Machine learning in science and society

From automated science to beneficial artificial intelligence

Christos Dimitrakakis

September 23, 2022

Contents

Chapter 2

Privacy

One interpretation of privacy is simply the ability to maintain a personal secret. Is it possible to maintain perfect secrecy? Can the thoughts in our head be perfectly safe, or can they be revealed indirectly through our actions?

While we certainly *can* securely store and transmit data using cryptography, the problems we will discuss in this chapter are *not* solvable solely through cryptography. We are interested in scenarios where we must make a public decision or release some summary statistics that depend on private data, in such a way as to minimise harm to the individuals contributing their data.

If we never have to reveal any of our computations, cryptography is what is needed. For example, through homomorphic computation, an untrusted party can even perform some computations on encrypted data, returning an encrypted result to us, while learning nothing about the original secret. As long as the data, and the results of any computation on it, are kept under lock and key, our personal information cannot be revealed.

However, sometimes we must make public decisions or release public information based on this data. Then it can be revealed indirectly. For example, you can trust your doctor to maintain confidentiality, but when you go to the pharmacy to get the prescribed medicine, somebody can infer the medical condition you suffer from.

It is even possible to learn personal information from aggregate statistics. Let us say your doctor publishes a list of cases of different diseases every week, together with some other information such as the approximate patient age. Even though your own data is mixed with that of all other patients, it is possible to infer your diagnosis, especially with some additional side-information: If somebody knows you were the only person in that age group visiting the doctor that week, they will learn your diagnosis.

From that point of view, it is not the data itself, but the complete process of data collection, treatment and public release that can be characterised as private or non-private. For that reason, we will emphasise an *algorithmic* view of privacy: participants entrusts their data to an algorithm, which produces a useful output in return. The algorithmic process is typically not fully automated, as it also depends on some human input: One example is a medical study examining different treatments for a disease. While humans select and administer the treatments, they will typically rely on a randomised strategy for assigning treatments to individuals, and use a statistical method to report their results. Given this mixture of ad-hoc decisions and formal algorithmic methods, is it possible to guarantee privacy in any sense? What kind of guarantees can we make?

Generally speaking, an algorithm has good privacy properties, if the amount of information that can be revealed through the algorithm's output about any individual contributing data

ALL I WANT IS A SOCIAL NETWORK SMART ENOUGH TO PUT A FOOD ITEM INTO MY MOUTH JUST AS I REALIZE THAT I WANT IT, BUT WHICH ALSO DOESN'T COLLECT ANY PERSONAL DATA! IS THAT TOO MUCH TO ASK?

Just because they're the problem, doesn't mean we aren't.

is bounded. In particular, we are interested in how much an adversary can learn about any individual's input to the algorithm from the algorithm's output.

This does not preclude learning general facts about individuals from the output. For example, a study about the use of steroids in sports may show that 90% of sprinters with times under 10 seconds are using steroids, while only 50% of slower sprinters do so. Any sprinter with a time under 10 seconds is thus suspected of using steroids by association. However, it does not matter if their data has been used in the study. The publication of the result does not impact their privacy, but the amount of harm it does to them does not depend in their participation: the same statistical result would have been obtained with or without them.

In this chapter, we will look at two formal concepts of privacy protection: k-anonymity and *differential privacy.* The first is a simple method for anonymising databases. However, it provides only limited resistance to identification. The latter is a more general concept, which provides full information-theoretic protection to individuals. A major problem with any privacy definition and method, however is correct interpretation of the privacy concept used, and correct implementation of the algorithm used.

2.1 Things we do with data

How do we use data in the first place? Sometimes we simply publish the data, perhaps after some initial processing. Publication of datasets is useful for researchers that want to do perform further analysis on the data. Usually though, we collect data in order to calculate specific statistics. For example the census collects data about the number of people in different households, wages, etc, and then publishes tables detailing the average age and number of people per household in different areas of the country. This demographic information is useful for policy makers, urban planning to organise voting centers and the distribution of police and fire stations, as well as other public services.

Fundamentally, privacy in statistics is an *issue of trust.* The analyst, be it a human, or an automated service, will use your data to make decisions. You must also decide who to trust and how much. Do we trust the data analyst? How much privacy are we willing to sacrifice to the analyst? How much to the public at large? What you want out of the service. Is the service important enough to sacrifice significant amounts of privacy? What is an acceptable trade-off between utility and privacy? These are difficult questions and are hard to quantify, hence we assume that we have already decided how to answer them, and we simply want to find an appropriate methodology for achieving a good result.

Consider a researcher wishing to collect data for a statistical analysis. If the analysis is eventually published,[1] this creates two possible scenarios for that may lead to privacy violations.

- Publication of "anonymised" data. Sometimes we may collect data in order to publish the dataset itself for other researchers to use. This is common practice in machine learning, with image classification datasets being a good example. However, there is always some privacy risk even if identifying information such as names are removed.

- Public data analysis. In this setting, we only publish summary statistics or models about the data. A prime example is a national census analysis, which may provide detailed demographic information for all towns and regions in a nation. Although only aggregate data is published, it is theoretically possible to infer personal information. For that reason, the US Census Bureau conducted its analysis in 2020 using differential private algorithms.

⚠ **Cryptography is not enough.**

Cryptography provides secure communication and computation, authentication and verification. These are used to establish secure channels with somebody that we trust. However, the privacy violations we are concerned with relate to *publicly released* outputs of algorithms. It is not important whether or not all the data and computation are encrypted: as long as the algorithm generates a public output, it is theoretically possible for somebody to learn something about the algorithm's input.

2.2 Statistical disclosure

"Data is everywhere", said the statistician, "data, data!".

In the past, statistical analysis was performed with laboriously collected and annotated datasets. Even as recently as in the early 21st century, databases for machine learning were limited to a few thousand entries at most. At the time of writing, not only has the size of datasets become extremely large, but the sources of data are much more diverse. Data are collected and commercialised whenever we visit a website and even as we walk around with our phone. To a limited extent, there is a tradeoff between what we can get out of a service and what we pay into it. Many free services such as navigation software rely on collecting user data to perform better: If you can tell that there is a traffic jam in Central Avenue, you can after all try and take another route. As long as informed consent exists, use of private data is generally regarded as unproblematic.[2]

However, even apparently benign data collected with appropriate consent can lead to serious and unexpected privacy violations. There are three famous examples of this: Firstly, the identification of people in supposedly anonymous health data in the 1990s in the state of Massachussets, which we will go over in detail in this chapter. Secondly, the identifications of users through anonymised movie ratings in the Netflix dataset. Finally, the ability to discover if any given individual's data is contained in a pooled genomic study.

Anonymisation

[1]If somebody knows that the analysis is being conducted, however, they could still learn something private from the fact that the analysis has *not* been published.

[2]This is actually underscored by the GDPR legislation, which focuses on consent and data use methods.

Data is collected for many reasons. Any typical service you might want to use will require a minimal amount of data. For example, a dating service will at a minimum require your age and location, as shown in the example below.

Birthday	Name	Height	Weight	Age	Postcode	Profession
06/07	Li Pu	190	80	60-70	1001	Politician
06/14	Sara Lee	185	110	70+	1001	Rentier
01/01	A. B. Student	170	70	40-60	6732	Time Traveller

Example 1 (Typical relational database in a dating website.). If somebody is a user in a dating website, you expect them to give some minimal personal information to be stored in the site's database. This might include their birthday, location and profession, for example.

When you submit your data to a service, you expect it to be used responsibly. For the dating service, you expect it to use the information to find good matches, based on your preferences and location. Whenever somebody uses the service, they obtain some information about you, at least indirectly. For example, if they make a query for similar singles in their neighbourhood, and they see your profile, then they have learned that you live nearby.

If we wish to publish a database, frequently we need to protect the identities of the people involved. A simple idea is to erase directly identifying information. However, this does not really work most of the time, especially since attackers can have side-information that can reveal the identities of individuals in the original data, when combined with the published dataset.

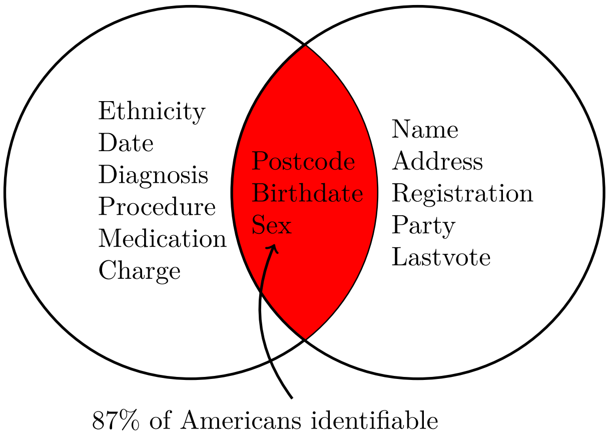
The simple act of hiding or using random identifiers is called anonymisation. However this is generally insufficient as other identifying information may be used to re-identify individuals in the data. In particular, even if somebody is unable to infer the information of any individual from the published, anonymised, dataset, they may be able to do so via some side-information. All that is needed is another dataset with some columns in common with the dataset they want to target.

Record linkage

As an example, in the 1990s a the governor of Massachussets, decided to publish anonymised information about the health records of individual state employs. They were careful to hide all obviously identifying information such as their name, and thus claimed that there are no potential privacy violations. However, they left in some information that they thought would be useful for researchers: the postcode, the birthdate, and the sex of each individual.

This allowed a PhD student, McSweeney, to perform the following linkage attack. She order a floppy disk containing database of voting records in the state. This contained names and addresses, as well as the postcode, birth date and sex of every voter. In that way, she was able to cross-reference this data, and so obtain the identities of individuals in that database. The first record she obtained was that of the governor himself, Bill Weld. Later, she estimated that approximately 87% of Americans are uniquely identifiable through those three attributes.

Clearly, anonymisation is not enough. So, is there a way to formally guarantee privacy? In the next section we will go over the solution of McSweeny, which provides us with an algorithmic definition of anonymity. While this provides some degree of protection against certain linkage attacks, it is sadly insufficient to protect privacy in general. Later, we will introduce the notion of differential privacy, which protects individuals against statistical disclosure in an information-theoretic manner.



(a) Voter registration

(b) Bill Weld

Figure 2.1: Linkage attack on a anonymised patient information database. A misguided attempt to release medical information as public data (left circle in figure (a)) by then-governor Bill Weld resulted in a simple linkage attack. This was done by accessing a party registration database (right circle in figure (b)) which contained three common fields with the medical database.

2.3 Algorithmic privacy.

The above discussion should help emphasise that *a dataset* cannot be characterised as private or non-private. We should actually start from the observation that *any data contributed by an individual* should be considered private, because it could be combined with other datasets to obtain sensitive personal information.

For that reason, we wish to ensure that, whenever individuals contribute data for processing, the result of the processing cannot be used to reveal whatever information they had contributed. Thus, the idea of privacy only applies on *algorithms* that are used on personal data. Generally speaking, we wish for algorithms to both be useful and have good privacy properties. Unfortunately, as we shall see in the sequel, there is a distinct trade-off between privacy and utility. However, let us specify more precisely what we mean by an algorithm.

What is an algorithm?

Any functional process is an algorithm. We typically identify inputs with observations or features $\mathcal{X}$, and outputs with actions $\mathcal{A}$ by the algorithm: i.e. we view the algorithm as taking actions that have an observable effect on its external environment. We generally consider *stochastic* algorithms.

Definition 2.3.1 (Stochastic algorithm π)**.** A stochastic algorithm π with input domain $\mathcal{X}$ and output domain $\mathcal{A}$ is a mapping $\pi : \mathcal{X} \to \boldsymbol{\Delta}(\mathcal{A})$ from observations $\mathcal{X}$ to distributions over outputs $\boldsymbol{\Delta}(\mathcal{A})$. When $\mathcal{A}$ is finite, we will write $\pi(a \mid x)$ to denote the conditional probability that the algorithm outputs a given input x.

The general case.

It is best to think of $\pi(\cdot|x)$ as a probability measure over $\mathcal{A}$. Then we write

$$\pi(A \mid x) \triangleq \mathbb{P}_\pi(a \in A \mid x), \qquad A \subset \mathcal{A},$$

for the conditional probability that algorithm's output is in some set $A \subset \mathcal{A}$ given input x. This allows us to treat the case when $\mathcal{A}$ is continuous or discrete with the same notation.

Algorithmic randomness.

We can construct a random algorithm through access to a random coin ω taking values in Ω. We can then define the output through a deterministic function $a_\pi(\omega, x)$. Since ω is random, the output of the function is also random. If P is the probability distribution of ω, then:

$$\pi(A \mid x) = P(\{\omega \in \Omega \mid a_\pi(\omega, x) \in A\}),$$

i.e. the probability that the algorithm's output is in A is equal to the measure of the values of ω for which the $a_\pi(\omega, x) \in A$.

What is private?

2.4 k-anonymity

k-anonymity

(a) Samarati

(b) Sweeney

The concept of k-anonymity was introduced by **?** and provides good guarantees against inferring personal information from a single database. This requires the analyst to first determine the variables of interest, and then determine which variables could be potentially used to identify somebody in the database. *It's the analyst's job to define quasi-identifiers.*

Definition 2.4.1 (k-anonymity)**.** A database provides k-anonymity if for every person in the database is indistinguishable from $k-1$ persons with respect to *quasi-identifiers*.

This hope is that, if the database satisfies k-anonymity it can be safely released, without revealing any private information directly. Let us now walk through an extended example.

k-anonymity example

In particular, let us say that the analyst simply wants to calculate some statistics about how different professions correlate with age, weight, height and where people live. Some areas of the country might produce more politicians, for example. And taller people may be more successful in politics. The initial data collected might look like the table below. It was obvious to the analyst, that even if he did remove all the names, somebody knowing where A. B. Student lived and saw the table would have little trouble recognising them.

Birthday	Name	Height	Weight	Age	Postcode	Profession
06/07	Li Pu	190	80	65	1001	Politician
06/14	Sara Lee	185	110	67	1001	Rentier
06/12	Nikos Karavas	180	82	72+	1243	Politician
01/01	A. B. Student	170	70	52	6732	Time Traveller
05/08	Li Yang	175	72	35	6910	Politician

Table 2.1: 1-anonymity.

After thinking about it for a bit, the analyst decides to remove the birthday and name, and broadly categorise people according to their height in increments of 10cm, the weight in increments of 20cm, and keep just the first digit of the postcode. Now that looked much more reasonable. Still, somebody that knows that Li Yang and A. B. Student are in the table, as well as their postcodes, and they also know that Li Yang is a politician, can infer that A. B. Student is a Time Traveller.

Height	Weight	Age	Postcode	Profession
180-190	80+	60+	1*	Politician
180-190	80+	60+	1*	Rentier
180-190	80+	60+	1*	Politician
170-180	60-80	20-60	6*	Time Traveller
170-180	60-80	20-60	6*	Politician

Table 2.2: 2-anonymity: the database can be partitioned in sets of at least 2 records

However, with enough information, somebody may still be able to infer something about the individuals. In the example above, it remains true that if somebody knows that both A. B. Student and Li Yang are in the database, as well as their postcodes, as well as that Li Yang is a politician, they can infer A. B. Student's profession. Fortunately, there is a way to protect individual information from adversaries with arbitrary side-information. This is given by differential privacy.

2.5 Differential privacy

This section introduces one of the main tools for giving formal guarantees about the privacy of any algorithm ran on a dataset, *differential privacy.* This will provide individual-level privacy, in the sense that an algorithm that is differentially private guarantees that no adversary can significantly increase their knowledge about any particular individual by observing the algorithm's output. This is independent of the adversary's existing knowledge, or computational power.

While k-anonymity can protect against specific re-identification attacks when used with care, it says little about what to do when the adversary has a lot of knowledge. For example, if the adversary knows the data of everybody that has participated in the database, it is trivial for them to infer what our own data is. For some particularly sensitive datasets, we may want for the adversary to be unable to tell whether or not your data was part of the base. Differential privacy offers protection against adversaries with unlimited side-information or computational power. Informally, an algorithmic computation is differentially-private if an adversary cannot distinguish two "neighbouring" database based on the result of the computation. Informally, two databases are neighbours when they are identical apart from the data of one person. A differentially private algorithm, because of its randomness, makes it impossible for somebody to tell from the algorithm's output whether any specific individual's data was in the database.

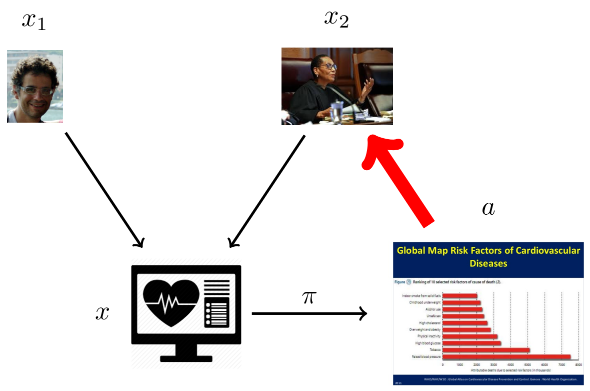


Figure 2.3: If two people contribute their data $x = (x_1, x_2)$ to a medical database, and an algorithm π computes some public output a from x, then it should be hard infer anything about the data from the public output.

Consider the example given in Figure 2.3, where two people contribute their data to a medical database. The i-th individual contributes data x_i, and the complete dataset is $x = (x_1, x_2)$. The algorithm π defines a distribution over the set of possible outputs $a \in \mathcal{A}$, with $\pi(a|x)$ being the probability that the algorithm outputs a if the data is x. If the algorith was deterministic, then it might be possible for an adversary to invert the computation and obtain x from a. But even if that is not possible, maybe they can learn *something* about the data from the output. In the section below, we will formalise this notion.

Privacy desiderata

Consider a scenario where n persons give their data $x_1, \ldots, x_n$ to an analyst. This analyst then performs some calculation on the data and publishes the result through a randomised algorithm π, where for any output a, and any dataset x, $\pi(a|x)$ is the probability that the algorithm generates a. The following properties are desirable from a general standpoint.

Anonymity. Individual participation in the study remains a secret. From the release of the calculations results, nobody can significantly increase their probability of identifying an individual in the database.

Secrecy. The data of individuals is not revealed. The release does not significantly increase the probability of inferring individual's information x_i.

Side-information. Even if an adversary has arbitrary side-information, he cannot use that to amplify the amount of knowledge he would have obtained from the release.

Utility. The released result has, with high probability, only a small error relative to a calculation that does not attempt to safeguard privacy.

The prevalence of drugs in sport

Let's say you need to perform a statistical analysis of the drug-use habits of athletes. Obviously, even if you promise the athlete not to reveal their information, you still might not convince them. Yet, you'd like them to be truthful. The trick is to allow them to randomly change their answers, so that you can't be *sure* if they take drugs, no matter what they answer.

Algorithm for randomising responses about drug use

1. Flip a coin.
2. If it comes heads, respond truthfully.
3. Otherwise, flip another coin and respond `yes` if it comes heads and `no` otherwise.

Calculating the true rate of responses.

Assume that the observed rate of positive responses in a sample is p, that everybody follows the protocol, and the coin is fair. Then, what is the true rate q of drug use in the population?

The problem with this approach, of course, is that we are effectively throwing away half of our data sources. In particular, if we repeated the experiment with a coin that came heads at a rate ϵ, then our error bounds would scale as $O(1/\sqrt{\epsilon n})$ for n data points.

This algorithm is very specific: it assumes binary responses, and it uses a fair coin, which introduces a lot of noise. Since the coin flips make the responses noisy, we may want to have some way of controlling it. In general, we want to consider an algorithm that takes data $x_1, \ldots, x_n$ from n users transforms it randomly to $a_1, \ldots, a_n$ using the following mapping.

The binary randomised response mechanism

Definition 2.5.1 (Randomised response)**.** The i-th user, whose data is $x_i \in \{0, 1\}$, responds with $a_i \in \{0, 1\}$ with probability

$$\pi(a_i = j \mid x_i = k) = p, \qquad \pi(a_i = k \mid x_i = k) = 1 - p,$$

where $j \neq k$.

Given the complete data x, the algorithm's output is $a = (a_1, \ldots, a_n)$. Since the algorithm independently calculates a new value for each data entry, the output probability is

$$\pi(a \mid x) = \prod_i \pi(a_i \mid x_i)$$

This mechanism satisfies the formal notion of ϵ-differential privacy, which will be given in Definition 2.5.2. In a more general setting, we may have multiple possible responses. While the algorithm can be trivially generalised to n-ary outputs, the special case of when the outputs are integers is deferred until later.

The original randomised response mechanism?

As first proposed by **?**, the mechanism distributes spinners to people. The spinner has a probability p of landing on A and $1 - p$ of landing on B. This can be easily arranged by having the corresponding areas to have the appropriate proportions. The interesting thing about this mechanism is that the responders must merely say whether or not the spinner landed on the group they identify with. They do *not* have to reveal a group. This makes the mechanism feel like you are revealing less, even if it is just a special case of a general randomised response mechanism.

What can we learn from the output?

In the Bayesian setting, we can think of the adversary as having some prior belief $\beta(x_i)$, expressed as a probability distribution over the secret value of each individual.

After the adversary observes the output, they can form a posterior belief $\beta(x_i \mid a_i)$, representing the information they collected. The following example quantifies the amount of information gained by the adversary.

Example 2. For simplicity, consider only one individual, and let the adversary have a prior $\beta(x = 0) = 1 - \beta(x = 1)$ over the values of the true response of an individual. We use the randomised response mechanism with parameter p and the adversary observes the randomised data $a = 1$ for that individual, then what is $\mathbb{P}^\pi_\beta(x = 1 \mid a = 1)$, assuming the adversary knows the mechanism?

Proof. Bayes's theorem states that[3]

$$\mathbb{P}^\pi_\beta(x \mid a) = \pi(a \mid x)\beta(x)/\,\mathbb{P}^\pi_\beta(a),$$

where

$$\mathbb{P}^\pi_\beta(a) = \sum_{x'} \pi(a \mid x')\beta(x').$$

Let $q = \beta(x = 1)$. Then we have:

$$\mathbb{P}^\pi_\beta(x = 1 \mid a = 1) = pq/[pq + (1 - p)(1 - q)].$$

It is particularly interesting to consider the case where $q = 1/2$. Then

$$\mathbb{P}^\pi_\beta(x = 1 \mid a = 1) = p,$$

so we only have limited evidence for whether $x = 1$. Now consider the case where we have some arbitrary prior q, and $p = 1/2$. This means that the output of the algorithm is completely random. Consequently:

$$\beta(x = 1 \mid a = 1) = q = \beta(x = 1).$$

So, in this scenario we learn nothing from the algorithm's output. □

[3]If there are too many symbols for you, you can write Bayes theorem simply with $P(x|a) = P(a|x)P(x)/P(a)$.

Differential privacy.

Now let us take a look at a way to characterise the the inherent privacy properties of algorithms. This is called differential privacy, and it can be seen as a bound on the information an adversary with arbitrary power or side-information could extract from the result of a computation π on the data. For reasons that will be made clear later, this computation has to be stochastic.

☕ ϵ-Differential Privacy

Definition 2.5.2. A stochastic algorithm $\pi : \mathcal{X} \to \boldsymbol{\Delta}(\mathcal{A})$, where $\mathcal{X}$ is endowed with a neighbourhood relation N, is said to be ϵ-differentially private if

$$\left|\ln \frac{\pi(A \mid x)}{\pi(A \mid x')}\right| \leq \epsilon, \qquad \forall x N x', \quad A \subset \mathcal{A}. \tag{2.5.1}$$

This form is related to standard notions of statistical distance such as the KL divergence $\sum_a \ln \frac{\pi(a|x)}{\pi(a|x')} \pi(a \mid x)$. However, the above inequality can be equivalently rewritten as

$$\pi(A \mid x) \leq e^{\epsilon} \pi(A \mid x').$$

Frequently (and particularly when $\mathcal{A}$ is finite) we can also use $\pi(a \mid x)$ without any technical difficulties.

Typically, algorithms are applied to datasets $x = (x_1, \ldots, x_n)$ composed of the data of n individuals. Thus, all privacy guarantees relate to the data contributed by these individuals.

Neighbourhoods

Differential privacy guarantees that it is hard to distinguish neighbouring datasets. Hence, the definition of neighbourhood we use reflects what we want to protect. It makes sense to define neighbourhoods in terms of changes in one individual's data, because then an adversary cannot learn about the values of any particular individual.

In this book we will use two definitions of neighbourhoods. The first is constructed so that the adversary cannot distinguish whether or not any particular individual's information is in the dataset. If not, then they cannot infer the individual's presence from the output of the algorithm.

Definition 2.5.3 (Addition/deletion neighbourhood)**.** If two datasets x, x' are neighbours, then we write xNx', and it holds that

$$x = (x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n), \qquad x' = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n),$$

for some i, i.e. if one dataset is missing an element.

This is an important concept if the participation in the dataset is itself sensitive. For example, if somebody is enlisted in study for experimental treatment of a rare disease, then the mere fact that they are part of the study is strong evidence that they have the disease.

The second definition is slightly weaker. It determines whether or not we can distinguish between different *values* of individual data. If not, then they cannot infer whether the data submitted by the individual has a particular value.

Definition 2.5.4 (Edit neighbourhood)**.** We say that two datasets x, x' are neighbours, and write xN_ex' if

$$x = (x_1, \dots, x_i, \dots, x_n), \qquad x' = (x_1, \dots, x'_i, \dots, x_n), \qquad x_i \neq x'_i.$$

i.e. if one dataset has an altered element.

If x, x' are 1-neighbours under the second definition, then they are 2-neighbours under the first definition. To see this, create a new dataset $\hat{x}$ from x, without the data of person i. Then $xN\hat{x}$ and $\hat{x}Nx'$, since we can change the data of one person by removing the original data x_i and then re-inserting the altered data x'_i. This is illustrated in the example below.

Neihgbourhood example

In this example, we have three datasets, x, x' and $hatx$. In the second dataset, $\hat{x}$, the highlighted row in x is missing. In the third dataset, x', another row with the same name is added, but the height and weight are changed.

Birthday	Name	Height	Weight
06/07	Li Pu	190	80
06/14	Sara Lee	185	110
06/12	*John Smith*	*170*	*82*
01/01	A. B. Student	170	70
05/08	Li Yang	175	72

Table 2.3: Data x

In fact, $\hat{x}$ is obtained through a deletion from x. Of course, the operation can be reversed: x can be obtained from an addition to $\hat{x}$.

Birthday	Name	Height	Weight
06/07	Li Pu	190	80
06/14	Sara Lee	185	110
01/01	A. B. Student	170	70
05/08	Li Yang	175	72

Table 2.4: $\hat{x}$, 1-Neighbour of x.

We can now instead add another row to $\hat{x}$, which will be similar to the row we had removed. This will have the effect of altering the original data in x.

Birthday	Name	Height	Weight
06/07	Li Pu	190	80
06/14	Sara Lee	185	110
06/12	John Smith	*180*	*80*
01/01	A. B. Student	170	70
05/08	Li Yang	175	72

Table 2.5: x', 2-Neighbour of x.

Since x, x' only differ in the contents of a single row, they are edit-neighbours.

As we hinted earlier, the randomised response mechanism satisfies ϵ-DP. The ϵ parameter is dependent on p, with higher values giving a smaller ϵ, and thus better privacy protection.

Remark 2.5.1. The randomised response mechanism with $p \leq 1/2$ is $(\ln \frac{1-p}{p})$-DP with respect to the edit neighbourhood N_e.

Proof. Consider $x = (x_1, \ldots, x_j, \ldots, x_n)$, $x' = (x_1, \ldots, x'_j, \ldots, x_n)$. Then

$$\begin{aligned}
\pi(a \mid x) &= \prod_i \pi(a_i \mid x_i) \\
&= \pi(a_j \mid x_j) \prod_{i \neq j} \pi(a_i \mid x_i) \\
&\leq \frac{1-p}{p} \pi(a_j \mid x'_j) \prod_{i \neq j} \pi(a_i \mid x_i) \\
&= \frac{1-p}{p} \pi(a \mid x')
\end{aligned}$$

$\pi(a_j = k \mid x_j = k) = 1 - p$ so the ratio is $\max\{(1-p)/p, p/(1-p)\} \leq (1-p)/p$ for $p \leq 1/2$. □

Moving to a new neighbourhood

Is the randomised-response mechanism for it to be differentially private with respect to the insertion-neighbourhood definition? If not, is it possible to modify it in order to satisfy that privacy definition? *Hint: The mechanism must be able to hide the participation of a single individual in the database.*

Finally, it may be convenient to the look at neighbourhoods in terms of a distance between daasets. Let $\mathbb{N}^{|\mathcal{X}|}$ be the set of all possible dataset histograms, i.e. counts of different possible rows in each dataset. Then the distance between two datasets is simply the total difference in counts:

Definition 2.5.5 (Hamming distance between datasets)**.**

$$\|\boldsymbol{x} - \boldsymbol{x}'\|_1 = \sum_j | \sum_{i=1}^{n} \mathbb{I}\{x_i = j\} - \sum_{i=1}^{n'} \mathbb{I}\{x_i = j\} | = \sum_j |N_j(\boldsymbol{x}) - N_j(\boldsymbol{x}')|, \tag{2.5.2}$$

where $N_j(\boldsymbol{x})$ is the number of elements in $\boldsymbol{x}$ equal to j.

In this definition, if $\boldsymbol{x}'$ has one less element than $\boldsymbol{x}$, then there is some j for which $N_j(\boldsymbol{x}) = N_j(\boldsymbol{x}) + 1$. Consequently, $\|\boldsymbol{x} - \boldsymbol{x}'\|_1 = 1$. On the other hand, if some elements $x_i = j$ but $x'_i = k$, then $N_j(\boldsymbol{x}) = N_j(\boldsymbol{x}) + 1$ and $N_k(\boldsymbol{x}) = N_k(\boldsymbol{x}) - 1$. Consequently $\|\boldsymbol{x} - \boldsymbol{x}'\|_1 = 2$.

2.6 The local and central privacy models

So far, we have only seen the concept of differential privacy applied as a method to generate a privatised version of a dataset. In general, however, we want to perform some specific calculations on the data. Is it possible to compute things privately in the first place? Let us assume that you have defined a function $f : \mathcal{X} \to \mathcal{Y}$, which you wish to compute on arbitrary data $x \in \mathcal{X}$. If you wish to preserve privacy, however, you need the computation to satisfy some formal guarantee like differential privacy. The simplest way to do this is by simply generating a dataset a with a DP algorithm and then processing the data with the function we have already specified. This corresponds to the local privacy model.

The local privacy model.

Given a function $f : \mathcal{X} \to \mathcal{Y}$, and a set of individual data $\boldsymbol{x} = (x_i)_{i=1}^n$, use a differentially private algorithm $\pi(\boldsymbol{a}|\boldsymbol{x})$ to generate $\boldsymbol{a} \in \mathcal{X}$. Then outptut $f(\boldsymbol{a})$.

The advantage of the local model is that the calculation f does not need to be modified. The individuals do not have to trust anybody, as they can modify their data locally. However, they must take care that the DP calculation is performed correctly.

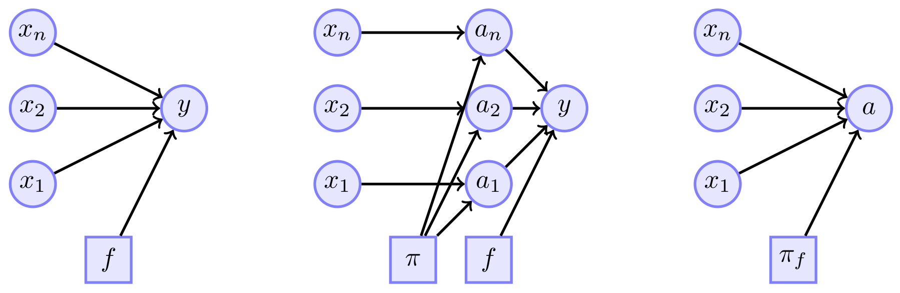
Typically, in the local privacy model, the i-th individual's data x_i is used to generate a private response a_i. This means that individual data is only available to a local private mechanism. This model allows us to publish a complete dataset of private responses. In the central privacy model, the data x is collected and the result a is published by a *trusted curator.*

The central privacy model.

A trusted curator obtains the data $\boldsymbol{x}$ of all individuals and selects a DP policy $\pi(\boldsymbol{a} \mid \boldsymbol{x})$ to generate the output a, so that $\boldsymbol{a} \approx f(\boldsymbol{x})$.

The main advantage of the centralised model is that approximating f with a differentially-private version can be much more accurate than simply using the original function with noisy data. However, obtaining such an approximation is not always easy.

The dependency diagrams in Figures 2.4 shows how the output depends on the individual data, the non-private and private algorithm.



(a) Non-private calculation (b) Local privacy (c) Central privacy

Figure 2.4: Non-privacy, local privacy, and centralised privacy models. In the first case, the value of y depends directly on the function f to be calculated, as well as the data x. In the local privacy model, noise is added to the individual data with a differentially private algorithm π, and the function is calculated on the output. In the central model, a stochastic version π_f of the original function f is calculated on the data.

Trust models.

Google uses a local privacy model to collect data from Android phone users. How does this compare to Google gathering data in the clear and then performing private computations? Who do the users have to trust? What are the privacy risks in either case?

2.6.1 Properties of differential privacy.

Remark 2.6.1. Any differentially private algorithm must be stochastic.

To prove that this is necessary, consider the example of counting how many people take drugs in a competition. If the adversary only doesn't know whether you in particular take drugs, but knows whether everybody else takes drugs, it's trivial to discover your own drug habits by looking at the total. This is because in this case, $f(x) = \sum_i x_i$ and the adversary knows x_i for all $i \neq j$. Then, by observing $f(x)$, he can recover $x_j = f(x) - \sum_{i \neq j} x_i$. Consequently, it is not possible to protect against adversaries with arbitrary side information without stochasticity.

Randomness is also necessary in cryptography. In that setting, Alice wants to communicate a message x to Bob. In a secret key cryptographic scheme, information theoretic security is achieved by making sure that the encrypted message a comes from a uniform distribution $\pi(a|x)$. This is achieved by uniformly selecting a key and selecting an appropriate hash function

Depending on the DP scheme, each query answered may leak privacy. In particular, if we always respond with an ϵ-DP mechanism, after T queries our privacy guarantee is $T\epsilon$. There exist mechanisms that do not respond to each query independently, which can bound the total privacy loss.

Definition 2.6.1 (T-fold composition)**.** In this privacy model, we compose a mechanism out of T differential private mechanisms $\pi_1, \ldots, \pi_T$. This composition is fixed ahead of time.

Theorem 2.6.1. *For any $\epsilon > 0$, the class of ϵ-differentially private mechanism satisfy $T\epsilon$-differential privacy under T-fold composition. More generally, if each mechanism is ϵ_i-DP, the composed mechanism is $\sum_{i=1}^{T} \epsilon_i$-DP.*

Theorem 2.6.2 (Post-processing)**.** *Let mechanism $\pi(a \mid x)$ be ϵ-DP. Applying any transformation $f : A \to Y$ to the output of the mechanism to obtain $y = f(a)$, results in another ϵ-DP mechanism.*

The composition theorem is a very useful tool, as it allows us to create new mechanisms that are composed of simpler parts. As a first example, we can look at how we can use it to create a randomised response algorithm when the respondents provide multiple attributes.

Randomised response for multiple attributes.

Up to now we have been discussing the case where each individual only has one attribute. However, in general each individual t contributes multiple data $x_{t.i}$, which can be considered as a row $\boldsymbol{x}_t$ in a database. Then the mechanism can release each $a_{t,i}$ independently.

For n users and k attributes, if the release of each attribute i is ϵ-DP then the data release is $k\epsilon$-DP. Thus to get ϵ-DP overall, we need ϵ/k-DP per attribute.

The result follows immediately from the composition theorem. We can see each attribute release as the result of an individual query. More generally, if each attribute i is released with an ϵ_i-DP mechanism, the overall mechanism is $\sum_i \epsilon_i$-DP.

Adversary knowledge

Assume that the adversary knows that the data is either $\boldsymbol{x}$ or $\boldsymbol{x}'$. For concreteness, assume the data is either

$$\boldsymbol{x} = (x_1, \ldots, x_j = 0, \ldots, x_n)$$

where x_i indicates whether or not the i-th person takes drugs, or

$$\boldsymbol{x}' = (x_1, \ldots, x'_j = 1, \ldots, x_n).$$

In other words, the adversary knows the data of all people apart from one, the j-th person. We can assume that the adversary has some prior belief

$$\beta(\boldsymbol{x}) = 1 - \beta(\boldsymbol{x}')$$

for the two cases. Assume the adversary knows the output a of a mechanism π. What can we say about the posterior distribution of the adversary $\beta(\boldsymbol{x} \mid a, \pi)$ after having seen the output, if π is ϵ-DP? How does it depend on ϵ?

Solution. We can write the adversary posterior as follows.

$$\begin{aligned}
\mathbb{P}^\pi_\beta(\boldsymbol{x} \mid a) &= \frac{\pi(a \mid \boldsymbol{x})\beta(\boldsymbol{x})}{\pi(a \mid \boldsymbol{x})\beta(\boldsymbol{x}) + \pi(a \mid \boldsymbol{x}')\beta(\boldsymbol{x}')} && (2.6.1)\\
&\geq \frac{\pi(a \mid \boldsymbol{x})\beta(\boldsymbol{x})}{\pi(a \mid \boldsymbol{x})\beta(\boldsymbol{x}) + \pi(a \mid \boldsymbol{x})e^\epsilon\beta(\boldsymbol{x}')} && \text{(from DP definition)}\\
&= \frac{\beta(\boldsymbol{x})}{\beta(\boldsymbol{x}) + e^\epsilon\beta(\boldsymbol{x}')}. && (2.6.2)
\end{aligned}$$

Note that $\pi(a \mid \boldsymbol{x}) \leq e^\epsilon \pi(a \mid \boldsymbol{x}')$ and conversely $\pi(a \mid \boldsymbol{x}') \leq e^\epsilon \pi(a \mid \boldsymbol{x})$. We can also then bound the quantity from above:

$$\mathbb{P}^\pi_\beta(\boldsymbol{x} \mid a) \leq \frac{\beta(\boldsymbol{x})}{\beta(\boldsymbol{x}) + e^{-\epsilon}\beta(\boldsymbol{x}')}.$$

Consequently, $\lim_{\epsilon \to 0} \mathbb{P}^\pi_\beta(\boldsymbol{x} \mid a) = \beta(\boldsymbol{x})$, hence the information gained by the adversary is bounded by the prior and the information loss ϵ. □

2.7 The Laplace mechanism

Many times we already have a function $f : \mathcal{X} \to \mathbb{R}$ we want to calculate on data, and we would like to make the function preserve privacy. This clearly falls within the *central privacy* model: The analyst has access to the data and function, and wishes for the algorithm generating the final output to be private. One solution, that can satisfy differential privacy, is to add noise to the function's output. additive noise. We start by first calculating the value of the function for the data we have, $f(x)$, and then we add some random noise ω, hence our calculation is random:

$$a = f(x) + \omega.$$

The amount and type of noise added, together with the smoothness of the function f, determine the amount of privacy we have. One of the simplest noise-adding mechanisms is to add zero-mean Laplace noise. Then we write $\omega \sim \mathit{Laplace}(\lambda)$.[4] The mechanism is defined below.

[4]When unspecified, the mean parameter is assumed to be zero.

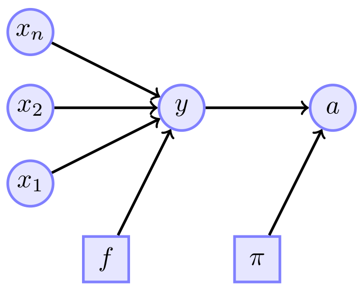


Figure 2.5: Laplace mechanism

Definition 2.7.1 (The Laplace mechanism)**.** For any function $f : \mathcal{X} \to \mathbb{R}$, the output a of the mechanism is sampled from a Laplace distribution with mean $f(x)$ and scaling parameter λ, i.e.

$$\pi(a \mid x) = \mathcal{Laplace}(f(x), \lambda). \tag{2.7.1}$$

The probability density function of the Laplace distribution with mean μ and scaling λ is given by:

$$p(\omega \mid \mu, \lambda) = \frac{1}{2\lambda} \exp\left(-\frac{|\omega - \mu|}{\lambda}\right).$$

and has mean μ and variance $2\lambda^2$.

The Laplace mechanism for averages

Here we have n individuals for which we wish to calculate the average salary.

- The i-th person receives salary x_i
- We wish to calculate the average salary in a private manner.

We can do this in two ways. By using a DP mechanism on each individual salary and then calculating the average, or by first calculating the average and then applying a DP mechanism to the result. In particular, we can add try adding Laplace noise in both cases.

The local model. In this case, $\pi(a \mid x)$ is obtained by independent Laplace noise ω_i for each individual:

- Obtain $y_i = x_i + \omega_i$, where $\omega_i \sim \mathcal{Laplace}(\lambda)$.
- Return $a = n^{-1} \sum_{i=1}^{n} y_i$.

The centralised model. In this case, $\pi(a \mid x)$ is obtained by averaging first and adding noise later.

- Calcualte $y = n^{-1} \sum_{i=1}^{n} x_i$.
- Return $a = y + \omega$, where $\omega \sim \mathcal{Laplace}(\lambda')$.

We must tune λ, λ' appropriately in to obtain the needed ϵ-DP guarantee.

In the centralised privacy model, the non-private calculation directly outputs $y = f(x)$. We can approximate this with a DP calculation with distribution $\pi(a \mid x)$. The Laplace mechanism does so by first calculating $y = f(x)$ and then generating $\pi(a \mid y)$ so that $\mathbb{E}_\pi[a \mid x] = f(x)$.

Let us now talk about how the Laplace mechanism that can be used both in the centralised and local model when $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}$.

DP properties of the Laplace mechanism

To use the Laplace mechanism in the centralised setting, we must relate it to a specific function f that we wish to compute. In particular, the mecnahism works by adding noise to the output of the function f in a carefully calibrated manner so that we achieve exactly ϵ-differential privacy.

Definition 2.7.2 (Sensitivity)**.** The sensitivity of a function f is

$$\mathbb{L}\left(f\right) \triangleq \sup_{xNx'} |f(x) - f(x')|$$

If we define a metric d, so that $d(x, x') = 1$ for xNx', then:

$$|f(x) - f(x')| \leq \mathbb{L}\left(f\right) d(x, x'),$$

i.e. f is $\mathbb{L}\left(f\right)$-Lipschitz with respect to d.

Example 3. If $f : \mathcal{X} \to [0, B]$, e.g. $\mathcal{X} = \mathbb{R}$ and $f(x) = \min\{B, \max\{0, x\}\}$, then $\mathbb{L}\left(f\right) = B$.

Example 4. If $f : [0, B]^n \to [0, B]$ is $f = \frac{1}{n}\sum_{t=1}^n x_t$, then $\mathbb{L}\left(f\right) = B/n$.

Proof. Consider two neighbouring datasets x, x' differing in example j. Then

$$f(x) - f(x') = \frac{1}{n}\left[f(x_j) - f(x'_j)\right] \leq \frac{1}{n}\left[B - 0\right]$$

□

Theorem 2.7.1. *The Laplace mechanism on a function f with sensitivity $\mathbb{L}\left(f\right)$, ran with $\mathcal{L}aplace(\lambda)$ is $\mathbb{L}\left(f\right)/\lambda$-DP. Consequently, if the Laplace mechanism is ran with $\lambda = \mathbb{L}\left(f\right)/\epsilon$, then it is ϵ-DP.*

Proof.

$$\frac{\pi(a \mid x)}{\pi(a \mid x')} = \frac{e^{|a-f(x')|/\lambda}}{e^{|a-f(x)|/\lambda}} \leq \frac{e^{|a-f(x)|/\lambda + \mathbb{L}(f)/\lambda}}{e^{|a-f(x)|/\lambda}} = e^{\mathbb{L}(f)/\lambda}$$

The first step follows from the definition of the Laplace mechanism. The inequality follows from the fact that for xNx', we have $|f(x) - f(x')| \leq \mathbb{L}\left(f\right)$. In particular,

(a) If $f(x') - a \geq 0$ then

$$|f(x') - a| = f(x') - a \leq \mathbb{L}\left(f\right) + f(x) - a \leq L + |a - f(x)|,$$

as $f(x') \leq f(x) + \mathbb{L}\left(f\right)$ from the Lipschitz property.

(b) If $f(x') - a < 0$ then

$$|f(x') - a| = a - f(x') \leq \mathbb{L}\left(f\right) - f(x) + a \leq L + |a - f(x)|,$$

as $-f(x') \leq \mathbb{L}\left(f\right) - f(x)$.

Replacing into the exponential gives us the required inequality. The final result is obtained with elementary algebra. □

DP properties of the average in the local and central model.

What is the effect of applying the Laplace mechanism in the local versus centralised model? Let us continue the average example. Here let us assume $x_i \in [0, B]$ for all i.

The Laplace mechanism in the local privacy model

The sensitivity of the individual data is B, so to obtain ϵ-DP we need to use $\lambda = B/\epsilon$. The variance of each component is $2(B/\epsilon)^2$, so the total variance is $2B^2/\epsilon^2 n$.

The Laplace mechanism in the centralised privacy model

The sensitivity of f is B/n, so we only need to use $\lambda = \frac{B}{n\epsilon}$. The variance of a is $2(B/\epsilon n)^2$.

Thus the two models have a significant difference in the variance of the estimates obtained, for the same amount of privacy. While the central mechanism has variance $O(n^{-2})$, the local one is $O(n^{-1})$ and so our estimates will need much more data to be accurate under this mechanism. In particular, we need square the amount of data in the local model as we need in the central model. Nevertheless, the local model may be the only possible route if we have no specific use for the data.

2.8 Interactive data access.

In a lot of applications, we cannot pre-define the computation that we want to perform. After we have collected the data, we need to be perform an adaptive data analysis. This requires performing a sequence of computations on the data, where the result of one computation determines what computation we will do next.

We can think of this as performing queries to the database. You may be familiar with database access languages such as SQL, where you ask a question such as “what is the sum of the attribute `age` in table `students`?”and obtain the exact sum back. It is possible to answer such queries through a differentially private mechanism. However, the more queries you answer, the more you reveal about the original data. In addition, since an adversary may be cleverly adjusting the queries to more efficiently discover a secret, we need the concept of adaptive composition.

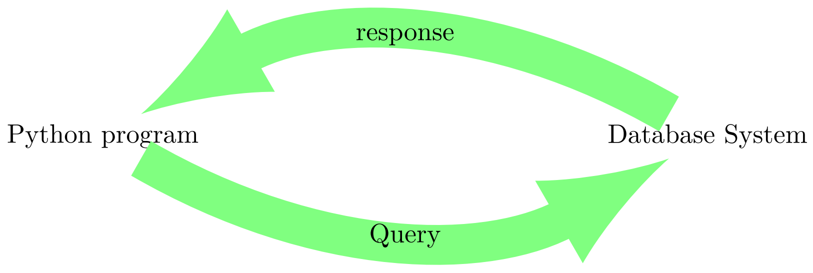


Figure 2.6: Database access model

2.8.1 Utility of queries

Rather than saying that we wish to calculate a private version of some specific function f, sometimes it is more useful to consider the problem from the perspective of the utility of different answers to queries. More precisely, imagine the interaction between a database system and a user:

Interactive queries

- System has data x.
- User asks query q.
- System responds with a.
- There is a common utility function $U : \mathcal{X}, \mathcal{A}, \mathcal{Q} \to \mathbb{R}$.

We wish to give the most useful answer, by return a that maximises U, i.e. $a = \arg\max_{a'} U(x, a', q)$, but are constrained by the fact that we also want to preserve privacy.

The utility $U(x, a, q)$ describes how appropriate each response a given by the system for a query r is given the data x. It can be seen as how useful the response is [5] It allows us to quantify exactly how much we would gain by replying correctly. The exponential mechanism, described below is a simple differentially private mechanism for responding to queries while trying to maximise utility for *any possible* utility function.

The Exponential Mechanism.

Here we assume that we can answer queries q, whereby each possible answer a to the query has a different utility to the DM: $U(q, a, x)$. The idea is that the best answer to the query should have the highest utility. The further away from the correct answer the response is, the lower the utility should be. This allows us to quantify how much we value correct answers to a query.

As an example, if the optimal response to a query is given by a real-valued function $f(q, x)$, then one possible utility function is $U(q, a, x) = -[f(q, x) - a]^2$. This results in the correct response having a utility of zero, and responses whose value is further away will have negative values.

The exponential mechanism allows us to answer queries in a "soft" manner, by using this utility function. It assigns lower probability to answers with lower utility, hence the better responses have a higher likelihood of being selected.

Definition 2.8.1 (The Exponential mechanism)**.** For any utility function $U : \mathcal{Q} \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}$, define the policy, which implicitly depends on U and ϵ, as:

$$\pi(a \mid x, q) \triangleq \frac{e^{\epsilon U(q,a,x)/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}}, \tag{2.8.1}$$

where $\mathbb{L}\left(U(q)\right) \triangleq \max_a \sup_{xNx'} |U(q, a, x) - U(q, a, x')|$ denotes the sensitivity of a query.

Clearly, when $\epsilon \to 0$, this mechanism is uniformly random. When $\epsilon \to \infty$ the action maximising $U(q, a, x)$ is always chosen.

Although the exponential mechanism can be used to describe most known DP mechanisms, its best use is in settings where there is a natural utility function.

[5] This is essentially the utility to the user that asks the query, but it could be the utility to the person that answers. In either case, the motivation does not matter the action should maximise it, but is constrained by privacy.

Theorem 2.8.1. *The exponential mechanism is ϵ-differentially private.*

Proof. We only need to look at the ratio between two different distributions of answers to queries. Then we have:

$$\begin{aligned}
\frac{\pi(a \mid x, q)}{\pi(a \mid x', q)} &= \frac{e^{\epsilon U(q,a,x)/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}} \times \frac{\sum_{a'} e^{\epsilon U(q,a',x')/\mathbb{L}(U(q))}}{e^{\epsilon U(q,a,x')/2\mathbb{L}(U(q))}} \\
&= e^{\epsilon U(q,a,x)/2\mathbb{L}(U(q)) - \epsilon U(q,a,x')/2\mathbb{L}(U(q))} \times \frac{\sum_{a'} e^{\epsilon U(q,a',x')/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}} \\
&= e^{\epsilon [U(q,a,x) - U(q,a,x')]/2\mathbb{L}(U(q))} \times \frac{\sum_{a'} e^{\epsilon U(q,a',x')/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}} \\
&\le e^{\epsilon/2} \times \frac{\sum_{a'} e^{\epsilon U(q,a',x')/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}} \\
&\le e^{\epsilon/2} \times \frac{\sum_{a'} e^{\epsilon [(U(q,a',x) + \mathbb{L}(U(q)))/2\mathbb{L}(U(q))}}{\sum_{a'} e^{\epsilon U(q,a',x)/2\mathbb{L}(U(q))}} \le e^{\epsilon}
\end{aligned}$$

□

2.9 Advanced topics.

In this section, we give a brief overview of some more advanced topics on privacy. The interested reader is urged to look into the given references.

2.9.1 Relaxations of differential privacy.

In practice, satisfying differential privacy results in mechanisms with low utility. It is possible to relax the definition of differential privacy to add an additive term, as shown below.

Definition 2.9.1. Approximate differential privacy. A stochastic algorithm $\pi : \mathcal{X} \to \boldsymbol{\Delta}(\mathcal{A})$, where $\mathcal{X}$ is endowed with a neighbourhood relation N, is said to be (ϵ, δ) differentially private if

$$\pi(A \mid x) \le \pi(A \mid x')\epsilon + \delta, \qquad \forall x N x', \quad A \subset \mathcal{A}. \tag{2.9.1}$$

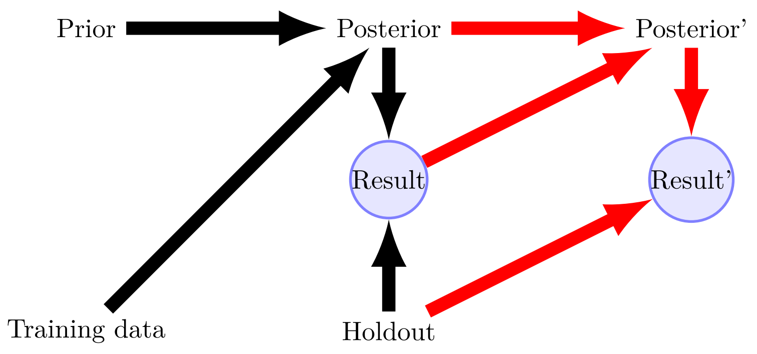
One simple interpretation of this definition is that the mechanism is $(\epsilon, 0)$-DP with probability $1 - \delta$. Conversely, this implies that with a very small probability δ, the complete dataset might be revealed. For that reason, the parameter δ must be as small as possible.

Approximate differential privacy is satisfied by the *Gaussian* mechanism. This is similar to the Laplace mechanism, but scales noise to the ℓ_2 sensitivity of the function.

The Gaussian mechanism

Given a function $f : \mathcal{X} \to \mathbb{R}^k$, the Gaussian mechanism with parameter σ outputs $a \sim \mathcal{N}(f(x), \sigma)$.

Let ℓ_2 sensitivity of f be $\Delta_2(f) = \max\{\|f(x) - f(y)\|_2 \mid xNy\}$. If we set $\sigma \ge 2\ln(1.25/\delta)\Delta_2(f)/\epsilon$, then the Gaussian mechanism is (ϵ, δ)-differentially private.



Renyi differential privacy.

The careful reader will have noticed that differential privacy is essentially a bound on the distributional distance of mechanism outputs for similar inputs. In particular, the distance used for ϵ-DP is $D_\infty(P\|Q) = \sup_A |\ln P(A)/Q(A)|$. While it is possible to use the KL divergence instead, this does not lead to very good properties—in particular, the resulting compositional properties are not so nice.

2.9.2 Reproducibility

Training and testing, overfitting on the test set.

The unfortunate practice of adaptive analysis

In the ideal data analysis, we start from some prior hypothesis, then obtain some data, which we split into training and holdout. We then examine the training data and obtain a posterior that corresponds to our conclusions. We can then measure the quality of these conclusions in the independent holdout set.

However, this is not what happens in general. Analysts typically use the same holdout repeatedly, in order to improve the performance of their algorithms. This can be seen as indirectly using the holdout data to obtain a new posterior, and so it is possible that you can overfit on the holdout data, even if you never directly see it. It turns out we can solve this problem if we use differential privacy, so that the analyst only sees a differentially private version of queries.

The reusable holdout? [6]

One idea to solve this problem is to only allow the analyst to see a private version of the result. In particular, the analyst will only see whether or not the holdout result is τ-close to the training result.

Algorithm parameters

- Performance measure f.
- Threshold τ. How close do we want f to be on the training versus holdout set?
- Noise σ. How much noise should we add?

[6] Also see `https://ai.googleblog.com/2015/08/the-reusable-holdout-preserving.html`

- Budget B. How much are we allowed to learn about the holdout set?

Algorithm idea

Run algorithm λ on data D_T and get e.g. classifier parameters θ.
Run a DP version of the function $f(\theta, D_H) = \mathbb{I}\{U(\theta, D_T) \geq \tau U(\theta, D_H)\}$.

So instead of reporting the holdout performance at all, you just see if you are much worse than the training performance, i.e. if you're overfitting. The fact that the mechanism is DP also makes it difficult to learn the holdout set. See the thresholdout link for more details.

2.10 Discussion

The definition of differential privacy

- First rigorous mathematical definition of privacy.
- Relaxations and generalisations possible.
- Connection to learning theory and reproducibility.

Current uses

- Apple. DP is used internally in the company to "protect user privacy". It is not clear exactly what they are doing but their efforts seem to be going in the right direction.
- Google. The company has a DP API available based on randomised response, RAPPOR.
- Uber. Elastic sensitivity for SQL queries, which is available as open source. This is a good thing, because it is easy to get things wrong with privacy.
- US 2020 Census. It uses differential privacy to protect the condidentiality of responders' information while maintaining data that are suitable for their intended uses.

Open problems

- Complexity of differential privacy.
- Verification of implementations and queries.

Available privacy toolboxes

***k*-anonymity**

- `https://github.com/qiyuangong/Mondrian` Mondrian k-anonymity

Differential privacy

- `https://github.com/bmcmenamin/thresholdOut-explorations`Threshold out
- `https://github.com/steven7woo/Accuracy-First-Differential-Privacy`Accuracy-constrained DP
- `https://github.com/menisadi/pydp`Various DP algorithms
- `https://github.com/haiphanNJIT/PrivateDeepLearning` Deep learning and DP

Learning outcomes

Understanding

- Linkage attacks and k-anonymity.
- Inferring data from summary statistics.
- The local versus centralised differential privacy model.
- False discovery rates.

Skills

- Make a dataset satisfy k-anonymity with respect to identifying attributes.
- Apply the randomised response and Laplace mechanism to data.
- Apply the exponential mechanism to simple decision problems.
- Use differential privacy to improve reproducibility.

Reflection

- How can potentially identifying attributes be chosen to achieve k-anonymity?
- How should the parameters of the two ideas, ϵ-DP and k-anonymity be chosen?
- Does having more data available make it easier to achieve privacy?

Further reading

- k-anonymity[?]
- Randomness, privacy, and the US 2020 census[?]
- The paper introducing differential privacy[?]
- Differential privacy book[?]
- Bayesian inference and privacy[?]
- Local differential privacy and statistics[?]

- Local differential privacy and applications$^?$
- The exponential mechanism$^?$

2.11 Exercises

Exercise 1. Show that the randomised response mechanism, as originally defined, is not differentially private with respect to the addition/deletion neighbourhood.

Proof. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ be three datasets with $\boldsymbol{x}_1 N \boldsymbol{x}_2 N \boldsymbol{x}_3$ and $\boldsymbol{x}_1 N_e \boldsymbol{x}_3$. First of all, an algorithm that is ϵ-DP with respect to the N, is 2ϵ-DP with respect to N_e. This follows from the fact that $\pi(a|\boldsymbol{x}_1) \leq e^{\epsilon}\pi(a|\boldsymbol{x}_2) \leq e^{2\epsilon}\pi(a|\boldsymbol{x}_3)$. What about the converse? Consider the case where $\boldsymbol{x}_1$ has n entries and $\boldsymbol{x}_2$ has $n+1$ entries. Let A_n be the set of all responses with n entries. Then clearly $\pi(A_n|\boldsymbol{x}_1) = 1$ and $\pi(A_n|\boldsymbol{x}_2) = 0$. Consequently $|\ln \frac{\pi(A_n|\boldsymbol{x}_1)}{\pi(A_n|\boldsymbol{x}_2)}|$ is not bounded by any constant, and differential privacy is not satisfied with respect to insertions and deletions. □

Exercise 2. Define a variant of the binary randomised response mechanism that satisfies differential privacy with respect to the insertion and deletion neighbourhood. More precisely, it should hold that for any dataset pair $\boldsymbol{x} = (x_1, \ldots, x_n)$, and $\boldsymbol{x}' = (x_1, \ldots, x_n)$, $\boldsymbol{x} = (x_1, \ldots, x_n, x_{n+1})$, the mechanism satisfies

$$\pi(A|\boldsymbol{x}) \leq e^{\epsilon}\pi(A|\boldsymbol{x}'), \qquad \forall A \subset \mathcal{A}.$$

Proof. First of all, for the mechanism to satisfy DP, it must be the case that an output size must have a non-zero probability: If with an n-size input we can only generate $\leq n$-size outputs, then we cannot generate a $n+1$-size output. However, we can do so with an input of size $n+1$. □

Exercise 3. Consider a relaxation of differential privacy to KL divergences, that is. That is, a mechanism is ϵ-KL-private if

$$\int_{\mathcal{A}} \ln \frac{d\pi(a|x)}{d\pi(a|x')} d\pi(a|x) \leq \epsilon.$$

Prove a non-interactive composition theorem for this mechanism.

Proof. If performing two queries, we can show that □