What Can We Learn Privately?

Sofya Raskhodnikova  
*Penn State University*

*Joint work with*  
Shiva Kasiviswanathan  
Homin Lee  
Kobbi Nissim  
Adam Smith  

Los Alamos  
UT Austin  
Ben Gurion  
Penn State

*To appear in SICOMP special issue for FOCS ’08*
Private Learning

• **Goal:** machine learning algorithms that protect the privacy of individual examples (people, organizations,...)

• **Desiderata**
  – **Privacy:** Worst-case guarantee (differential privacy)
  – **Learning:** Distributional guarantee (e.g., \(\text{PAC learning}\))

• **This work**
  – Characterize classification problems learnable privately
  – Understand power of popular models for private analysis
What Can We Compute Privately?

Prior work:

• Function evaluation
  [DiNi,DwNi,BDMN,EGS,DMNS,\ldots]
  – Statistical Query (SQ) Learning [Blum Dwork McSherry Nissim 05]
  – Learning Mixtures of Gaussians [Nissim Raskhodnikova Smith 08]

• Mechanism design [McSherry Talwar 07]

This work: PAC Learning (in general, not captured by function evaluation)

Some subsequent work:

• Learning [Chaudhuri Monteleoni 08, McSherry Williams 09, Beimel Kasiwisanathan Nissim 10, Sarwate Chaudhuri Monteleoni]

• Statistical inference [Smith 08, Dwork Lei 09, Wasserman Zhou 09]

• Synthetic data [Machanavajjhala Kifer Abowd Gehrke Vilhuber 08,Blum Ligget Roth 08, Dwork Naor Reingold Rothblum Vadhan 09, Roth Roughgarden 10]

• Combinatorial optimization [Gupta Ligett McSherry Roth Talwar 10]
Our Results 1: What is Learnable Privately

PAC* = PAC learnable with poly samples, not necessarily efficiently

PAC* = Private PAC*
Basic Privacy Models

Local Noninteractive

Local (Interactive)

Centralized

- Most work in data mining
- “randomized response”, “input perturbation”, “Post Randomization Method” (PRAM), “Framework for High-Accuracy Strict-Privacy Preserving Mining” (FRAPP) [W65, AS00, AA01, EGS03, HH02, MS06]

Advantages:
- private data never leaves person’s hands
- easy distribution of extracted information (e.g., CD, website)
Our Results 2: Power of Private Models

$\text{Centralized} = \text{PAC}^*$

$\text{Privately PAC-learnable}$

$\text{Local} = \text{SQ}$

$\text{Local Noninteractive} = \text{Nonadaptive SQ}$

Painted Parity

Masked Parity

Parity

PAC* = PAC learnable with poly samples, not necessarily efficiently
Definition: Differential Privacy [DMNS06]

**Intuition:** Users learn roughly the same thing about me whether or not my data is in the database.

A randomized algorithm \( A \) is \( \varepsilon \)-differentially private if

- for all databases \( x, x' \) that differ in one element
- for all sets of answers \( S \)

\[
\Pr[A(x) \in S] \leq e^\varepsilon \Pr[A(x') \in S]
\]
Properties of Differential Privacy

• Composition:
  If algorithms $A_1$ and $A_2$ are $\varepsilon$-differentially private then the algorithm that outputs $(A_1(x), A_2(x))$ is $2\varepsilon$-differentially private.

• Meaningful in the presence of arbitrary external information.
Learning: An Example*

- Bank needs to decide which applicants are bad credit risks
- **Goal:** given sample of past customers (*labeled examples*), produce good prediction rule (*hypothesis*) for future loan applicants

| % down | Recent delinquency? | High debt? | Mmp/inc | Good Risk? |
|--------|---------------------|------------|---------|------------|
| 10     | No                  | No         | 0.32    | Yes        |
| 10     | No                  | Yes        | 0.25    | Yes        |
| 5      | Yes                 | No         | 0.30    | No         |
| 20     | No                  | No         | 0.31    | Yes        |
| 5      | No                  | No         | 0.32    | No         |
| 10     | Yes                 | Yes        | 0.38    | No         |

**Reasonable hypotheses given this data:**
- Predict YES iff (!Recent Delinquency) AND (% down > 5)
- Predict YES iff $100 \times (\frac{\text{Mmp}}{\text{inc}}) - (%) \text{ down} < 25$

*Example taken from Blum, FOCS03 tutorial*
PAC Learning: The Setting

Algorithm draws independent examples from some distribution $P$, labeled by some target function $c$. 
Algorithm outputs hypothesis $h$ (a function from points to labels).
PAC Learning: The Setting

- Hypothesis $h$ is good if it mostly agrees with target $c$:
  $$\Pr_{y \sim \mathcal{P}} [h(y) \neq c(y)] \leq \alpha.$$  
- Require that $h$ is good with probability at least $1 - \beta$. 

new point drawn from $\mathcal{P}$
**PAC**\* Learning Definition

A concept class $C$ is a set of functions \{c : D→\{0,1\}\} together with their representation.

**Definition.** Algorithm A **PAC**\* learns concept class $C$ if, for all $c$ in $C$, all distributions $P$ and all $\alpha$, $\beta$ in $(0,1/2)$

- Given $\text{poly}(1/\alpha,1/\beta,\text{size}(c))$ examples drawn from $P$, labeled by some $c$ in $C$
- A outputs a good hypothesis (of accuracy $\alpha$) of poly length with probability $\geq 1- \beta$ in poly-time
Private Learning

**Input:** Database: \( x = (x_1, x_2, \ldots, x_n) \) where
\[ x_i = (y_i, z_i), \] where \( y_i \sim P, \) \( z_i = c(y_i) \) (\( z_i \) is the label of example \( y_i \))

**Output:** a hypothesis

\[ \text{e.g., Predict Yes if } 100 \times (\text{Mmp/inc}) - (\% \text{ down}) < 25 \]

| % down | Recent delinquency? | High debt? | Mmp/inc | Good Risk? |
|--------|---------------------|------------|---------|------------|
| 10     | No                  | No         | 0.32    | Yes        |
| 10     | No                  | Yes        | 0.25    | Yes        |
| 25     | No                  | No         | 0.30    | Yes        |
| 20     | No                  | No         | 0.31    | Yes        |

- Algorithm A **privately PAC learns** concept class \( C \) if:
  - **Utility:** Algorithm A PAC learns concept class \( C \)
  - **Privacy:** Algorithm A is \( \varepsilon \)-differentially private
How Can We Design Private Learners?

• Previous privacy work focused on function approximation

• **First attempt:** View non-private learner as function to be approximated
  – Problem: “Close” hypothesis may mislabel many points
\[ \text{PAC}* = \text{Private PAC}* \]

**Theorem.** Every PAC* learnable concept class can be learned privately, using a poly number of samples.

**Proof:** Adapt exponential mechanism [MT07]:

\[ \text{score}(x,h) = \text{# of examples in } x \text{ correctly classified by hypothesis } h \]

Output hypothesis \( h \) from \( C \) with probability \( \sim e^{\varepsilon \cdot \text{score}(x,h)} \)

- may take exponential time

**Privacy:** for any hypothesis \( h \),

\[
\frac{\Pr[h \text{ is output on input } x]}{\Pr[h \text{ is output on input } x']} = \frac{e^{\varepsilon \cdot \text{score}(x,h)}}{e^{\varepsilon \cdot \text{score}(x',h)} \cdot \frac{\sum_h e^{\varepsilon \cdot \text{score}(x',h)}}{\sum_h e^{\varepsilon \cdot \text{score}(x,h)}}} \leq e^{2\varepsilon}
\]

\[ \text{score}(x,h)=4 \]
\textbf{PAC* = Private PAC*}

**Theorem.** Every PAC* learnable concept class can be learned privately, using a poly number of samples.

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

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

**Utility (learning):**

- Best hypothesis correctly labels all examples: $\Pr[h] \sim e^{\varepsilon \cdot n}$
- Bad hypotheses mislabel $> \alpha$ fraction of examples: $\Pr[h] \sim e^{\varepsilon (1-\alpha) n}$

$$\Pr[\text{output } h \text{ is bad}] \leq e^{\varepsilon (1-\alpha) n} \cdot \frac{\# \text{ bad hypothesis}}{e^{\varepsilon n}} \leq \frac{|C|}{e^{\varepsilon \alpha n}}$$

Sufficient to ensure $n \geq (\ln |C| + \ln(1/\beta)) / (\varepsilon \alpha)$. Then w/ probability $\geq 1-\beta$, output $h$ labels $\geq 1-\alpha$ fraction of examples correctly.

- “Occam’s razor”: If $n \geq (\ln |C| + \ln(1/\beta)) / \alpha$ then $h$ does well on examples $\Rightarrow$ it does well on distribution $P$
Our Results: What is Learnable Privately

- Parity
- PAC
- PAC-learnable
- Privately PAC-learnable
- SQ [BDMN05]
  - Halfplanes
  - Conjunctions
  - ...
**Efficient Learner for Parity**

**Parity Problems**

**Domain:** $D = \{0,1\}^d$

**Concepts:** $c_r(x) = \langle r, x \rangle \mod 2$

**Input:** $x = ((y_1, c_r(y_1)), \ldots, (y_n, c_r(y_n)))$

- Each example $(y_i, c_r(y_i))$ is a linear constraint on $r$
  - $(1101, 1)$ translates to $r_1 + r_2 + r_4 \mod 2 = 1$

- **Non-private learning algorithm:**
  - Find $r$ by solving the set of linear equations over $\text{GF}(2)$ imposed by input $x$
The Effect of a Single Example

• Let $V_i$ be space of feasible solutions for the set of equations imposed by $(y_1, c_r(y_1)), \ldots, (y_i, c_r(y_i))$

• Add a fresh example $(y_{i+1}, c_r(y_{i+1}))$
  – Consider the new solution space $V_{i+1}$

• Then
  – $|V_{i+1}| \geq |V_i|/2$, or
  – $|V_{i+1}| = 0$ (system becomes inconsistent)

The solution space changes drastically only when the non-private learner fails

new constraint: shared coordinate is 0
Private Learner for Parity

Algorithm A

1. With probability $\frac{1}{2}$ output “fail”.

2. Construct $x_s$ by picking each example from $x$ with probability $\varepsilon$.

3. Solve the system of equations imposed by examples in $x_s$.
   - Let $V$ be the set of feasible solutions.

4. If $V = \emptyset$, output “fail”.
   Otherwise, choose $r$ from $V$ uniformly at random; output $c_r$.

Lemma [utility]. Our algorithm PAC-learns parity with
\[ n = O((\text{non-private-sample-size})/\varepsilon) \]

Proof idea: Conditioned on passing step 1, get the same utility as with $\varepsilon n$ examples. By repeating a few times, pass step 1 w.h.p.
Lemma. Algorithm $A$ is $4\varepsilon$-differentially private.

Proof: For inputs $x$ and $x'$ that differ in position $i$, show that for all outcomes probability ratio $\leq 1+4\varepsilon$.

- Changed input $x_i$ enters the sample with probability $\varepsilon$.
- Probability of “fail” goes up or down by $\leq \varepsilon/2$.

$$\frac{\Pr[A(x) \text{ fails}]}{\Pr[A(x') \text{ fails}]} \leq \frac{\Pr[A(x') \text{ fails}]+\varepsilon/2}{\Pr[A(x') \text{ fails}]} \leq 1+\varepsilon \text{ as } \Pr[A(x') \text{ fails}] \geq 1/2.$$ 

- For hypothesis $r$:

$$\frac{\Pr[A(x) = r]}{\Pr[A(x') = r]} \leq \frac{\varepsilon \Pr[A(x) = r \mid i \in S] + (1-\varepsilon)\Pr[A(x) = r \mid i \notin S]}{0} + (1-\varepsilon)\Pr[A(x') = r \mid i \notin S]$$

$$\leq 2 \frac{\varepsilon}{(1-\varepsilon)} + 1 \leq 4\varepsilon+1 \quad \text{for } \varepsilon \leq \frac{1}{2}$$

- The 2$^{nd}$ $\leq$ uses:

$$\frac{\Pr[A(x) = r \mid i \in S]}{\Pr[A(x') = r \mid i \notin S]} = \frac{\Pr[A(x) = r \mid i \in S]}{\Pr[A(x) = r \mid i \notin S]} \leq 2.$$ 

Intuitively, this follows from $|V_i| \geq |V_{i-1}|/2$. 


Our Results: What is Learnable Privately

Note: Parity with noise is thought to be hard

Private PAC ≠ learnable with noise
Our Results 2: Power of Private Models

Local Noninteractive = Nonadaptive SQ

Local = SQ

Centralized = PAC* ✓

PAC* = PAC learnable ignoring computational efficiency
Reminder: Local Privacy Preserving Protocols

- Interactive
- Non-interactive
Statistical Query (SQ) Learning [Kearns 93]

- Same guarantees as PAC model, but algorithm no longer has access to individual examples

\[ g : D \times \{0,1\} \rightarrow \{0,1\} \quad \text{Algorithm} \]

Probability that a random labeled example \((\sim P)\) satisfies \(g\)

**Theorem** [BDMN05]. Any SQ algorithm can be simulated by a private algorithm

**Proof:** [DMNS06] Perturb query answers using Laplace noise.
**Theorem.** Any (non-adaptive) SQ algorithm can be simulated by a (non-interactive) local algorithm

- **Local protocol for SQ:**
  - For each $i$, compute bit $R(x_i) = \begin{cases} g(x_i) & \text{w.p. } \frac{1}{2} + \epsilon \\ 1 - g(x_i) & \text{w.p. } \frac{1}{2} - \epsilon \end{cases}$
  - Sum of noisy bits allows approximation to answer
- Participants can compute noisy bits on their own
- $R$ (applied by each participant) is differentially private
- If all SQ queries are known in advance (non-adaptive), the protocol is non-interactive
Theorem. Any (non-interactive) local algorithm can be simulated by a (non-adaptive) SQ algorithm.

Technique: Rejection sampling

Proof idea [non-interactive case]:

To simulate randomizer $R: D \rightarrow W$ on entry $z_i$, need to output $w$ in $W$ with probability $p(w)=\Pr_{z \sim p}[R(z)=w]$. Let $q(w)=\Pr[R(0)=w]$. (Approximates $p(w)$ up to factor $e^{\epsilon}$).

1. Sample $w$ from $q(w)$
2. With probability $p(w)/(q(w)e^{\epsilon})$, output $w$
3. With the remaining probability repeat from (1)

Use SQ queries to estimate $p(w)$.

Idea: $p(w) = \Pr_{z \sim p}[R(z)=w] = E_{z \sim p}[\Pr[R(z)=w]].$
Our Results 2: Power of Private Models

Local Noninteractive = Nonadaptive SQ

Local = SQ

Centralized = PAC*

PAC* = PAC learnable ignoring computational efficiency
Non-interactive Local $\not\equiv$ Interactive Local

Masked Parity Problems

Concepts: $c_{r,a} : \{0,1\}^{d+\log d+1} \rightarrow \{+1,-1\}$

indexed by $r \in \{0,1\}^d$ and $a \in \{0,1\}$

$$c_{r,a}(y,i,b) = \begin{cases} 
(-1)^r \cdot y \pmod 2 + a & \text{if } b=0 \\
(-1)^r i & \text{if } b=1 
\end{cases}$$

• (Adaptive) SQ learner: Two rounds of communication

• Non-adaptive SQ learner: Needs $\geq 2^d-1$ samples
  – Proof uses Fourier analytic argument
    similar to proof that parity is not in SQ
Summary

• PAC* is privately learnable
  – Non-efficient learners
• Known problems in PAC are efficiently privately learnable
  – Parity
  – SQ [BDMN05]
  – What else is in PAC?
• Equivalence of local model and SQ:
  – Local = SQ
  – Local non-interactive = non-adaptive SQ
• Interactivity helps in local model
  – Local non-interactive ⋙ Local
  – SQ non-adaptive ⋙ SQ

Open questions

• Separate efficient learning from efficient private learning
• Better private algorithms for SQ problems
• Other learning models