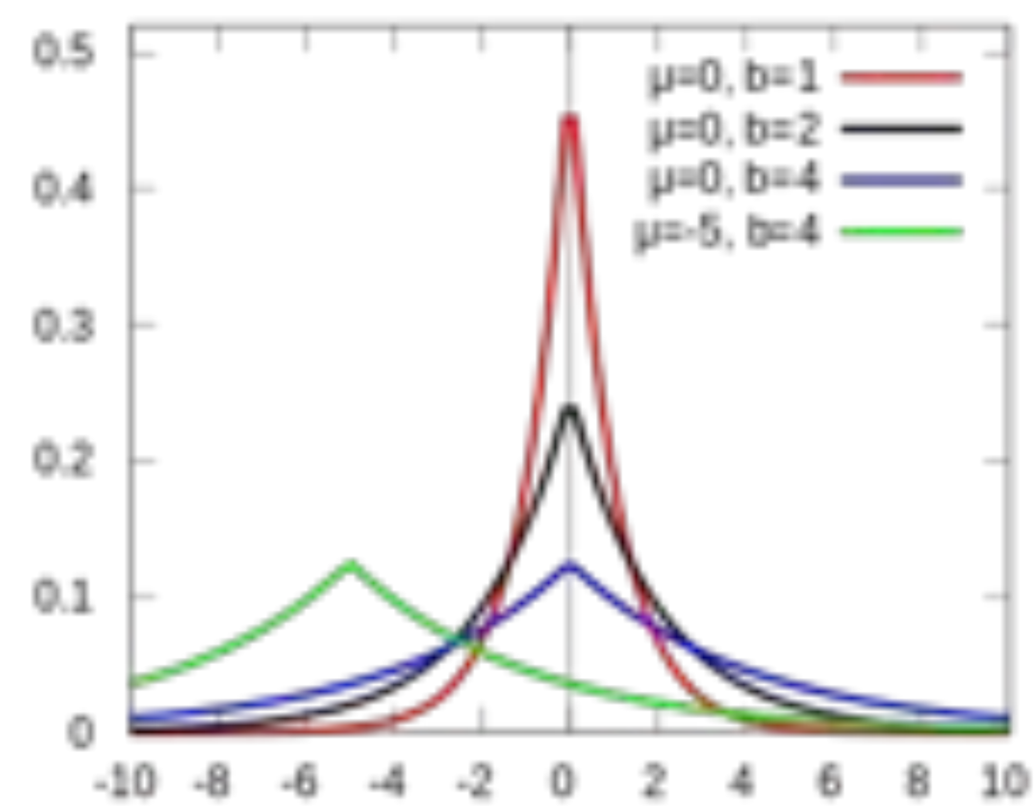
$n \sim \mathrm{Lap}(b)$ has density $p_n(n) = \frac{1}{2b} e^{-\frac{|n|}{b}}$

Variance: $\sigma_n^2 = 2b^2$

"How much does one sample affect the output?"

Sensitivity

$$\Delta g := \max_{X \sim X'} \|g(X) - g(X')\|_1$$



DP how-to: Laplacian mechanism

(A) standard way to achieve DP: add randomness as additive Laplacian noise

The Laplacian mechanism

If $g(\cdot)$ is the target task, then

$$f(X) = g(X) + n \quad \text{with} \quad n \sim \mathrm{Lap}\left(\frac{\Delta g}{\epsilon}\right)$$

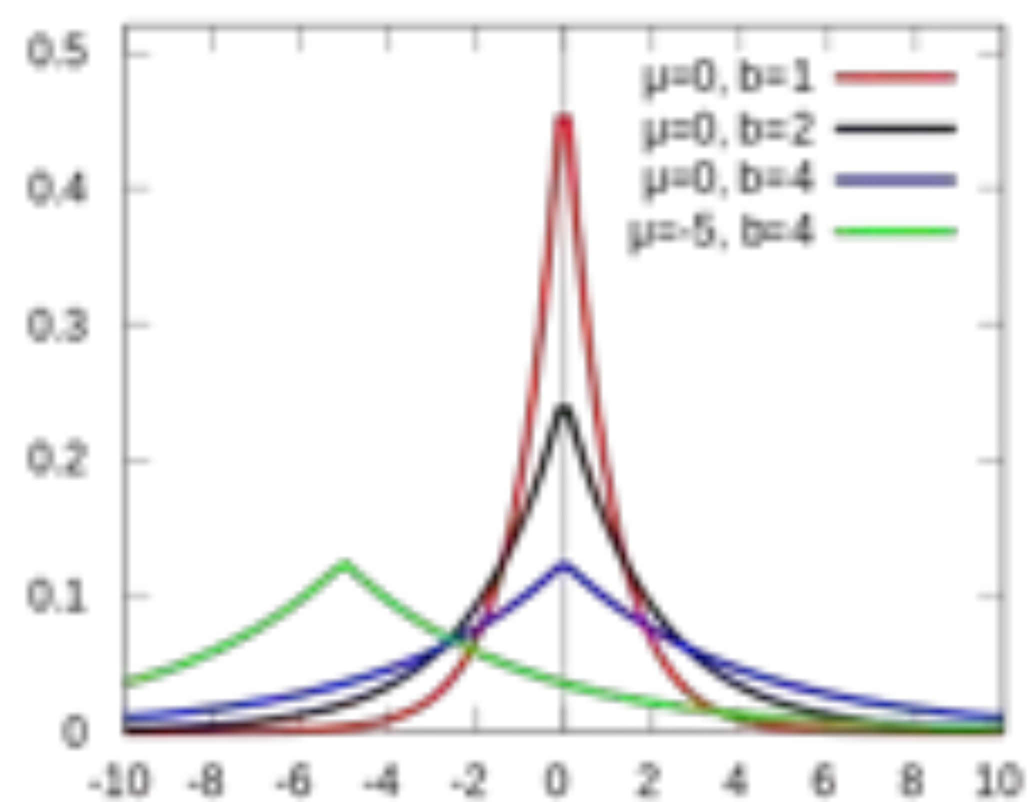
is $\epsilon - \mathrm{DP}$ Variance: $\sigma_n^2 = 2(\Delta g/\epsilon)^2$

Laplace random variable

$n \sim \mathrm{Lap}(b)$ has density $p_n(n) = \frac{1}{2b} e^{-\frac{|n|}{b}}$

Variance: $\sigma_n^2 = 2b^2$

"How much does one sample affect the output?"

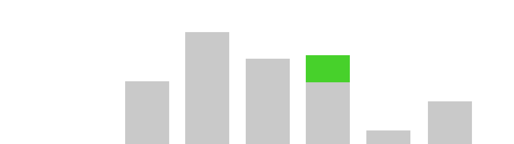


Sensitivity

$$\Delta g := \max_{X \sim X'} \|g(X) - g(X')\|_1$$

Example: histogram

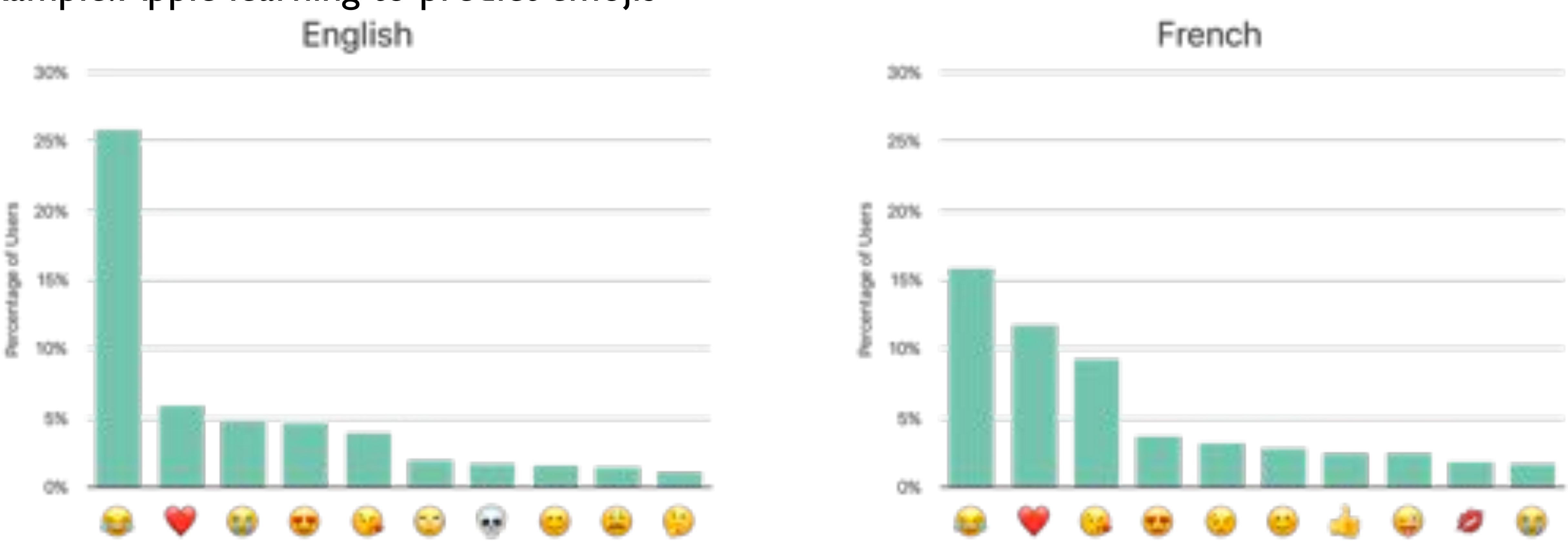
$$\Delta g = 1$$



Differential Privacy: pros/cons

- **Extensively studied, widely accepted standard** (2008-present)
- Very strong guarantee (robust to, e.g., side-information…)
- Composition property (robust to post-processing)
- Often easy to implement (Laplacian mechanism)

Example: Apple learning to predict emojis



Differential Privacy: pros/cons

- Extensively studied, widely accepted standard (2008-present)
- Very strong guarantee (robust to, e.g., side-information…)
- Composition property (robust to post-processing)
- Often easy to implement (Laplacian mechanism)

- How to pick epsilon? Not easy to interpret!
- A "too strong" (restrictive) guarantee? (cfr privacy-utility tradeoff)

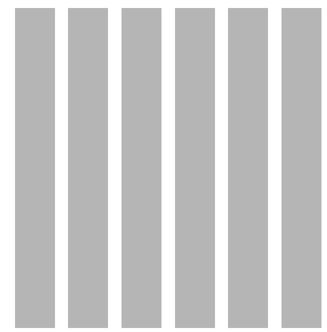
In this talk…

Part 2

Compressive Learning

Machine Learning recap'

(Unsupervised) Machine Learning

$X =$  n features

dataset

N examples $\boldsymbol{x}_N$

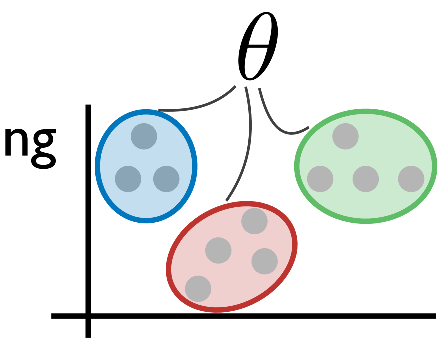
Learning $\longrightarrow$ θ

Mathematical model

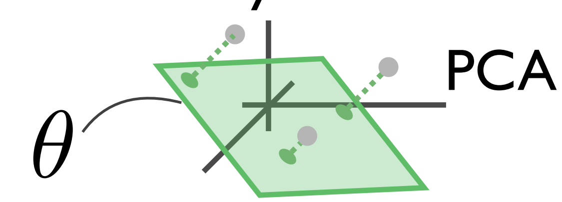
"explain" the data

E.g.,

- Clustering

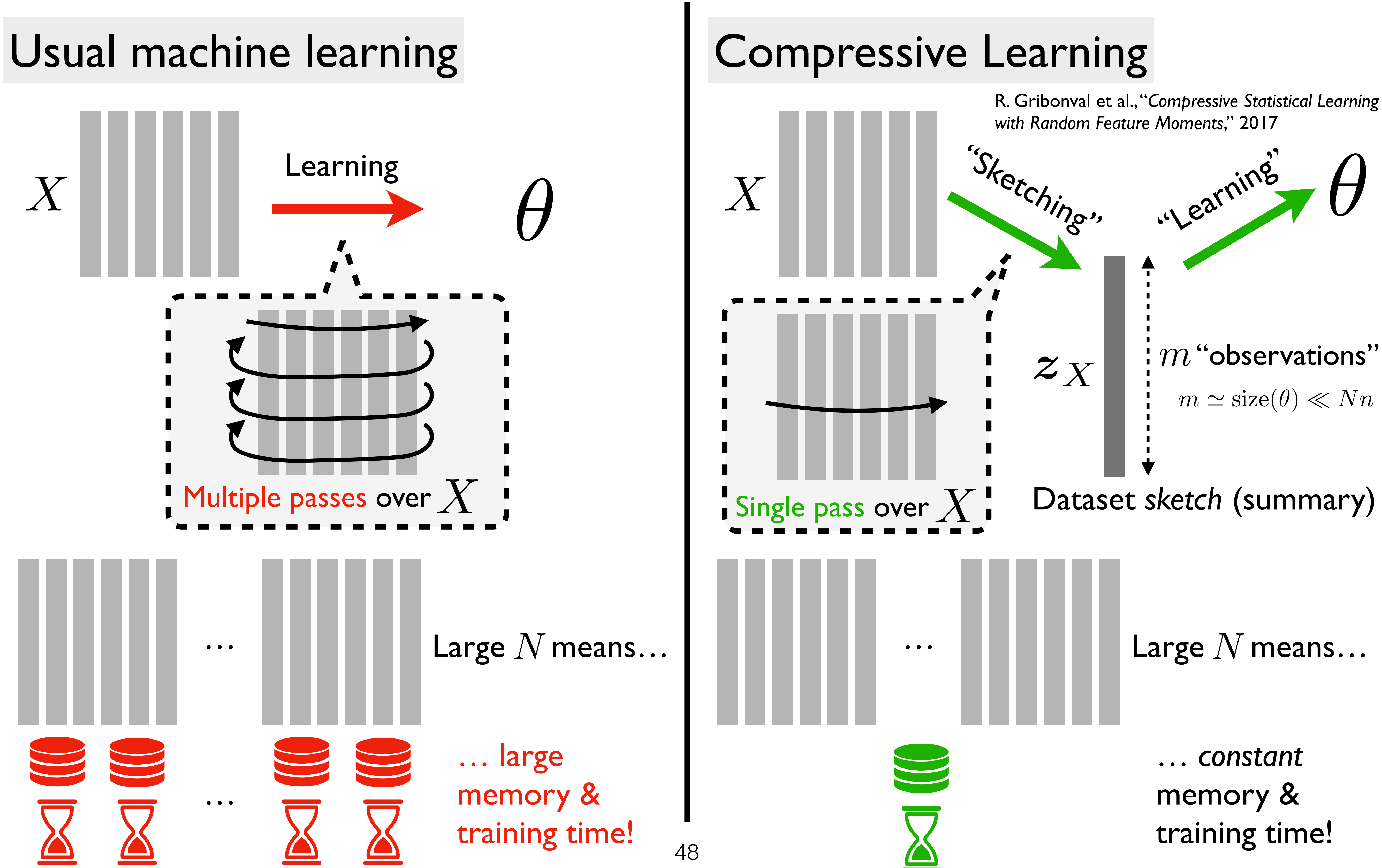


- Dimensionality reduction

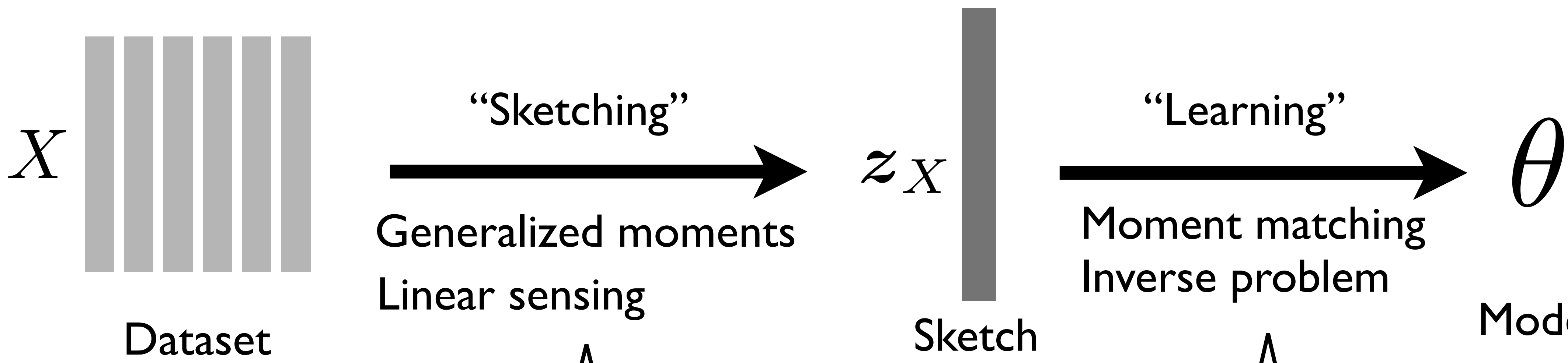


- Autoencoder, GAN, SOM…

Compressive Learning



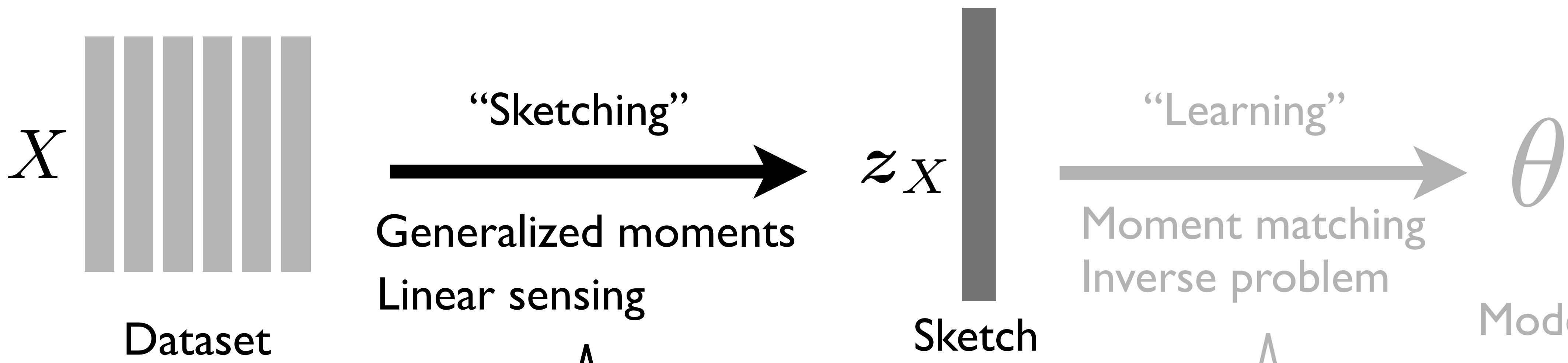
CL challenges



Goal:
- Preserve sufficient information
- Compress as much as possible
- Efficient computation (fast transform, quantized sketch)

Goal:
- Recovery procedure
- Tractable algorithm

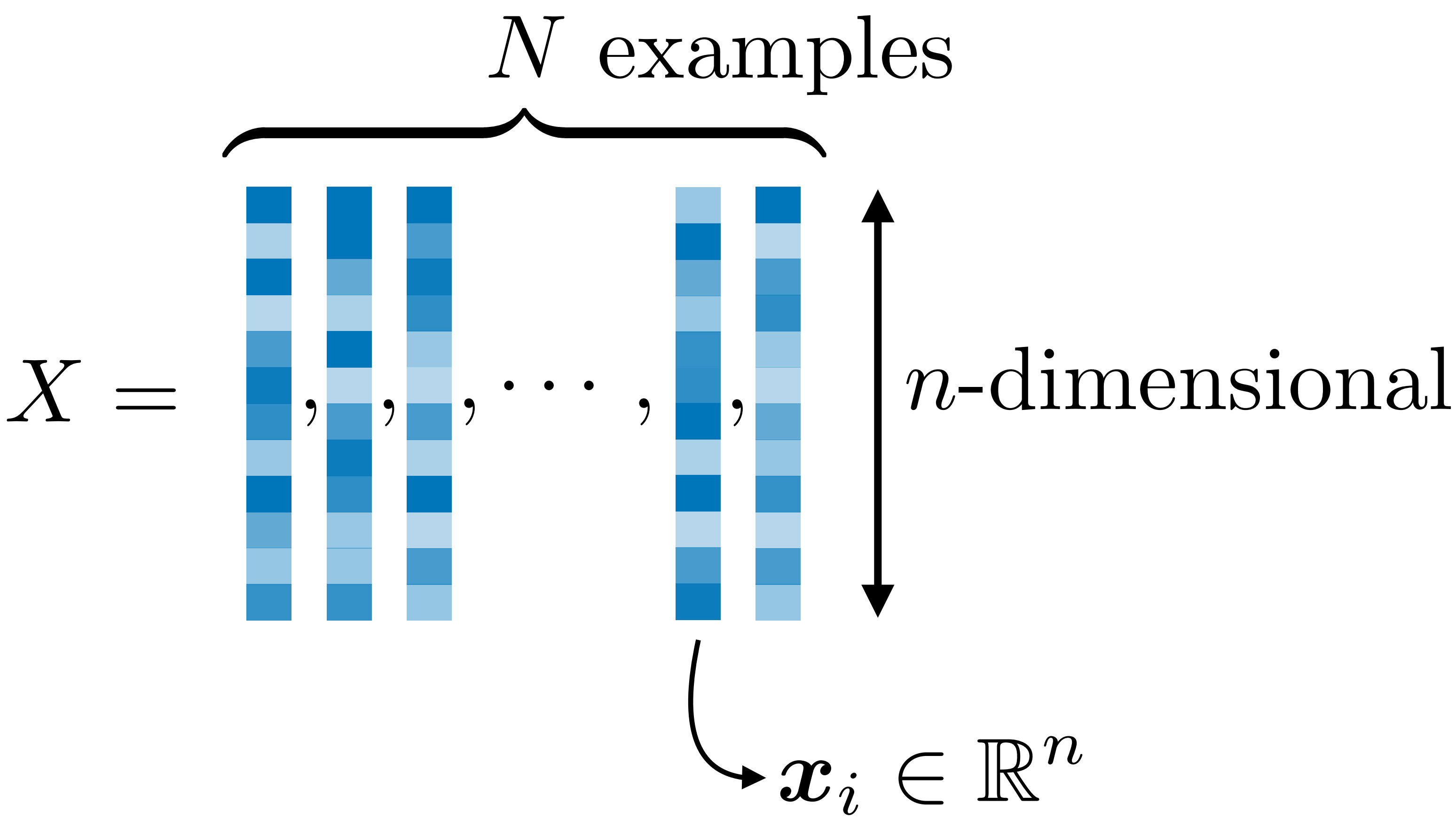
Sketching



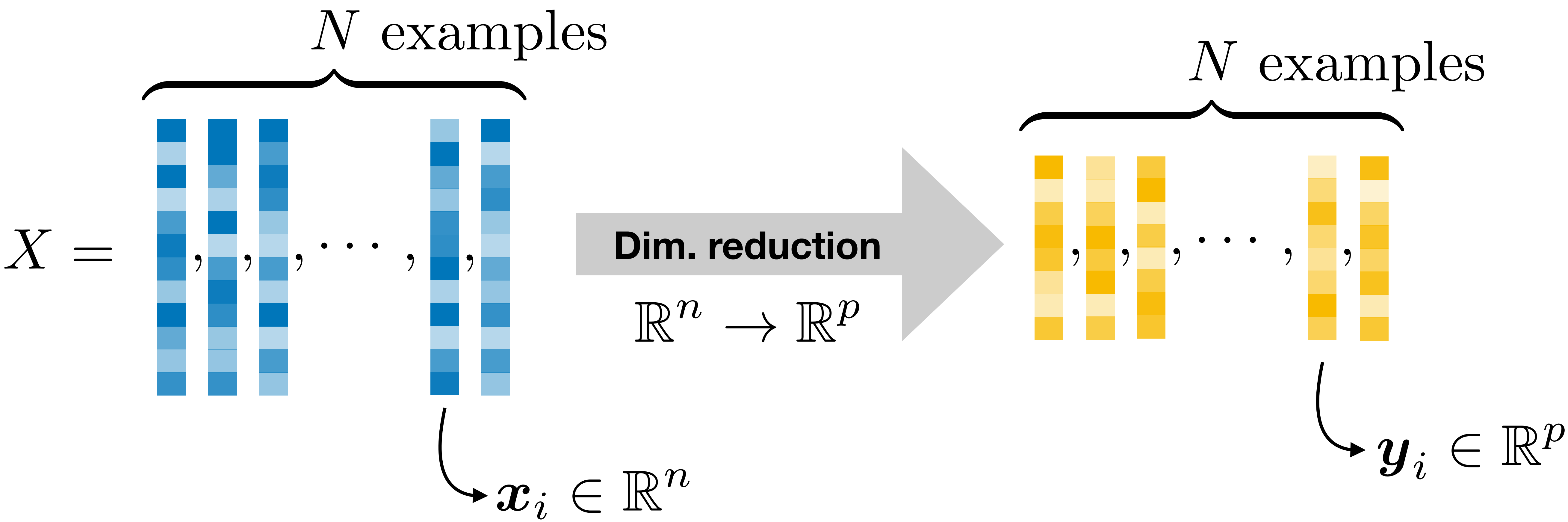
Goal:
- Preserve sufficient information
- Compress as much as possible
- Efficient computation (fast transform, quantized sketch)

Goal:
- Recovery procedure
- Tractable algorithm

Compressing a dataset?

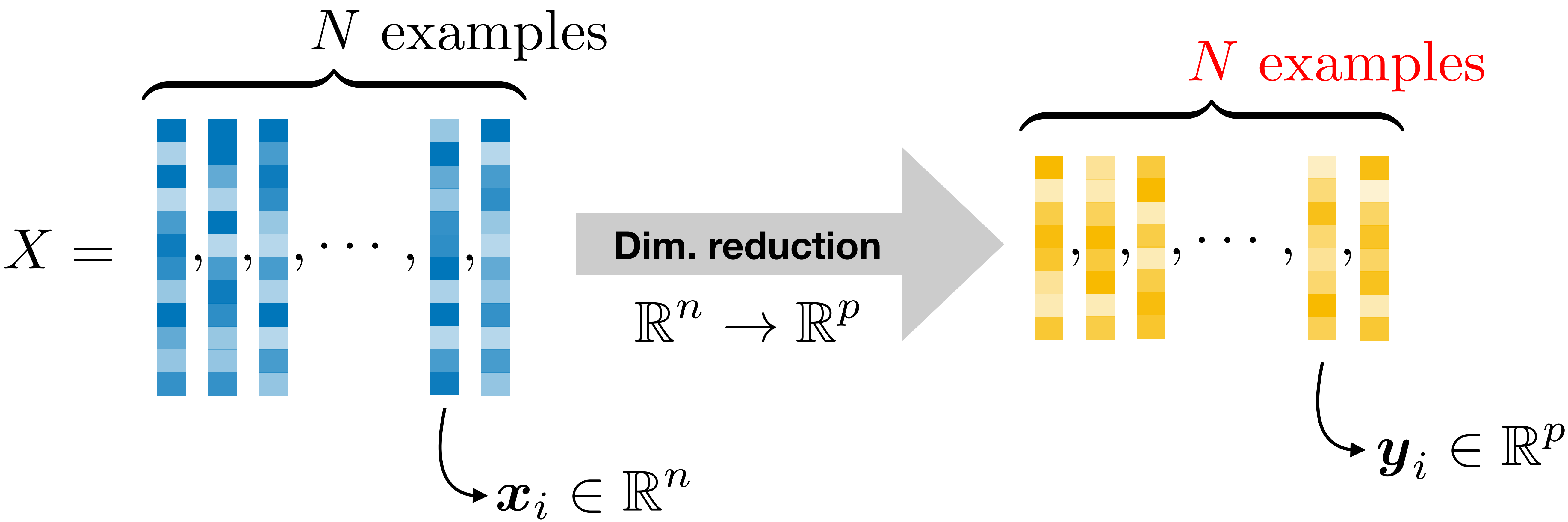


Compressing a dataset?



- **Compressed representation** ✓
- **Preserves relevant information** ✓

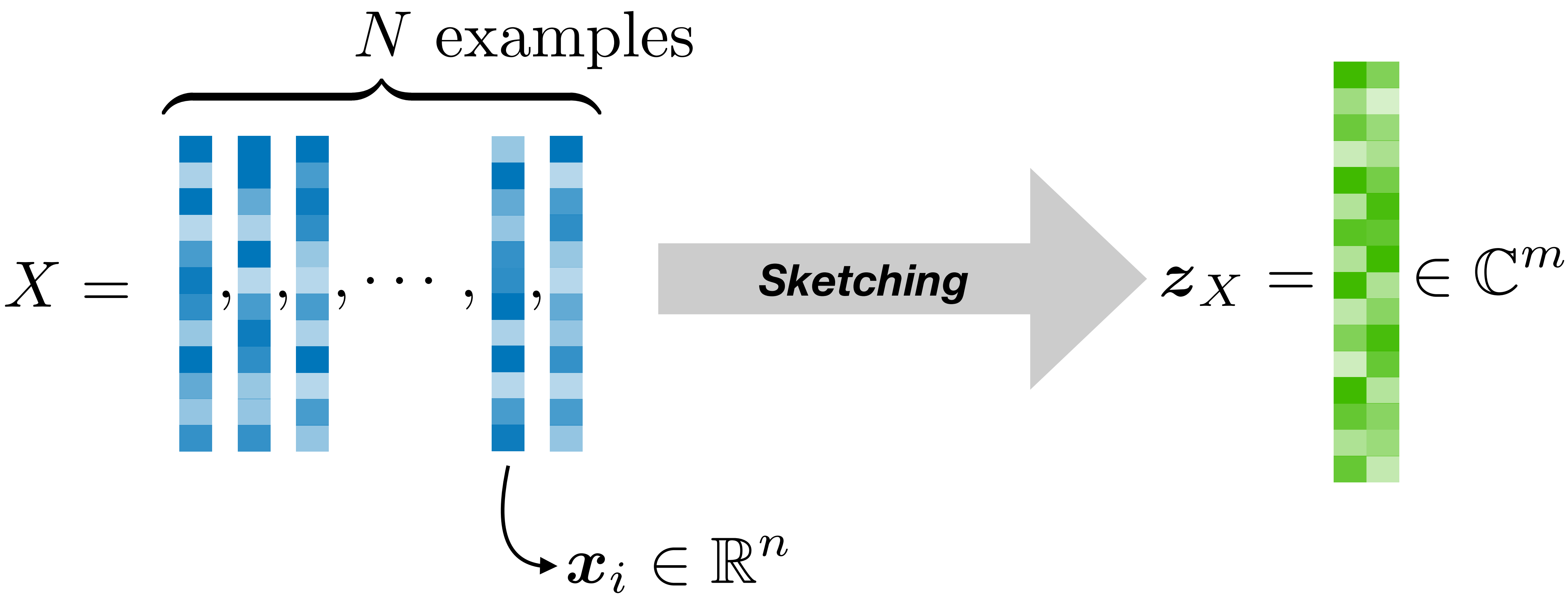
Compressing a dataset?



- **Compressed representation** ✓
- **Preserves relevant information** ✓
- **Constant number of examples** ✗

N **can be VERY large ("big data")!**

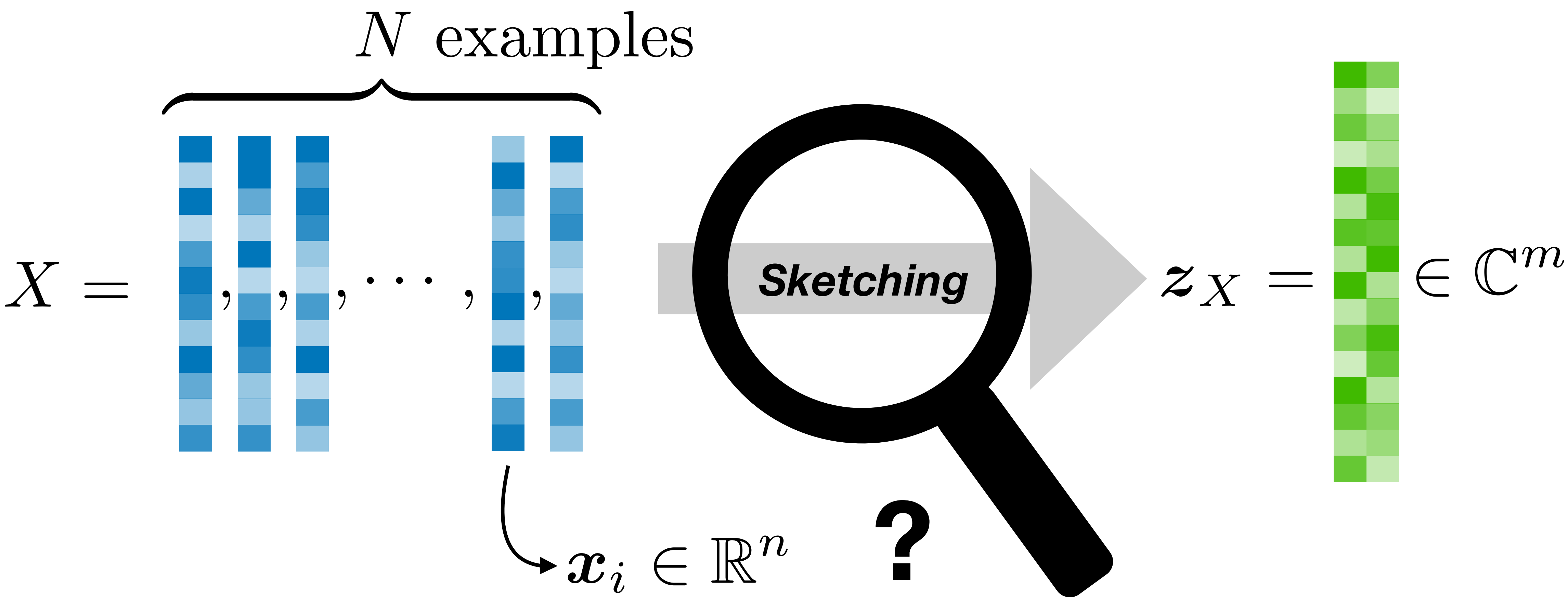
Compressing a dataset!



- **Compressed representation** ✓
- **Preserves relevant information** ✓
- **Dataset summary = single vector** ✓

[Gribonval17]

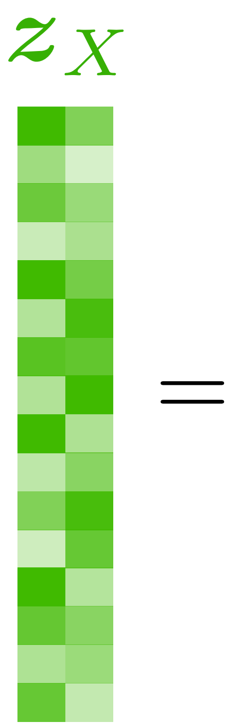
Compressing a dataset!

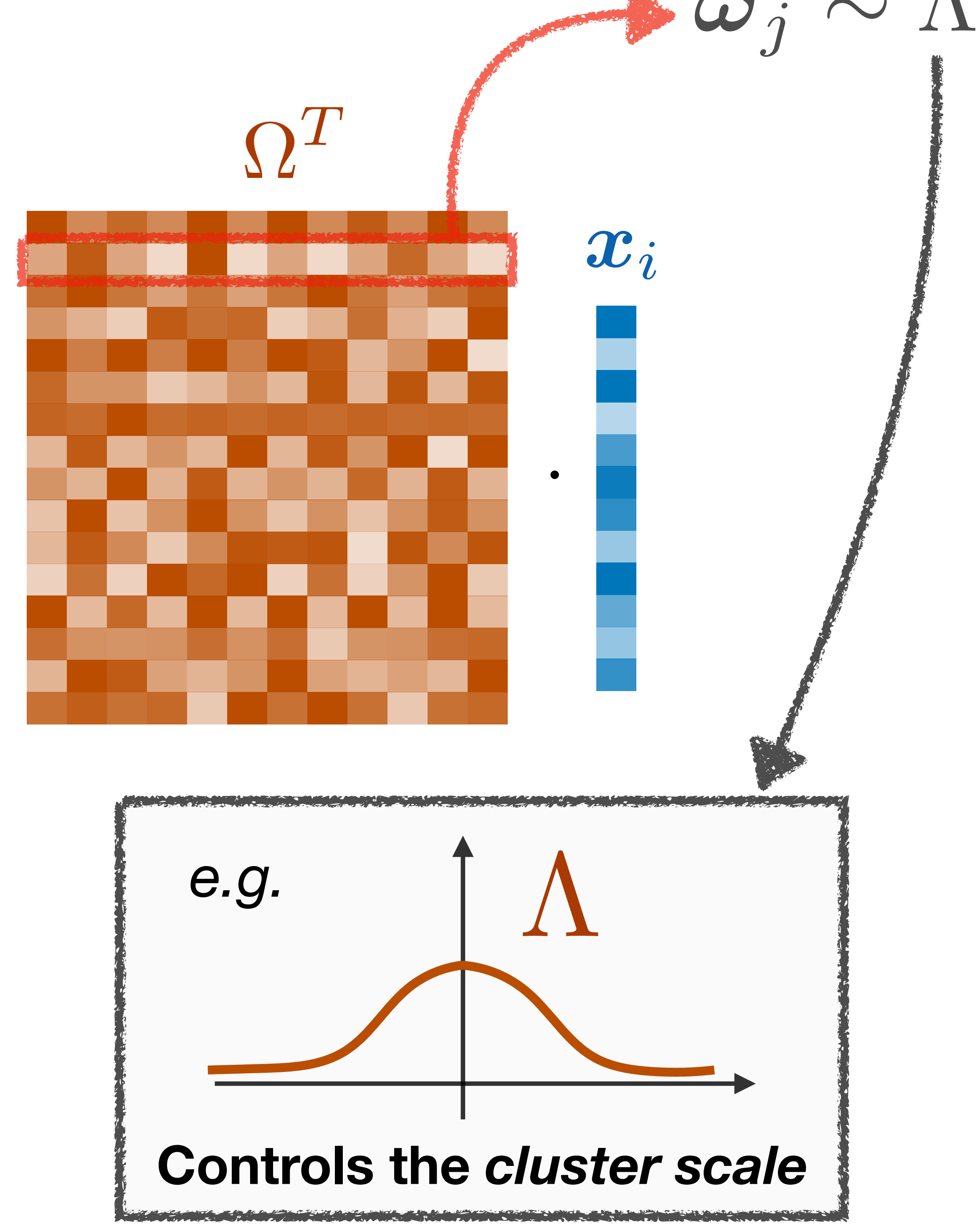


- **Compressed representation** ✓
- **Preserves relevant information** ✓
- **Dataset summary = single vector** ✓

[Gribonval17]

Dataset sketch





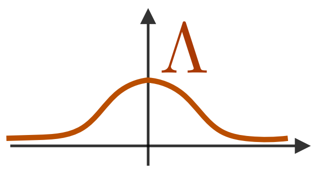
1. Project on m (random) vectors

Dataset sketch

$$\boldsymbol{z}_X = \exp\left(\mathrm{i}\, \Omega^T \cdot \boldsymbol{x}_i \right)$$

$\boldsymbol{\omega}_j \sim \Lambda$

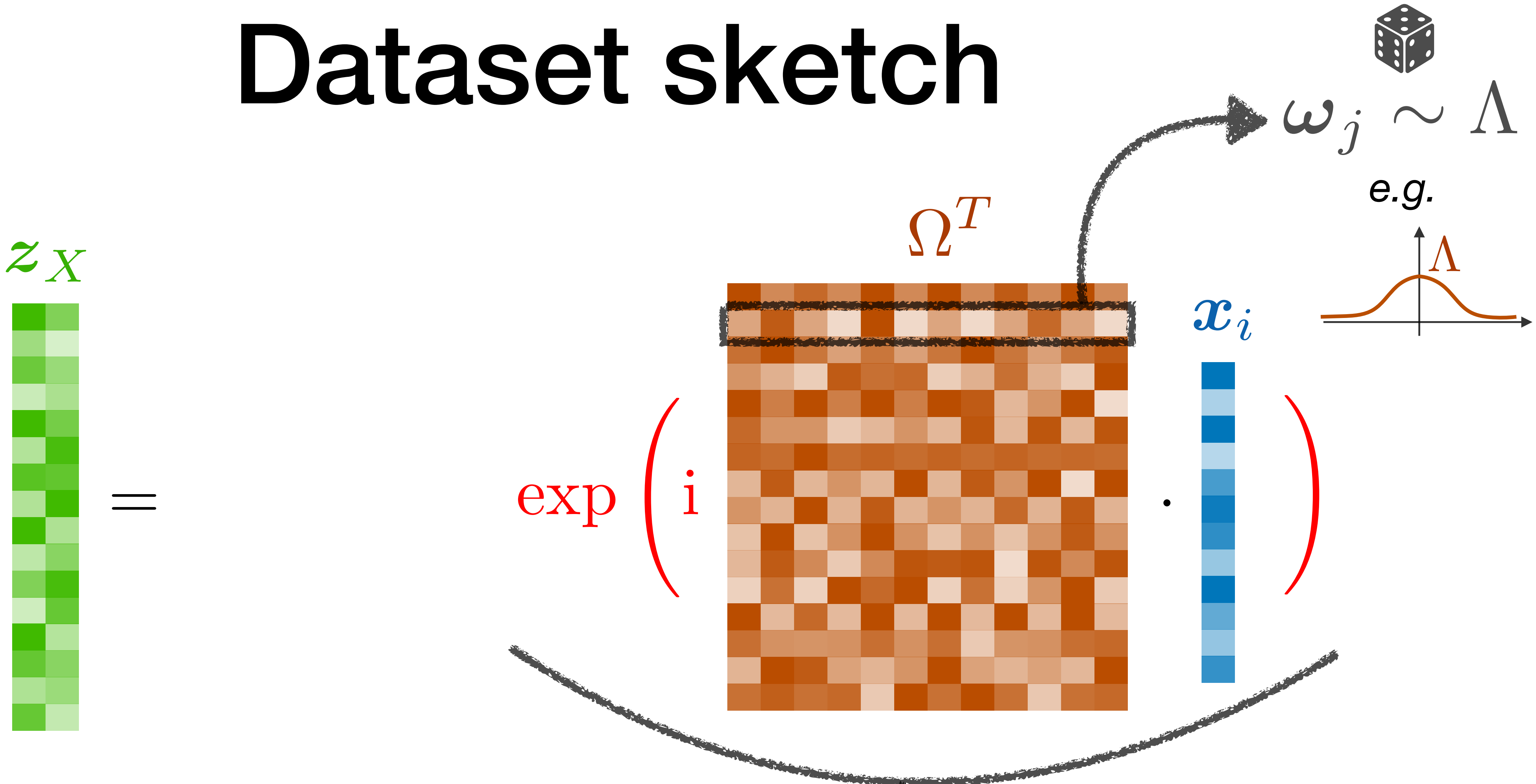
e.g.



1. Project on m (random) vectors

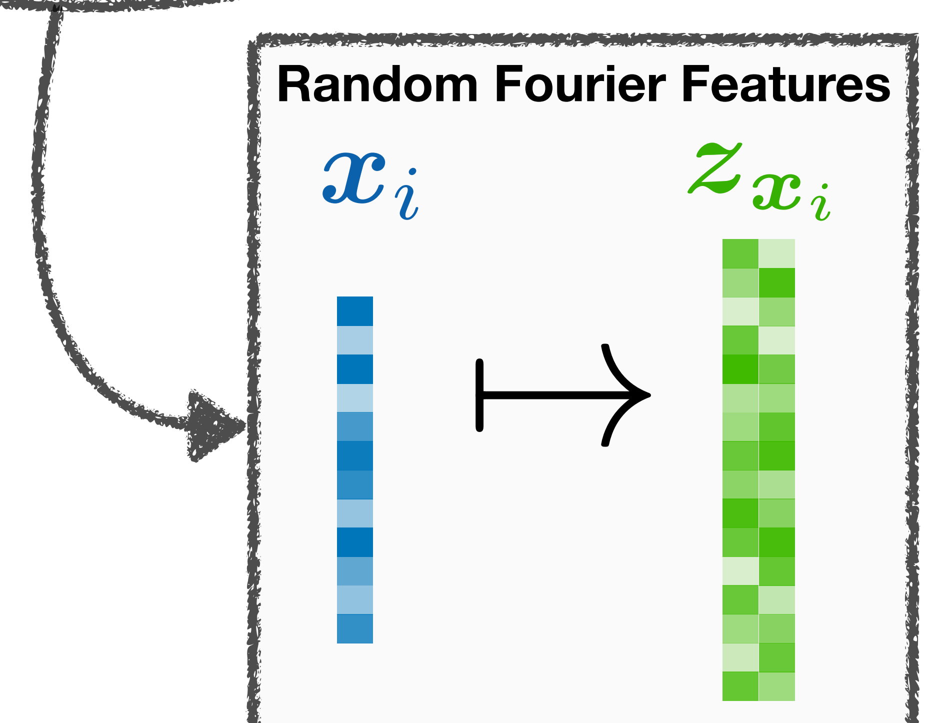
2. Nonlinear periodic signature function

Dataset sketch



1. Project on m (random) vectors

2. Nonlinear periodic signature function

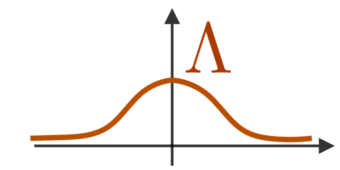


[A. Rahimi, B. Recht, "*Random Features for Large-scale Kernel Machines*",NIPS, 2007]

Dataset sketch

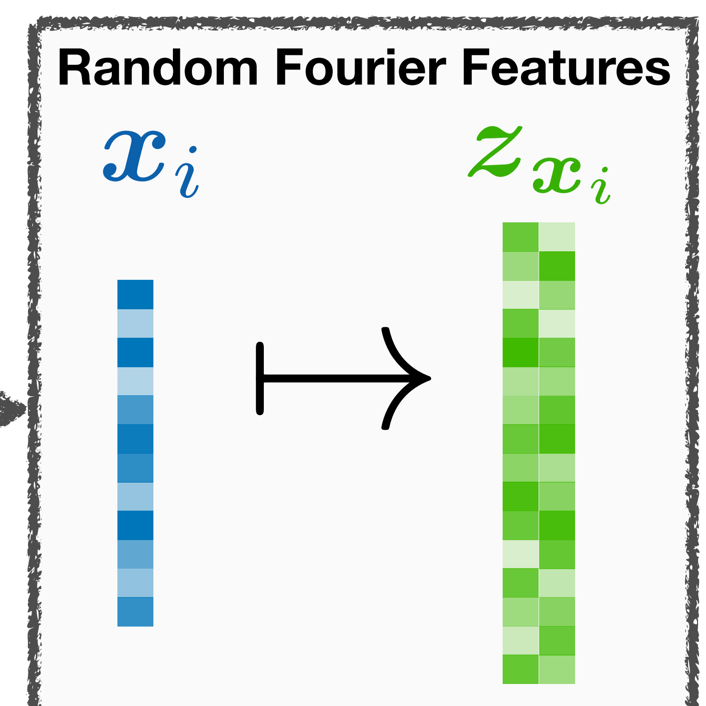
$$\omega_j \sim \Lambda$$

e.g.



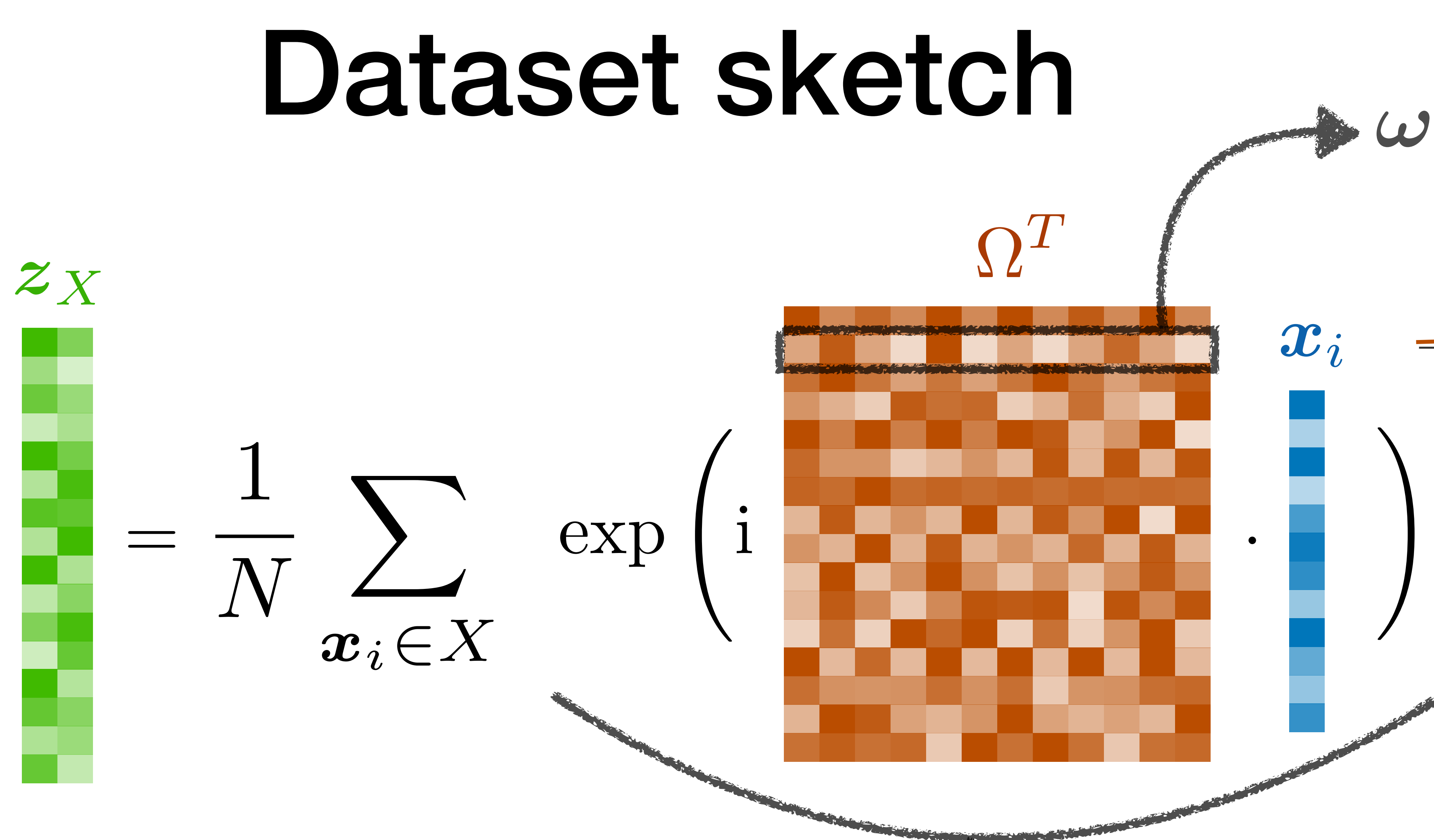
$$z_X = \frac{1}{N} \sum_{x_i \in X} \exp\left(\mathrm{i}\, \Omega^T \cdot x_i \right)$$

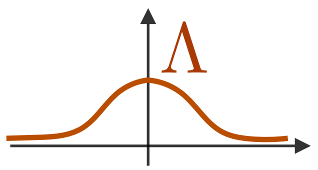
1. Project on m (random) vectors
2. Nonlinear periodic signature function
3. Pooling (average)

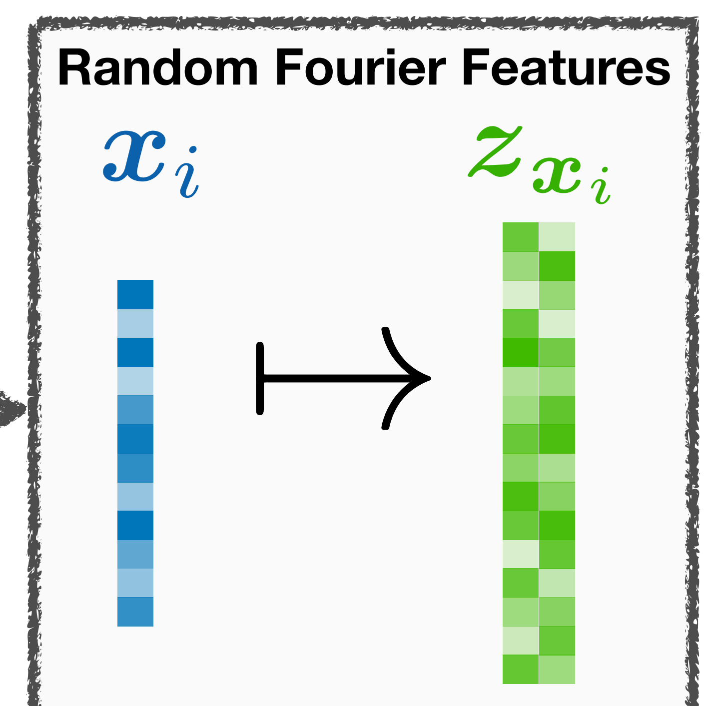


[A. Rahimi, B. Recht, "*Random Features for Large-scale Kernel Machines*",NIPS, 2007]

Dataset sketch







1. Project on m (random) vectors

2. Nonlinear periodic signature function

3. Pooling (average)

$$\boldsymbol{z}_X = \left[\frac{1}{N} \sum_{\boldsymbol{x}_i \in X} e^{\mathrm{i}\boldsymbol{\omega}_j^T \boldsymbol{x}_i} \right]_{j=1}^{m} \in \mathbb{C}^m$$

[A. Rahimi, B. Recht, "*Random Features for Large-scale Kernel Machines*", NIPS, 2007]

Sketched learning

X

Dataset

"Sketching"

Generalized moments
Linear sensing

z_X

Sketch

"Learning"

Moment matching
Inverse problem

θ

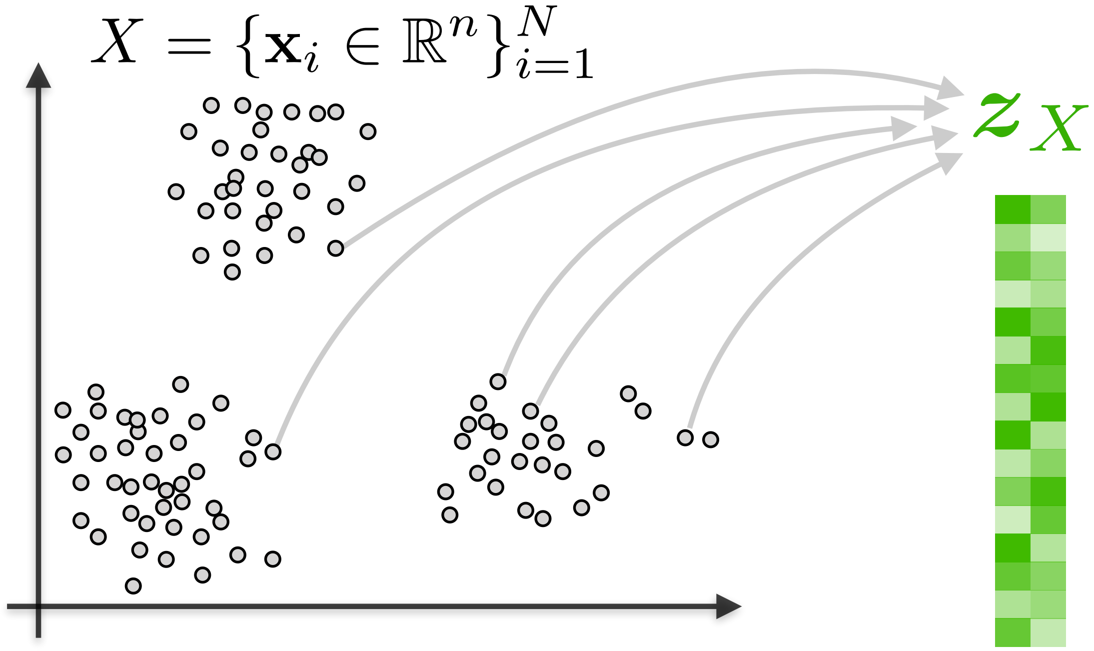
Model

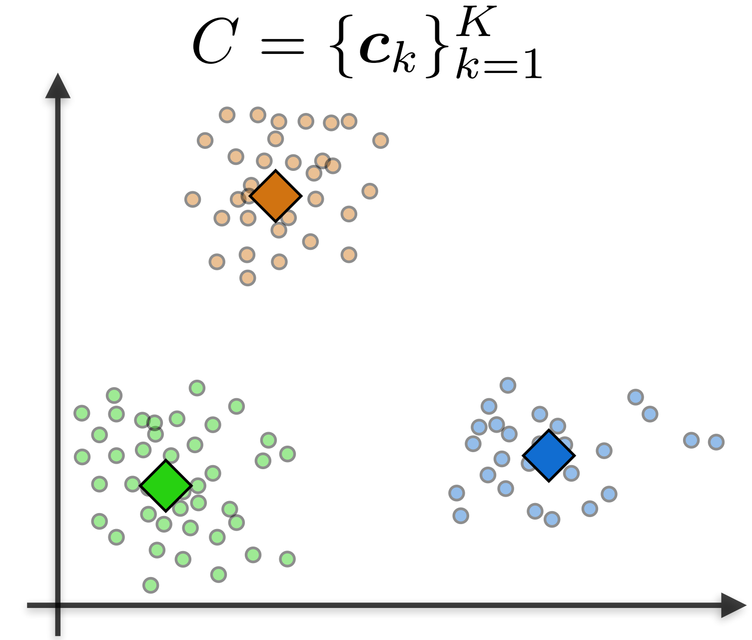
Goal:
- Preserve sufficient information
- Compress as much as possible
- Efficient computation (fast transform, quantized sketch)

Goal:
- Recovery procedure
- Tractable algorithm

Illustration here: Compressive K-Means

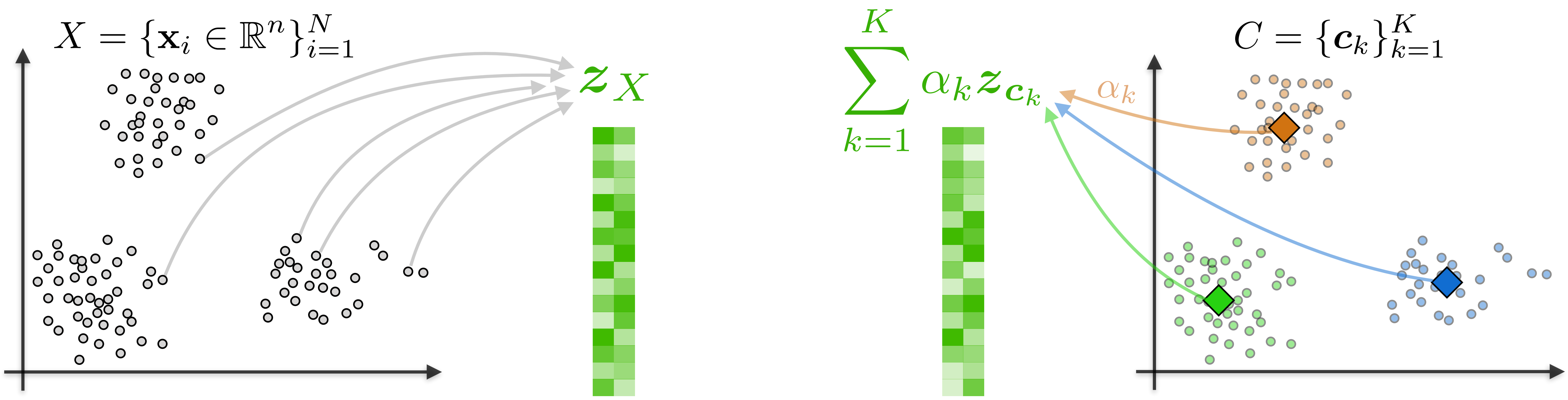
Compressive K-Means





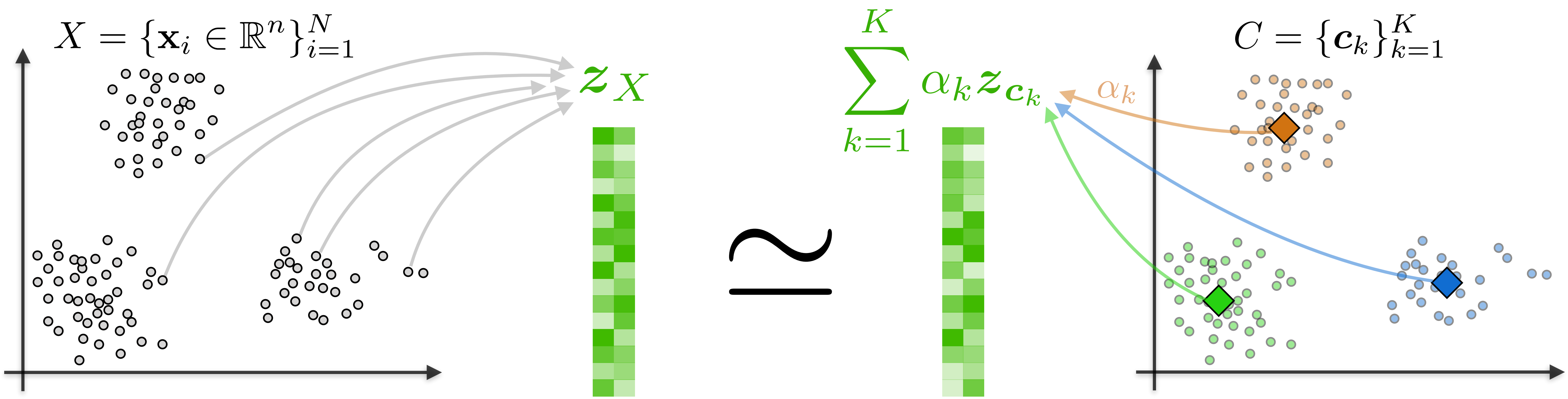
[N. Keriven et al., “*Compressive K-Means*”, IEEE ICASSP, 2017]

Compressive K-Means



[N. Keriven et al., "*Compressive K-Means*", IEEE ICASSP, 2017]

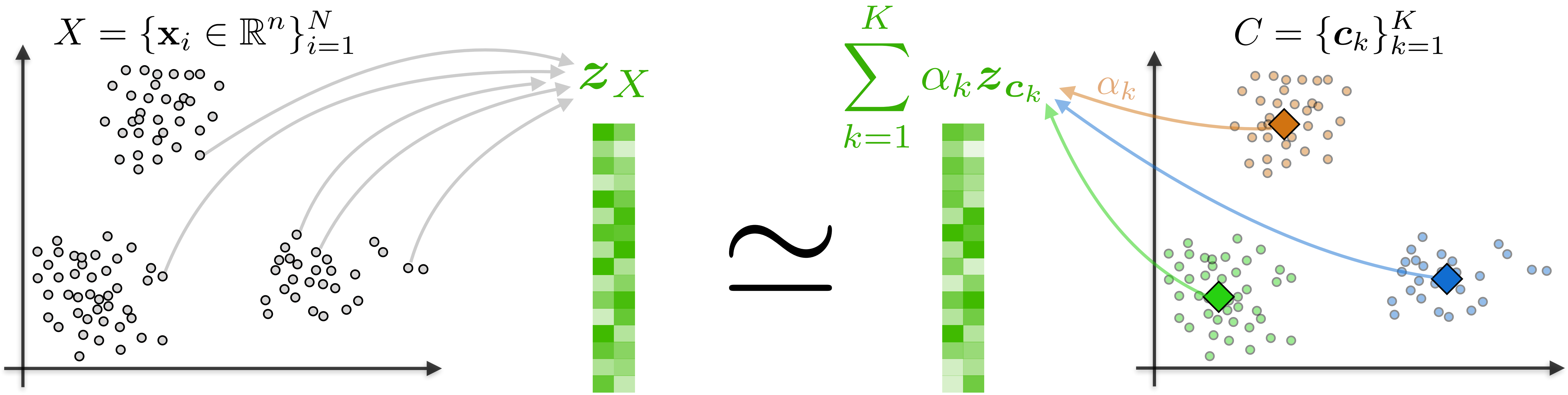
Compressive K-Means



Idea: sketch matching (inverse problem)

[N. Keriven et al., "*Compressive K-Means*", IEEE ICASSP, 2017]

Compressive K-Means



Idea: sketch matching (inverse problem)

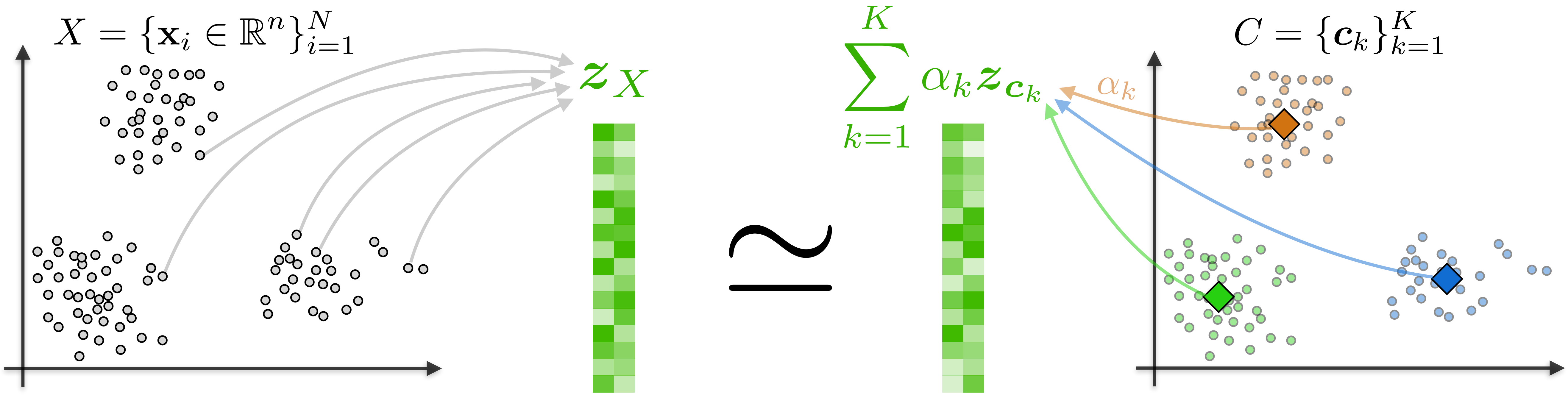
$$\min_{C,\boldsymbol{\alpha}} \|\boldsymbol{z}_X - \sum_{k=1}^{K} \alpha_k \boldsymbol{z}_{\boldsymbol{c}_k}\|_2^2$$

Nonconvex optimization

Approximatively solved by greedy algorithm

[N. Keriven et al., "*Compressive K-Means*", IEEE ICASSP, 2017]

Compressive K-Means

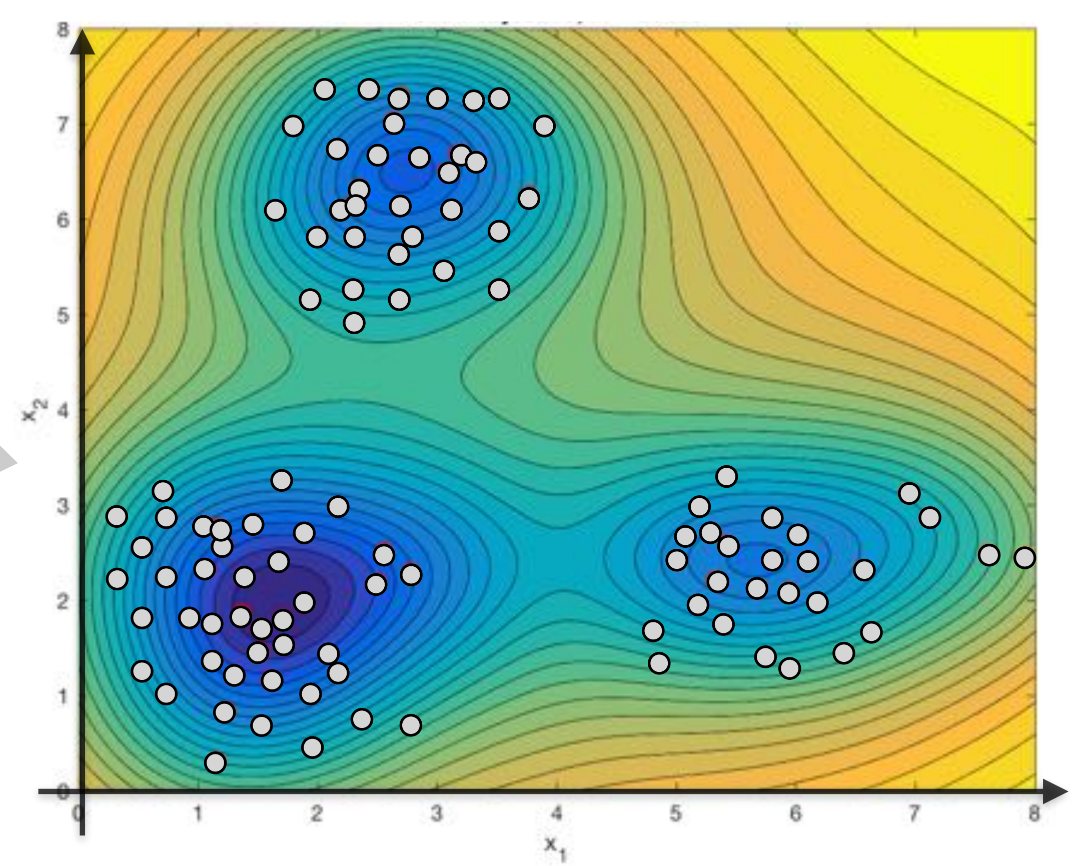


Idea: sketch matching (inverse problem)

$$\min_{C,\boldsymbol{\alpha}} \|\boldsymbol{z}_X - \sum_{k=1}^{K} \alpha_k \boldsymbol{z}_{\boldsymbol{c}_k}\|_2^2$$

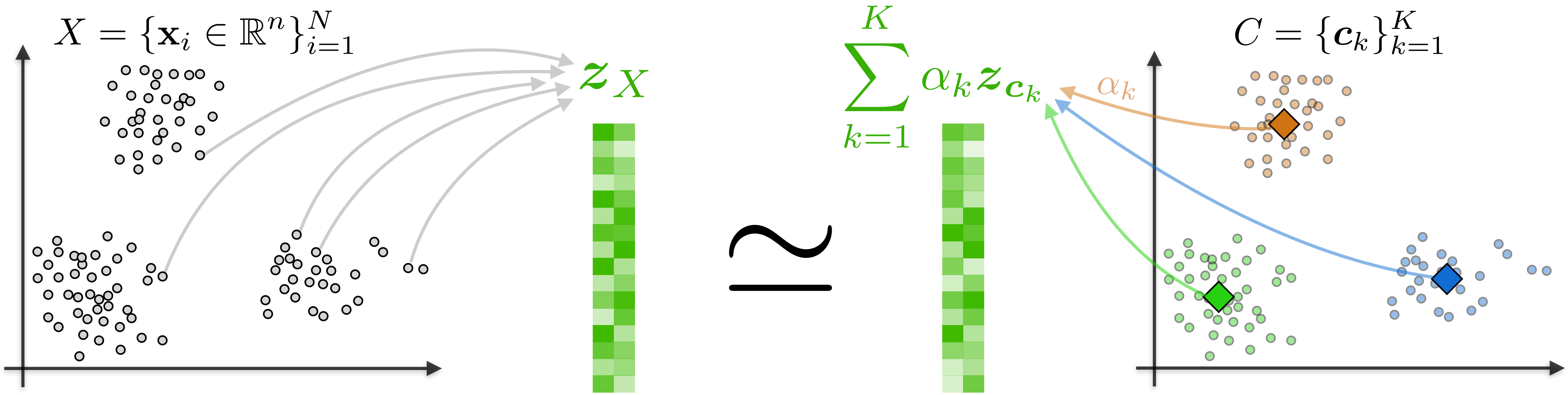
Nonconvex optimization

$m \to \infty$



[N. Keriven et al., "*Compressive K-Means*", IEEE ICASSP, 2017]

Compressive K-Means

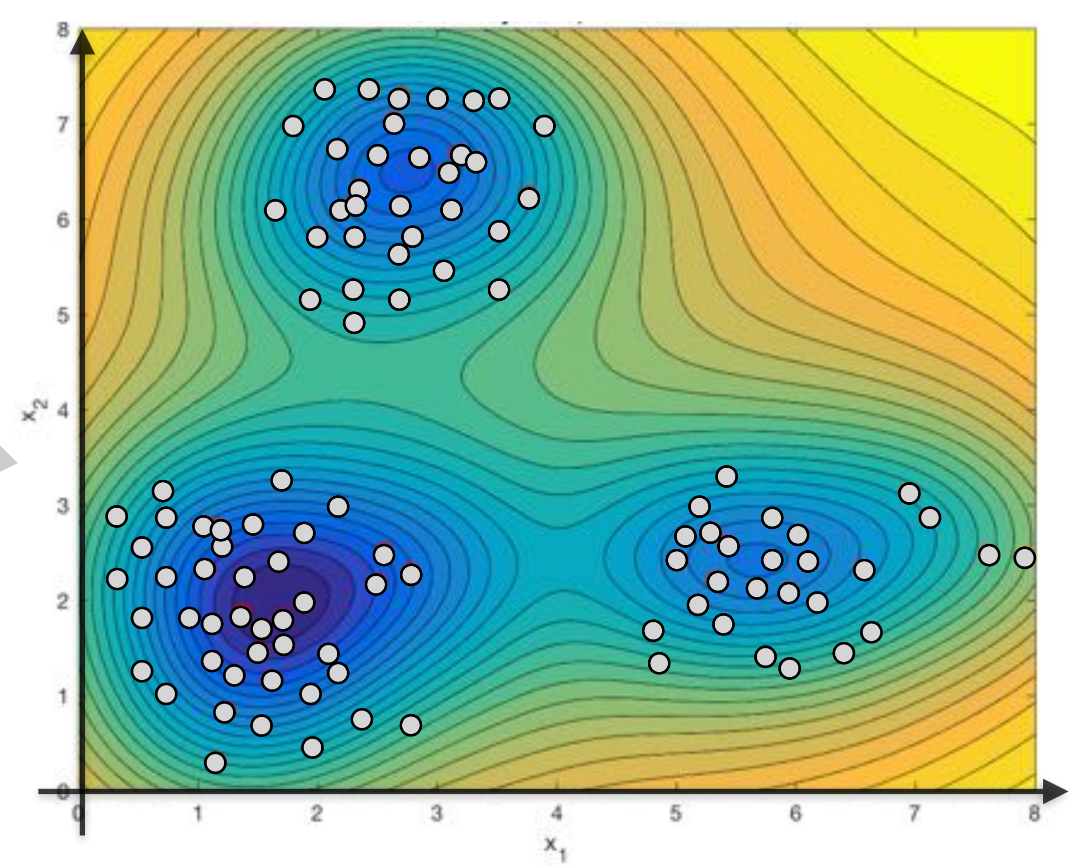


Idea: sketch matching (inverse problem)

$$\min_{C,\boldsymbol{\alpha}} \|\boldsymbol{z}_X - \sum_{k=1}^{K} \alpha_k \boldsymbol{z}_{\boldsymbol{c}_k}\|_2^2$$

Nonconvex optimization

$m \to \infty$



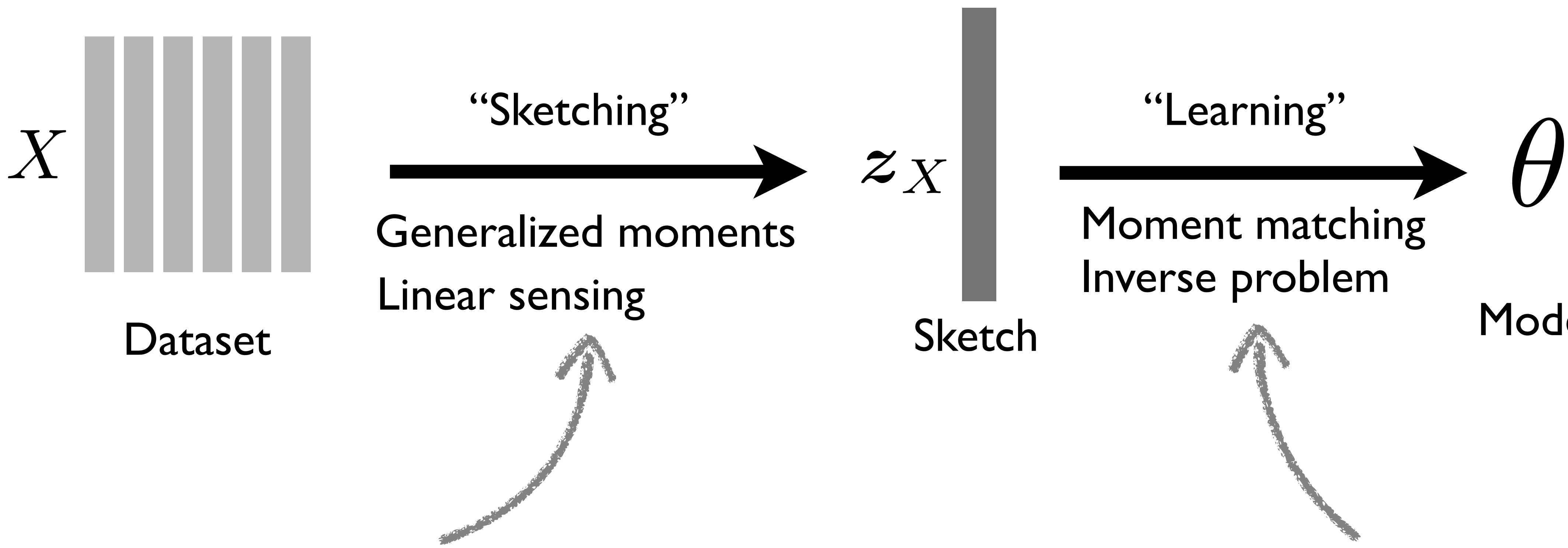
Empirically: ok when

$$m = \mathcal{O}(\underbrace{nK}_{\text{Model size}})$$

No dependence on N!

[N. Keriven et al., "*Compressive K-Means*", IEEE ICASSP, 2017]

CL in a nutshell



Sketch operator

$$\boldsymbol{z}_X = \frac{1}{N} \sum_{\boldsymbol{x}_i \in X} \exp(\mathrm{i}\Omega^T \boldsymbol{x}_i)$$

Optimal decoder

$$\Delta[\boldsymbol{y}] \in \arg\min_{\pi \in \mathcal{G}} \|\boldsymbol{y} - \mathcal{A}(\pi)\|_2$$

- Can be done in one pass, online, in //…
- Privacy-preserving (??)

- Nonconvex optimization
- Complexity independent of N

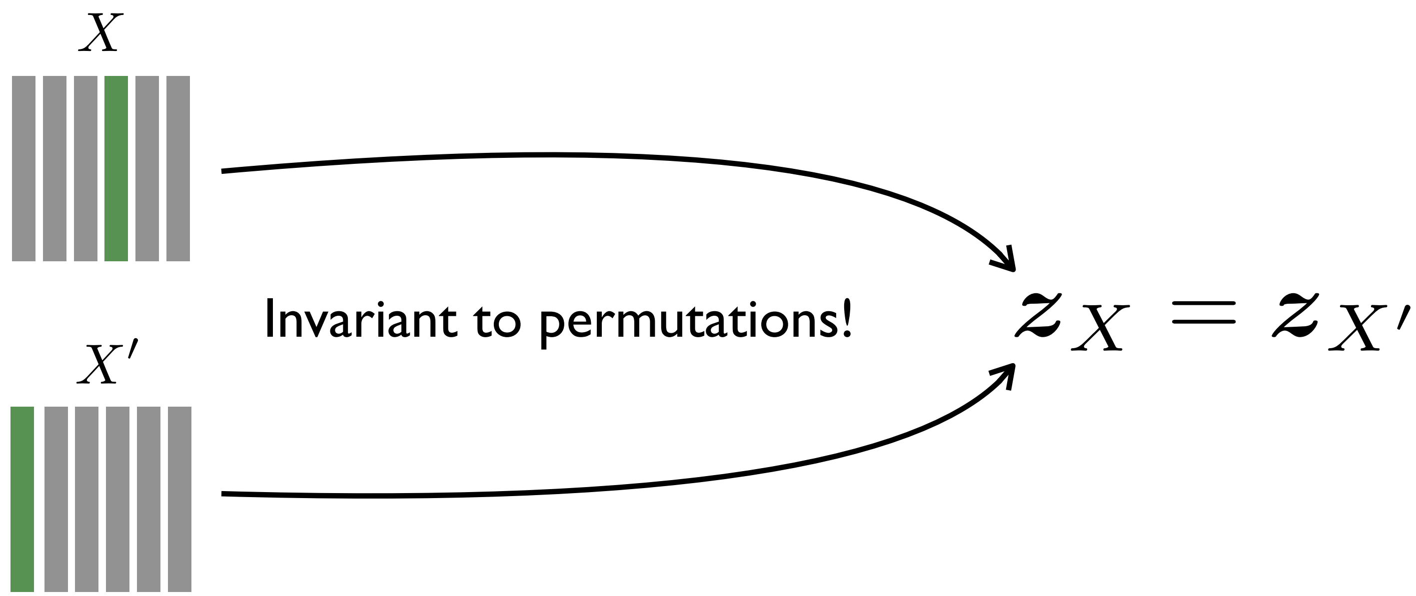
In this talk…

Part 3

Privacy-Preserving Compressive Learning

Compressive Learning and Privacy

Intuitively, releasing only the sketch provides some form of *(N-)anonymity*…

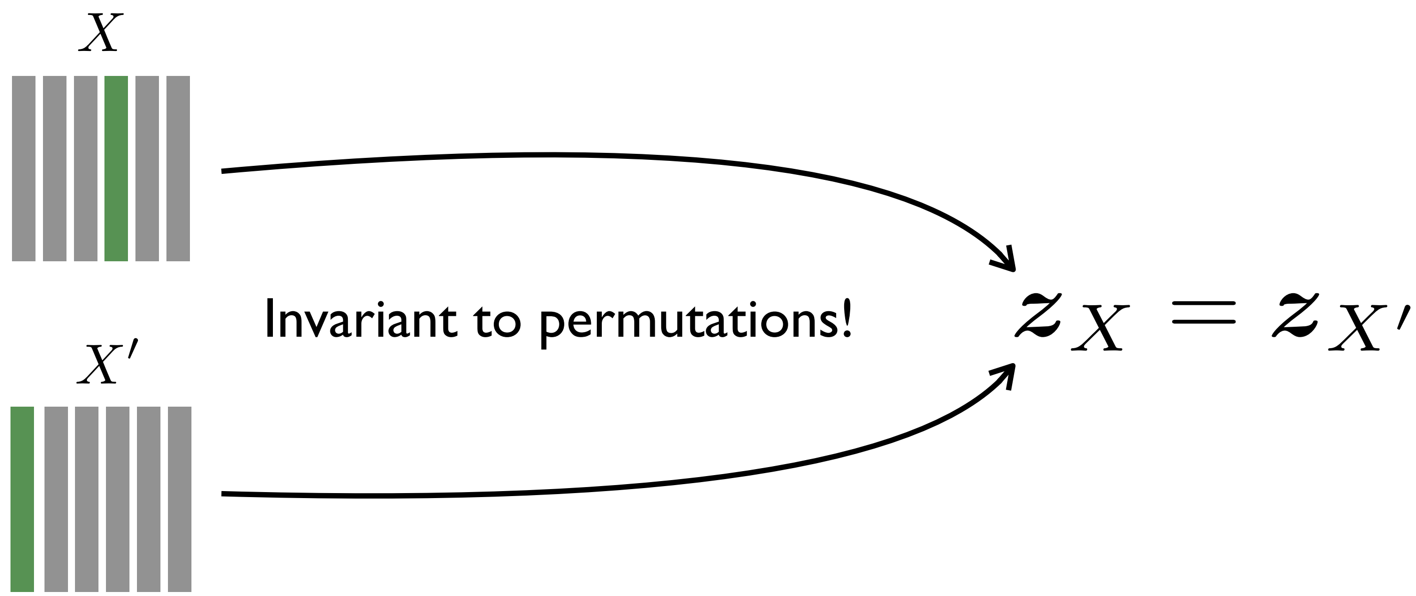


Sketch operator

$$\boldsymbol{z}_X = \frac{1}{N} \sum_{\boldsymbol{x}_i \in X} \exp(\mathrm{i}\Omega^T \boldsymbol{x}_i)$$

Compressive Learning and Privacy

Intuitively, releasing only the sketch provides some form of *(N-)anonymity*…



Sketch operator

$$\boldsymbol{z}_X = \frac{1}{N} \sum_{\boldsymbol{x}_i \in X} \exp(\mathrm{i}\Omega^T \boldsymbol{x}_i)$$

A stronger, formal privacy guarantee for Compressive Learning? >>> DP!

Besides DPs advantages, a good match:

CL: "we <u>forget the individual signals</u> and store only statistics of the dataset"

DP: "the output is <u>not much influenced by one signal</u>"

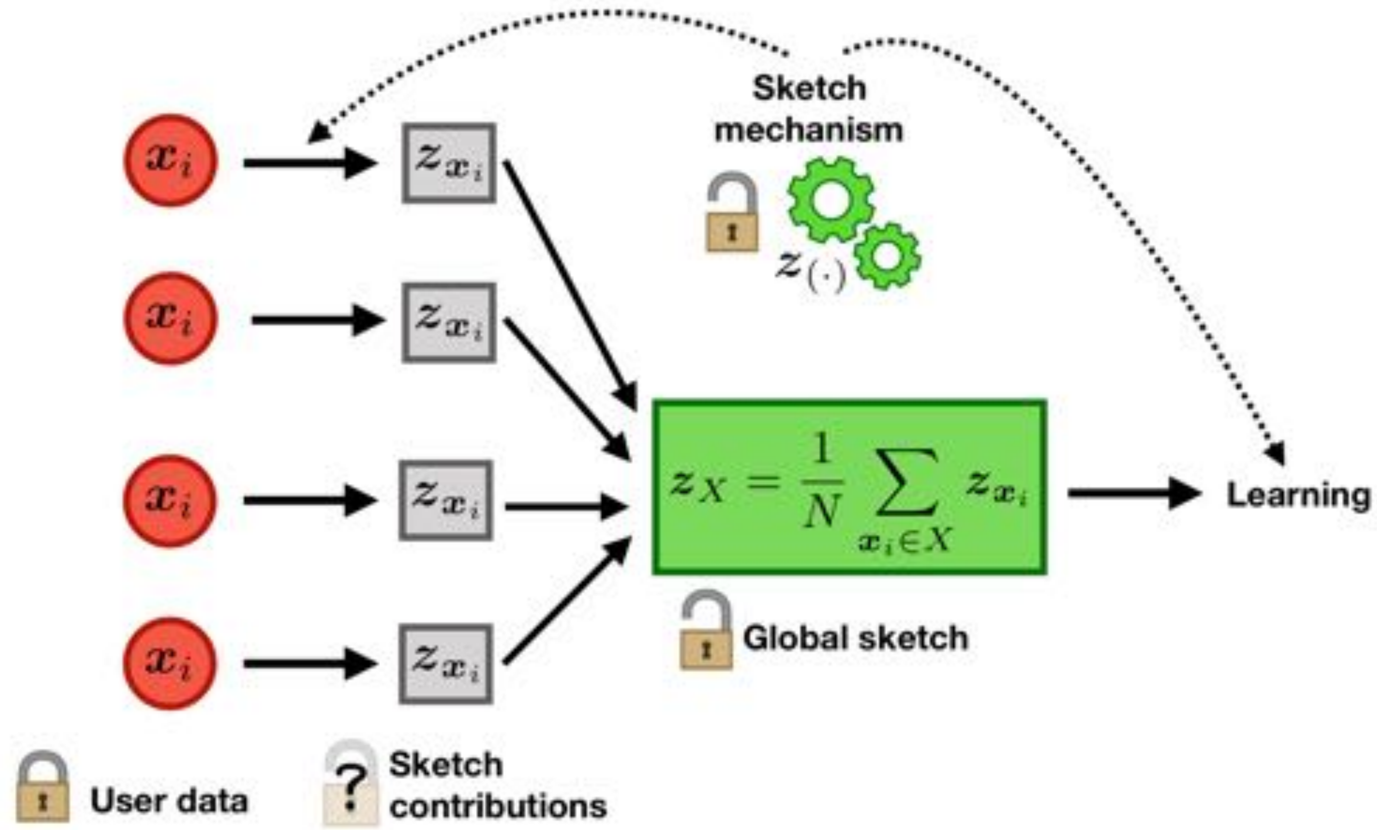
$\epsilon - \mathrm{DP}$

f satisfies $\epsilon - \mathrm{DP}$ if: $\forall S$, $\forall X \sim X'$

$$\mathbb{P}[f(X) \in S] \leq e^{\epsilon} \cdot \mathbb{P}[f(X') \in S]$$

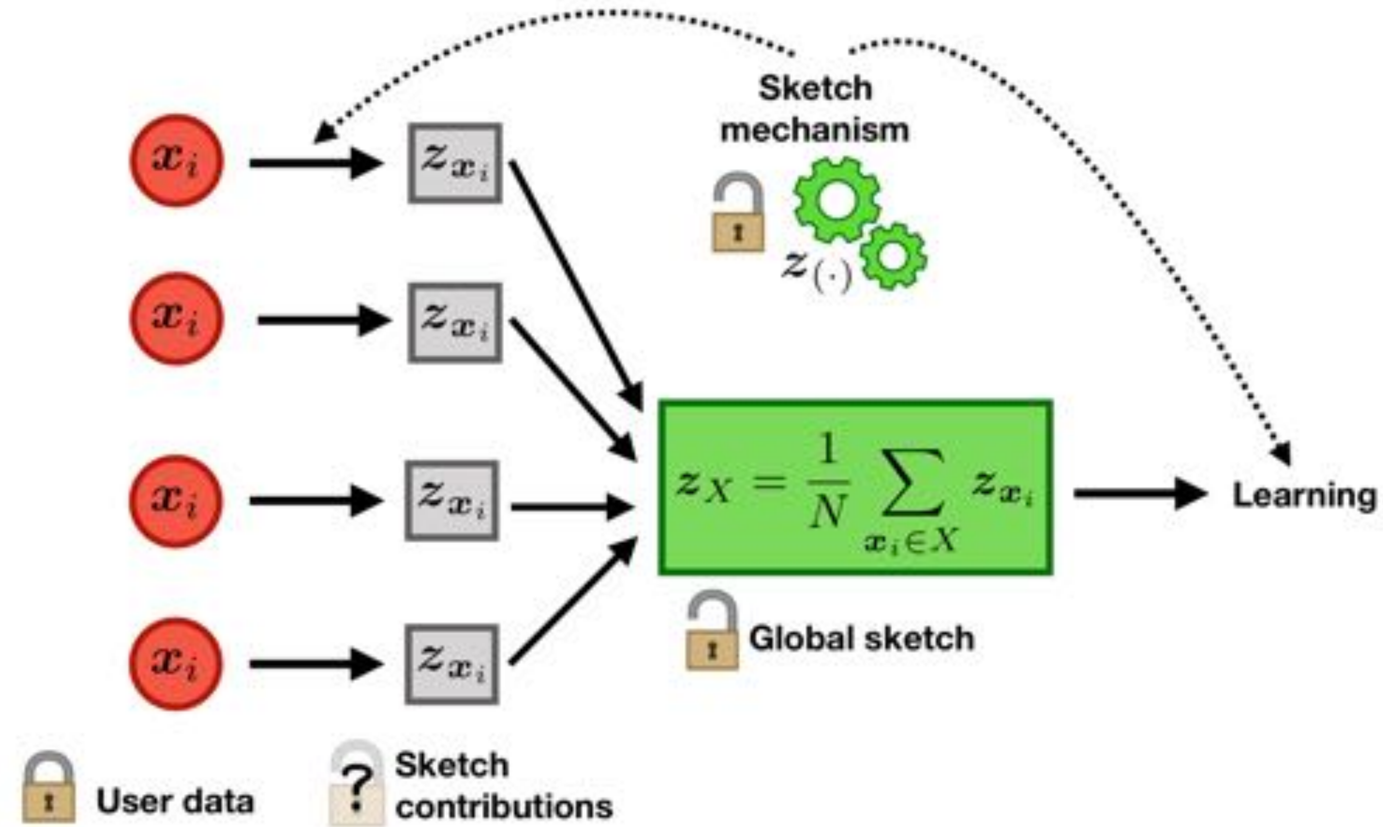
Private CL: attack model

What is publicly available and what is kept secret?

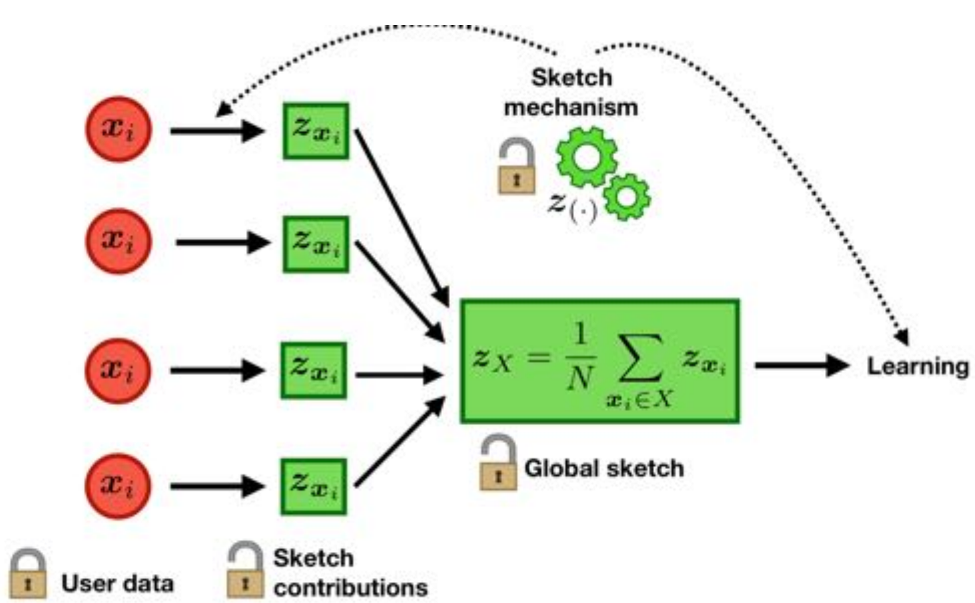


Private CL: attack model

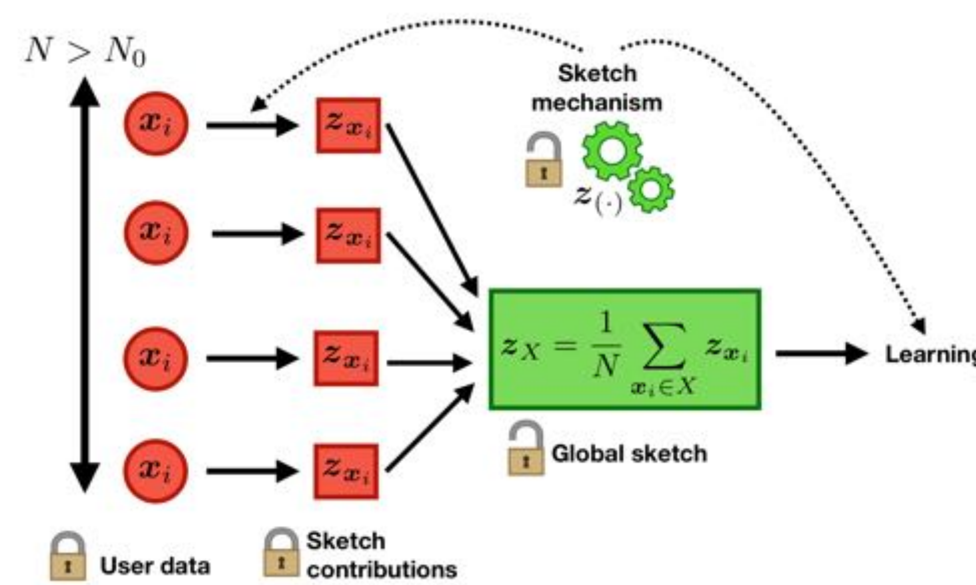
What is publicly available and what is kept secret?



Two extreme cases:



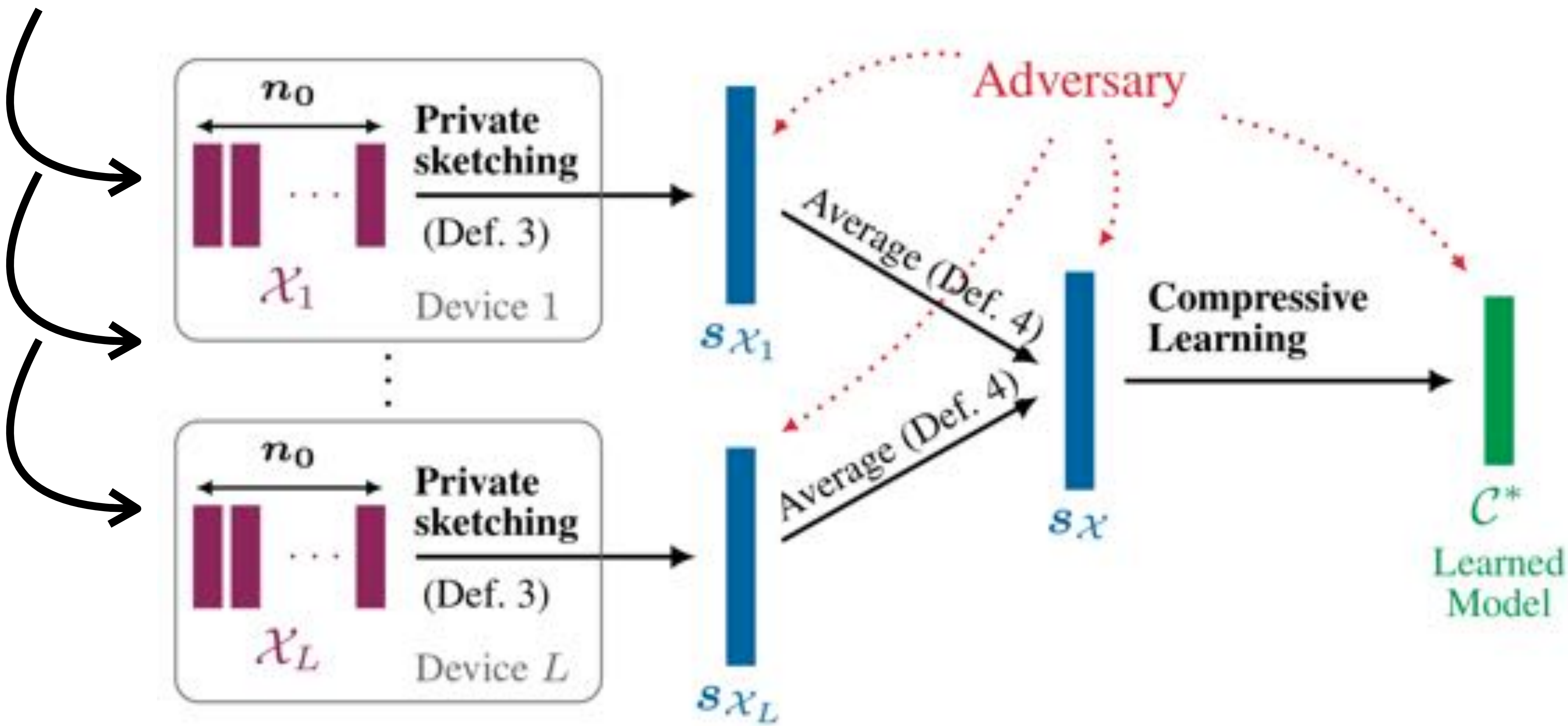
VS



Private CL: attack model

Model combining the two extreme cases:

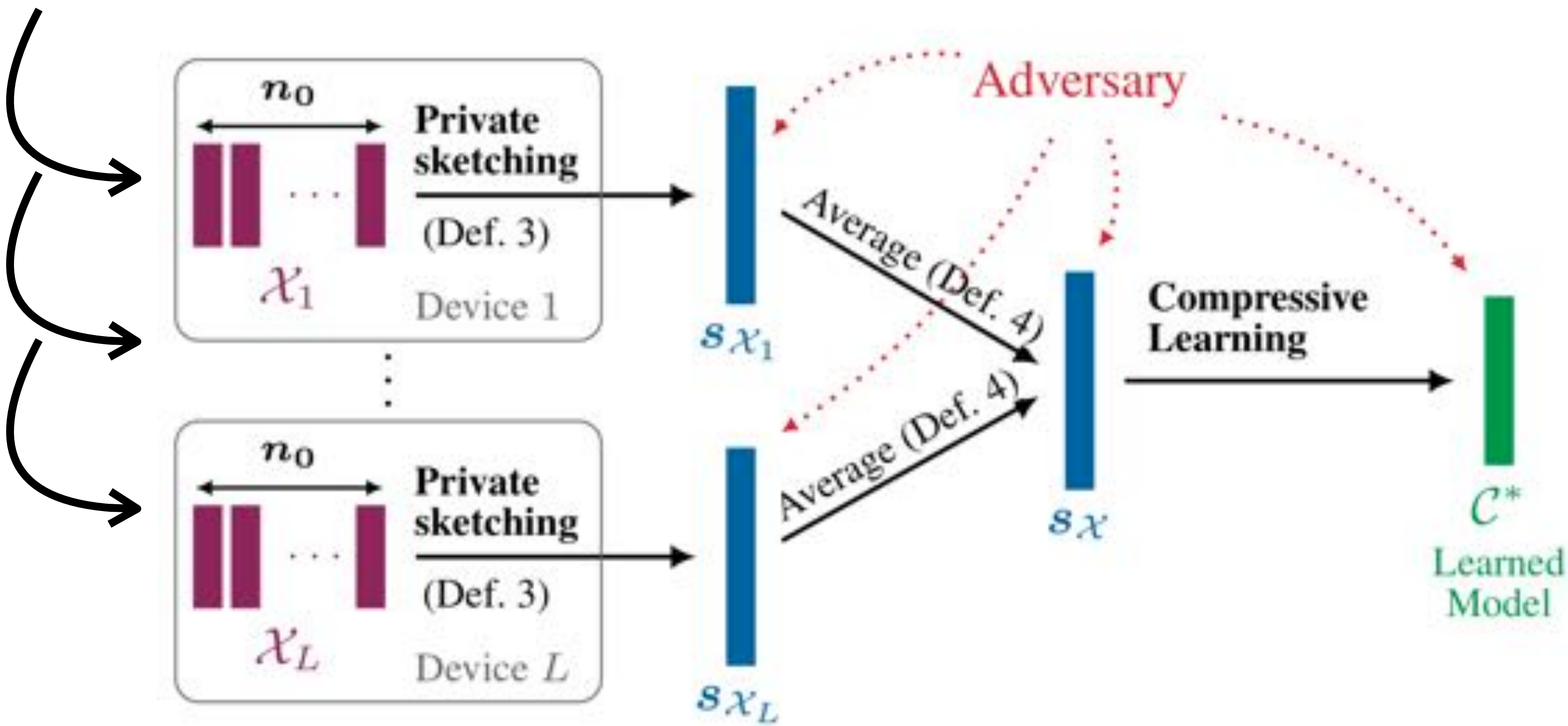
Dataset is shared across L devices…



Private CL: attack model

Model combining the two extreme cases:

Dataset is shared across L devices…



…each device holds n_0 signals… …and releases a (privacy-preserving) local sketch!

$$N = L \times n_0$$

Private CL: attack model

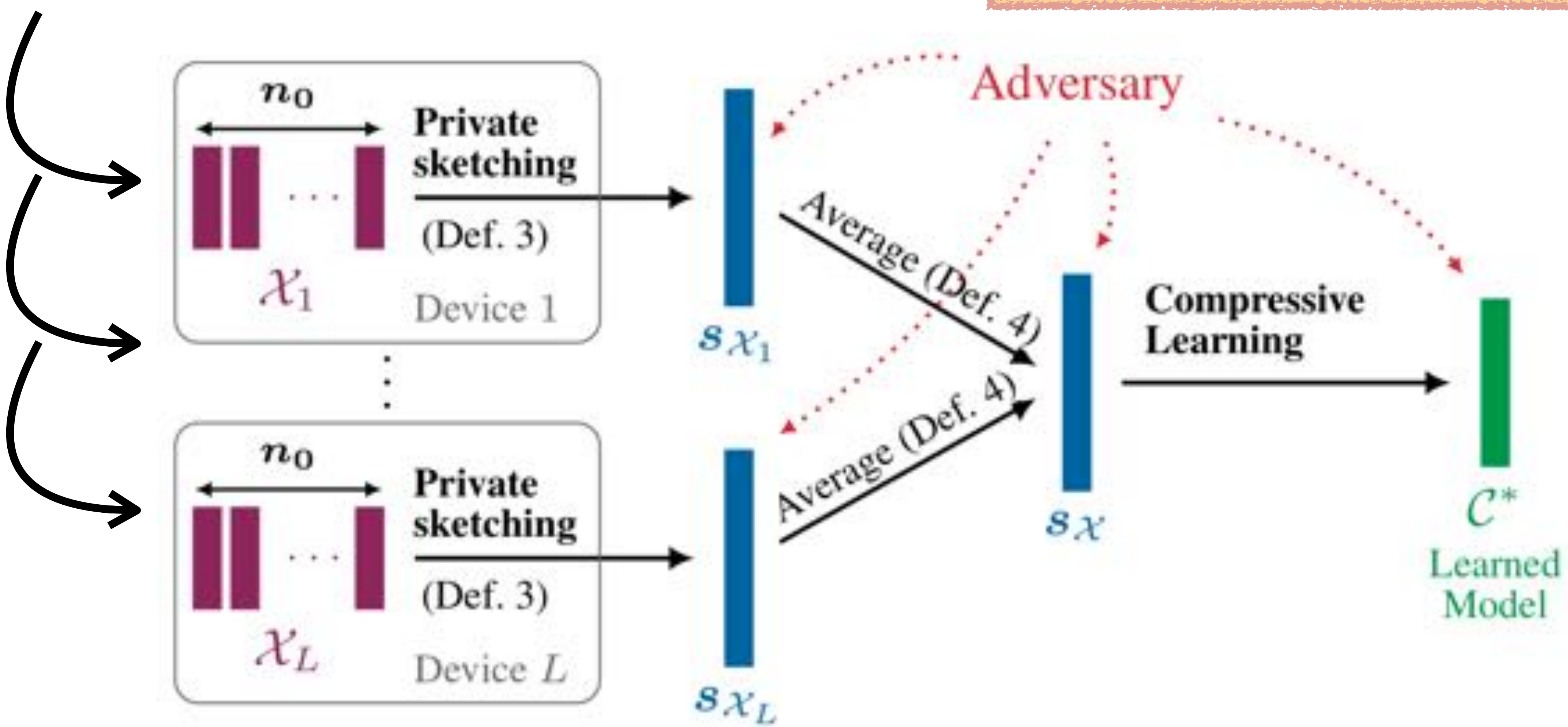
Model combining the two extreme cases:

Important remarks

1) The adversary can know the sketch operator!

2) It is randomly drawn but *fixed*,
i.e., additional noise is necessary!

Dataset is shared across L devices…



…each device holds n_0 signals… …and releases a (privacy-preserving) local sketch!

$$N = L \times n_0$$

Differentially Private Sketching

Local private sketches are obtained by Laplacian mechanism *and subsampling*

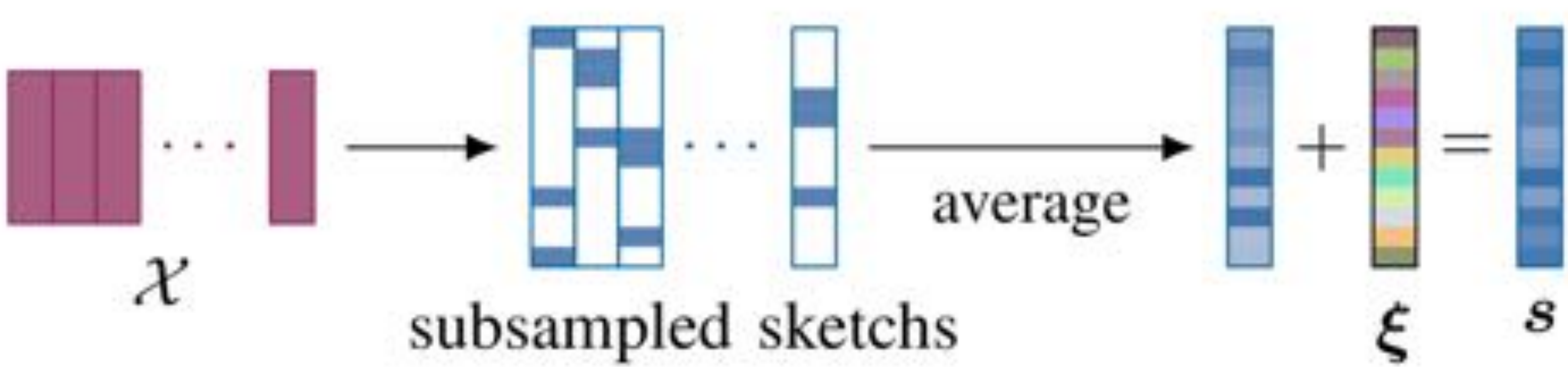
See later

Private sketch mechanism

$$\boldsymbol{s}_X := \frac{1}{N}\sum_{\boldsymbol{x}_i \in X} \left(\exp(\mathrm{i}\Omega^T \boldsymbol{x}_i) \odot \boldsymbol{b}_i\right) + \boldsymbol{\xi}$$

Subsampling: binary mask, keeps r values

$\xi_j \sim \mathrm{Lap}(\sigma_\xi/\sqrt{2})$



Differentially Private Sketching

Local private sketches are obtained by Laplacian mechanism *and subsampling*

See later

Private sketch mechanism

$$\boldsymbol{s}_X := \frac{1}{N} \sum_{\boldsymbol{x}_i \in X} (\exp(\mathrm{i}\Omega^T \boldsymbol{x}_i) \odot \boldsymbol{b}_i) + \boldsymbol{\xi}$$

Subsampling: binary mask, keeps r values

$\xi_j \sim \mathrm{Lap}(\sigma_\xi / \sqrt{2})$

Theorem: the proposed mechanism is private:

If $\sigma_\xi \propto \frac{\sqrt{rm}}{\sqrt{n_0 \epsilon}}$, then $\boldsymbol{s}_X$ provides $\epsilon-\mathrm{DP}$ to the contributors of X

Differentially Private Sketching: proof

Theorem: the proposed mechanism

$$\boldsymbol{s}_X := \frac{1}{N} \sum_{\boldsymbol{x}_i \in X} (\exp(\mathrm{i}\Omega^T \boldsymbol{x}_i) \odot \overset{\text{Keeps r values}}{\boldsymbol{b}_i}) + \boldsymbol{\xi}$$

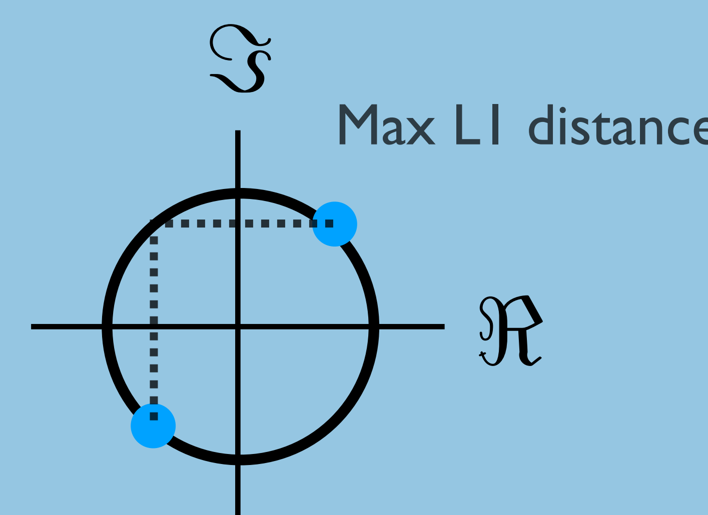
$\xi_j \sim \mathrm{Lap}(\sigma_\xi/\sqrt{2})$

where $\sigma_\xi \propto \frac{\sqrt{rm}}{\sqrt{n_0}\epsilon}$

is $\epsilon - \mathrm{DP}$

Proof idea:

$$\frac{p(\boldsymbol{s}_X)}{p(\boldsymbol{s}_{X'})} \leq \exp\left(\frac{1}{\sigma_\xi N} \underbrace{\|\boldsymbol{z}_{\boldsymbol{x}} \odot \boldsymbol{b} - \boldsymbol{z}_{\boldsymbol{x}'} \odot \boldsymbol{b}\|_1}_{r \text{ nonzero entries}}\right)$$



Remark

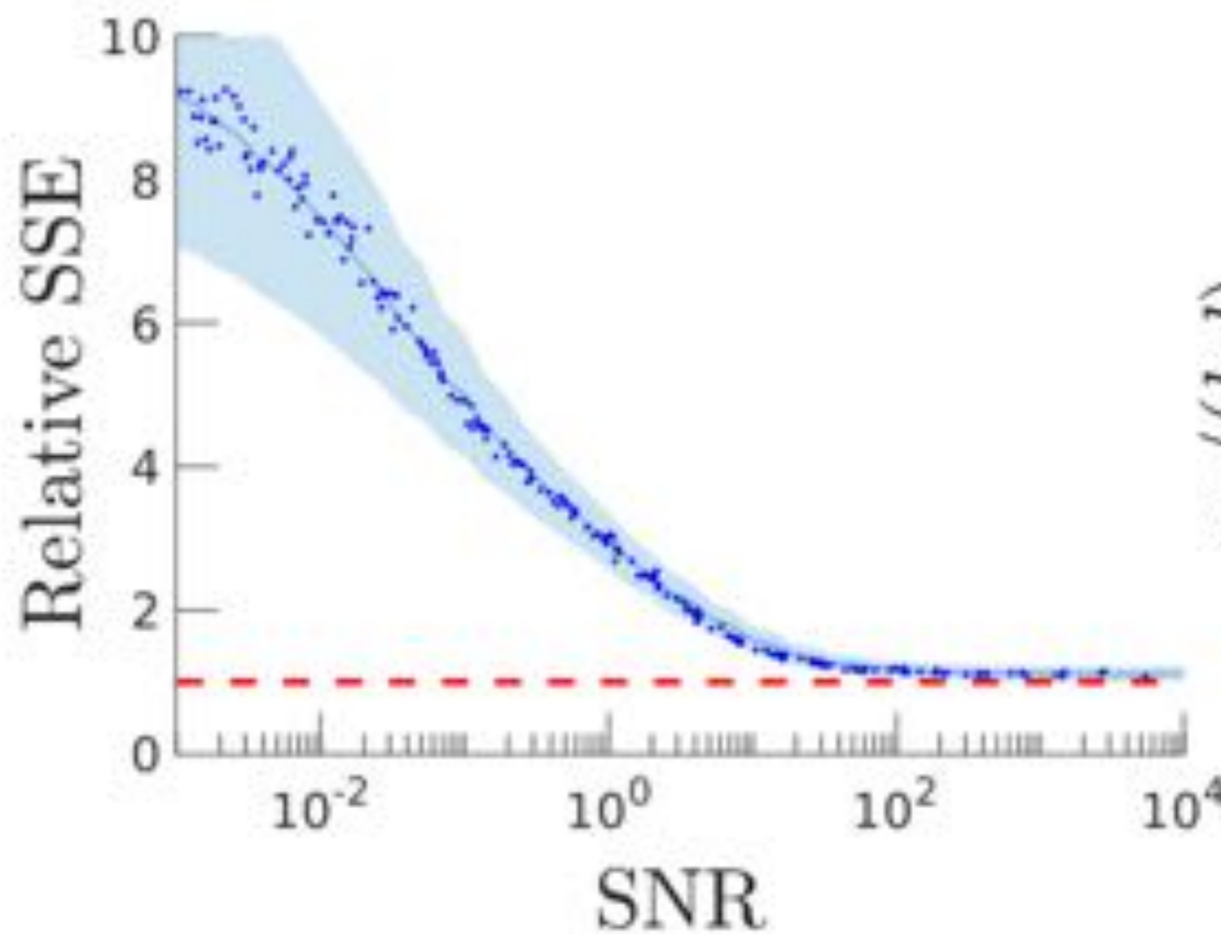
It can be argued that the bound above is sharp (without additional constraints)

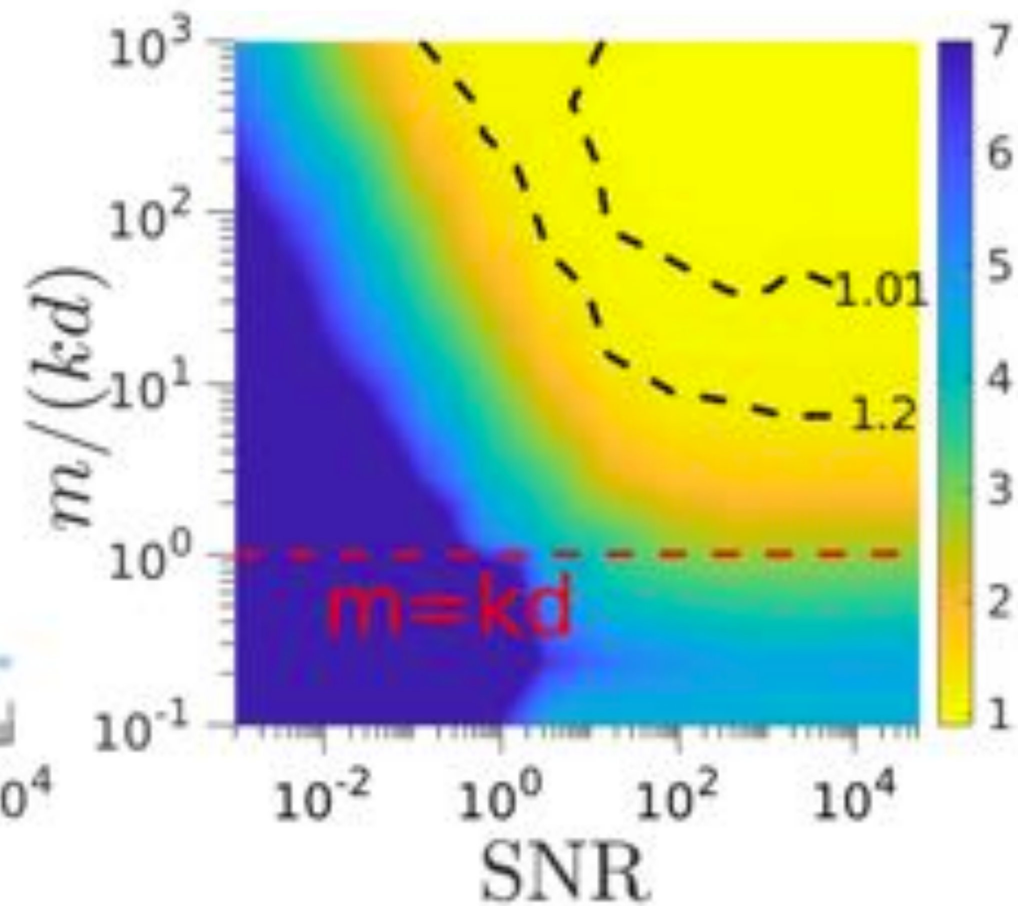
But… can we still learn?

I.e., what about “utility”??

How does the addition of noise and subsampling affect learning?

$$\mathrm{SNR} \triangleq \frac{\|\boldsymbol{z}\|^2}{\sum_{j=1}^{m} \mathrm{Var}((\boldsymbol{s}_{\mathcal{X}})_j)} = \frac{\alpha_r n_0 L \|\boldsymbol{z}\|^2}{1-\alpha_r\|\boldsymbol{z}\|^2+\sigma_\xi^2}.$$



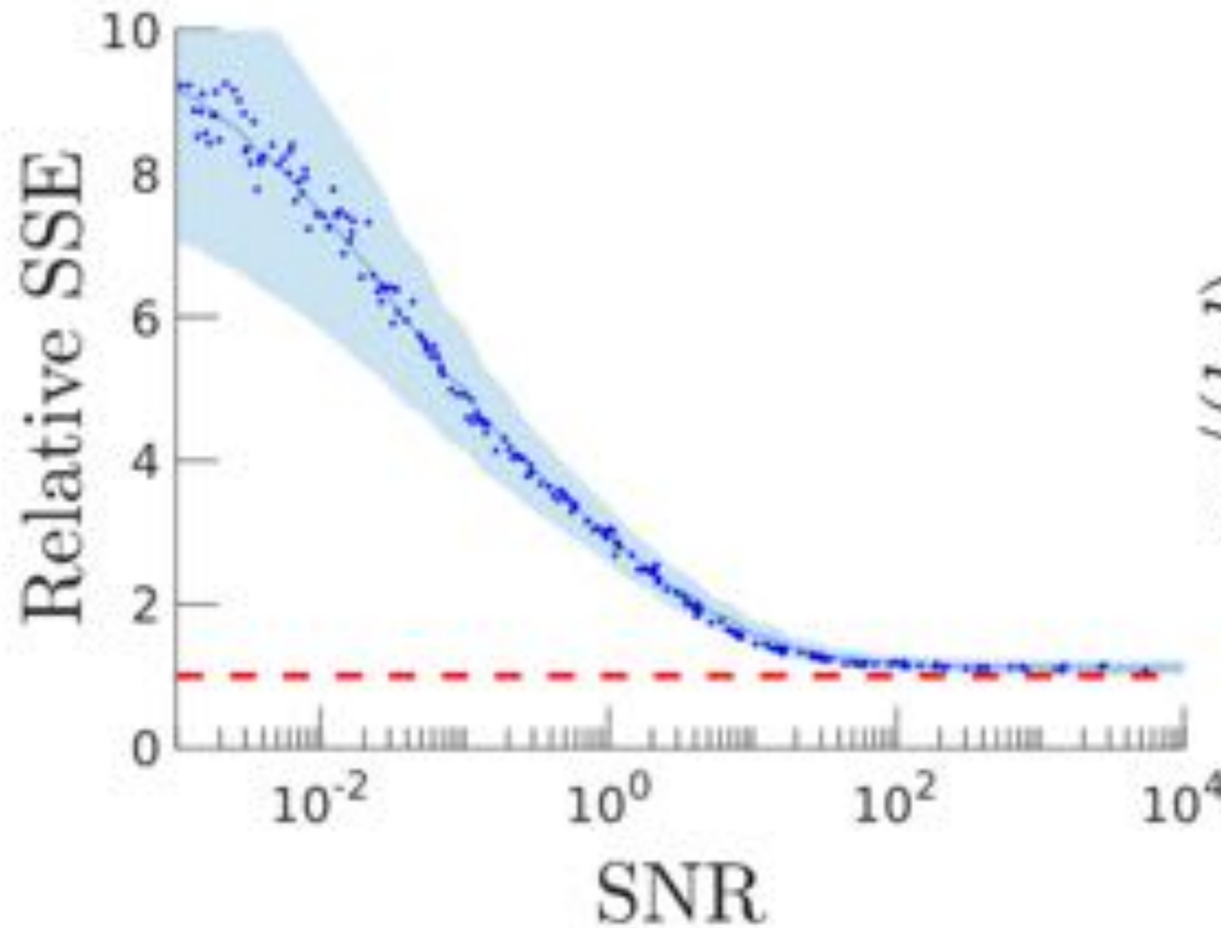


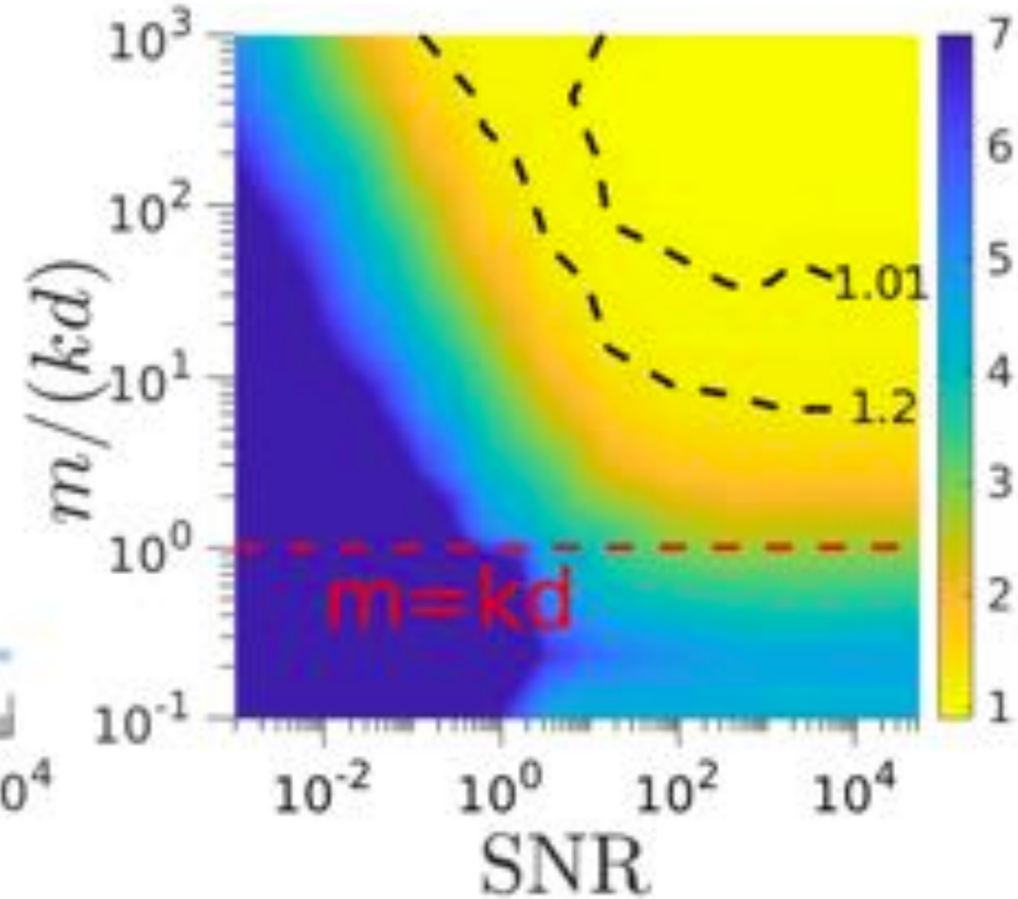
But… can we still learn?

I.e., what about "utility"??

How does the addition of noise and subsampling affect learning?

$$\mathrm{SNR} \triangleq \frac{\|\boldsymbol{z}\|^2}{\sum_{j=1}^{m} \mathrm{Var}((\boldsymbol{s}_{\mathcal{X}})_j)} = \frac{\alpha_r n_0 L \|\boldsymbol{z}\|^2}{1 - \alpha_r \|\boldsymbol{z}\|^2 + \sigma_\xi^2}.$$



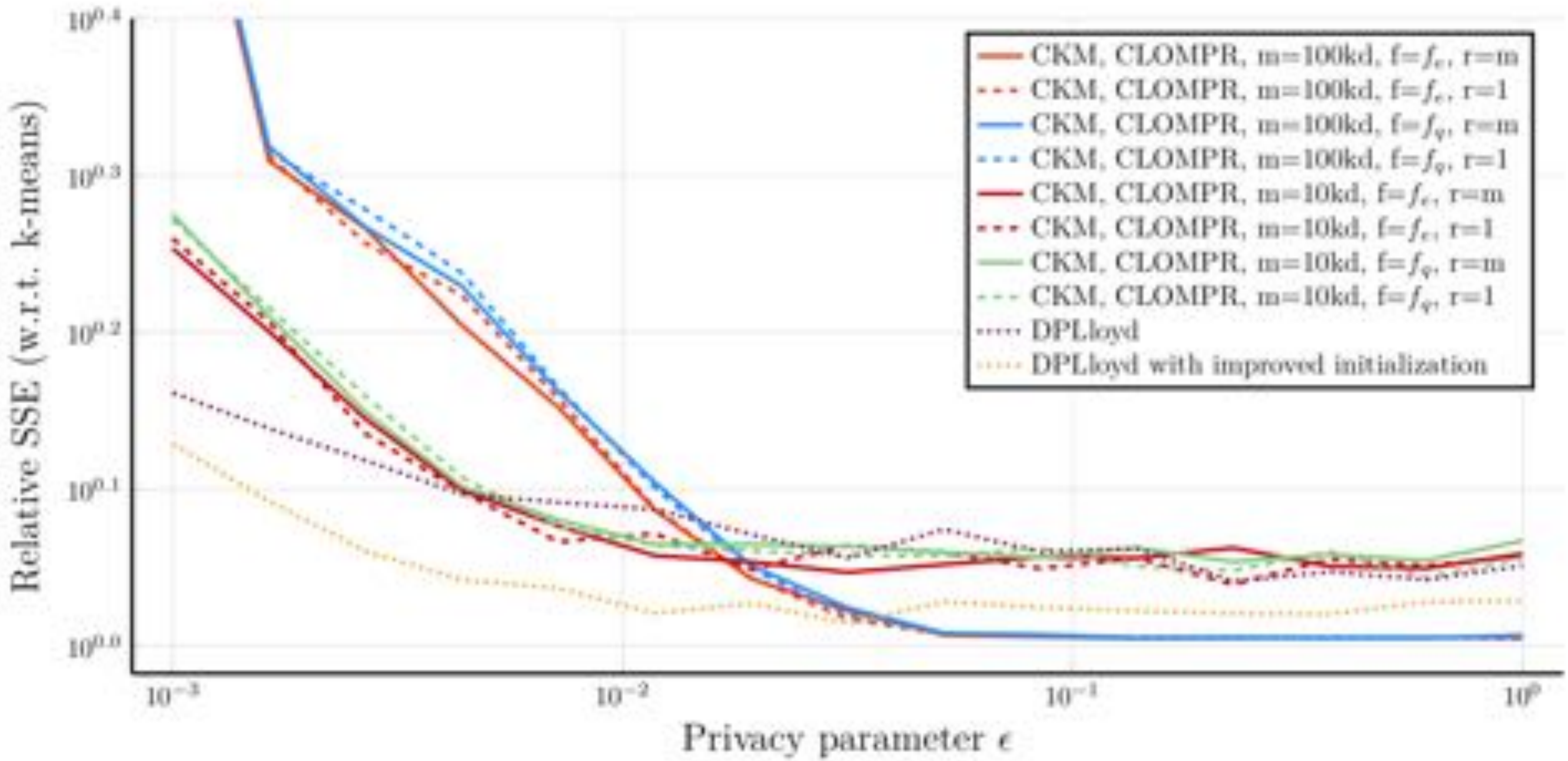


$$\mathrm{SNR}(\epsilon; n_0, L, \alpha_r, m) = \frac{\alpha_r\, n_0\, L\, \delta}{1 - \alpha_r \delta + \frac{32 \alpha_r m^2}{n_0 \epsilon^2}}$$

The SNR helps to understand the effect of the parameters

Privacy-utility tradeoff (case study)

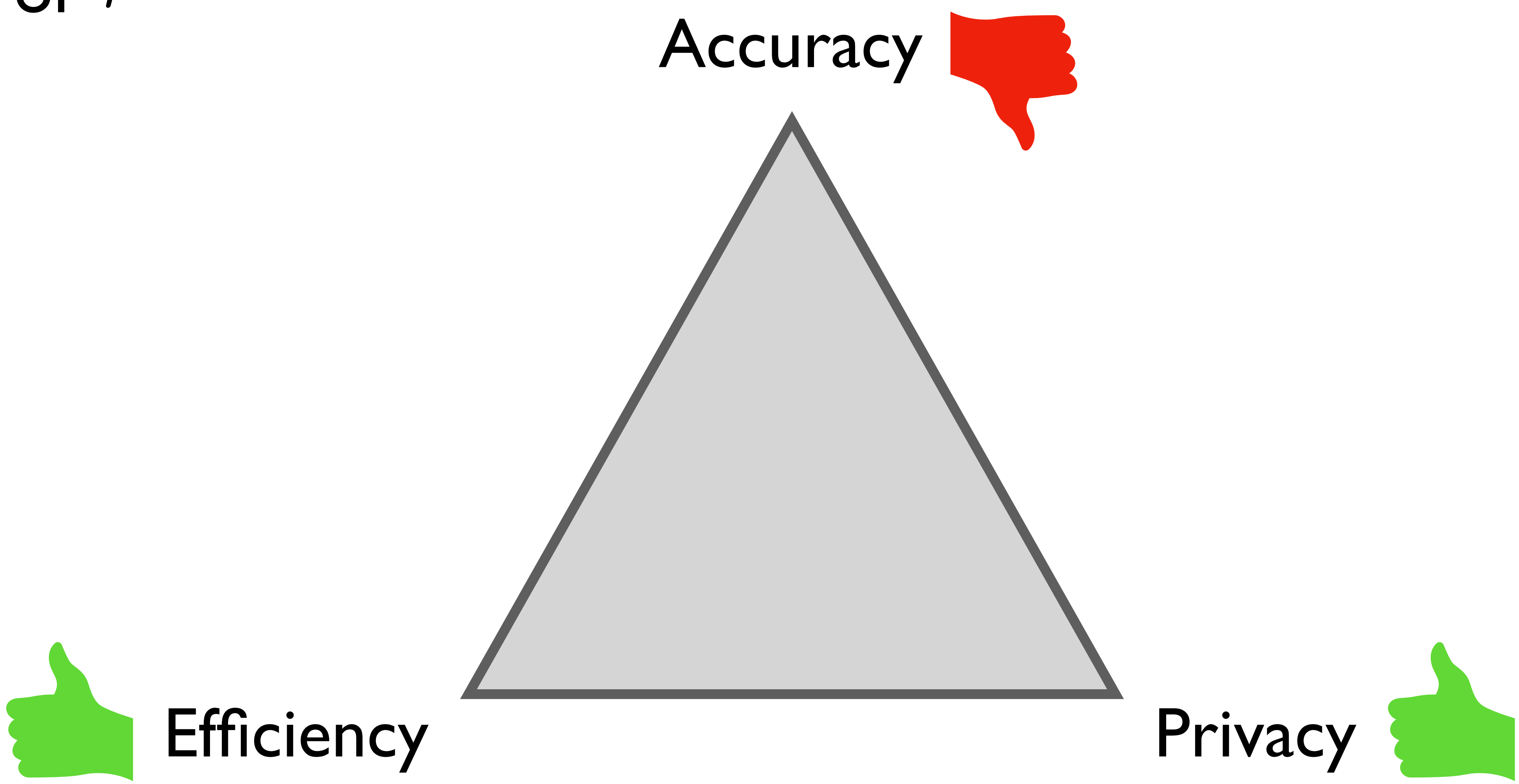
Some experimental privacy-utility curves (in a well-controlled environment)



… competitive with state-of-the-art Differentially Private K-Means :-)

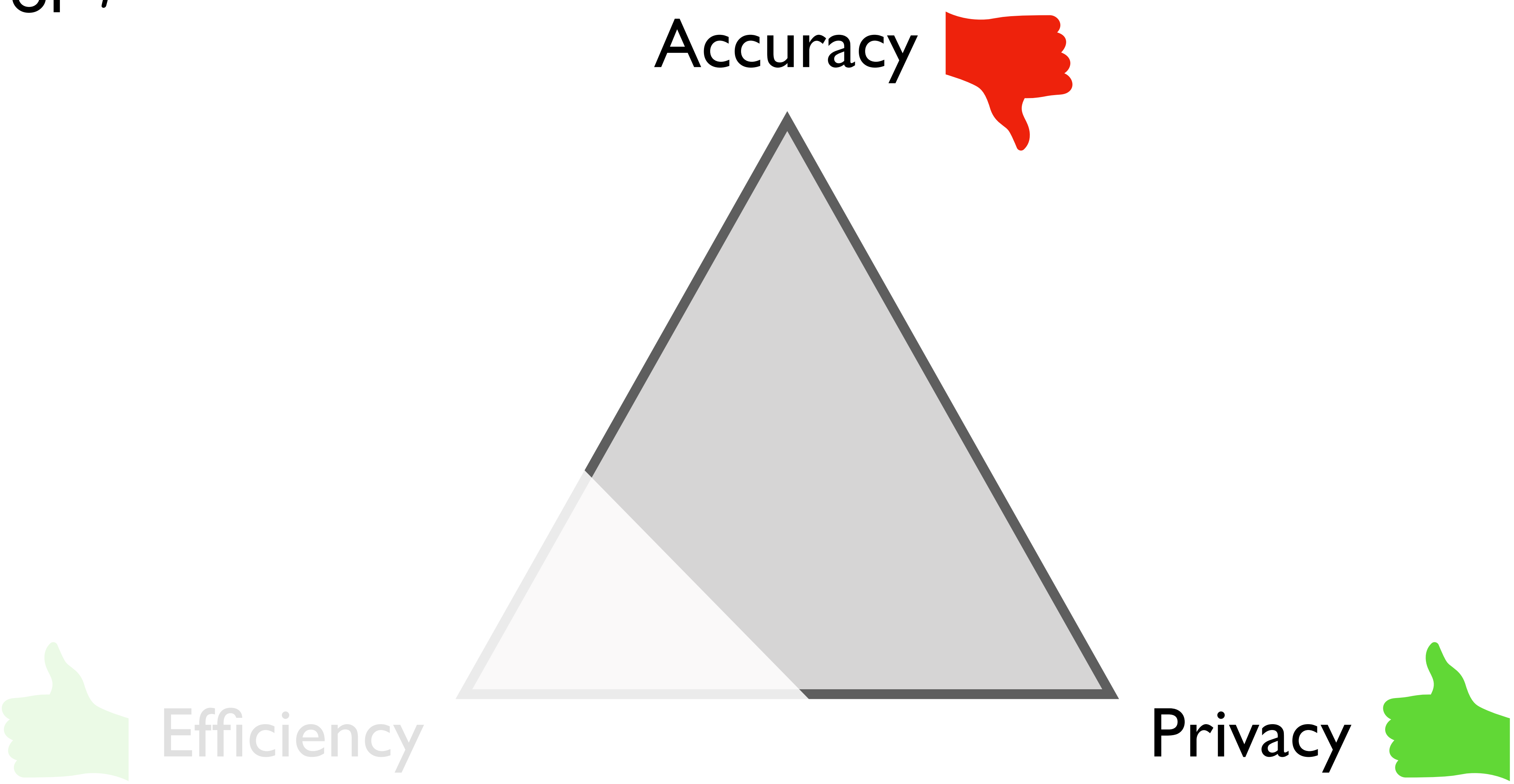
Discussion

Role of r



Discussion

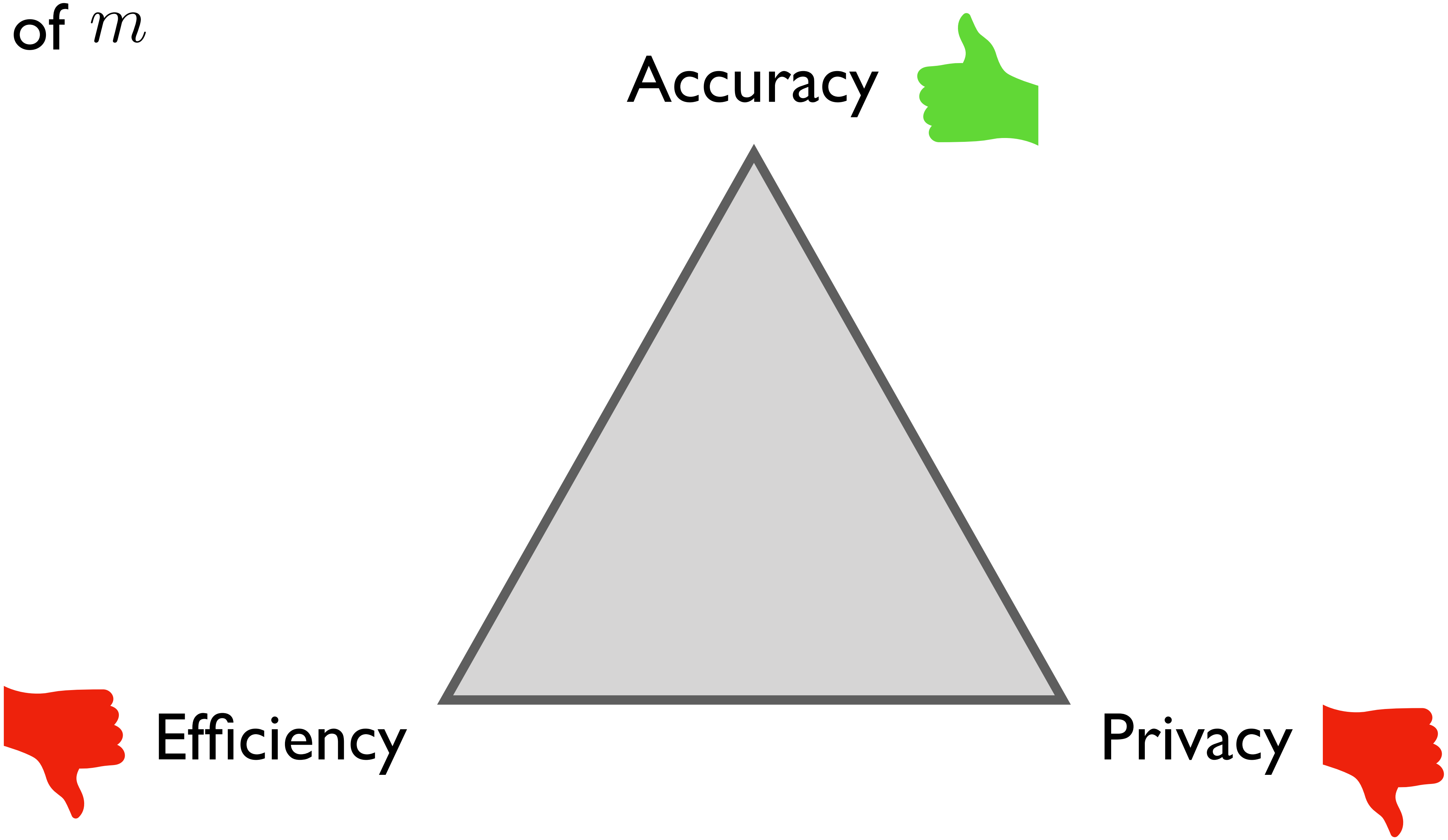
Role of r



Here r=m is the best, but has negligible impact

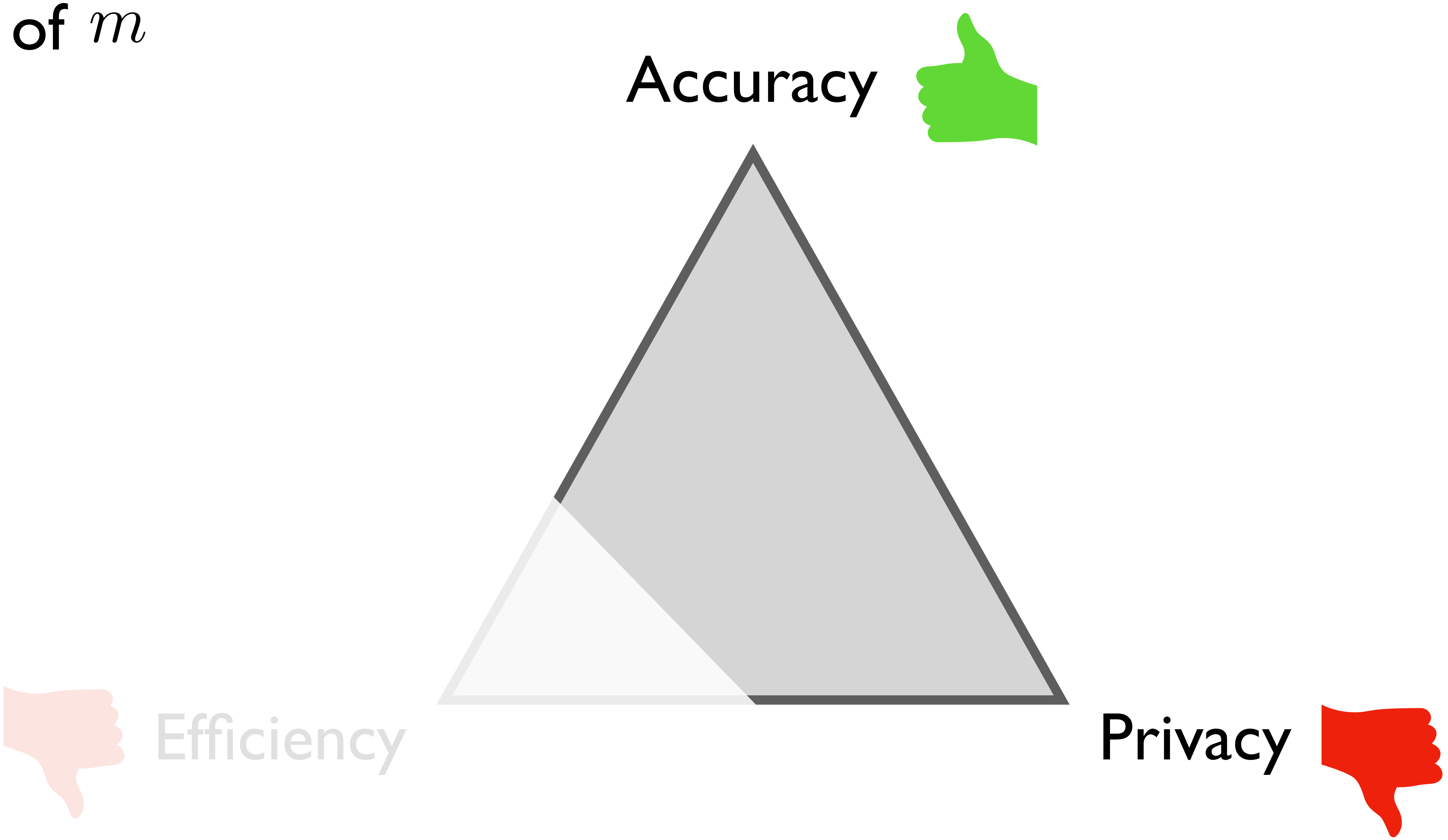
Discussion

Role of m



Discussion

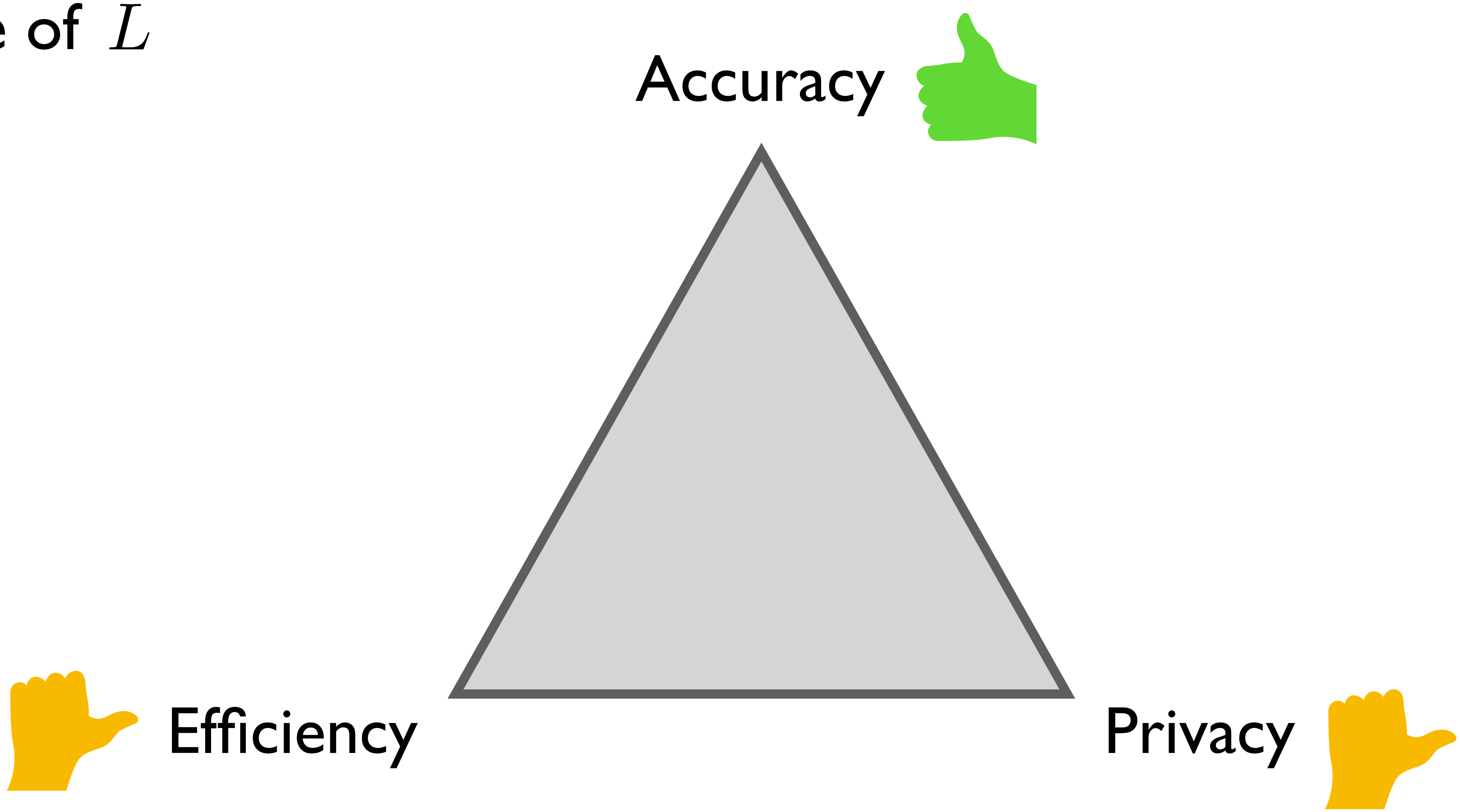
Role of m



For given epsilon, there is an optimal m

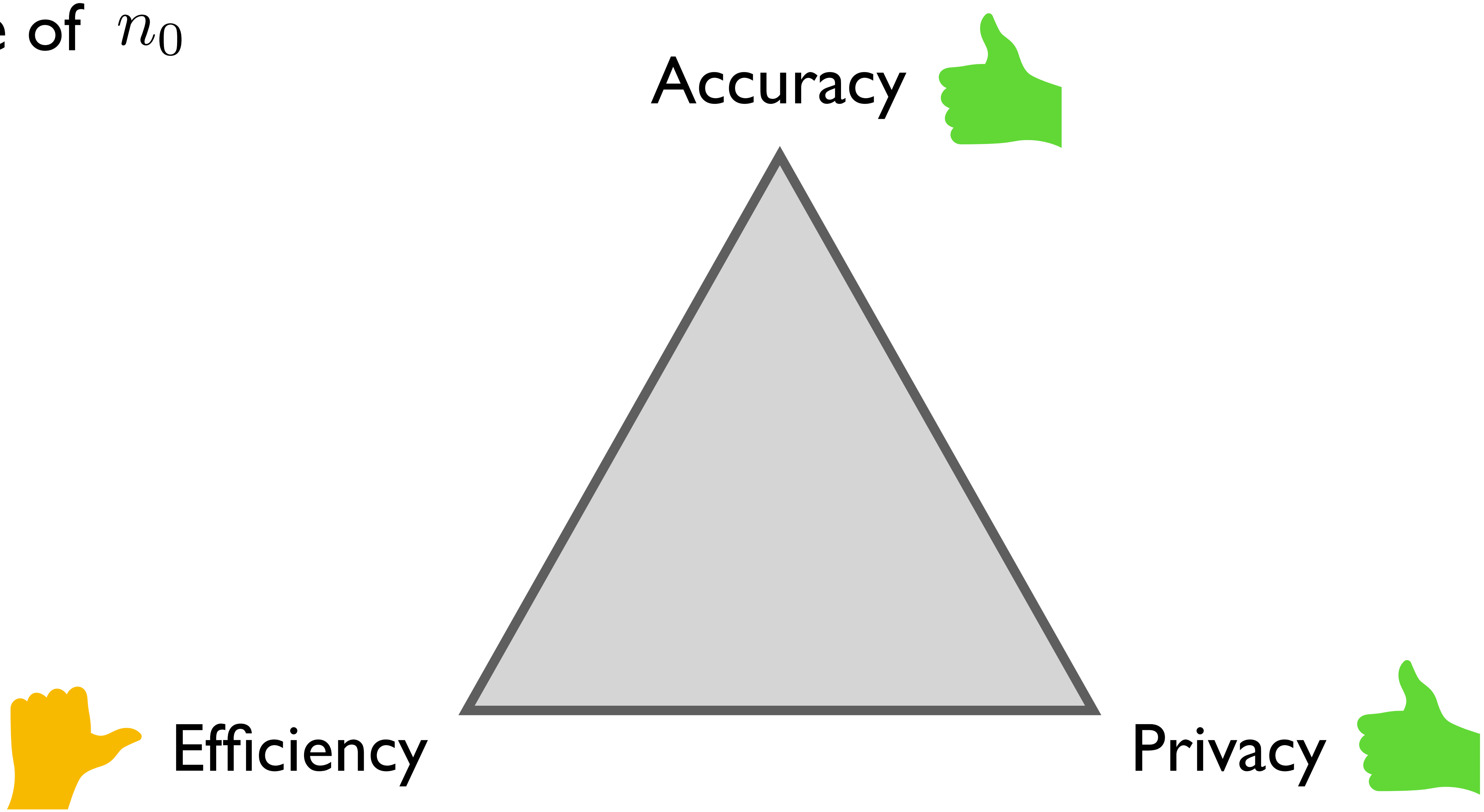
Discussion

Role of L

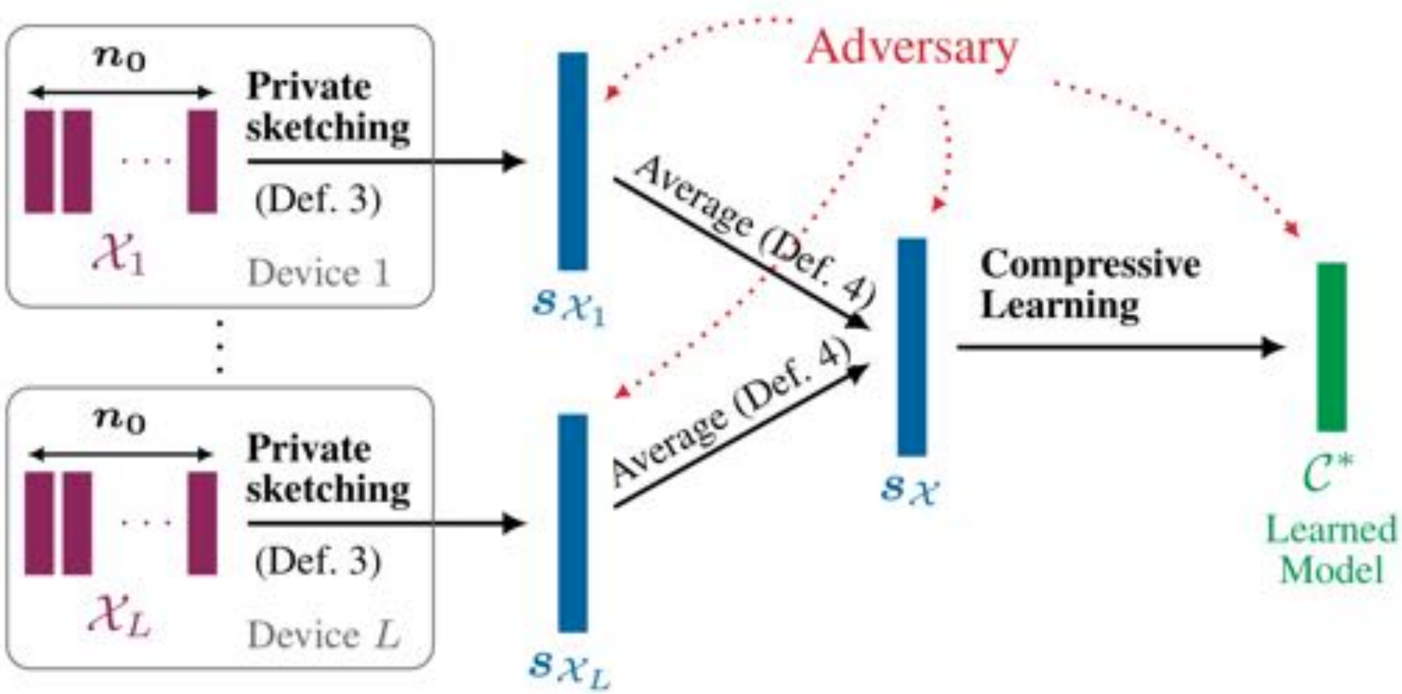


Discussion: the huge advantage

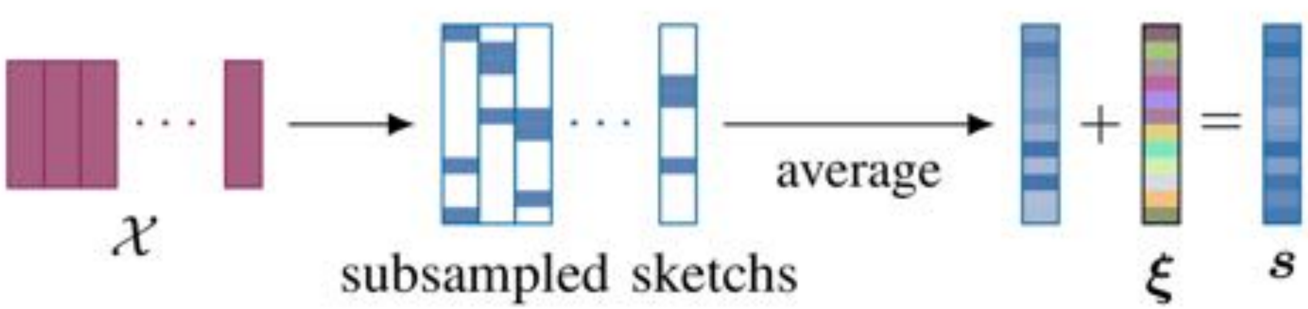
Role of n_0



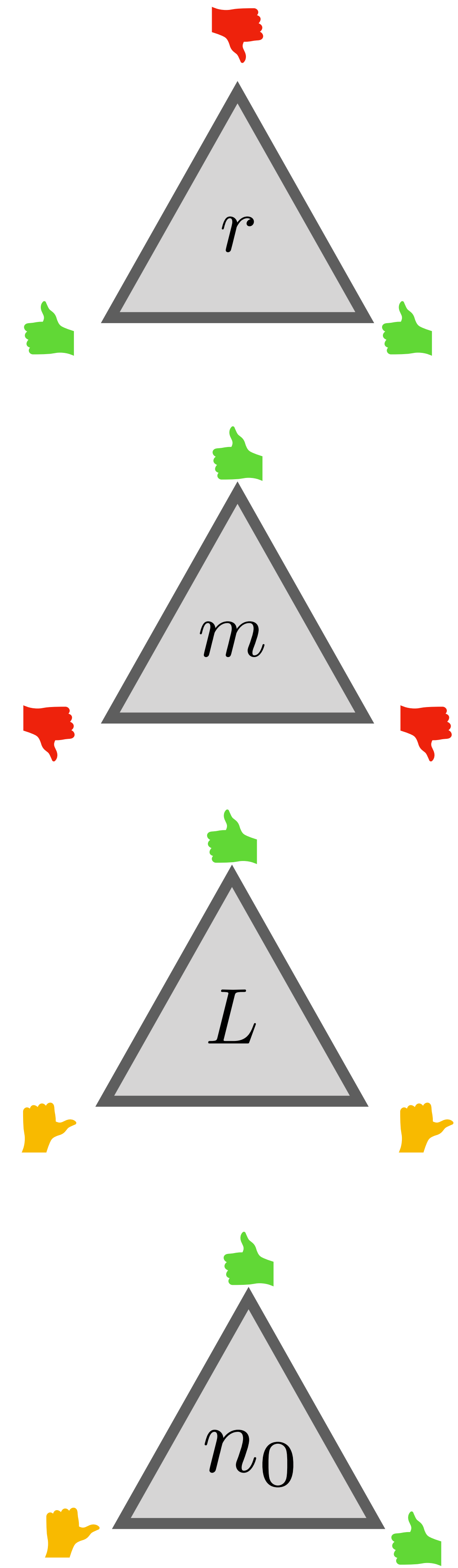
Recap'



n_0
Private sketching
(Def. 3)
$\mathcal{X}_1$
Device 1
n_0
Private sketching
(Def. 3)
$\mathcal{X}_L$
Device L
Adversary
$s_{\mathcal{X}_1}$
$s_{\mathcal{X}_L}$
Average (Def. 4)
Average (Def. 4)
$s_{\mathcal{X}}$
Compressive Learning
$\mathcal{C}^*$
Learned Model



$\mathcal{X}$
subsampled sketchs
average
$+$
$=$
ξ
s



r
m
L
n_0