10-806 Foundations of Machine Learning and Data Science

Lecturer: Maria-Florina Balcan Lecture 25: December 07, 2015

---

1 Differential Privacy and Statistical Query Learning

1.1 Differential Privacy

Suppose we are performing machine learning on sensitive data such as patient medical records. One concern we might have is that even if the person running the machine learning algorithm can be trusted, they hypothesis h itself produced by the algorithm could leak sensitive information. For example, suppose we are running a decision list algorithm, and suppose feature j is "has-green-hair". Then if h contains the rule "if $(x_j = 1)$ then positive" and there is only one person in town with green hair, then we know that person must have been in the study and have been positive. This kind of problem can happen for standard linear classifier algorithms as well. Notice that it occurs even if we have stripped names, ages, etc., from the data in advance.

We would like to be able to assure participants in a medical study or other data sample that their privacy will be protected in a formal, meaningful way. An important notion that will allow us to design algorithms and prove strong privacy properties for them is that of *differential privacy*, introduced in [7, 4], which we describe now. See also [5, 6] for further discussion.

1.1.1 The Privacy Definition

Let A be a probabilistic learning algorithm, that takes in a training sample and then outputs some hypothesis. What we are going to do is imagine running A on two different training samples that differ in just one of their data points. That is, let $S = \{z_1, z_2, \ldots, z_m\}$ and let S' be the same as S except replacing z_i with z_i'. We will say that A is differentially private if for any possible hypothesis h, the ratio $\mathbb{P}(A(S) = h)/\mathbb{P}(A(S') = h)$ is close to 1, where probabilities are over randomization internal to algorithm A. Formally, to allow discussion of densities and real values, as well as algorithms with different kinds of outputs, we define differential privacy as follows. Let A be a probabilistic algorithm that takes in a dataset as input and produces outputs in some space $Range(A)$. E.g., if A is a learning algorithm then $Range(A) = \mathcal{H}$.

Definition 1. *A probabilistic algorithm A satisfies α-**differential privacy** if for any two samples S, S', that differ in just one example, for any set $\mathcal{O} \subseteq Range(A)$ of possible outputs of A,*

$$\mathbb{P}(A(S) \in \mathcal{O})e^{-\alpha} \;\leq\; \mathbb{P}(A(S') \in \mathcal{O}) \;\leq\; \mathbb{P}(A(S) \in \mathcal{O})e^{\alpha},$$

where note that for small α we have $e^{-\alpha} \approx 1 - \alpha$ and $e^{\alpha} \approx 1 + \alpha$.

Note that if $Range(A)$ was discrete then we could replace "for any $\mathcal{O} \subseteq Range(A)$" with "for any $h \in Range(A)$". Or if $Range(A)$ was continuous with a density then we could have required that all densities have a ratio between $e^{-\alpha}$ and e^{α}. The definition as stated allows one to capture both cases.

One way to think about this definition is to imagine some individual deciding whether to submit their correct information z_i to the data set, or instead some fake information z_i'. Differential privacy guarantees that for any event B that might occur, the probability of B occurring if they submit z_i will be close to the probability of B occurring if they submit z_i'. We can also think in terms of an observer wondering if the individual's true information is z_i or z_i'. By Bayes rule, we have:

$$\frac{\mathbb{P}(z_i|output)}{\mathbb{P}(z_i'|output)} = \frac{\mathbb{P}(output|z_i)}{\mathbb{P}(output|z_i')} \cdot \frac{\mathbb{P}(z_i)}{\mathbb{P}(z_i')},$$

meaning that if A is differentially private then the posterior ratio will be close to the prior ratio.

1.1.2 The Laplace Mechanism

One simple way to achieve differential privacy is to have an algorithm that ignores the sample S and always outputs some default value. Of course, this is useless: we want algorithms that preserve privacy *and* do something useful with the data. A basic technique for doing this is the *Laplace Mechanism*. This will be a mechanism for outputting a single number in a differentially-private manner, but then we will build on it to do more interesting tasks.

Suppose we have a set S of n inputs in the range $[0, b]$, and we want to output an estimate of their average while preserving differential privacy. For example, these could be medical test. Note that changing any one of the inputs can change the true answer by at most b/n. The Laplace mechanism computes the true average and then adds noise from a Laplace distribution as follows.

Definition 2. *Given a set S of n inputs in the range $[0, b]$, the* **Laplace mechanism** *for the average computes the true average value a and then outputs $v = a + x$ where x is drawn from the Laplace density with scale parameter $\alpha n/b$:*

$$Lap_{\alpha n/b}(x) = \frac{\alpha n}{2b} e^{-|x|\alpha n/b}.$$

Let $p_S(v)$ denote the density on output v of this mechanism with input set S.

Theorem 3. *The Laplace mechanism satisfies α-differential privacy, and moreover has the property that with probability $\geq 1 - \delta$, the error added to the true average is*

$$O\left(\frac{b}{\alpha n} \log\left(\frac{1}{\delta}\right)\right).$$

Proof. Let's first analyze privacy. To do so, we consider the maximum and minimum possible ratios $p_S(v)/p_{S'}(v)$ over v, S, S' such that S and S' differ in a single input. So, fix v, S, S'. Let a be the correct average for S and let a' be the correct average for S'. We know $a' \in [a - b/n, a + b/n]$. Therefore, to produce $v = a + x$ from S' we must add noise x' such that $a' + x' = a + x = v$, which implies $x' = x + (a - a') \in [x - b/n, x + b/n]$. This implies that the ratio $p_S(v)/p_{S'}(v)$ is between

$$\inf_x \frac{e^{-|x|\alpha n/b}}{e^{-(|x|+b/n)\alpha n/b}} = e^{-\alpha} \quad \text{and} \quad \sup_x \frac{e^{-|x|\alpha n/b}}{e^{-(|x|-b/n)\alpha n/b}} = e^{\alpha}.$$

Since the ratio of densities is between $e^{-\alpha}$ and e^{α}, if we integrate over any $\mathcal{O} \subseteq \mathbb{R}$, the ratio of probabilities of lying in $\mathcal{O}$ will also be between $e^{-\alpha}$ and e^{α}. So this mechanism satisfies α-differential privacy.

We want to argue that with high probability for the given value of n, the magnitude of the error x added to the true answer a is small. Specifically, let us solve for $\epsilon > 0$ such that

$$\mathbb{P}[x \leq -\epsilon] \leq \delta/2, \quad \text{and} \quad \mathbb{P}[x \geq \epsilon] \leq \delta/2.$$

To do so, we solve:

$$\mathbb{P}[x \geq \epsilon] = \int_{\epsilon}^{\infty} \frac{\alpha n}{2b} e^{-x\alpha n/b} dx = \frac{1}{2} e^{-\epsilon\alpha n/b}.$$

Similarly,

$$\mathbb{P}[x \leq -\epsilon] = \int_{-\infty}^{-\epsilon} \frac{\alpha n}{2b} e^{x\alpha n/b} dx = \frac{1}{2} e^{-\epsilon\alpha n/b}.$$

Setting $\frac{1}{2} e^{-\epsilon\alpha n/b} = \delta/2$ and solving for ϵ we have that with probability $\geq 1 - \delta$ the magnitide of the error x added is at most ϵ for

$$\epsilon = \frac{b}{\alpha n} \ln\left(\frac{1}{\delta}\right).$$

□

Notice that in terms of the error added, if this were a random sample of size n from some population, we would have a standard deviation on the order of $b/\sqrt{n}$ anyway, so the additional error added is fairly small.

Generalization: The Laplace mechanism is actually more general than the description above. Given any function f over a dataset of size n, let Δf denote the maximum change in f that can be produced by changing a single element in the dataset (for the average, this was b/n). Then the Laplace mechanism computes f and then adds noise x from the density

$$Lap_{\frac{\alpha}{\Delta f}}(x) = \frac{\alpha}{2\Delta f}\, e^{-|x|\alpha/(\Delta f)}.$$

The proof of privacy is exactly as above.

Composition: One further point to make: if we answer k questions about a dataset and for each one we satisfy α-differential privacy (e.g., we run k Laplace mechanisms on different features) then the overall vector of results will satisfy $k\alpha$-differential privacy. This can be seen directly from the differential privacy definition. This will be useful when we aim to simulate learning algorithms.

1.2 The Statistical Query Model

The *Statistical Query* (SQ) model is a learning model that restricts how the learning algorithm can interact with data. It was introduced by Kearns [9] in order to address the problem of learning in the presence of a type of noise called *random classification noise*. We will also see that such algorithms can be made differentially private via the Laplace mechanism. The SQ model is discussed in detail in Chapter 5 of the KV book.

1.2.1 Definitions

Let us first recall the standard distributional learning setting (e.g., lecture 2) focusing on the realizable case. There is a target function c^* and a marginal distribution $\mathcal{D}$ over the instance space

X. The algorithm requests a labeled sample S, where each example is drawn i.i.d. from $\mathcal{D}$ and labeled by c^*. The goal of the algorithm is to produce a hypothesis h of error at most ϵ with respect to c^*, where error is defined as $\mathbb{P}_{x\sim\mathcal{D}}[h(x) \neq c^*(x)]$. We say that an algorithm PAC-learns a class $\mathcal{C}$ using hypothesis class $\mathcal{H}$ if for any $\epsilon, \delta > 0$, for any distribution $\mathcal{D}$, and target function $c^* \in \mathcal{C}$, with probability $\geq 1 - \delta$ the algorithm produces a hypothesis $h \in \mathcal{H}$ of error at most ϵ. Moreover, we want time and sample size to be polynomial in $1/\epsilon$, $1/\delta$, the size d of the examples and the description length of c^*.

The Statistical Query model has the same setup as in the distributional learning setting described above. However, the algorithm cannot ask for labeled data. Instead, the algorithm can ask an SQ oracle for statistical properties of the overall distribution D_{XY} on labeled examples, which it will then receive approximate answers to. Specifically, a *statistical query* consists of a function $\phi : X \times \{-1, 1\} \to [0, 1]$ together with a *tolerance* $0 < \tau \leq 1$. The response to this query is a value $\hat{P}_\phi$ such that

$$\hat{P}_\phi \in [\mathbf{E}_{x\sim\mathcal{D}}[\phi(x, c^*(x))] - \tau, \mathbf{E}_{x\sim\mathcal{D}}[\phi(x, c^*(x))] + \tau] .$$

For example, given a hypothesis h we might define $\phi(x, y) = 1$ if $h(x) \neq y$ and $\phi(x, y) = 0$ otherwise; in this case the query is a request for the error rate of h with respect to target c^*, up to $\pm\tau$. E.g., queries of this form might be useful for an algorithm that makes small changes to its current hypothesis and then tests to see if they improve performance. Or we could ask for, say, the probability that an example is positive and has $x_i = 0$, up to tolerance $\pm\tau$, by using $\phi(x, y) = \mathbf{1}_{x_i=0\wedge y=1}$. We will see that queries of this form are useful for learning the class of conjunctions.

The idea is that given a large sample we could simulate such queries by computing the empirical average over the sample, which will be within $\pm\tau$ of the true expectation with respect to $\mathcal{D}$ with high probability, if our sample is large enough compared to $1/\tau$. However, by restricting the algorithm to only accessing its data in this way, we will also be able to do many useful things with it in a black-box manner.

We now define learnability in the SQ model formally.

Definition 4. *A class $\mathcal{C}$ over instance space X is efficiently learnable by class $\mathcal{H}$ in the SQ model if there is a learning algorithm A such that for any target concept $f \in \mathcal{C}$, and distribution $\mathcal{D}$ over X, any $\epsilon < 1/2$, we have*

1. *The algorithm makes a polynomial number of statistical queries (polynomial in the usual parameters: $1/\epsilon$, the number of features d, and the description length of the target function).*
2. *Each query uses a tolerance τ such that $1/\tau$ is also polynomially bounded.*
3. *The algorithm outputs a hypothesis $h \in \mathcal{H}$ of error at most ϵ.*
4. *The algorithm runs in polynomial time, and each query can be evaluated in polynomial time.*

1.2.2 Example: Learning Monotone Conjunctions

As an example, consider the class $\mathcal{C}$ of monotone conjunctions over $\{0, 1\}^d$, such as $x_1x_4x_5$. We can learn this class $\mathcal{C}$ (using hypotheses also in $\mathcal{C}$) in the SQ model.

Theorem 5. *The class of monotone conjunctions over $\{0, 1\}^d$ can be learned in the SQ model by the class of conjunctions using d queries of tolerance $\tau = \frac{\epsilon}{2d}$.*

Proof. Recall that in the PAC model, we learned monotone conjunctions by taking a large sample S, deleting variables set to 0 in any positive example in S, and taking the conjunction of the remainder. This produces a hypothesis that contains all variables in the target function, and so only makes one-sided error, with a low overall error rate if S is large enough.

In the SQ model, we will do something similar. For each variable x_i, note that $\mathbb{P}[x_i = 0 \wedge c^*(x) = 1]$ is an upper bound on the additional error on positive examples incurred by including x_i in our hypothesis. For all variables in the target, this probability is 0. So, if we include all variables where this quantity is 0 and don't include any variables where this is greater than ϵ/d, our overall error rate will be at most ϵ.

We can implement this in the SQ model as follows. For each feature x_i, we ask the query $\phi_i(x, y) = \mathbf{1}_{x_i=0 \wedge y=1}$ with tolerance $\tau = \frac{\epsilon}{2d}$. We then output the conjunction h of all features x_i such that $\hat{P}_{\phi_i} \leq \frac{\epsilon}{2d}$. Note that h will contain all x_i that are in the target function c^* since for those features, the true probability is 0 so they will satisfy $\hat{P}_{\phi_i} \leq \frac{\epsilon}{2d}$. Additionally, while h may contain some x_i that are not in the target function, they will have the property that $\mathbb{P}(x_i = 0 \wedge c^*(x) = 1) \leq \frac{\epsilon}{d}$. So, the total amount of error they introduce is at most ϵ by the union bound. □

A number of examples of formulating popular learning algorithms in the SQ model are given in [9, 3].

1.2.3 SQ learnable implies PAC learnable

Note the the SQ model has no "δ" parameter. However, we can still simulate an SQ algorithm in the distributional learning model by giving empirical estimates from a large sample. It's just that will some small probability, this simulation will fail. So the δ parameter comes back in the simulation. Specifically, we have:

Theorem 6. *If class $\mathcal{C}$ is is efficiently learnable by class $\mathcal{H}$ in the SQ model, then it is also efficiently learnable by $\mathcal{H}$ in the PAC model. In particular, if there is an algorithm that makes M queries of tolerance τ to learn $\mathcal{C}$ to error ϵ in the SQ model, then a sample size*

$$O\left(\frac{M}{\tau^2}\log\left(\frac{M}{\delta}\right)\right)$$

is sufficient to learn $\mathcal{C}$ to error ϵ with probability $\geq 1 - \delta$ in the PAC model.

Proof. Suppose that algorithm A learns class $\mathcal{C}$ making at most M Statistical Queries of tolerance τ. To learn in the PAC model, we request a sample S of size $O(\frac{M}{\tau^2}\log(\frac{M}{\delta}))$ and break it into M pieces $S_1, \ldots, S_M$ of size $O(\frac{1}{\tau^2}\log(\frac{M}{\delta}))$ each, using S_i to answer query i. Since each S_i is an independent sample, we can apply Hoeffding bounds. Specifically, we are estimating the average value of a $[0, 1]$-bounded random variable, so a sample of size $O(\frac{1}{\tau^2}\log(\frac{M}{\delta}))$ is sufficient to produce an estimate within $\pm\tau$ with probability $\geq 1 - \delta/M$. By the union bound, with probability $\geq 1 - \delta$ we answer all queries to within $\pm\tau$ and therefore A outputs a hypothesis of error $\leq \epsilon$ as desired. □

In fact, not only do we get PAC learning, but we also get differential privacy via the Laplace mechanism and learnability in the presence of large amounts of random classification noise.

1.3 Statistical Queries and the Laplace Mechanism for Privacy

The SQ model and the Laplace mechanism are a natural fit, allowing us to make any SQ learning algorithm differentially private. This analysis is from [2, 8].

Theorem 7. *If class $\mathcal{C}$ is is efficiently learnable by class $\mathcal{H}$ in the SQ model, then it is also efficiently learnable by $\mathcal{H}$ in the PAC model by an algorithm satisfying α-differential privacy, with time and sample size polynomial in $1/\alpha$. In particular, if there is an algorithm that makes M queries of tolerance τ to learn $\mathcal{C}$ to error ϵ in the SQ model, then a sample size*

$$O\left(\left[\frac{M}{\alpha\tau}+\frac{M}{\tau^2}\right]\log\left(\frac{M}{\delta}\right)\right)$$

is sufficient to learn $\mathcal{C}$ to error ϵ with probability $\geq 1-\delta$ in the PAC model while satisfying α-differential privacy.

Proof. The argument is similar to the proof of Theorem 6. Suppose that algorithm A learns class $\mathcal{C}$ making at most M Statistical Queries of tolerance τ. We request a sample S of size

$$O\left(\left[\frac{M}{\alpha\tau}+\frac{M}{\tau^2}\right]\log\left(\frac{2M}{\delta}\right)\right)$$

and break it into M pieces $S_1,\ldots,S_M$ of size

$$O\left(\left[\frac{1}{\alpha\tau}+\frac{1}{\tau^2}\right]\log\left(\frac{2M}{\delta}\right)\right)$$

each. We use sample S_i to answer the ith statistical query (by taking the average over the sample), and then add additional noise from the Laplace distribution with scale parameter $\alpha|S_i|/M$, giving the result to algorithm A.

Let us first analyze privacy. For each statistical query, the empirical average is an average of $|S_i|$ values in the range $[0,1]$. So, by Theorem 3 (with $n=|S_i|, b=1$) the Laplace noise provides α/M-differential privacy to each query. Using the composition property of differential privacy, this implies that over all M queries the algorithm satisfies α-differential privacy.

Now, let us analyze the error. For query i, just as in the proof of Theorem 6, $|S_i|$ is sufficiently large so that *before* adding in the Laplace noise, with probability $\geq 1-\delta/(2M)$ the sample average will be within $\tau/2$ of the true expectation for the query.

Next, by Theorem 3 with $n=|S_i|, b=1$, and "δ" $=\delta/(2M)$, with probability $\geq 1-\delta/(2M)$, the amount of noise added by the Laplace query is at most

$$O\left(\frac{1}{\alpha|S_i|}\log\left(\frac{2M}{\delta}\right)\right).$$

Using the fact that our sample size satisfies

$$|S_i| \geq \left[\frac{c}{\alpha\tau}\right]\log\left(\frac{2M}{\delta}\right),$$

for sufficiently large c, this is at most $\tau/2$.

So, with probability $\geq 1-\delta/M$, the total error for the ith query (due to both sampling and the Laplace noise) is at most τ.

So, by the union bound, with probability $\geq 1-\delta$, all queries have error at most τ. Thus with probability $\geq 1-\delta$, the algorithm produces a hypotheis of error at most ϵ, as desired. □

1.4 Random Classification Noise

The *Random Classification Noise* (RCN) model is a model for PAC learning when labels have been corrupted by random noise. Specifically, we consider learning a binary target function f and we assume that for some *noise rate* $\eta < 1/2$. each time we request a labeled example, with probability $1 - \eta$ we see $(x, c^*(x))$ and with probability η we see $(x, -c^*(x))$. In either case, x is drawn from the marginal distribution $\mathcal{D}$, and the noise process is independent of x. That is, we can imagine first drawing a labeled example $(x, c^*(x))$ and then flipping a coin of bias η such that if the coin comes up heads, the label is reversed. Let us call this distribution D^{η}_{XY}.

Our goal in the RCN model will be to produce a hypothesis h such that $d(h, c^*) \leq \epsilon$. Note that we may have $\epsilon < \eta$, thus we wish to be closer to c^* than the observed labels. Below we will see how to convert any SQ algorithm to have this guarantee.

Theorem 8. *If class $\mathcal{C}$ is is efficiently learnable by class $\mathcal{H}$ in the SQ model, then it is also efficiently learnable by $\mathcal{H}$ in the PAC model with random classification noise. In particular, if there is an algorithm that makes M queries of tolerance τ to learn $\mathcal{C}$ to error ϵ in the SQ model, then a sample size*

$$O\left(\frac{M}{\tau^2(1-2\eta)^2}\log\left(\frac{M}{\delta}\right)\right)$$

is sufficient to learn $\mathcal{C}$ to error ϵ with probability $\geq 1-\delta$ in the PAC model with random classification noise of rate η.

Proof. We will use a proof method from [1]; see Chapter 5 in the KV book for a different proof. Consider a statistical query defined by function $\phi : X \times \{-1, 1\} \to [0, 1]$ and tolerance τ. It will be convenient to rewrite $\phi(x, y)$ as follows:

$$\begin{aligned}\phi(x,y) &= \phi(x,1)\frac{1+y}{2} + \phi(x,-1)\frac{1-y}{2} \\ &= \frac{\phi(x,1)-\phi(x,-1)}{2}\cdot y + \frac{\phi(x,1)+\phi(x,-1)}{2} \qquad (1)\end{aligned}$$

We wish to estimate $\mathbf{E}_{x\sim\mathcal{D}}[\phi(x, c^*(x))]$ to tolerance τ, so it suffices to estimate the expected value of each of the two terms on the RHS of (1) to tolerance $\tau/2$, for $y = c^*(x)$.

Let us first consider the rightmost term. We wish to estimate

$$\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)+\phi(x,-1)}{2}\right].$$

This term does not depend on c^* at all, so noisy data is equivalent to non-noisy data. This means we can simply estimate the two expectations $\mathbf{E}_{x\sim\mathcal{D}}[\phi(x, 1)]$ and $\mathbf{E}_{x\sim\mathcal{D}}[\phi(x, -1)]$ to tolerance $\tau/2$, which by Hoeffding bounds can be done from a sample of size $O(\frac{1}{\tau^2}\log(\frac{M}{\delta}))$ just as in the proof of Theorem 6.

Now, let us consider the leftmost term. We wish to estimate

$$\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot c^*(x)\right],$$

given access only to data from D^{η}_{XY}. However, we can use the following fact:

$$\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot c^*(x)\right] = (1-2\eta)^{-1}\mathbf{E}_{(x,y)\sim D^\eta_{XY}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot y\right].$$

The reason is that

$$\begin{aligned}\mathbf{E}_{(x,y)\sim D^\eta_{XY}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot y\right] &= (1-\eta)\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot c^*(x)\right]\\ &+ \quad \eta\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot -c^*(x)\right]\\ &= (1-2\eta)\mathbf{E}_{x\sim\mathcal{D}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot c^*(x)\right].\end{aligned}$$

Therefore, it suffices to estimate

$$\mathbf{E}_{(x,y)\sim D^\eta_{XY}}\left[\frac{\phi(x,1)-\phi(x,-1)}{2}\cdot y\right]$$

to a tolerance of $\frac{\tau(1-2\eta)}{2}$. By Hoeffding bounds, this can be done from a sample of size

$$O\left(\frac{1}{\tau^2(1-2\eta)^2}\log\left(\frac{M}{\delta}\right)\right).$$

We use fresh samples for each estimation, and perform a total of $O(M)$ estimations, resulting in a total sample size

$$O\left(\frac{M}{\tau^2(1-2\eta)^2}\log\left(\frac{M}{\delta}\right)\right)$$

as desired. □

References

[1] M.-F. Balcan and V. Feldman. Statistical active learning algorithms for noise tolerance and differential privacy. *Algorithmica*, 2015. (special issue on Learning Theory).

[2] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the SuLQ framework. In *Proc. 24th ACM Symposium on Principles of Database Systems (PODS)*, pages 128–138, 2005.

[3] C.T. Chu, S.K. Kim, Y. Lin, Y. Yu, G. Bradski, K. Olukotun, and A.Y. Ng. Map-reduce for machine learning on multicore. In B. Schölkopf, J.C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 281–288. MIT Press, 2007.

[4] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.

[5] C. Dwork. Differential privacy: A survey of results. In *TAMC*, pages 1–19, 2008.

[6] C. Dwork. The differential privacy frontier (extended abstract). In *TCC*, pages 496–502, 2009.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. *Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284, 2006.

[8] S. Kasiviswanathan, H. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What Can We Learn Privately? In *FOCS*, pages 531–540, 2008.

[9] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998.