21 Privacy

As companies, and governments, and researchers are getting better at collecting data, what if this data is about you?! Google, Facebook, Yahoo! and most other web companies make money by understanding their customer base, and applying targeted advertising. The government would like to identify terrorists before they strike. Researchers would like to study properties of large data sets. But if *you* are the subject of the data, are you ok with your data being available to be mined?

How can we preserve the "privacy" of individual data points while also being able to mine the data?

This goal has a long history. Since privacy is hard to define, there have been many attempts which have come up short in achieving their intended goal. As researchers devise new standards by which data should be private, others find weaknesses that allow more information than one might have thought. Yet, these attempts are useful for demonstrating the problems and issues.

Ethics and Empathy. Before we get into technical definitions and examples, its worth noting that what follows will not necessary provide solutions to ethical dilemmas – just tools and examples to think about them. As data scientists, **you will at some point face an ethical dilemma!** The only mechanism to prevent bad ethical behavior is societal pressure (shame people for doing bad things, or at least let them know it is bad), and laws. The other way of course, is to hold yourselves to high ethical standards. But one may argue these standards may be evolving, and there is no way to know what the right choice is. I argue that there is a simple rule that you can follow, "if you were the data point, would you want to be treated that way." Basically don't hide behind technology to justify your ethical decisions.

21.1 Attempts to Define Privacy

What does it mean to have privacy?

Here is the prototypical problem. A company C wants to release a dataset D about its costumers habits, first to its own data scientists, but then they may want to share with researchers in universities. The goal is to develop new ways to extract information to improve business (e.g. advertising, product pricing).

So the data scientists want to compute some statistics / mine some structure from D, but not to be able to identify individual's information.

Example: health records *Consider an "anonymized" survey of hospital patients. Only provide zip code to identify location. Some have cancer. What if a CEO went for routine treatment on a treatable cancer, but was the only person in that zip code (e.g. the one with big houses ... she is a CEO after all). If someone knows the CEO was part of the study, they may know they have cancer, and it can cause the stock to plummet!*

STORY TIME:

- In 2000, Massachusettes released all stated employee's medical records in an effort for researchers to be able to study them.
- They wiped all ids, but kept zip codes, birthday, gender. Was declared anonymized by the government.
- In Massachusettes, it was possible to buy voter data for $20. It has names, birthday, zip codes, and birthdays of all voters.
- A grad student, Latanya Sweeney combined the two to identify the governor of Massachusettes. Story is, she mailed him his own health records!
- Dr. Sweeney now teaches at Harvard.

This demonstrates the danger of other data sets D' which are released or available independently of the one D you release. How can we circumvent this? There was a series of several attempts:

k-anonymity: With datasets D, one can only identify someone up to k other people, for some parameter k. That means, there must be at least k people with the same public set of tests in the same zip code / birthday / gender if D is released. Otherwise, some data is withheld.

Teacher evaluations work like this, I only see them if there are at least $k = 4$ people who respond.

ℓ-diversity: In this setting, for each of the anonymization groups (with at least k people), these exists someone from each one of ℓ-well separated classes. If you cannot identify a single person (the governor), but know they are one of k people, and *all* of those people have cancer, then you still know the governor has cancer. Instead, ℓ-diversity ensures that some person in each such set of k is in one of ℓ distinct clusters, and thus has the hidden attributes different.

t-closeness: This starts with ℓ-diversity, but also ensures that within each anonymization group, the set needs to look like the full data set D among the hidden parameters (less than distance t in Wasserstein distance).

But none of these can completely prevent corruption from outside data. You never know what other information is available.

What if height was an important quantity? (say for movie star, Silvester Stallone?)

- Information: *Sly Stallone is same height as average New Jersey man*
- Independent survey: *Average New Jersey man is 5' 8"*

This gives away Stallone's height? (Or did he have surgery to become taller ?)

STORY TIME: Netflix Prize

- In 2006, Netflix released awesome data sets $D_1 = \{\langle$user-id, movie, date of grade, grade$\rangle\}$. And another set $D_2 = \{\langle$user-id, movie, date of grade$\rangle\}$. Wants researchers to develop algorithm to predict grade on D_2. (Had another similar private data D_3 to evaluate grades – cross validation.)
- If certain improvement over Netflix's algorithm, get \$1 million!
- Led to lots of cool research!

- Raters of movies also rate on IMDB (with user id, time stamp)
- Researchers showed that by linking who rated similar sets of movies, with similar scores and times, they could identify many people.
- (maybe watched embarrassing films on Netflix, not listed on IMDB)
- Class action lawsuit filed (lated dropped) against Netflix.
- Netflix Prize had proposed sequel, dropped in 2010 for more privacy concerns.

21.2 Differential Privacy

These scenarios led to a formal approach to release data, called *differential privacy*.

The goal is to take a data set X and create a data structure S which represents the data through a mechanism M. So $M(X) \to S$, this mechanism is often random. We can then make queries on this data structure $q(X)$ in the same way (or similar ways) we would do to the data set.

To define differential privacy, we need to consider two very similar data sets X_1 and X_2. These should only differ in one data point x_i, e.g., the data point that contains *our* data. So in particular $|X_1 \triangle X_2|$

Now we say that a mechanism is *ε-differentially private* (or ε-DP) if for *any* query q and any range R over the outputs, the following holds:

$$\frac{\mathbf{Pr}[q(S_1) \in R]}{\mathbf{Pr}[q(S_2) \in R]} \leq \exp(\varepsilon) \approx 1 + \varepsilon.$$

If its easier to intuitively think about just this condition for any value r

$$\frac{\mathbf{Pr}[q(S_1) = r]}{\mathbf{Pr}[q(S_2) = r]} \leq \exp(\varepsilon) \approx 1 + \varepsilon.$$

its probably ok, except one needs to be careful about using likelihood, since we'll be working with continuous random values, and the "probability" is 0. But the ratio cancels out the normalizing factors anyway.

OK, so what does this mean? That is, if we observe a value r on a query, then which data set X_1 or X_2 is more likely. If the ratio is 2, this means that between the two options for X_1 or X_2, it is twice as likely to be X_1 than X_2. If the ratio is even larger, say 99, then between the two, we can think its X_1 with 0.99 probability. And we can suppose our data point x_i is in the data set. On the other hand, if $\varepsilon = 0.1$ so the ratio is 1.01, then we are only slightly more likely to think it is X_1 (and out data point is included), but are not very confident.

The Laplacian Mechanism. The simplest way to achieve ε-differential privacy is to add appropriate Laplacian noise to each data element. The Laplacian distribution is $\mathsf{Lap}(x; w) = \frac{1}{2w} \exp(-|x|/w)$. When discussing Laplacian noise $\mathsf{Lap}(w)$ we draw $x \sim \mathsf{Lap}(x; w)$.

To achieve ε-differential privacy on a set $X \subset \mathbb{R}$, we can let $S = \{s_1, s_2, \ldots, s_n\} \leftarrow M(X)$ where $s_i = x_i + \mathsf{Lap}(1/\varepsilon)$. That is, each data point x_i is given independent Laplacian noise with bandwidth $w = 1/\varepsilon$. Lets see this in a few examples.

Height example. Lets say the dataset X is a single data point $x = 68$ (e.g., the supposed true height of Sylvester Stallone in inches), and we consider another data set X' which is a single data point $x' = 69$ (listed height on Wikipedia). Now consider these two hypothetical data sets, passed through the Laplacian mechanism with $\mathsf{Lap}(w = 1/\varepsilon)$ noise to get s and s', respectively.

Lets evaluate how this holds up; what is the probability that $r = 70$.

$$\frac{\mathbf{Pr}[s = 70]}{\mathbf{Pr}[s' = 70]} = \frac{\frac{1}{2w} \exp(|68 - 70|/w)}{\frac{1}{2w} \exp(|69 - 70|/w)} = \exp(\frac{1}{w}(|68 - 70| - |69 - 70|)) = \exp(\frac{1}{w}) = \exp(\varepsilon).$$

Reviewing this calculation, this $\varepsilon(\varepsilon)$ holds up for any r and for any two values values x and x' such that $|x - x'| = 1$.

That is, if we know that we applied a $\mathsf{Lap}(1/\varepsilon)$-mechanism, we only believe with confidence $(1+\varepsilon)/(2+\varepsilon)$ what was the original value.

Binary example. Another example data set is just a bit $X = \{x\}$ where $x \in \{0, 1\}$. We want to know if something is there or not there. So if $x = 1$, then our alternative data set is $X' = \{x'\}$ with $x' = 0$. Again, we apply the Laplacian mechanism (adding noise $\mathsf{Lap}(1/\varepsilon)$) and return a value $s \in \mathbb{R}$ (it may be negative. Now lets check what confidence is if the item is there $r = 1$.

$$\frac{\mathbf{Pr}[s = 1]}{\mathbf{Pr}[s' = 1]} = \frac{\exp(|1 - 1|/(1/\varepsilon))}{\exp(|0 - 1|/(1/\varepsilon))} = \exp(\varepsilon(|0| - |1|)) = \exp(\varepsilon).$$

Binary database example. Now lets consider that we have a larger data set $X = \{x_1, x_2, \ldots, x_n\}$, where each $x_j = \{0, 1\}$ indicating if the jth person has cancer ($x_j = 1$) or not ($x_j = 0$). We apply the Laplacian mechanism on each point to get $S = \{s_j = x_i + \mathsf{Lap}(1/\varepsilon)\}$. If there query is does a certain person have cancer, the calculation is the same as in the binary example.

But now we can ask more complicated queries. We can ask in an interval range $I = [a, b]$ so $I \cap X = \{x_a, x_{a+1}, \ldots, x_n\}$, how many people in this interval range have cancer $q(X) = \sum_{j \in I} x_i$.

Only asking the query on the data structure S, we could just return $q_I(S) = |I \cap X| + \mathsf{Lap}(1/\varepsilon)$. On each query, if there is another database X' that swaps one bit x_i, then by the same calculation

$$\frac{\mathbf{Pr}[\sum_{i \in I} s_i = r]}{\mathbf{Pr}[\sum_{i \in I} s'_i = r]} = \frac{\exp(|\sum_{i \in I} x_i - r|\varepsilon)}{\exp(|\sum_{i \in I} x'_i - r|\varepsilon)} = \exp(|\sum_{i \in I} x_i - r|\varepsilon - |\sum_{i \in I} x'_i - r|\varepsilon) \leq \exp(\varepsilon(|\sum_{i \in I} x_i - \sum_{i \in I} x'_i|) \leq \exp(\varepsilon).$$

But we must store this noise, since if it was independently applied to each instance of the query, they could ask over and over again, and take the average.

Alternatively, they could combine queries so $q_I(S) = \sum_{i \in I} q(i)$. We need to be able to apply $\mathsf{Lap}(1/\varepsilon)$ to each query, otherwise they could use just the result to $q(i)$ to determine x_i. But then applying $\mathsf{Lap}(1/\varepsilon)$ to each x_i, the answer becomes

$$q_I(S) = \sum_{i \in I} (x_i + \mathsf{Lap}(1/\varepsilon)) \in q(X) + |I| \exp(\varepsilon),$$

which is $|I|$ times more error than we would like. If the interval is size $|I| = n$ (or nearly as large), then this would give $n \exp(\varepsilon)$ error. Its not quite so bad since the error concentrates, so using linearity of variance, we can argue that the total error acts like $\sqrt{n} \exp(\varepsilon)$, but still fairly large.

Can we achieve much less dependence on n for interval queries, while still achieving ε-DP? It turns out, Yes!

Build a binary tree on X, so each node j of the binary tree corresponds to an interval I_j query. For each node we can store additional data $s_j = q_{I_j} + \mathsf{Lap}(1/\varepsilon)$. This is still linear space in n. Now any query interval I can be composed as the disjoint union of at most $2 \log n$ disjoint intervals $J = \{I_j\}$ defined by subtrees. We can then use the *data structure* S to compose queries as

$$q_I(S) = \sum_{I_j \in J} q(s_i) = \sum_{I_j \in J} [|I_j \cap x| + \mathsf{Lap}(1/\varepsilon)] = |I \cap x| + \sqrt{|J|}\mathsf{Lap}(1/\varepsilon) \leq \sqrt{2 \log n} \exp(\varepsilon),$$

and this still achieves ε-DP on each interval query.

Big Picture. The final picture should be clear.

More noise implies that data is more private, but less informative.

The underlying hypothesis, is that if you are trying to learn a model, you should want your model to be robust. If it is robust, then adding removing a small amount of low-level noise should hopefully not affect the result. Then you should be able to apply differential privacy to hide individual elements without spoiling the global analysis.

It is currently the subject of active research to understand the effective of this trade-off, and when and how this can be made practical and useful.