Advances in Privacy-Preserving Machine Learning

Claire Monteleoni
Center for Computational Learning Systems
Columbia University

Challenges of real-world data

We face an explosion in data from e.g.:

- Internet transactions
- Satellite measurements
- Environmental sensors
- …

Real-world data can be:

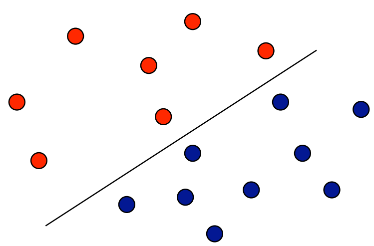
- Vast (many examples)
- High-dimensional
- Noisy (incorrect/missing labels/features)
- Sparse (relevant subspace is low-dim.)
- Streaming, time-varying
- Sensitive/private

Machine learning

Given labeled data points, find a good classification rule.

- Describes the data
- Generalizes well

E.g. linear separators:



Principled ML for real-world data

Goal: design algorithms to detect patterns in real-world data.

Want efficient algorithms, with performance guarantees.

Learning with online constraints:

Algorithms for streaming, or time-varying data.

Active learning:

Algorithms for settings in which unlabeled data is abundant, and labels are difficult to obtain.

Privacy-preserving machine learning:

Algorithms to detect cumulative patterns in real databases, while maintaining the privacy of individuals.

New applications of machine learning:

E.g. Climate Informatics: Algorithms to detect patterns in climate data, to answer pressing questions.

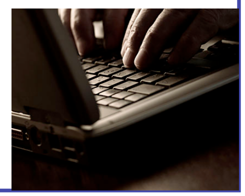
Privacy-preserving machine learning

Sensitive personal data is increasingly being digitally aggregated and stored.

Problem: How to maintain the privacy of individuals, when detecting patterns in cumulative, real-world data?

E.g.

- Disease studies, insurance risk
- Economics research, credit risk
- Analysis of social networks



Anonymization: not enough

Anonymization does not ensure privacy.

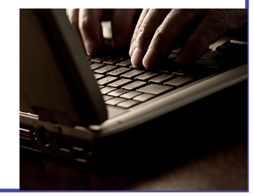
Attacks may be possible *e.g.* with:

- Auxiliary information
- Structural information

Privacy attacks:

[Narayanan & Shmatikov '08] identify Netflix users from anonymized records, IMDB.

[Backstrom, Dwork & Kleinberg '07] identify LiveJournal social relations from anonymized network topology and minimal local information.



Related work

Data mining:

Algorithms, often lacking strong privacy guarantees. Subject to various attacks.

Cryptography and information security:

Privacy guarantees, but machine learning less explored.

Learning theory

Learning guarantees for algorithms that adhere to strong privacy protocols, but are not efficient.

Related work

Data mining:

k-anonymity [Sweeney '02], ℓ-diversity [Machanavajjhala et al. '06], t-closeness [Li et al. '07]. Each found privacy attacks on previous.

All are subject to composition attacks [Ganta et al. '08].

Cryptography and information security:

[Dwork, McSherry, Nissim & Smith, TCC 2006]: Differential privacy, and sensitivity method. Extensions, [Nissim et al. '07].

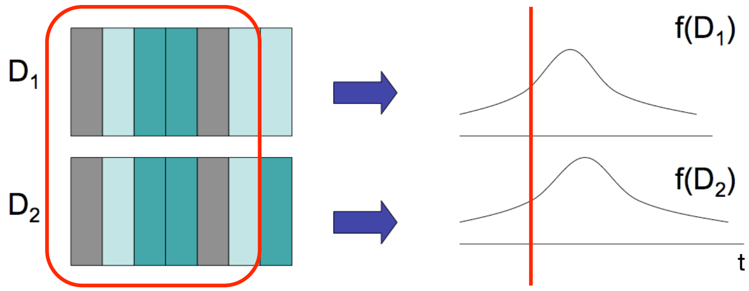
Learning theory

[Blum et al. '08] method to publish data that is differentially private under certain query types. (Can be computationally prohibitive.)

[Kasiviswanathan et al. '08] exponential time (in dimension) algorithm to find classifiers that respect differential privacy.

ε−differential privacy

[DMNS '06]: Given two databases, D_1, D_2 that differ in one element:



A random function f is ε-private, if, for any t

$$\Pr[\, f(D_1) = t\,] \le (1 + \varepsilon) \Pr[\, f(D_2) = t\,]$$

Idea: Effect of one person's data on the output: low.

The sensitivity method

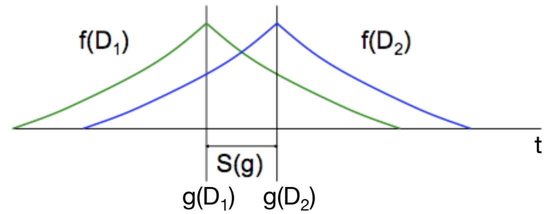
[DMNS '06]: method to produce ε−private approximation to any function of a database.

Sensitivity: For function g, *sensitivity* S(g) is the maximum change in g with one input.

$$S(g) \;=\; \max_{(a,a')} |g(x_1, \ldots, x_{n-1}, x_n = a) - g(x_1, \ldots, x_{n-1}, x_n = a')|$$

[DMNS '06]: Add noise, proportional to sensitivity. Output:

$f(D) = g(D) + \mathrm{Lap}(0, S(g)/\varepsilon)$



Motivations and contributions

Goal: machine algorithms that maintain privacy yet output good classifiers.

- Adapt canonical, widely-used machine learning algorithms
- Learning performance guarantees
- Efficient algorithms with good practical performance

[Chaudhuri & Monteleoni, NIPS 2008]:

A new privacy-preserving technique: perturb the optimization problem, instead of perturbing the solution.

Applied both techniques to logistic regression, a canonical ML algorithm.

Proved learning performance guarantees that are significantly tighter for our new algorithm.

Encouraging results in simulation.

Regularized logistic regression

We apply sensitivity method of [DMNS '06] to regularized logistic regression, a canonical, widely-used algorithm for learning a linear separator.

```
Regularized logistic regression:
Input: (x_1,y_1),...,(x_n,y_n).
 x_i in R^d, norm at most 1.  y_i in {-1, +1}.
Output:
```

$$w^* = \arg\min_w \frac{\lambda}{2} w^T w + \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i w^T x_i))$$

- Derived from model: $p(y|x;w) = \dfrac{1}{1 + \exp(-y w^T x)}$
- First term: regularization.
- w in R^d predicts $\mathrm{SIGN}(w^T x)$ for x in R^d.

Sensitivity method applied to LR

Sensitivity method [DMNS '06] applied to logistic regression:

Lemma: The sensitivity of regularized logistic regression is 2/nλ.

```
Algorithm 1 [Sensitivity-based PPLR]:
1. Solve w = regularized logistic regression with
   parameter λ.
2. Pick a vector h:
   Pick |h| from Γ(d, 2/nλε),
   Pick direction of h uniformly.
3. Output w + h.
```

```
Where density of
Γ(d,t) at x ~
x^(d-1) e^(-|x|/t)
```

Theorem 1: Algorithm 1 is ε-private.

New method for PPML

A new privacy-preserving technique: perturb the optimization problem, instead of perturbing the solution.

No need to bound sensitivity (may be difficult for other ML algorithms)
Noise does not depend on (the sensitivity of) the function to be learned.
Optimization happens *after* perturbation.

Application to regularized logistic regression:

```
Algorithm 2 [New PPLR]
1. Pick a vector b:
   Pick |b| from Γ(d, 2/ε),
   Pick direction of b uniformly.
2. Output:
```

$$w^* = \arg\min_w \frac{\lambda}{2} w^T w + \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i w^T x_i)) + \frac{1}{n} b^T w$$

New method for PPML

Theorem 2: Algorithm 2 is ε-private.

Remark: Algorithm 2 solves a convex program similar to standard, regularized LR, so similar running time.

General PPML for a class of convex loss functions:

Theorem 3: Given database $X=\{x_1,\ldots,x_n\}$, to minimize functions of the form: $F(w) = G(w) + \sum_{i=1}^{n} l(w, x_i)$

If G(w), $l(w, x_i)$ everywhere differentiable, have continuous derivatives G(w) strongly convex, $l(w, x_i)$ convex $\forall i$ and $\|\nabla_w l(w,x)\| \leq \kappa$, for any x,

then outputting $w^* = \arg\min_w G(w) + \sum_{i=1}^{n} l(w, x_i) + b^T w$

where b = B r, s.t. B is drawn from Γ(d, 2κ/ε), r is a random unit vector,

is *ε-private*.

Privacy of Algorithm 2

Proof outline (Theorem 2):

Want to show Pr[f(D_1) = w*] ≤ (1 + ε) Pr[f(D_2) = w*].

$$D_1 = \{(x_1,y_1),\ldots,(x_{n-1},y_{n-1}),(a,y)\} \qquad \forall i,\ ||x_i|| \leq 1$$
$$D_2 = \{(x_1,y_1),\ldots,(x_{n-1},y_{n-1}),(a',y')\} \qquad ||a||, ||a'|| \leq 1$$

$$\Pr[f(D_1) = w^*] = \Pr[w^* | x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1},x_n = a, y_n = y]$$
$$\Pr[f(D_2) = w^*] = \Pr[w^* | x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1},x_n = a', y_n = y']$$

We must bound the ratio:

$$\frac{\Pr[w^*|x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1},x_n=a,y_n=y]}{\Pr[w^*|x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1},x_n=a',y_n=y']} = \frac{h(b_1)}{h(b_2)} = e^{-\frac{\epsilon}{2}(||b_1||-||b_2||)}$$

Where b_1 is the unique value of b that yields w* on input D_1. (Likewise b_2)
- b's are unique because both terms in objective differentiable everywhere.

Where $h(b_i)$ is Γ density function at b_i.

Bound RHS, using optimality of w* for both problems, and bounded norms.

Learning guarantees

Theorem 4: For iid data, w.r.t. any classifier w_0 with loss $L(w_0)$, Algorithm 2 outputs a classifier with loss $L(w_0) + \delta$ if:

$$n > C \cdot \max\left(\frac{||w_0||^2}{\delta^2}, \frac{||w_0||d}{\epsilon\delta}\right)$$

where $L(w) = E[\log(1 + \exp(-y\, w^T x))]$.

Theorem 5: Bound for Algorithm 1 in identical framework:

$$n > C \cdot \max\left(\frac{||w_0||^2}{\delta^2}, \frac{||w_0||d}{\epsilon\delta}, \frac{||w_0||^2 d}{\epsilon\delta^{3/2}}\right)$$

The bound for Algorithm 2 is tighter than that of Algorithm 1, for cases in which (non-private) regularized logistic regression is useful, i.e. $||w_0|| \geq 1$ (otherwise $L(w_0) \geq \log(1 + 1/e)$).

Learning guarantees

Proof ideas for Theorems 4 and 5:

- Lemmas bounding the approximation to (non-private) regularized LR:

1. Lemma (Algorithm 1):

$$f(w_1) \leq f(w') + \frac{2d^2(1+\lambda)\log^2(d/\delta)}{\lambda^2 n^2 \epsilon^2}$$

2. Lemma (Algorithm 2):

$$f(w_2) \leq f(w') + \frac{8d^2\log^2(d/\delta)}{\lambda n^2 \epsilon^2}$$

where w' optimizes regularized LR objective, *f*, with parameter λ.

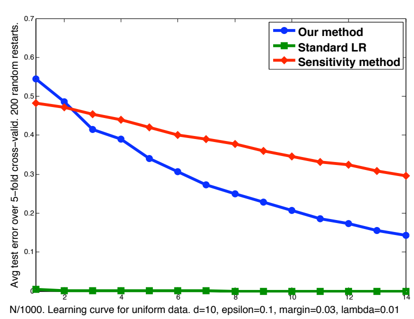
- Use techniques of:
  - [Shalev-Schwartz & Srebro, ICML 2008]
  - [Sridharan, Srebro, & Shalev-Schwartz, NIPS 2008].

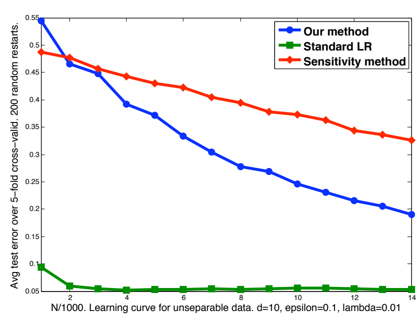
  to obtain generalization guarantees from these *approximate* optimization guarantees (*vs.* ERM).

Experiments

	Uniform, margin=0.03	Unseparable (uniform with noise 0.2 in margin 0.1)
Sensitivity method	0.2962 ± 0.0617	0.3257 ± 0.0536
New method	0.1426 ± 0.1284	0.1903 ± 0.1105
Standard LR	0 ± 0.0016	0.0530 ± 0.1105

Figure 1: Test error: mean ± standard deviation over five folds. N=17,500.

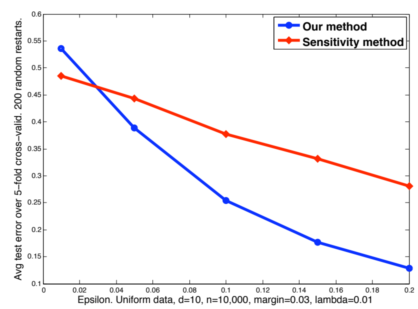


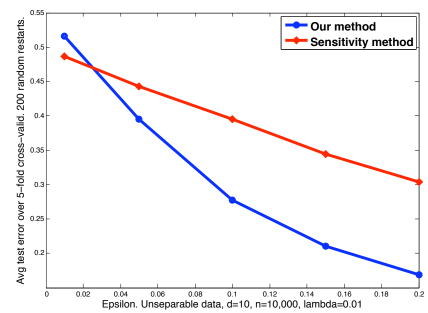


Learning curves

Experiments

Dependence on ε





PP Support Vector Machine

Support vector machine (SVM) enjoys extensive use and empirical success in ML and data-mining applications.

- Good generalization, robust to unseparable data.
- Classifier is the result of a convex optimization.

[Chaudhuri, Monteleoni, & Sarwate, manuscript 2009]

- Addresses the following challenges:

1. Non-differentiability of SVM objective (hinge-loss).
   ➔ Upper bound by a differentiable function with similar learning utility.
2. Standard SVM prediction (in RKHS) involves releasing part of database.
   ➔ Create random kernel, using [Rahimi & Recht, NIPS 2008].

- Algorithm is differentially private.
- Learning performance guarantees stronger than Sensitivity method.
- Good empirical performance.

Future work

Other standard ML algorithms, *e.g.*

Boosting, clustering, approximate *k*-nearest neighbor, *etc.*

Privacy-preserving optimization

A general technique to turn a convex optimization problem into a privacy-preserving version (by extending our results to fewer assump.s)

With increasing reliance on the internet for day-to-day tasks, emerging, necessary synergy between security/privacy and machine learning research, *e.g.*

PPML
Spam filtering
Identity theft detection
Fraud/anomaly/phishing detection

Thank you!

And many thanks to my collaborators:

Kamalika Chaudhuri (UC San Diego)
Anand Sarwate (UC San Diego)