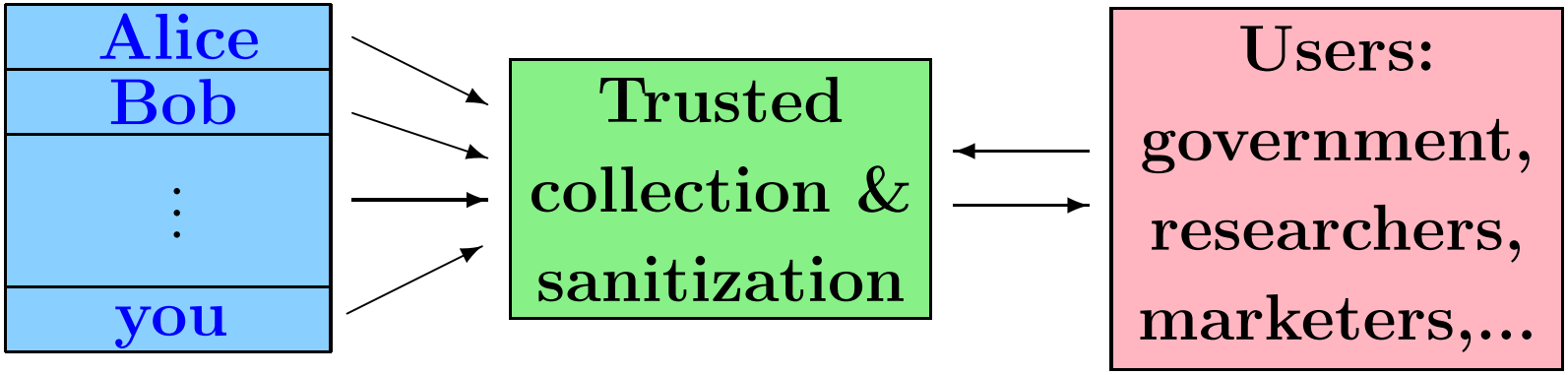
Techniques for Private Data Analysis

Sofya Raskhodnikova
Penn State University

Based on joint work with Shiva Kasiviswanathan, Homin Lee, Kobbi Nissim and Adam Smith

Private data analysis



Collections of personal and sensitive data

- census
- medical and public health data
- social networks
- recommendation systems
- trace data: search records, click data
- intrusion-detection

Meta Question

What information can be released?

- Two conflicting goals
  - utility: users can extract "global" statistics
  - privacy: individual information stays hidden

Related work

Other fields: huge amount of work

- in statistics (statistical disclosure limitation)
- in data mining (privacy-preserving data mining)
- largely: no precise privacy definition
  (only security against specific attacks)

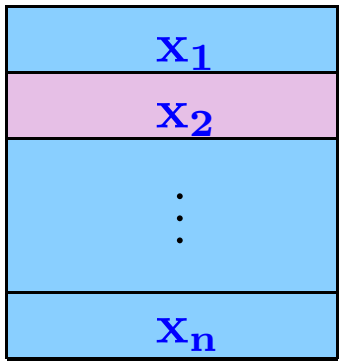
In cryptography (private data analysis)

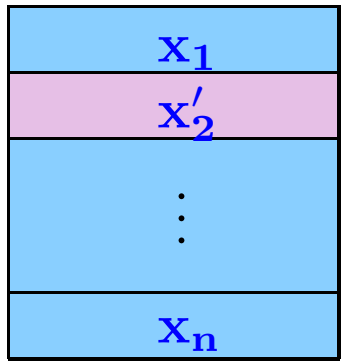
- [Dinur Nissim 03, Dwork Nissim 04,
  Chawla Dwork McSherry Smith Wee 05,
  Blum Dwork McSherry Nissim 05,
  Chawla Dwork McSherry Talwar 05,
  Dwork McSherry Nissim Smith 06, ...]
- rigorous privacy guarantees

Differential privacy [DMNS06]

Intuition: Users learn the same thing about me
whether or not I participate in the census

Two databases are *neighbors* if they differ in one row
(arbitrarily complex information supplied by one person).

$\mathbf{x} =$ 

$\mathbf{x}' =$ 

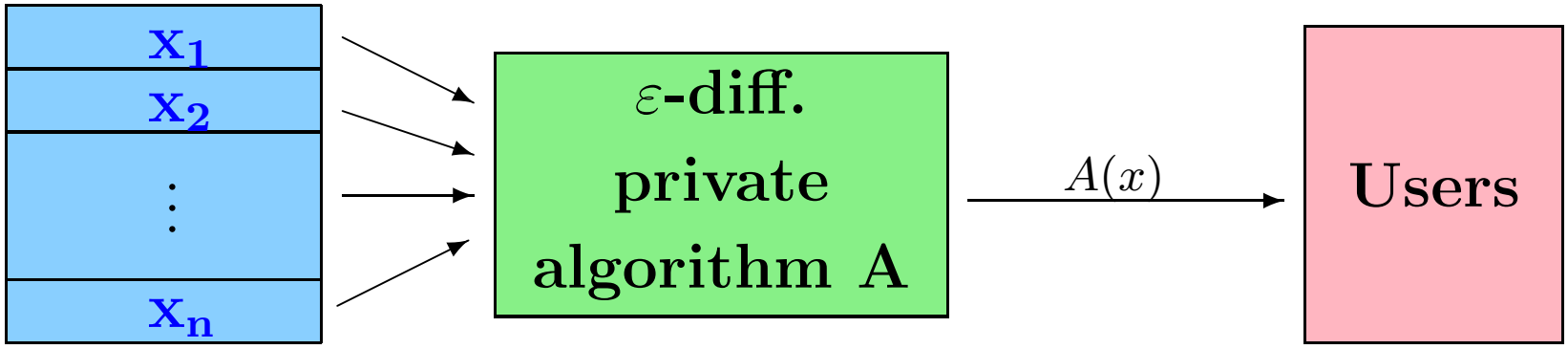
Privacy definition

Algorithm A is ε-differentially private if

- *for all neighbor databases x, x'*
- *for all sets of answers S*

$$\Pr[A(x) \in S] \leq (1+\varepsilon) \cdot \Pr[A(x') \in S]$$

Properties of differential privacy

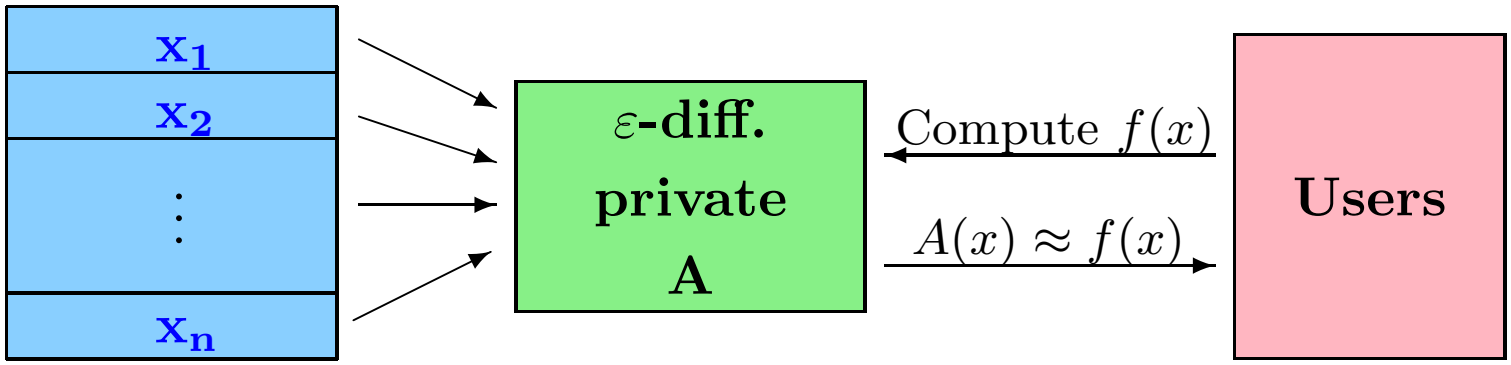


- ε is non-negligible (at least $\frac{1}{n}$).
- *Composition:* If A_1 and A_2 are ε-differentially private then (A_1, A_2) is 2ε-differentially private
- robust in the presence of arbitrary side information

What can we compute privately?

Research so far:

- Definitions [DiNi,DwNi,EGS,DMNS,DwNa,DKMMN,GKS]
- Function approximation



  - Protocols [DiNi,DwNi,BDMN,DMNS,NRS,BCDKMT]
  - Impossibility results [DiNi,DMNS,DwNa,DwMT,DwY]
  - Distributed protocols [DKMMN,BNiO]
- Mechanism design [McSherry Talwar 07]
- Learning [Blum Dwork McSherry Nissim 05, KLNRS08]
- Releasing classes of functions [Blum Ligett Roth 08]
- Synthetic data [Machanavajjhala Kifer Abowd Gehrke Vilhuber 08]

Road map

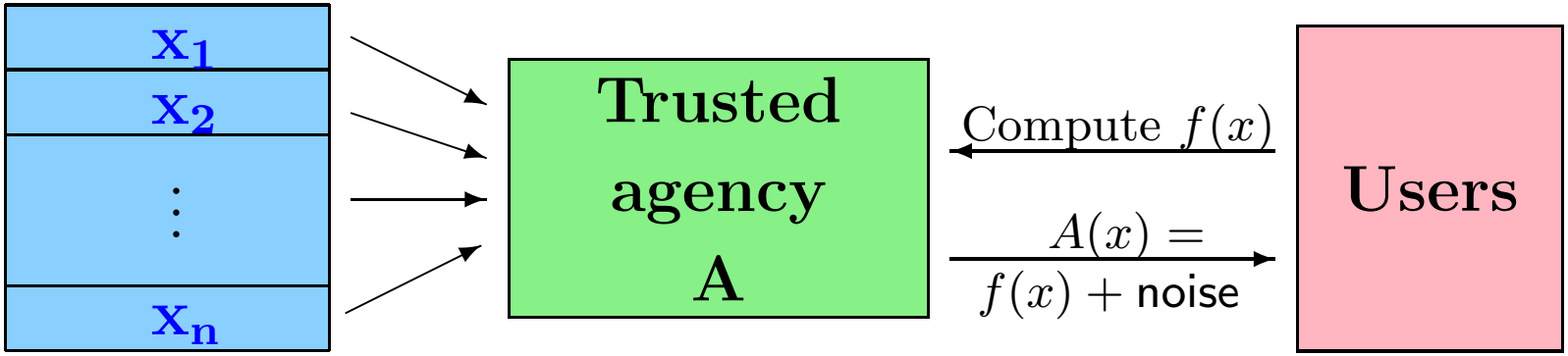
I. Function approximation

- Global sensitivity framework [DMNS06]
- Smooth sensitivity framework [NRS07]
- Sample-and-aggregate [NRS07]

II. Learning

- Exponential mechanism [MT07,KLNRS08]

Function Approximation



For which functions f can we have:

- privacy: differential privacy [DMNS06]
- utility: output $A(x)$ is close to $f(x)$

Global sensitivity framework [DMNS06]

Intuition: f can be released accurately when it is insensitive to individual entries $x_1, \ldots, x_n$.

Global sensitivity $\mathsf{GS}_f = \max_{\text{neighbors } x,x'} \|f(x) - f(x')\|_1$.

Example: $\mathsf{GS}_{\text{average}} = \frac{1}{n}$ if $x \in [0,1]^n$.

Theorem

If $A(x) = f(x) + \mathsf{Lap}\left(\frac{\mathsf{GS}_f}{\varepsilon}\right)$ *then* A *is* ε*-diff. private.*

Global sensitivity framework [DMNS06]

Intuition: f can be released accurately when it is insensitive to individual entries $x_1, \ldots, x_n$.

Global sensitivity $\mathsf{GS}_f = \max_{\text{neighbors } x,x'} \|f(x) - f(x')\|_1$.

Example: $\mathsf{GS}_{\text{average}} = \frac{1}{n}$ if $x \in [0,1]^n$. Noise $= \mathsf{Lap}\left(\frac{1}{\varepsilon n}\right)$.

Compare to: Estimating frequencies (e.g., proportion of people with blue eyes) from n samples: sampling error $\frac{1}{\sqrt{n}}$.

Theorem

If $A(x) = f(x) + \mathsf{Lap}\left(\frac{\mathsf{GS}_f}{\varepsilon}\right)$ *then* A *is* ε*-diff. private.*

Global sensitivity framework [DMNS06]

Intuition: f can be released accurately when it is insensitive to individual entries $x_1, \ldots, x_n$.

Global sensitivity $\mathsf{GS}_f = \max_{\text{neighbors } x,x'} \|f(x) - f(x')\|_1$.

Example: $\mathsf{GS}_{\text{average}} = \frac{1}{n}$ if $x \in [0,1]^n$. Noise $= \mathsf{Lap}\left(\frac{1}{\varepsilon n}\right)$.

Compare to: Estimating frequencies (e.g., proportion of people with blue eyes) from n samples: sampling error $\frac{1}{\sqrt{n}}$.

Theorem

If $A(x) = f(x) + \mathsf{Lap}\left(\frac{\mathsf{GS}_f}{\varepsilon}\right)$ *then* A *is* ε*-diff. private.*

Functions with low global sensitivity

- *Means, variances* for data in a bounded interval
- *histograms, contingency tables*
- *singular value decomposition*

Instance-Based Noise

Big picture for global sensitivity framework:

- add enough noise to cover the worst case for f
- noise distribution depends only on f, not database x

Problem: for some functions that's too much noise

Smooth sensitivity framework [Nissim Smith Raskhodnikova 07]:

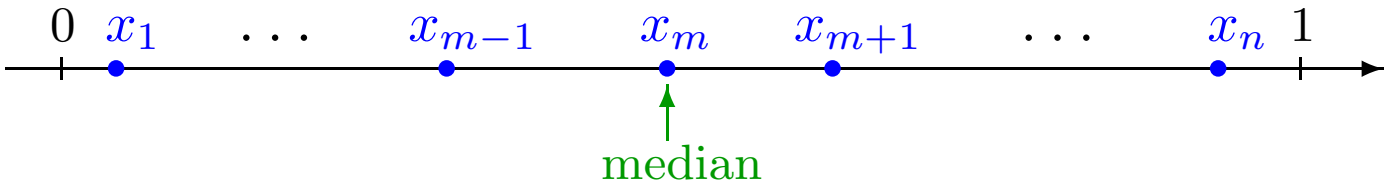
noise tuned to database x

Local sensitivity

Local sensitivity $\mathsf{LS}_f(x) = \max_{x' \text{: neighbor of } x} \|f(x) - f(x')\|$

Reminder: $\mathsf{GS}_f = \max_x \mathsf{LS}_f(x)$

Example: median for $0 \le x_1 \le \cdots \le x_n \le 1$, odd n



$$\mathsf{LS}_{\text{median}}(x) = \max(x_m - x_{m-1}, x_{m+1} - x_m)$$

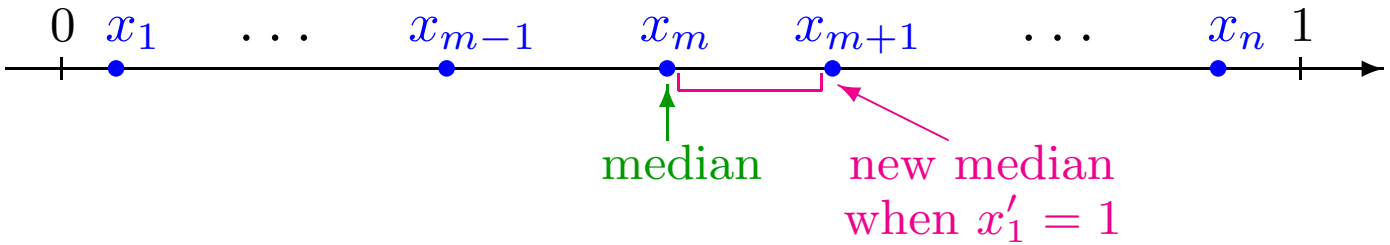
Goal: Release $f(x)$ with less noise when $\mathsf{LS}_f(x)$ is lower.

Local sensitivity

Local sensitivity $\mathsf{LS}_f(x) = \max_{x' \text{: neighbor of } x} \|f(x) - f(x')\|$

Reminder: $\mathsf{GS}_f = \max_x \mathsf{LS}_f(x)$

Example: median for $0 \le x_1 \le \cdots \le x_n \le 1$, odd n



$$\mathsf{LS}_{\text{median}}(x) = \max(x_m - x_{m-1}, x_{m+1} - x_m)$$

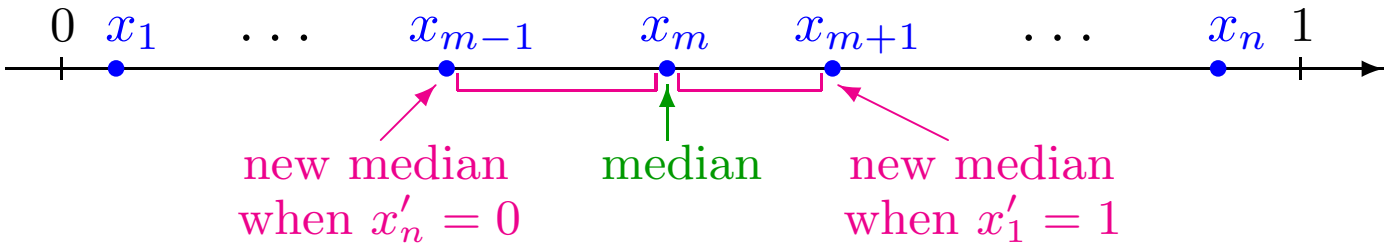
Goal: Release $f(x)$ with less noise when $\mathsf{LS}_f(x)$ is lower.

Local sensitivity

Local sensitivity $\mathsf{LS}_f(x) = \max_{x' \text{: neighbor of } x} \|f(x) - f(x')\|$

Reminder: $\mathsf{GS}_f = \max_x \mathsf{LS}_f(x)$

Example: median for $0 \le x_1 \le \cdots \le x_n \le 1$, odd n



$$\mathsf{LS}_{\text{median}}(x) = \max(x_m - x_{m-1}, x_{m+1} - x_m)$$

Goal: Release $f(x)$ with less noise when $\mathsf{LS}_f(x)$ is lower.

Instance-based noise: first attempt

Noise magnitude proportional to $\mathsf{LS}_f(x)$ instead of GS_f?

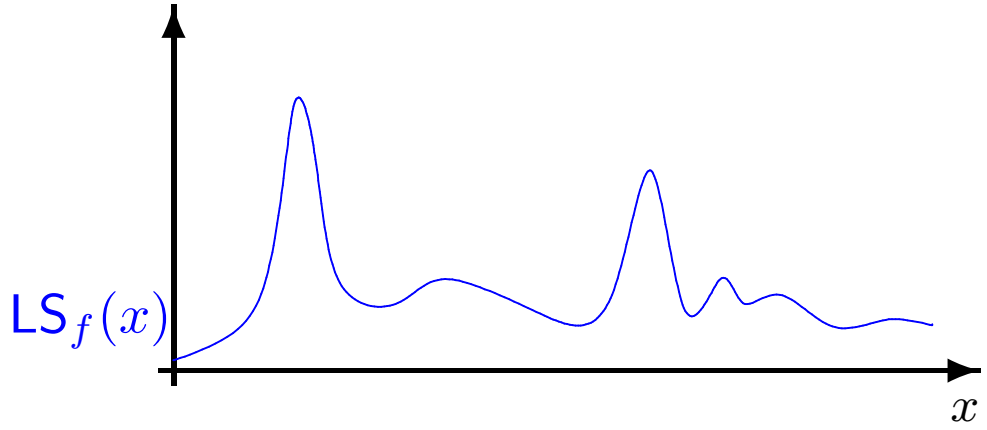
No! Noise magnitude reveals information.

Lesson: Noise magnitude must be an insensitive function.

Smooth bounds on local sensitivity

Design sensitivity function $S(x)$

- $S(x)$ is an ε-smooth upper bound on $\mathsf{LS}_f(x)$ if:
  - for all x: $S(x) \geq \mathsf{LS}_f(x)$
  - for all neighbors x, x' : $S(x) \leq e^{\varepsilon} S(x')$



Theorem

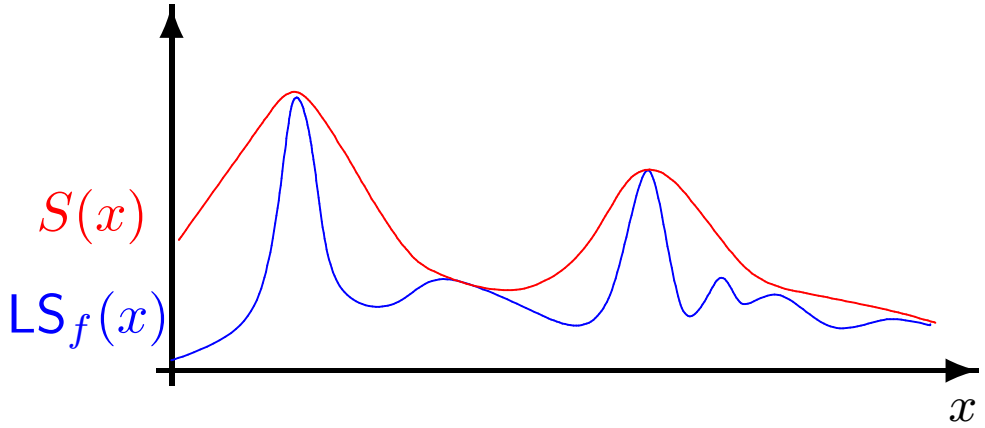
If $A(x) = f(x) + \mathsf{noise}\left(\dfrac{S(x)}{\varepsilon}\right)$ *then* A *is* ε'*-differentially private.*

Example: GS_f is always a smooth bound on $\mathsf{LS}_f(x)$

Smooth bounds on local sensitivity

Design sensitivity function $S(x)$

- $S(x)$ is an ε-smooth upper bound on $\mathsf{LS}_f(x)$ if:
  - for all x: $S(x) \geq \mathsf{LS}_f(x)$
  - for all neighbors x, x' : $S(x) \leq e^{\varepsilon} S(x')$



Theorem

If $A(x) = f(x) + \mathsf{noise}\left(\dfrac{S(x)}{\varepsilon}\right)$ *then* A *is* ε'*-differentially private.*

Example: GS_f is always a smooth bound on $\mathsf{LS}_f(x)$

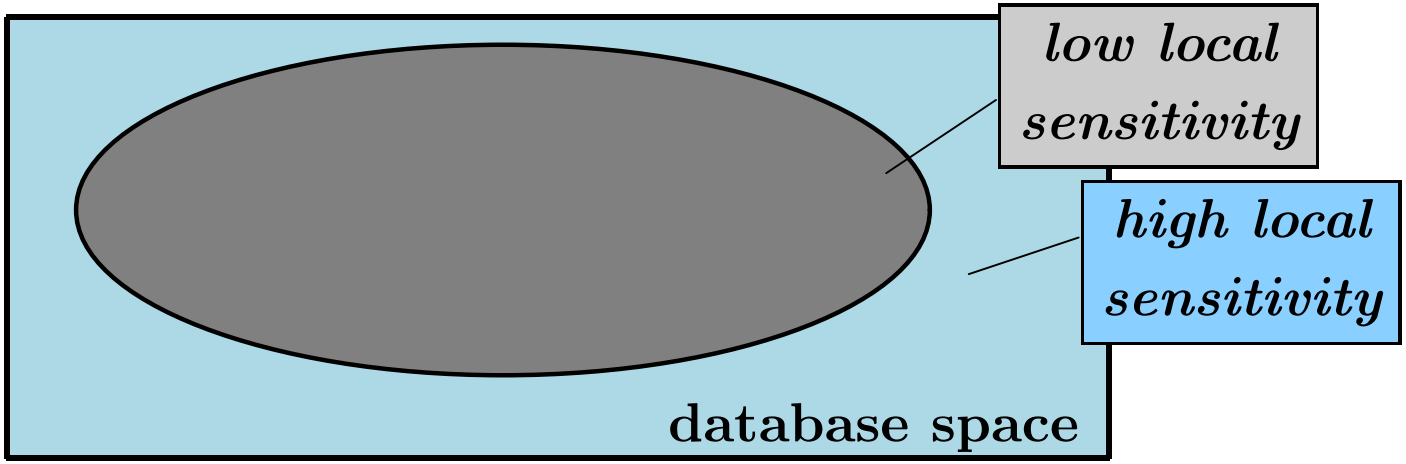
Smooth Sensitivity

Smooth sensitivity $\mathsf{S}^*_f(x) = \max_y \left(\mathsf{LS}_f(y) e^{-\varepsilon \cdot \mathsf{dist}(x,y)}\right)$

Lemma

For every ε-smooth bound S: $\mathsf{S}^*_f(x) \leq S(x)$ *for all x.*

Intuition: little noise when **far** from sensitive instances



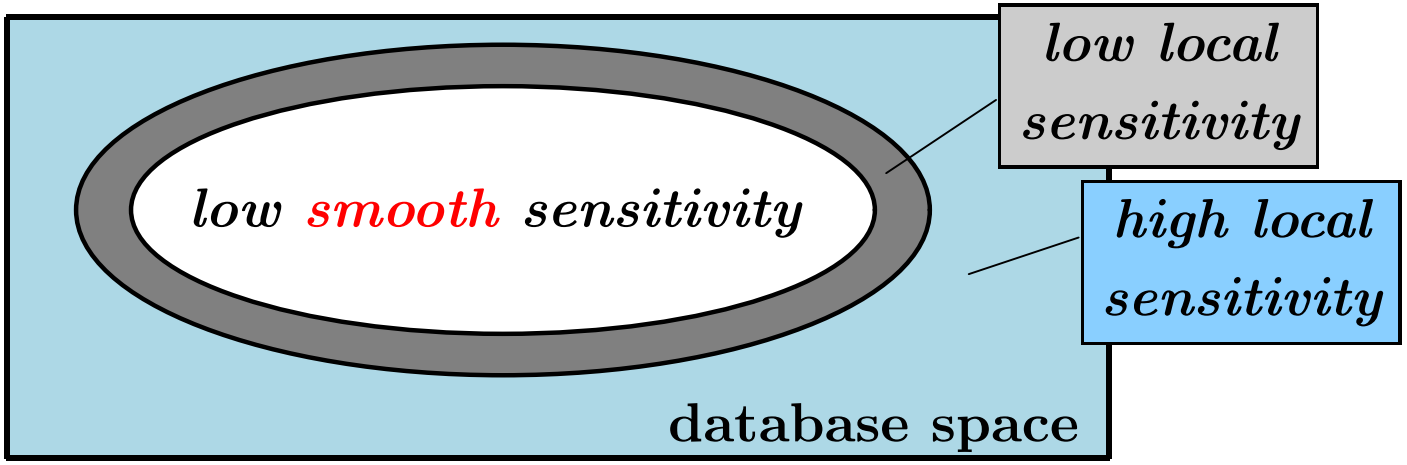
Smooth Sensitivity

Smooth sensitivity $\mathsf{S}^*_f(x) = \max_y \left(\mathsf{LS}_f(y) e^{-\varepsilon \cdot \mathsf{dist}(x,y)}\right)$

Lemma

For every ε-smooth bound S: $\mathsf{S}^*_f(x) \leq S(x)$ *for all x.*

Intuition: little noise when **far** from sensitive instances



Computing smooth sensitivity

Example functions with computable smooth sensitivity

- *Median* & *minimum* of numbers in a bounded interval
- *MST cost* when weights are bounded
- *Number of triangles* in a graph

Approximating smooth sensitivity

- only smooth upper bounds on LS are meaningful
- simple generic methods for smooth approximations
  - work for *median* and *1-median in* L_1^d

Road map

I. Function approximation

- Global sensitivity framework [DMNS06]
- Smooth sensitivity framework [NRS07]
- Sample-and-aggregate [NRS07]

II. Learning

- Exponential mechanism [MT07,KLNRS08]

New goal

- Smooth sensitivity framework requires understanding combinatorial structure of f

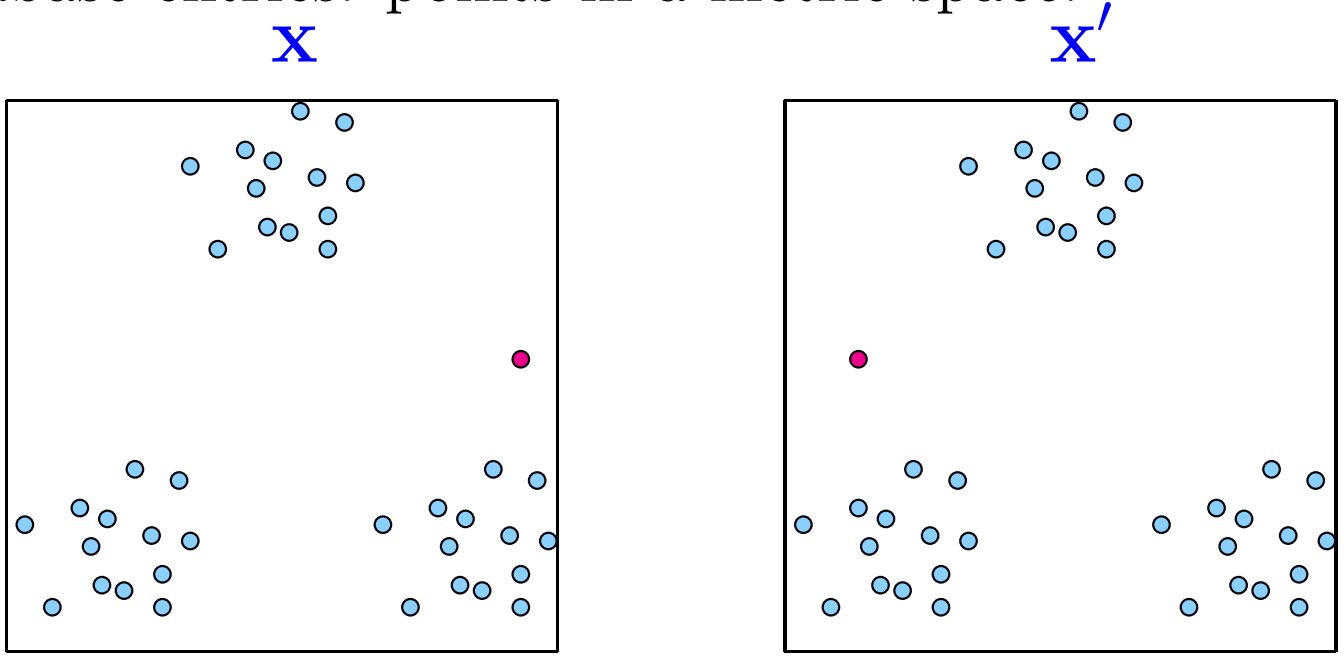
  – hard in general

- Goal: an automatable transformation from an arbitrary f into an ε-diff. private A

  – $A(x) \approx f(x)$ for "good" instances x

Example: cluster centers

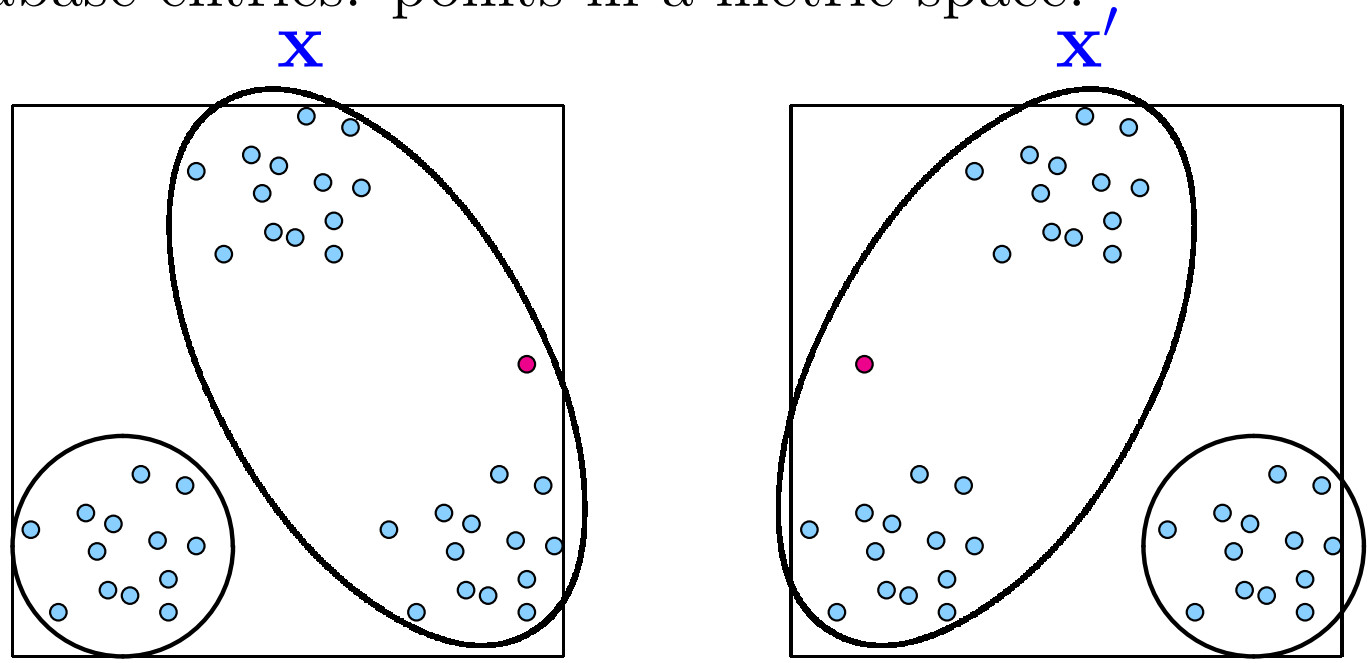
Database entries: points in a metric space.



- Comparing sets of centers: Earthmover-like metric
- Global sensitivity of cluster centers is roughly the diameter of the space. But intuitively, if clustering is "good", cluster centers should be insensitive.
- No efficient approximation for smooth sensitivity

Example: cluster centers

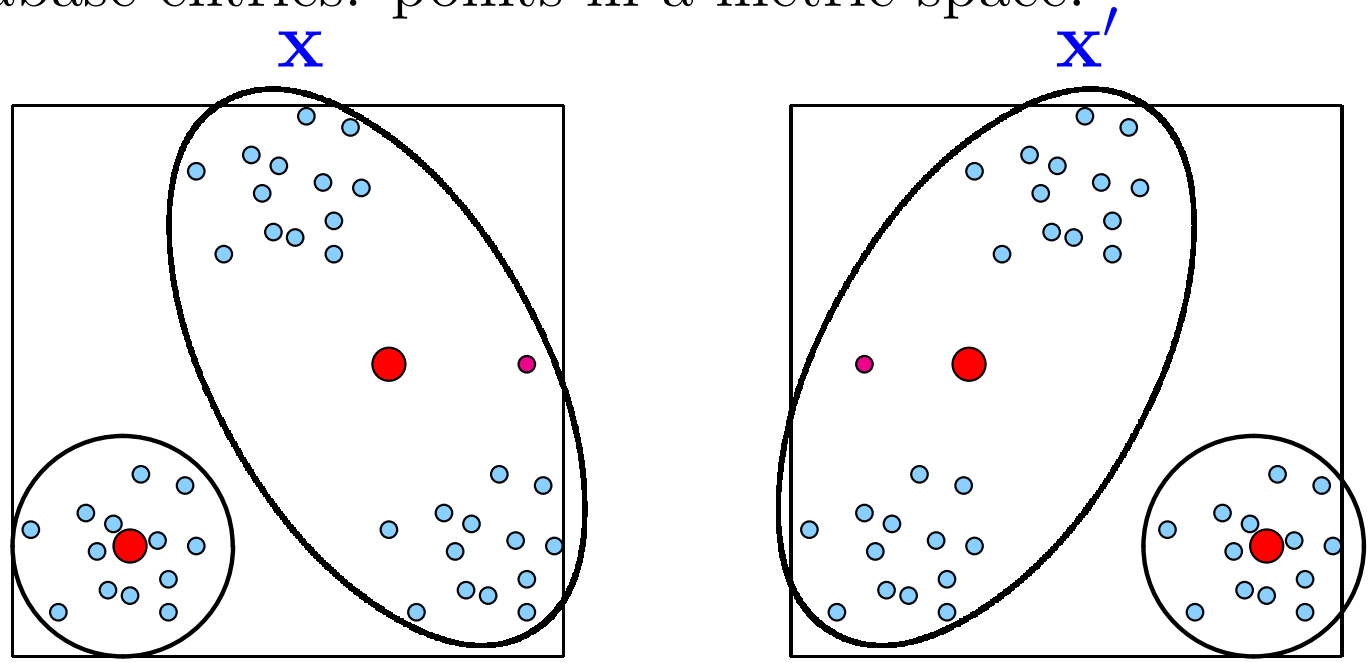
Database entries: points in a metric space.



- Comparing sets of centers: Earthmover-like metric
- Global sensitivity of cluster centers is roughly the diameter of the space. But intuitively, if clustering is "good", cluster centers should be insensitive.
- No efficient approximation for smooth sensitivity

Example: cluster centers

Database entries: points in a metric space.

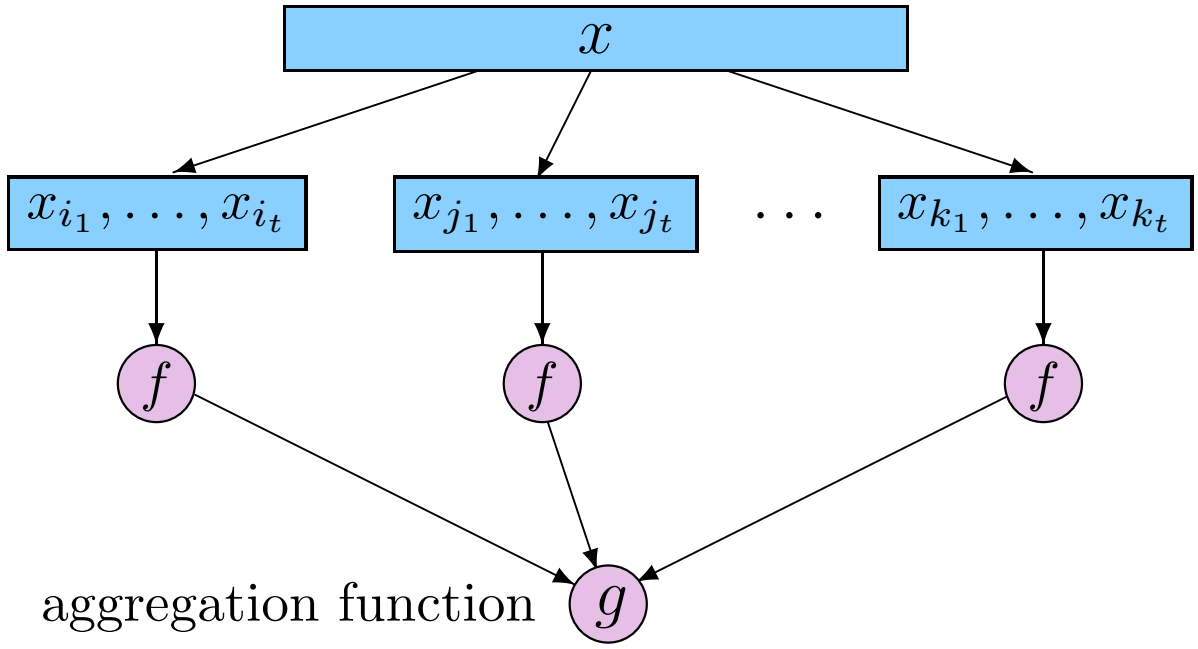


- Comparing sets of centers: Earthmover-like metric
- Global sensitivity of cluster centers is roughly the diameter of the space. But intuitively, if clustering is "good", cluster centers should be insensitive.
- No efficient approximation for smooth sensitivity

Sample-and-aggregate framework

Intuition: Replace f with a less sensitive function $\tilde{f}$.

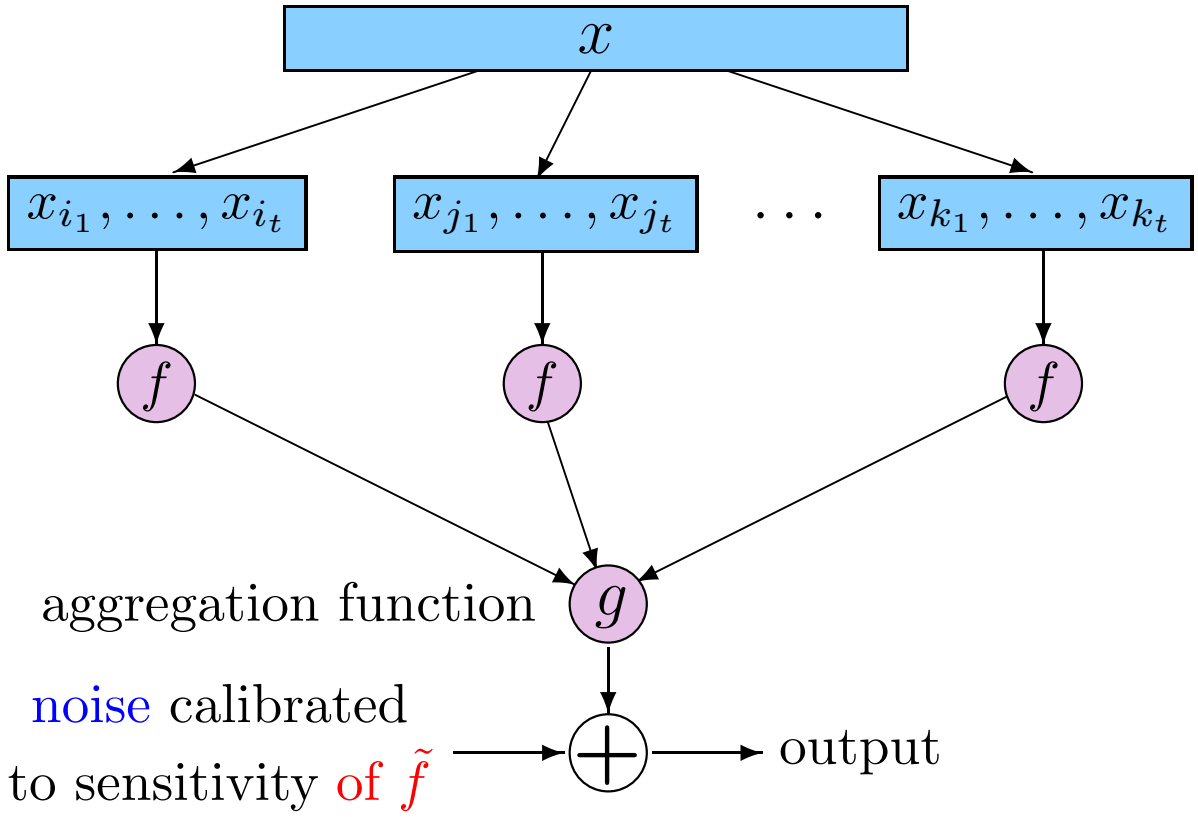
$$\tilde{f}(x) = g(f(sample_1), f(sample_2), \ldots, f(sample_s))$$



Sample-and-aggregate framework

Intuition: Replace f with a less sensitive function $\tilde{f}$.

$$\tilde{f}(x) = g(f(sample_1), f(sample_2), \ldots, f(sample_s))$$



Good aggregation functions

- average
  - works for L_1 and L_2
- center of attention
  - the center of a smallest ball containing a strict majority of input points
  - works for arbitrary metrics (in particular, for Earthmover)
  - gives lower noise for L_1 and L_2

Sample-and-aggregate method

Theorem

If f can be approximated on x
from small samples
then f can be released with little noise

Sample-and-aggregate method

Theorem

If f can be approximated on x within distance r
from small samples of size $n^{1-\delta}$
then f can be released with little noise $\approx \frac{r}{\varepsilon} + negl(n)$

Sample-and-aggregate method

Theorem

If f can be approximated on x within distance r
from small samples of size $n^{1-\delta}$
then f can be released with little noise $\approx \frac{r}{\varepsilon} + negl(n)$

- Works in all "interesting" metric spaces
- Example applications
  - k-means cluster centers (if data is separated a.k.a. [Ostrovsky Rabani Schulman Swamy 06])
  - fitting mixtures of Gaussians (if data is i.i.d., using [Achlioptas McSherry 05])
  - PAC concepts (if uniquely learnable,
    i.e., if learning algorithm always outputs
    the same hypothesis or something close to it)

Road map

I. Function approximation

- Global sensitivity framework [DMNS06]
- Smooth sensitivity framework [NRS07]
- Sample-and-aggregate [NRS07]

II. Learning

- Exponential mechanism [McSherry Talwar 07, Kasiviswanathan Lee Nissim Raskhodnikova Smith 08]

Learning: the setting

Bank needs to decide which applicants are bad credit risks*

Goal: given sample of labeled data (past customers), produce good prediction rule (hypothesis) for future loan applicants

*Example taken from Blum, FOCS03 tutorial

Learning: the setting

Bank needs to decide which applicants are bad credit risks*

Goal: given sample of labeled data (past customers), produce good prediction rule (hypothesis) for future loan applicants

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

*Example taken from Blum, FOCS03 tutorial

Learning: the setting

Bank needs to decide which applicants are bad credit risks*

Goal: given sample of labeled data (past customers), produce good prediction rule (hypothesis) for future loan applicants

	% down	high debt	other accts	mmp/ inc	good risk?
	10	No	Yes	0.32	**Yes**
example y_i	10	No	No	0.25	**Yes**
	5	Yes	No	0.30	**No**
	20	No	Yes	0.31	**Yes**
	10	No	No	0.25	**Yes**

*Example taken from Blum, FOCS03 tutorial

Learning: the setting

Bank needs to decide which applicants are bad credit risks*

Goal: given sample of labeled data (past customers), produce good prediction rule (hypothesis) for future loan applicants

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

example y_i — label z_i

*Example taken from Blum, FOCS03 tutorial

Learning: the setting

Bank needs to decide which applicants are bad credit risks*

Goal: given sample of labeled data (past customers), produce good prediction rule (hypothesis) for future loan applicants

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

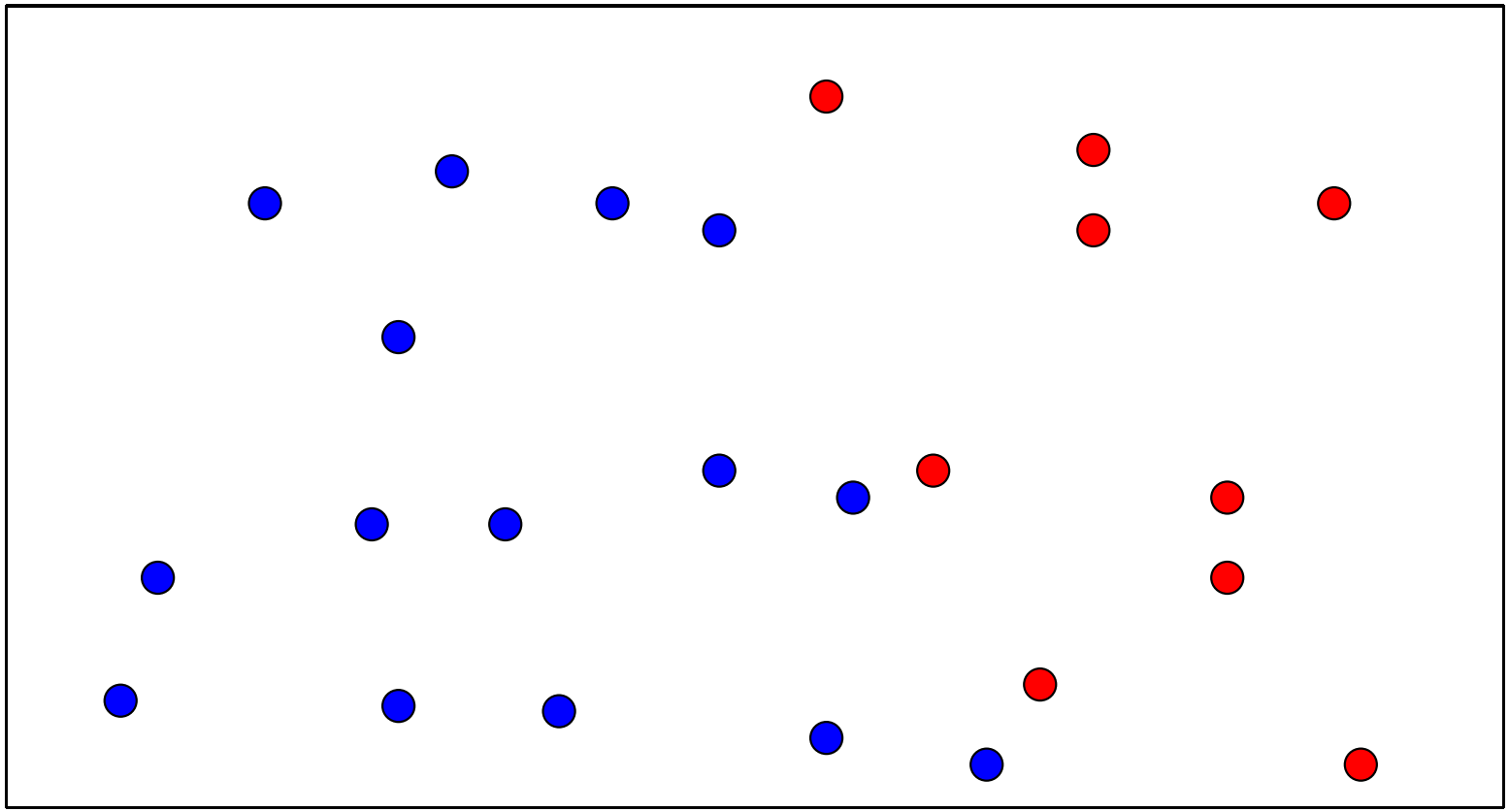
example y_i — label z_i

Reasonable rules given this data:

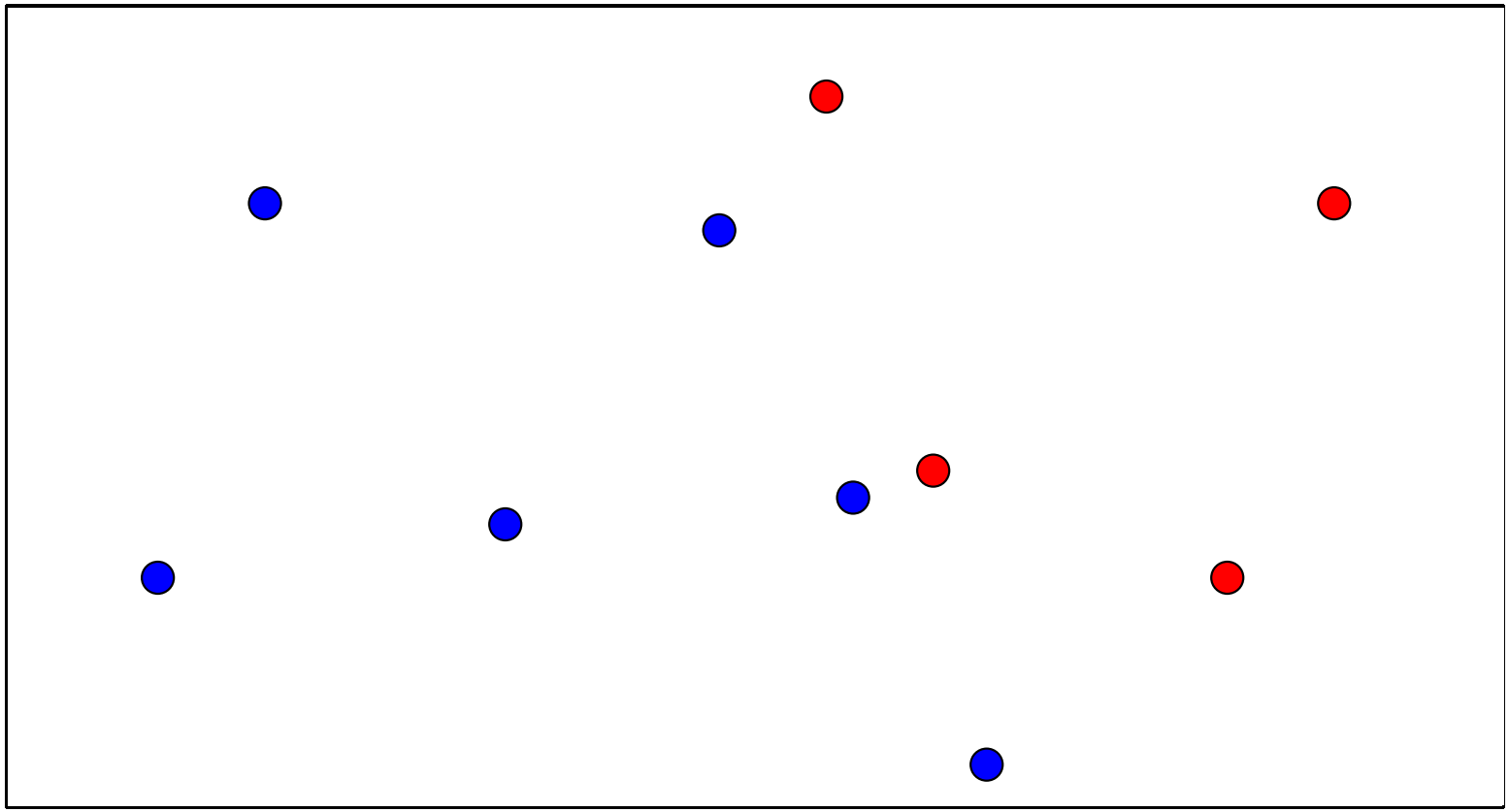
- Predict YES iff $100 \times \frac{\text{mmp}}{\text{inc}} - (\%\ \text{down}) < 25$
- Predict YES iff (!high debt) AND (% down > 5)

*Example taken from Blum, FOCS03 tutorial

Learning: the setting

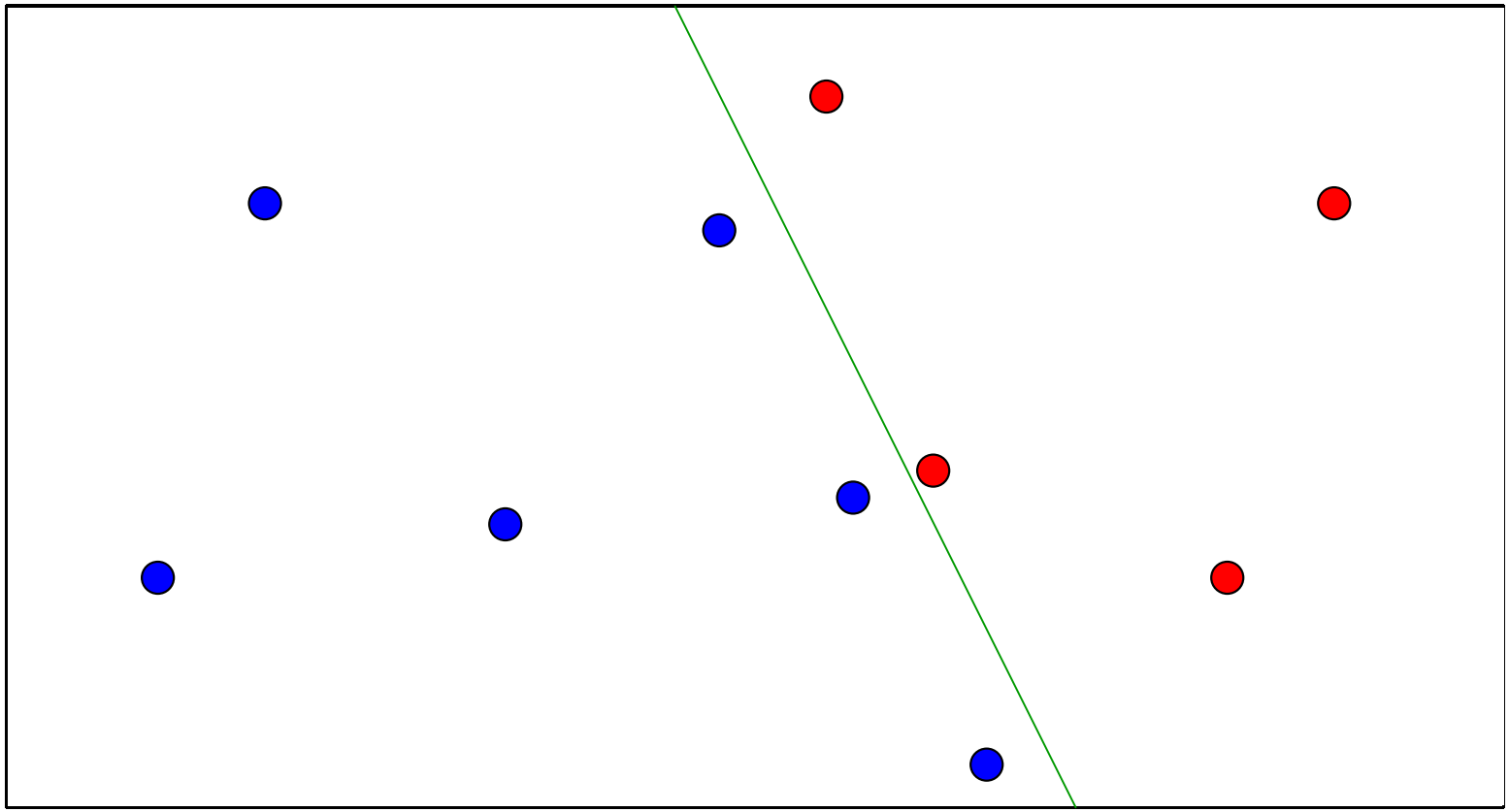


Learning: the setting



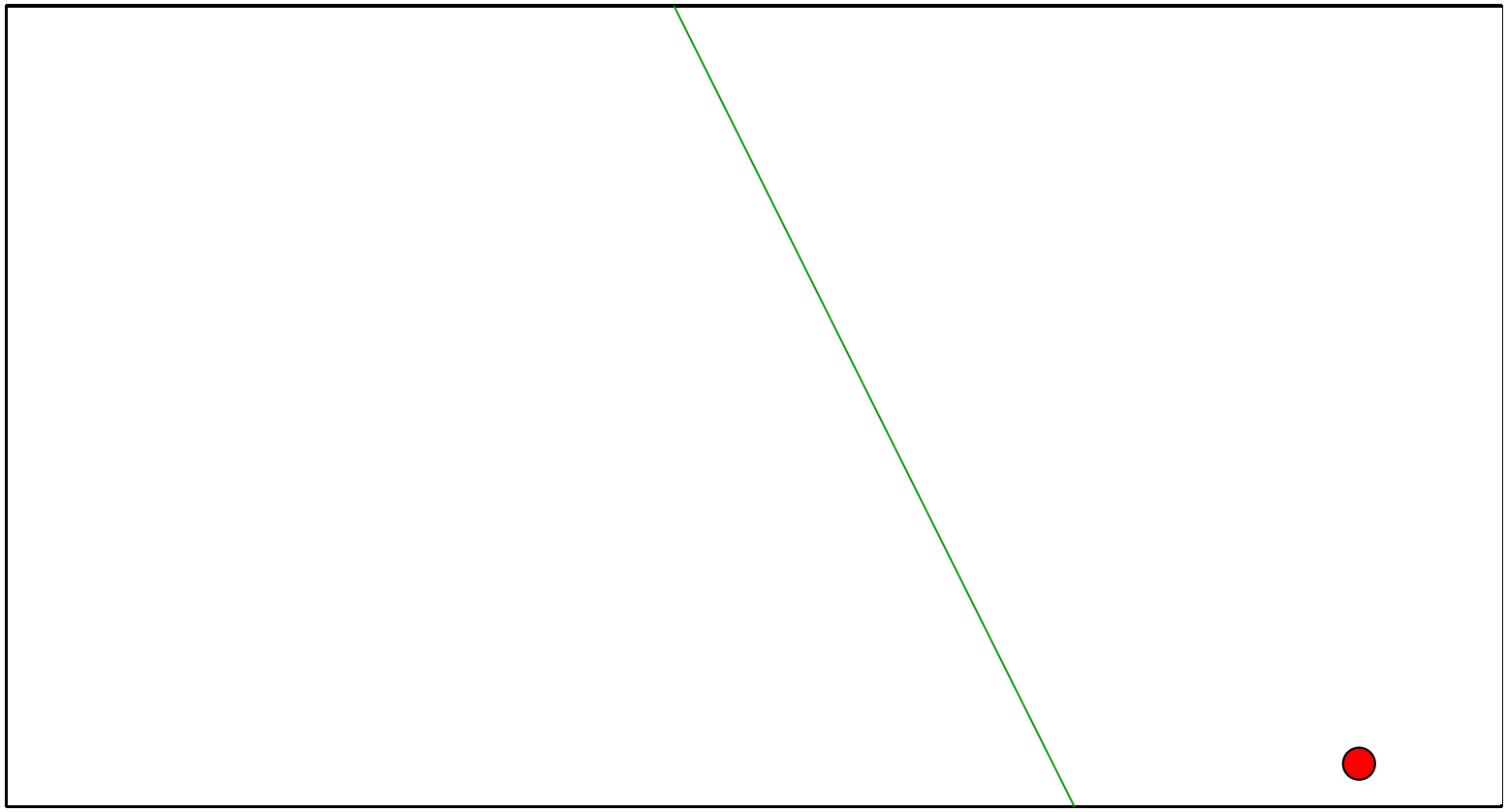
- Examples drawn according to distribution $\mathcal{D}$

Learning: the setting



- Examples drawn according to distribution $\mathcal{D}$

Learning: the setting



- Examples drawn according to distribution $\mathcal{D}$
- A point drawn according to $\mathcal{D}$ has to be classified correctly w.h.p. (over learner randomness and $\mathcal{D}$)

PAC learning [Valiant 84]

Given distribution $\mathcal{D}$ over examples, labeled by function c, hypothesis h is *good* if it mostly agrees with c:

$$\Pr_{y \sim \mathcal{D}}[h(y) = c(y)] \text{ is close to 1.}$$

PAC learning [Valiant 84]

Given distribution $\mathcal{D}$ over examples, labeled by function c, hypothesis h is *good* if it mostly agrees with c:

$$\Pr_{y \sim \mathcal{D}}[h(y) = c(y)] \text{ is close to 1.}$$

Definition of PAC learning

Algorithm A PAC learns a concept class C if

- *given polynomially many examples, drawn from $\mathcal{D}$, labeled by some $c \in C$*
- *A outputs a good hypothesis with high probability in polynomial time*

PAC learning [Valiant 84]

Given distribution $\mathcal{D}$ over examples, labeled by function c, hypothesis h is *good* if it mostly agrees with c:

$$\Pr_{y \sim \mathcal{D}}[h(y) = c(y)] \text{ is close to 1.}$$

Definition of PAC* learning

Algorithm A PAC* learns a concept class C if

- *given polynomially many examples, drawn from $\mathcal{D}$, labeled by some $c \in C$*
- *A outputs a good hypothesis of polynomial length with high probability ~~in polynomial time~~*

Private learning

Input: database $x = (x_1, ..., x_n)$

$x_i = (y_i, z_i)$, where $y_i \sim \mathcal{D}$ and $z_i = c(y_i)$ is the label of example y_i

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

Private learning

Input: database $x = (x_1, ..., x_n)$

$x_i = (y_i, z_i)$, where $y_i \sim \mathcal{D}$ and $z_i = c(y_i)$ is the label of example y_i

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

Output: hypothesis

e.g.

"Predict **Yes** if $100 \times \frac{\text{mmp}}{\text{inc}} - (\%\ \text{down}) < 25$ "

Private learning

Input: database $x = (x_1, ..., x_n)$

$x_i = (y_i, z_i)$, where $y_i \sim \mathcal{D}$ and $z_i = c(y_i)$ is the label of example y_i

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

Output: hypothesis

e.g.

"Predict **Yes** if $100 \times \frac{\text{mmp}}{\text{inc}} - (\%\ \text{down}) < 25$ "

Definition

Algorithm A privately learns concept class C if

- **Utility:** *Algorithm A PAC learns class C*
- **Privacy:** *Algorithm A is differentially private*

Private learning

Input: database $x = (x_1, ..., x_n)$

$x_i = (y_i, z_i)$, where $y_i \sim \mathcal{D}$ and $z_i = c(y_i)$ is the label of example y_i

% down	high debt	other accts	mmp/ inc	good risk?
10	No	Yes	0.32	**Yes**
10	No	No	0.25	**Yes**
5	Yes	No	0.30	**No**
20	No	Yes	0.31	**Yes**
10	No	No	0.25	**Yes**

Output: hypothesis

e.g.

"Predict **Yes** if $100 \times \frac{\text{mmp}}{\text{inc}} - (\%\ \text{down}) < 25$ "

Definition

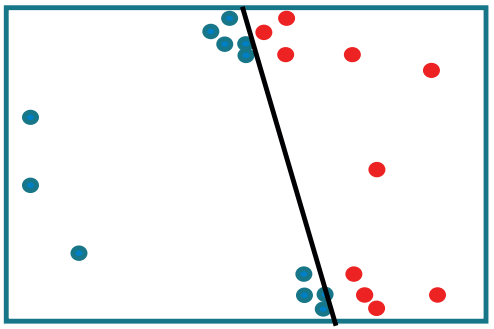
Algorithm A privately learns concept class C if

- **Utility:** *Algorithm A PAC learns class C* **(average-case)**
- **Privacy:** *Algorithm A is differentially private* **(worst-case)**

Designing private learners: baby steps

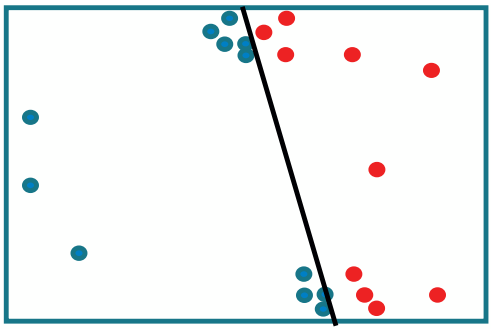
View non-private learner as function to be approximated

- First attempt: add noise
  - Problem: Close hypothesis may mislabel many points

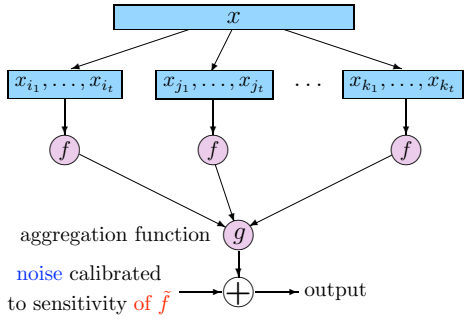


Designing private learners: baby steps

View non-private learner as function to be approximated



- First attempt: add noise
  - Problem: Close hypothesis may mislabel many points
- Second attempt:
  apply sample-and-aggregate
  to non-private learning algorithm



  - Works when good hypothesis is essentially unique
  - Problem: there may be many good hypotheses – different samples may produce different-looking hypotheses

$PAC^* = Private\ PAC^*$

Theorem **(Private analogue of "Occam's razor")**

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

$PAC^* = Private\ PAC^*$

Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

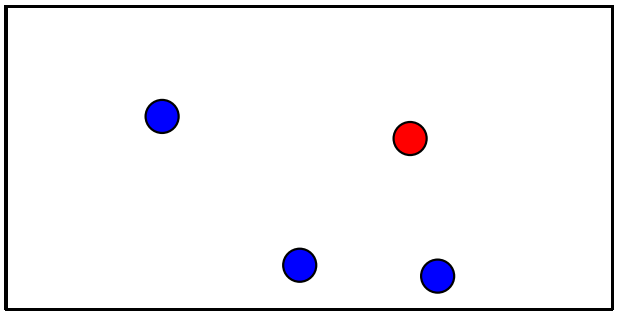
$PAC^* = Private\ PAC^*$

Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h



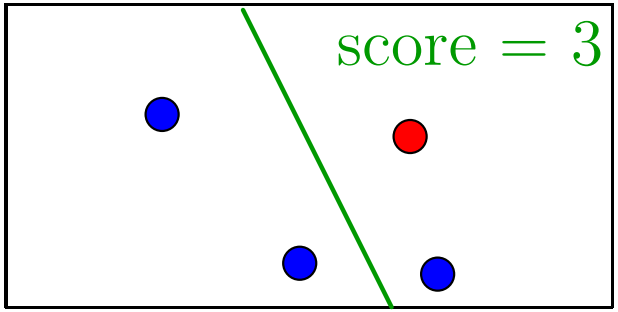
$PAC^* = Private\ PAC^*$

Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h



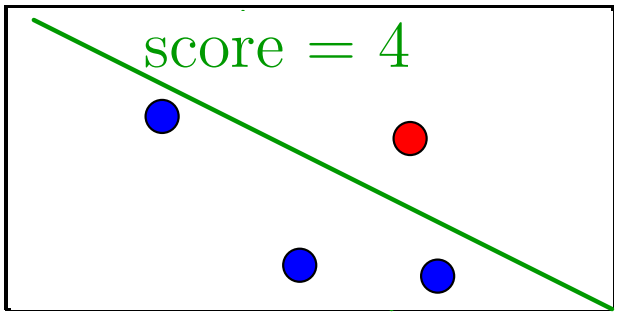
$PAC^* = Private\ PAC^*$

Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h



$PAC^* = Private\ PAC^*$

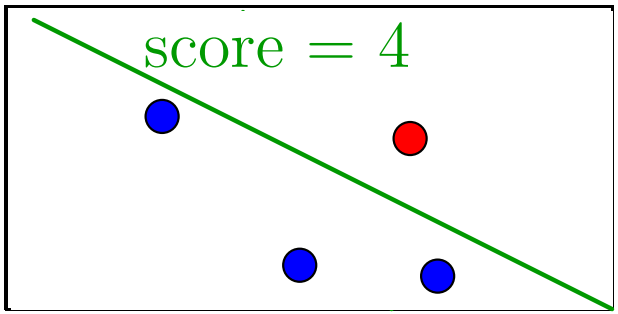
Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$
  - may take exponential time



$PAC^* = Private\ PAC^*$

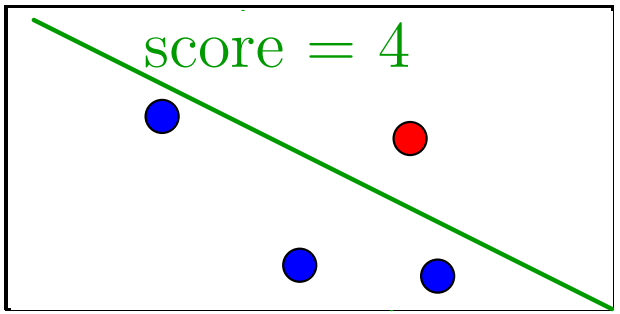
Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: Adapt the exponential mechanism of [MT07]:

$\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$
  - may take exponential time



Privacy: for any hypothesis h:

$$\frac{\Pr[h \text{ is output on input } x]}{\Pr[h \text{ is output on input } x']} = \frac{e^{\varepsilon \cdot \mathsf{score}(x,h)}}{e^{\varepsilon \cdot \mathsf{score}(x',h)}} \leq e^{\varepsilon}$$

PAC^* = Private PAC^*

Theorem (Private analogue of "Occam's razor")

Each PAC^* learnable concept class can be learned privately, using polynomially many samples.

Proof: $\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$

Utility (learning):

PAC = Private PAC**

Theorem (Private analogue of "Occam's razor")

Each PAC learnable concept class can be learned privately, using polynomially many samples.*

Proof: $\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$

Utility (learning):

Good h correctly label all examples: $\Pr[h] \sim e^{\varepsilon \cdot n}$

Bad h mislabel $\geq 10\%$ of examples: $\Pr[h] \sim e^{\varepsilon \cdot 0.9n}$

PAC = Private PAC**

Theorem (Private analogue of "Occam's razor")

Each PAC learnable concept class can be learned privately, using polynomially many samples.*

Proof: $\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$

Utility (learning):

Good h correctly label all examples: $\Pr[h] \sim e^{\varepsilon \cdot n}$

Bad h mislabel $\geq 10\%$ of examples: $\Pr[h] \sim e^{\varepsilon \cdot 0.9n}$

Sufficient to ensure $n \gg \underbrace{\log(\#\ \textbf{bad}\ \text{hypotheses})}_{\leq \text{output length of non-private learner}} = \text{polynomial}$

Then w.h.p. output h labels 90% of examples correctly.

PAC = Private PAC**

Theorem (Private analogue of "Occam's razor")

Each PAC learnable concept class can be learned privately, using polynomially many samples.*

Proof: $\mathsf{score}(x, h) = \#$ of examples in x correctly classified by h

- Output h from C with probability $\sim e^{\varepsilon \cdot \mathsf{score}(x,h)}$

Utility (learning):

Good h correctly label all examples: $\Pr[h] \sim e^{\varepsilon \cdot n}$

Bad h mislabel $\geq 10\%$ of examples: $\Pr[h] \sim e^{\varepsilon \cdot 0.9n}$

Sufficient to ensure $n \gg \underbrace{\log(\#\ \textbf{bad}\ \text{hypotheses})}_{\substack{\leq \text{output length of} \\ \text{non-private learner}}} = \text{polynomial}$

Then w.h.p. output h labels 90% of examples correctly.

By "Occam's razor", if $n \gg \log(\#\ \text{hypotheses})$, then

h does well on examples $\implies$ h does well on distrib. $\mathcal{D}$

Road map

I. Function approximation

- Global sensitivity framework [DMNS06]
- Smooth sensitivity framework [NRS07]
- Sample-and-aggregate [NRS07]

II. Learning

- Exponential mechanism [MT07,KLNRS08]

Conclusions

This talk: partial picture of techniques

- current techniques best for
  - function approximation
  - learning
- New ideas needed for
  - combinatorial search problems
  - text processing
  - graph data (definitions?)
  - high-dimensional outputs