

Monitoring Coronavirus and the need of privacy

Monitoring Coronavirus

- <https://www.technologyreview.com/s/615329/coronavirus-south-korea-smartphone-app-quarantine/>
- <http://theconversation.com/coronavirus-south-koreas-success-in-controlling-disease-is-due-to-its-acceptance-of-surveillance-134068>
- https://www.ansa.it/sito/videogallery/italia/2020/03/19/coronavirus-controlli-celle-telefoniche-cosa-sono-e-come-funzionano_3610d248-e14d-495f-aefe-9d03333362b5.html
- http://dangrover.com/blog/2020/04/05/covid-in-ui.html?fbclid=IwAR318H6XUAdxOSJjjwBdl6nDpOqAx_EcJ_AmSkNAtssFuC3pt5jG0y5NqNg
- <https://www.wired.com/story/apple-google-bluetooth-contact-tracing-covid-19/>

Privacy

- <https://www.ft.com/content/19d90308-6858-11ea-a3c9-1fe6fedcca75>
- <https://blog.openmined.org/covid-app-privacy-advice/>
- <https://www.covid-watch.org/>
- <https://www.covid-watch.org/article>



Private Set Intersection (PSI)

- Why?
 - To find out the people that met infected subjects
 - “... any analytic where you want to compare a user’s data (on a phone) with a patient’s data (in the cloud), use PSI to avoid centralizing a massive amount of data to a single location, which is a prime target for hacking and intentional or accidental mis-use”

PSI (toy example from [wikipedia](#))

Name	Hash
Alice	64489c85dc2fe0787b85cd87214b3810
Eve	d3f791f59cbeff0ec06afb94bb23e772

Name	Hash
Eve	d3f791f59cbeff0ec06afb94bb23e772
Bob	2fc1c0beb992cd7096975cfefb9d5c3b

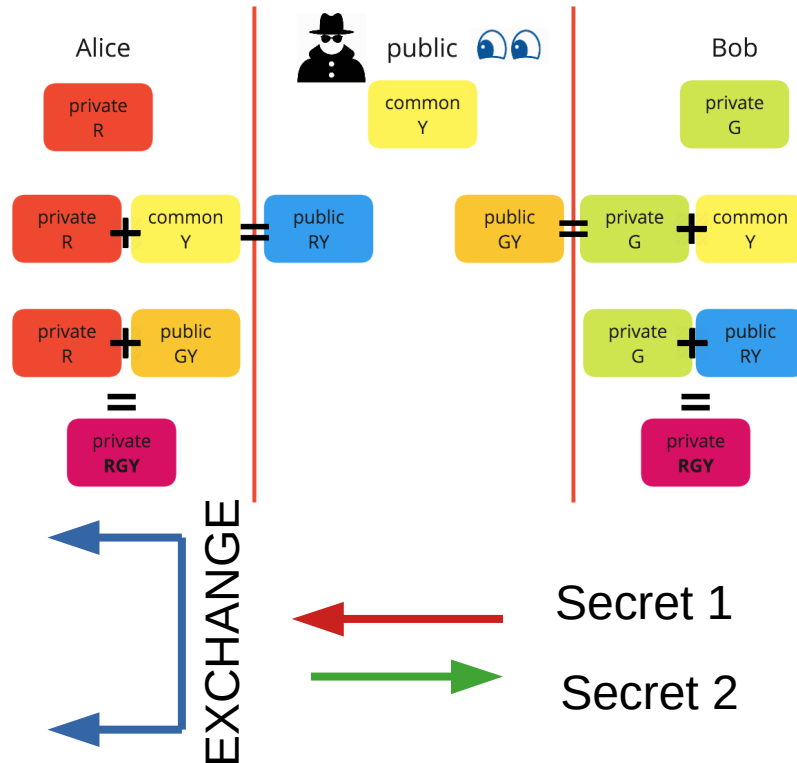
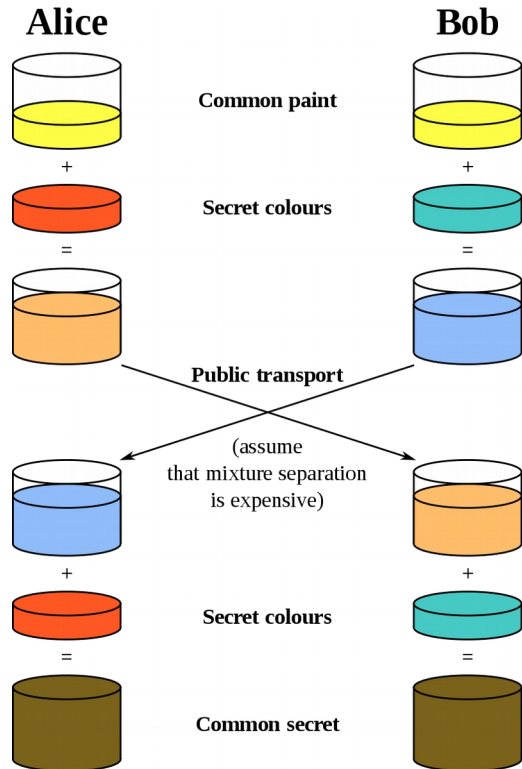
EXCHANGE

Eve has been in both places, but you don't know anything about Bob and Alice

Man-in-the-middle attack to obtain the encrypted data → one record matched (the intersection), although the name of shared record cannot be learned.

PSI (toy example from wikipedia)

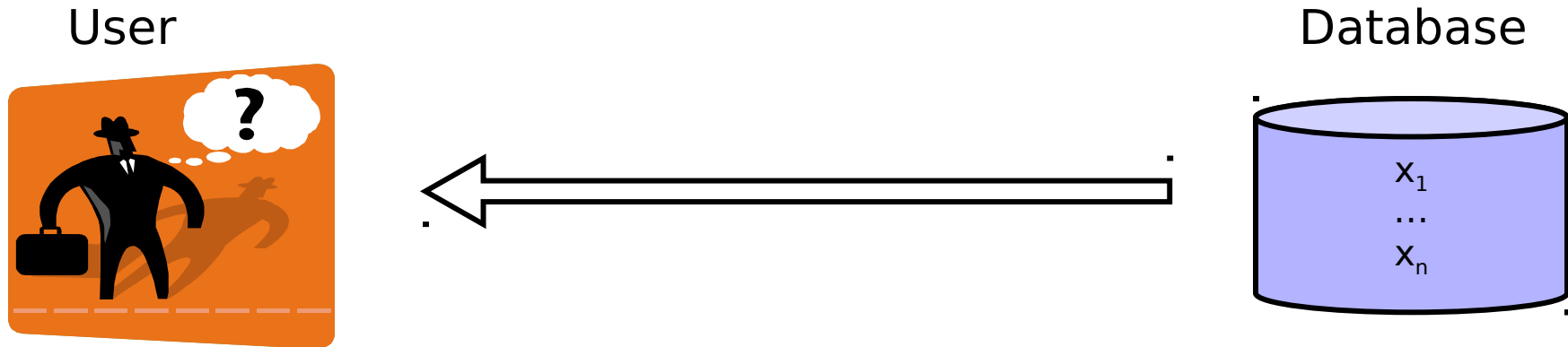
Man-in-the-middle attack to obtain the encrypted data → one record matched (the intersection), although the name of shared record cannot be learned.



Problem definition

A user makes query to a DB

- ◆ He is allowed to make statistical queries (privacy reasons)
- ◆ He is smart so he is able to infer information he should not know by repeatedly making queries questions
- ◆ We want to allow him to query but we want to preserve privacy of the DB



K-Anonymity

The information for each person contained in the release cannot be distinguished from at least $k - 1$ individuals whose information also appear in the release.

ID	Age	Zipcode	Diagnosis
1	28	13053	Heart Disease
2	29	13068	Heart Disease
3	21	13068	Viral Infection
4	23	13053	Viral Infection
5	50	14853	Cancer
6	55	14853	Heart Disease
7	47	14850	Viral Infection
8	49	14850	Viral Infection
9	31	13053	Cancer
10	37	13053	Cancer
11	36	13222	Cancer
12	35	13068	Cancer

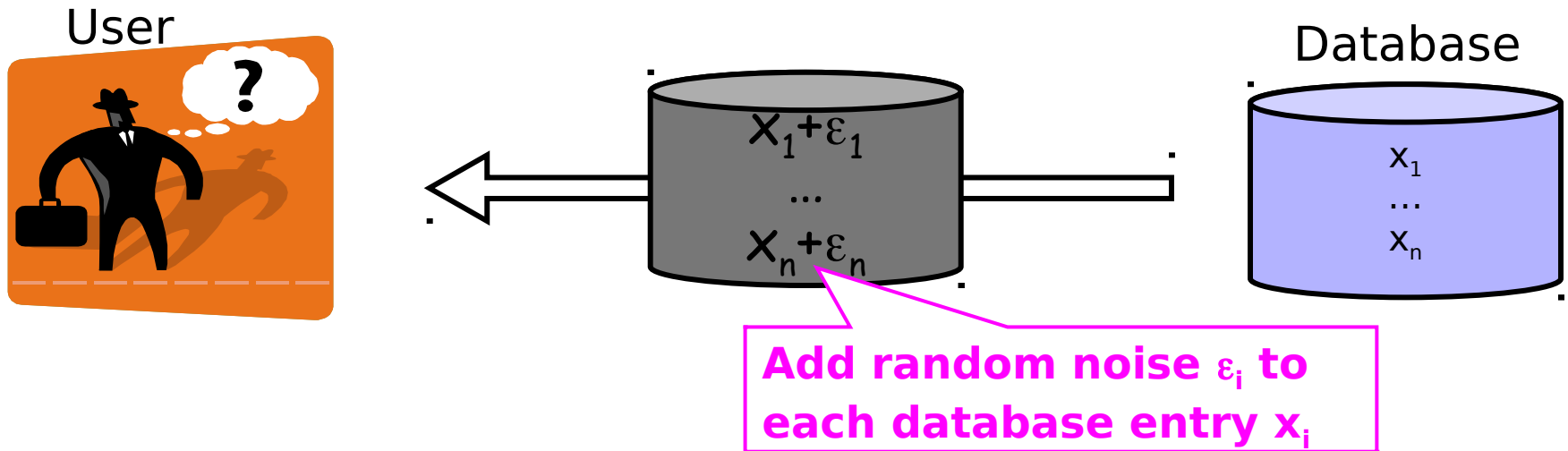
k-anonymization



ID	Age	Zipcode	Diagnosis
1	[20-30]	130**	Heart Disease
2	[20-30]	130**	Heart Disease
3	[20-30]	130**	Viral Infection
4	[20-30]	130**	Viral Infection
5	[40-60]	148**	Cancer
6	[40-60]	148**	Heart Disease
7	[40-60]	148**	Viral Infection
8	[40-60]	148**	Viral Infection
9	[30-40]	13***	Cancer
10	[30-40]	13***	Cancer
11	[30-40]	13***	Cancer
12	[30-40]	13***	Cancer

Input Perturbation

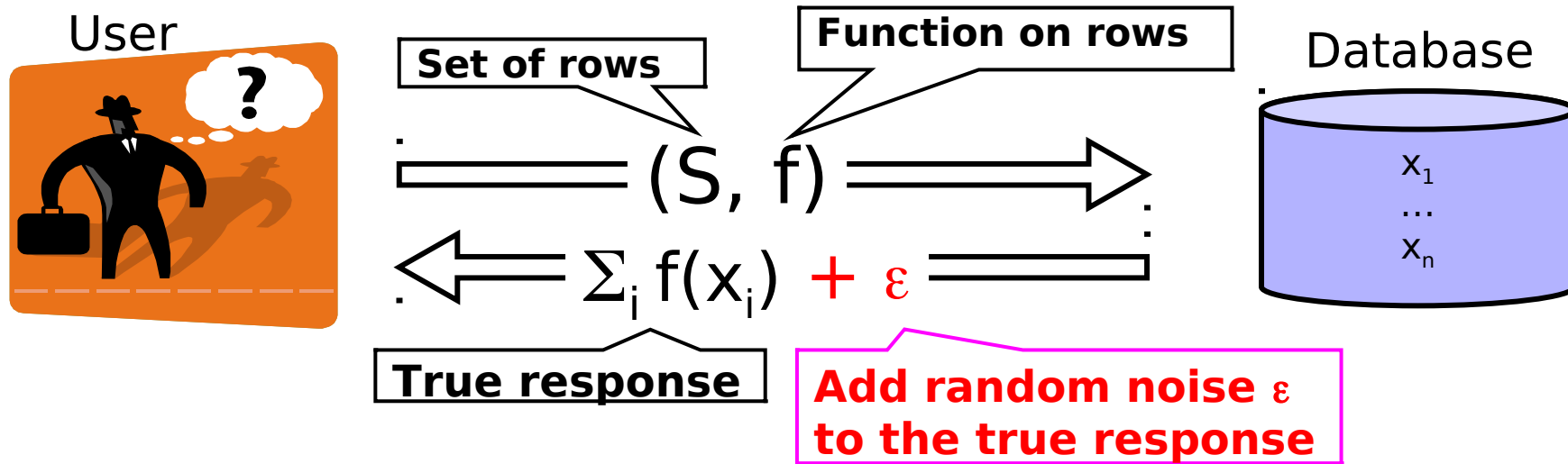
- Reveal entire database, but randomize entries



For example, if distribution of noise has mean 0, user can compute average of x_i

Output Perturbation

- Randomize response to each query



Limits of Output Perturbation

- Let n be the size of the database (# of entries)
- If $O(n^{1/2})$ perturbation applied, adversary can extract entire database after $\text{poly}(n)$ queries
- ...but even with $O(n^{1/2} \log n)$ perturbation, it is unlikely that user can learn anything useful from the perturbed answers (too much noise)

Revealing Information while Preserving Privacy

Irit Dinur Kobbi Nissim*
NEC Research Institute
4 Independence Way
Princeton, NJ 08540
(iritd,kobbi)@research.nj.nec.com

ABSTRACT

We examine the tradeoff between privacy and usability of statistical databases. We model a statistical database by an n -bit string d_1, \dots, d_n , with a query being a subset $q \subseteq [n]$ to be answered by $\sum_{i \in q} d_i$. Our main result is a polynomial reconstruction algorithm of data from noisy (perturbed) subset sums. Applying this reconstruction algorithm to statistical databases we show that in order to achieve privacy one has to add perturbation of magnitude $\Omega(\sqrt{n})$. That is, smaller perturbation always results in a strong violation of privacy. We show that this result is tight by exemplifying access algorithms for statistical databases that preserve privacy while adding perturbation of magnitude $O(\sqrt{n})$.

For time- T bounded adversaries we demonstrate a privacy-preserving access algorithm whose perturbation magnitude is $\approx \sqrt{T}$.

Keywords

Integrity and Security, Data Reconstruction, Subset-sums with noise.

1. INTRODUCTION

Let us begin with a short story. Envision a database of a hospital containing the medical history of some population. On one hand, the hospital would like to advance medical research which is based (among other things) on statistics of the information in the database. On the other hand, the hospital is obliged to keep the privacy of its patients, i.e. leak no medical information that could be related to a specific patient. The hospital needs an access mechanism to the database that allows certain 'statistical' queries to be answered, as long as they do not violate the privacy of any single patient.

*Work partly done when the author was at DIMACS, Rutgers University, and while visiting Microsoft Research Silicon Valley Lab.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

PRODS 2003, June 9-12, 2003, San Diego, CA.
Copyright 2003 ACM 1-58113-670-6/03/06 ...\$5.00.

One simple tempting solution is to remove from the database all 'identifying' attributes such as the patients' names and social security numbers. However, this solution is not enough to protect patient privacy since there usually exist other means of identifying patients, via *indirectly identifying* attributes stored in the database. Usually, identification may still be achieved by coming across just a few 'innocuous' looking attributes¹.

The topic of this work is to explore the conditions under which such a privacy preserving database access mechanism can exist.

A Threshold for Noisy Reconstruction. Viewing query-answer pairs as an 'encoding' of the bits d_1, \dots, d_n , the goal of the privacy-breaking adversary is to efficiently 'decode' this encoding i.e. to obtain values of some d_i s. In our setting, the 'decoding' algorithm is given access to subset sums of the d_i s perturbed by adding some random noise of magnitude $\leq \epsilon$. We show an interesting threshold phenomenon where either almost all of the d_i s can be reconstructed, in case $\epsilon \ll \sqrt{n}$, or none of them, when $\epsilon \gg \sqrt{n}$.

1.1 A Brief Background

The problem of protecting sensitive information in a database while allowing *statistical* queries (i.e. queries about sums of entries, and the like) has been studied extensively since the late 70's, (see surveys in [2, 18]).

In their comparative survey of privacy methods for statistical databases, Adam and Wortmann [2] classified the approaches taken into three main categories: (i) query restriction, (ii) data perturbation, and (iii) output perturbation. We give a brief review of these approaches below, and refer the reader to [2] for a detailed survey of the methods and their weaknesses.

Query Restriction. In the query restriction approach, queries are required to obey a special structure, supposedly to prevent the querying adversary from gaining too much information about specific database entries. The limit of this approach is that it allows for a relatively small number of queries.

A related idea is of query auditing [7], i.e. a log of the queries is kept, and every new query is checked for possible compromise, allowing/disallowing the query accordingly.

¹A patient's gender, approximate age, approximate weight, ethnicity, and marital status – may already suffice for a complete identification of most patients in a database of a thousand patients. The situation is much worse if a relatively 'rare' attribute of some patient is known. For example, a patient having Cystic Fibrosis (frequency $\approx 1/3000$) may be uniquely identified within about a million patients.

A more practical definition of Privacy

IMPOSSIBLE

Privacy means that anything that can be learned about a respondent from the statistical database can be learned without access to the database

SECOND APPROACH (DIFFERENTIAL PRIVACY)

Whatever is learned about one respondent A would be learned regardless of whether or not A participates to the database

Differential Privacy

- Promise: an individual will not be affected, adversely or otherwise, by allowing his/her data to be used in any study or analysis, no matter what other studies, datasets, or information sources, are available
- Paradox: learning nothing about an individual while learning useful statistical information about a population

Differential Privacy: ϵ parameter

For every pair of inputs $D1$ $D2$
that differ in one row \leftarrow
presence or absence of a
record

For every output O

Controls the degree to which $D1$ and $D2$
can be distinguished.

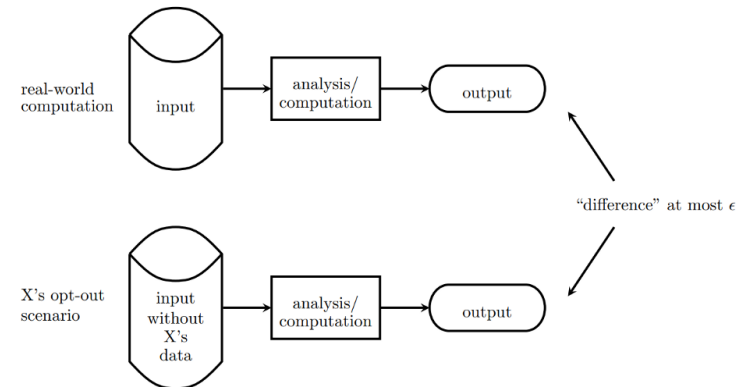
Smaller ϵ gives more privacy
and worse utility

If algorithm A satisfies differential privacy then

$$\Pr [A(D1) = O] \leq \exp(\epsilon) \Pr [A(D2) = O] \quad (\epsilon > 0)$$

$$\Pr [A(D2) = O]$$

Intuition: adversary should not be able to use
output O to distinguish between any $D1$ and $D2$



A randomized algorithm K gives ϵ -differential privacy if for all data sets D and D' differing on at most one row, and any $S \subseteq \text{Range}(K)$,

$$\Pr[K(D) \in S] \leq \exp(\epsilon) \times \Pr[K(D') \in S]$$

These are 2 quantities must be considered in DP algorithms are:

- Epsilon (ϵ): A metric of privacy loss at a differentially change in data (adding, removing 1 entry). The smaller the value is, the better privacy protection.
- Accuracy: The closeness of the output of DP algorithms to the pure output. In the *Private Machine Learning with PATE* part, I will use the classification accuracy on the test set as a statistic for evaluating accuracy.

Calculate the average age of a group of people.

Instead of having each person send you their true age, you have them send you their true age + random number between -100 and 100. So, if someone was 42, they might send you $42 + (-50) = -8$.

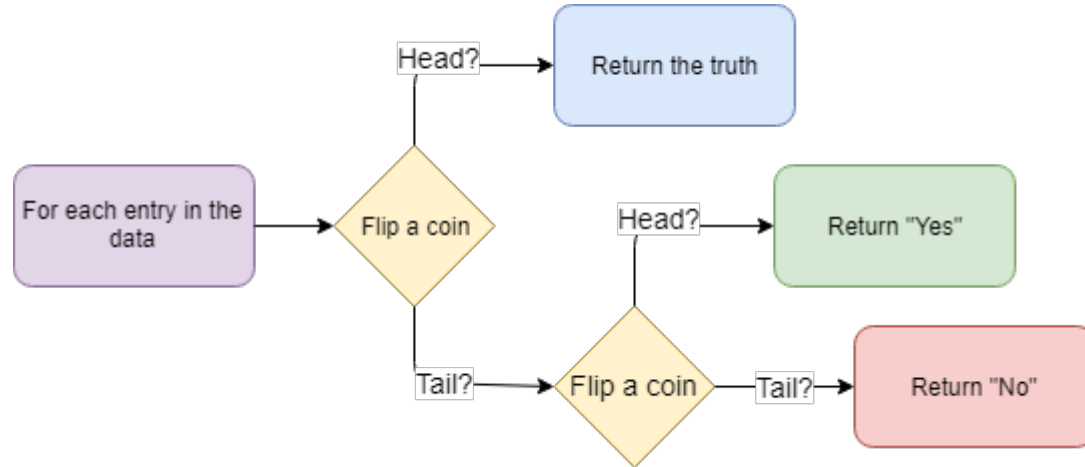
we can generate random numbers so that if you average over enough of them, they cancel each other out. Thus, if 10,000 people all add a random number (pulled from a distribution with a mean of 0) to their age before reporting it, the average age reported will still be similar to the underlying raw data despite the fact that nobody revealed their true age.

The bigger the random numbers (on average), the more privacy protection we give people, but the larger the group of people we need to average over before we can get aggregate statistics

This approach is useful for allowing app users to transform their local data in a way that protects it so that the central server can collect useful statistics without the central server being able to reverse engineer any specific person's personal data.

Differential Privacy

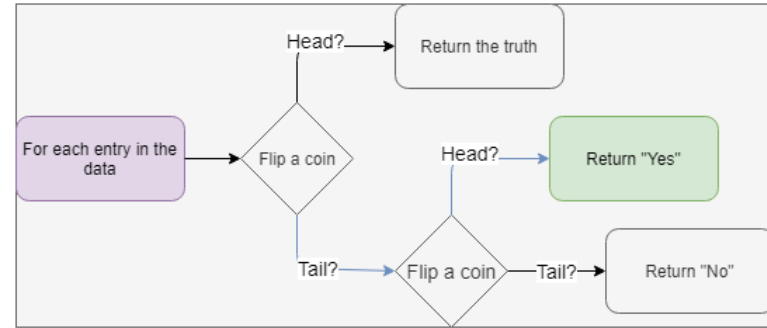
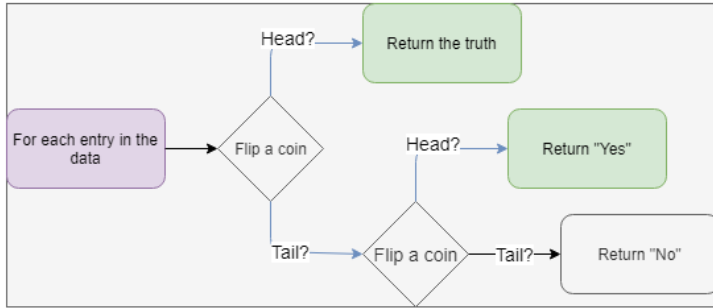
<https://towardsdatascience.com/understanding-differential-privacy-85ce191e198a>



Assume you want to infer the percentage of innocents in the population (p_{innocent}) from that noisy data.

- Compute probability of returning “yes” given that individual isn’t an innocent: $P(\text{“yes”} | \text{not innocent}) = p_{\text{head}} + (1 - p_{\text{head}}) * p_{\text{head}}$.

- Compute probability of returning “yes” given that individual is an innocent: $P(\text{“yes”} | \text{innocent}) = (1 - p_{\text{head}}) * p_{\text{head}}$.



- Compute p_{innocent} :

$$p_{\text{innocent}} = (P(\text{“yes”}) - P(\text{“yes”} | \text{not innocent})) / (P(\text{“yes”} | \text{innocent}) - P(\text{“yes”} | \text{not innocent})) = 1 - (P(\text{“yes”}) - (1 - p_{\text{head}}) * p_{\text{head}}) / p_{\text{head}}$$
 (proof by deduction)

Note: Above result is an asymptotic result guaranteed by Law of Large Numbers.

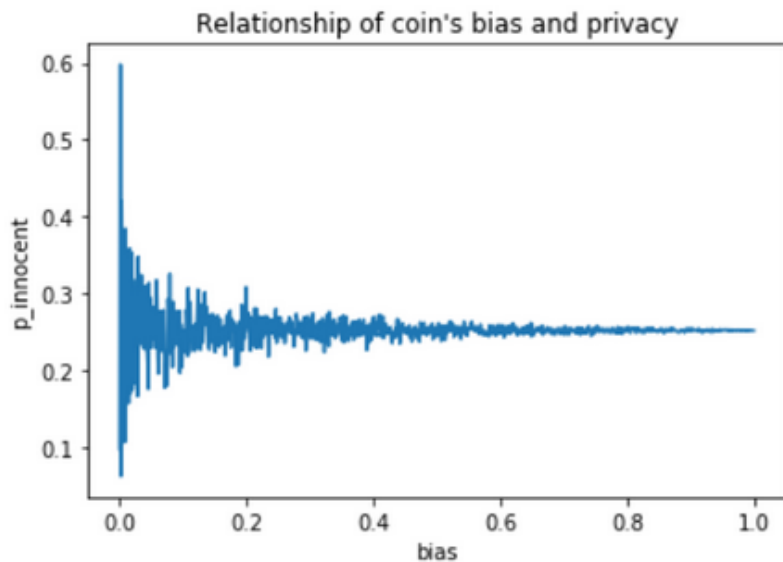
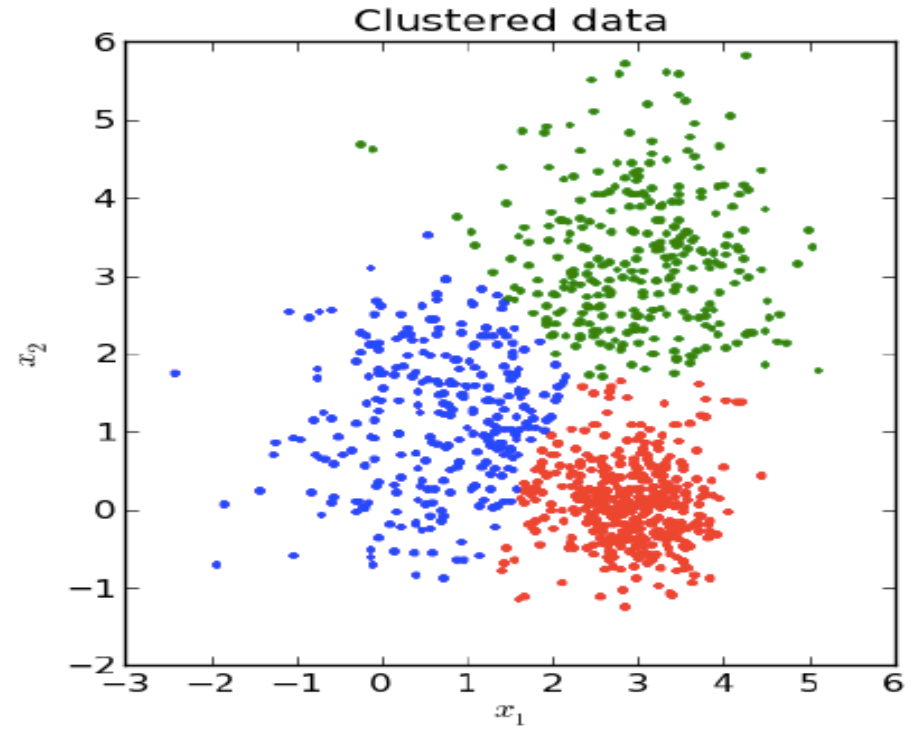
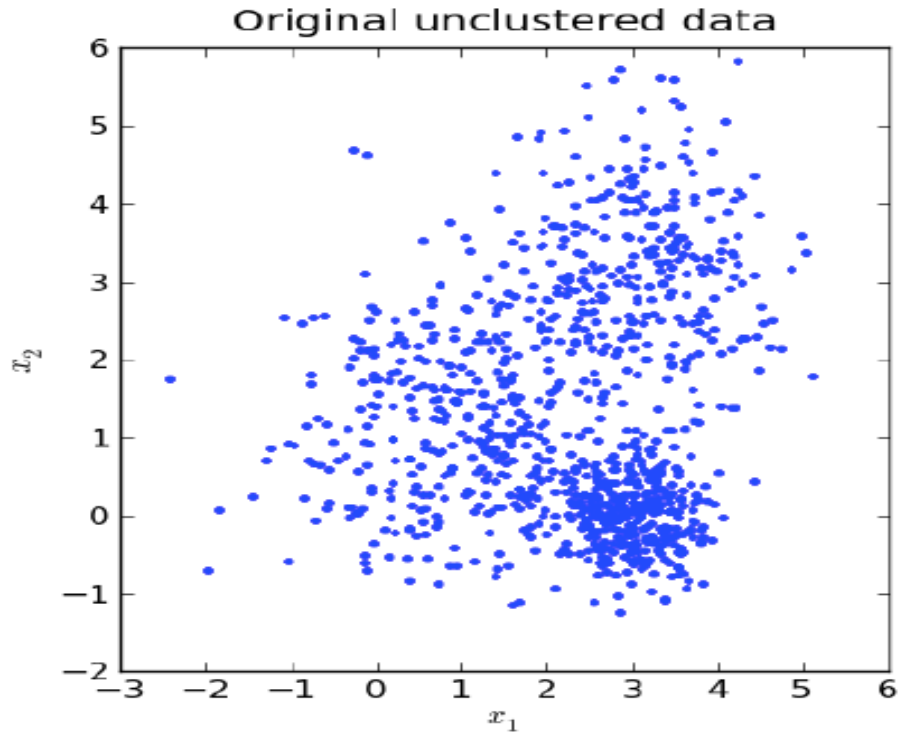


Figure 4: Compute on a simulated survey data of 20,000 individuals. Low bias coins add more noise to the data. A consequence of adding noise is the decrease in privacy loss.

- What does DP tell us?
As you can see in Figure 4, the variance of $p_innocent$ increases dramatically and approaches infinity when p_head approaches 0, lead to a rapid decrease in privacy loss. DP also gives us the same conclusion. Thus, when p_head is 0, the distribution of returned result is identical, no matter an individual is an innocent or not (the distance of 2 distributions is $P(\text{"yes"} | \text{not innocent}) -$

$P(\text{"yes"} | \text{innocent}) = p_head$, the bias). If the number of innocents participates in the data changed, it does not lead to any changes in information in the noisy returned data. It means that there is no private information in the noisy returned data.

Case Study: K-means Clustering



K-means Clustering

- Partition a set of points x_1, x_2, \dots, x_n into k clusters S_1, S_2, \dots, S_k such that the following is minimized:

$$\sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|_2^2$$



Mean of the cluster S_i

K-means Clustering

Algorithm (Lloyd):

- ◆ Initialize a set of k centers
- ◆ Repeat
 - Assign each point to its nearest center
 - Recompute the set of centers

Until convergence ...

- ◆ Output final set of k centers

Differentially Private K-means

- Suppose we fix the number of iterations to T
- In each iteration (given a set of centers):
 - 1. Assign the points to the new center to form clusters
 - 2. Noisily compute the size of each cluster
 - 3. Compute noisy sums of points in each cluster

- k-Means Clustering
- True means (leaky)
 - Noisy means (private)

Many data-mining tasks can be written as (few) noisy statistical queries such as

- k-means,
- association rules,
- PCA, Perceptron, etc.

Noise is independent of the dB size such that privacy is ensured but accuracy varies. The larger the dB the higher the accuracy.

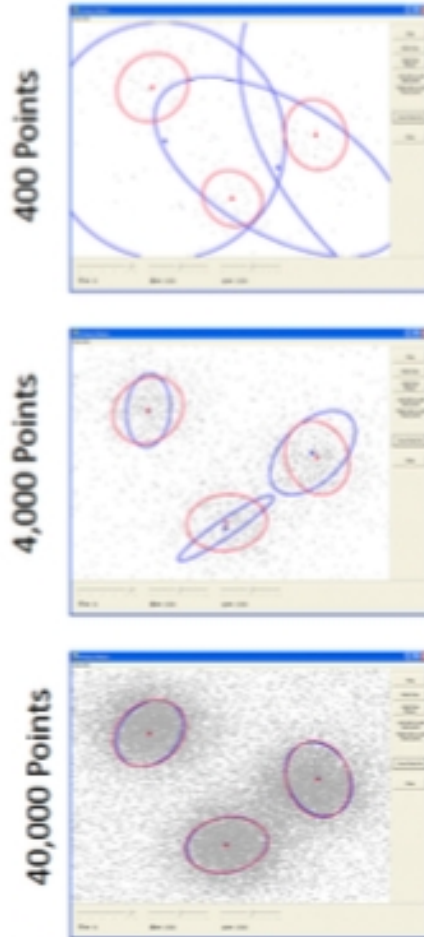


Figure 6: Effect of noise

Differentially Private Kmeans

- Suppose we fix the number of iterations to T
- In each iteration (given a set of centers):
 - 1. Assign the points to the new center to form clusters
 - 2. Noisily compute the size of each cluster
 - 3. Compute noisy sums of points in each cluster

Each iteration uses ϵ/T privacy budget, total privacy loss is ϵ

Differentially Private Kmeans

- Question: Which of these steps expands privacy budget?
- In each iteration (given a set of centers):
 - 1. Assign the points to the new center to form clusters
 - 2. Noisily compute the size of each cluster
 - 3. Compute noisy sums of points in each cluster

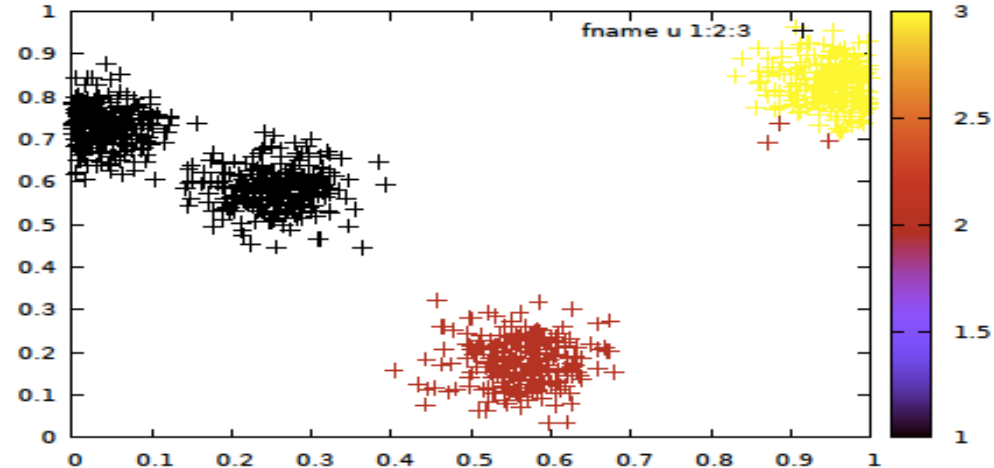
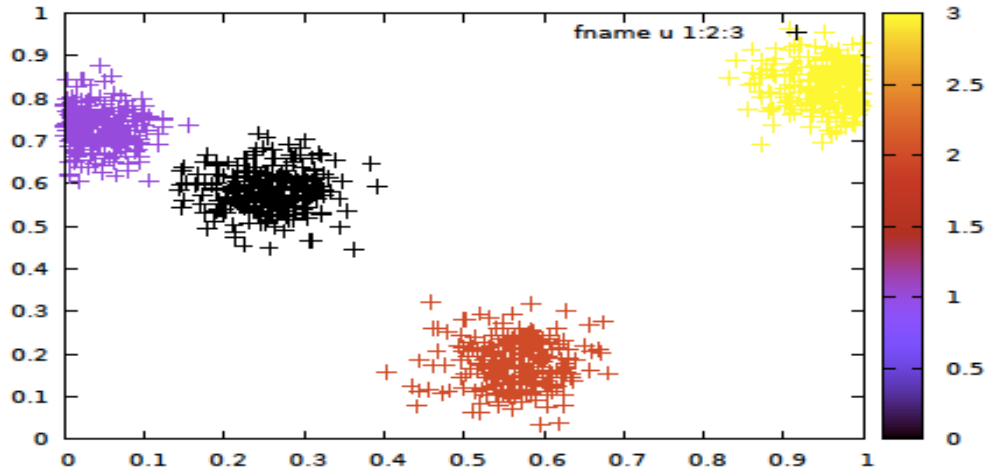
Differentially Private Kmeans

- In each iteration (given a set of centers):
 - 1. Assign the points to the new center to form clusters
 - 2. Noisily compute the size of each cluster
 - 3. Compute noisy sums of points in each cluster
- If each cluster has large size then we expect that the error in computing size is small

Differentially Private Kmeans

Original K-means alg.

Laplace Kmeans algorithm



clusters that are far apart.

- ◆ Since we add noise to the sums with sensitivity proportional to $|\text{dom}|$, Laplace k-means can't distinguish small clusters that are close by.