Big Data Privacy

Open DataBase

Deterministic database:

→ Database with concrete information

pseudo.	job	sex	age	disease
u1	engineer	m	35	hepatitis
u2	engineer	m	38	hepatitis
u3	lawyer	m	38	HIV
u4	writer	f	30	flu
u5	writer	f	30	HIV
u6	dancer	f	30	HIV
u7	dancer	f	30	HIV

Issue: if Emily → (30 yo, fem, danc) we know she has HIV

k anonymity

→ $k := \min_{o \in \mathcal{O}} |A(o)|$ (number of users having the non sensible characteristics o)

→ Can be done by:

→ Aggregating data
→ Grouping users
→ Noise addition

pseudo.	job	sex	age	disease
u1	professional	m	35-40	hepatitis
u2	professional	m	35-40	hepatitis
u3	professional	m	35-40	HIV
u4	artist	f	30-35	flu
u5	artist	f	30-35	HIV
u6	artist	f	30-35	HIV
u7	artist	f	30-35	HIV

k=3 → good, otherwise I would know group + no Pet

k=4

k=3

Issue: lack of diversity in sensible

l-diversity

→ sensible attributes are "well distributed"

→ Ideal: uniformly

→ l = # of different sensible in each group

pseudo.	job	sex	age	disease
u1	professional	m	35-40	hepatitis
u2	professional	m	35-40	hepatitis
u3	professional	m	35-40	HIV
u4	artist	f	30-35	flu
u5	artist	f	30-35	HIV
u6	artist	f	30-35	HIV
u7	artist	f	30-35	HIV

2 ≥ l
2 ≥ l

t - closeness

The distribution of sensitive attributes in an anonymity set is "close" to the distribution of the full table

Probabilistic database

↳ Can capture real world complexity
$A(o) = 2 \ \forall o$

pseudo.	comedy	drama
u1	... **65** (num 60%)	... **35** (num 40%)
u2	... **50** (num 55%)	... **50** (num 45%)

→ If we have enough observations we can guess who is the user
→ How much / observation ?

Hypothetical Information:

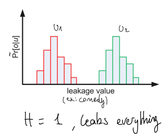
Idea: → How much information 1 observation leaks
→ Based on internal structure $\hat{\Pr}$
→ If db follows this prob model, how much info would a leakage carry ?

$$\hat{\mathrm{HI}}(U;O) = \mathrm{H}[U] + \sum_{u \in \mathcal{U}} \Pr[u] \cdot \sum_{o \in \mathrm{DB}} \hat{\Pr}[o|u] \cdot \log_2 \hat{\Pr}[u|o],$$

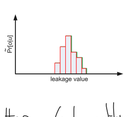
- with $\Pr[X = x] := \Pr[x]$ & $\mathrm{H}[U] = -\sum_u \Pr[u] \cdot \log_2 \Pr[u]$

→ Re-Identification-Success-Rate = $1 - (1-H)^n$

Two examples:



H = 1, leaks everything



H=0, leaks nothing

Perceived Information

important bc one might act on his believes

↳ Given what the attacker believes how much info do they third they gained

→ even if they actually did not gain much knowledge PI might be high

- Split the DB in two parts: DB_1 and DB_2 → simulates attacker's model; → simulates the leakage (true)
  - Where p stands for profiling and t for testing
- Build a model $\hat{\Pr}[o|u] \leftarrow DB_p$ for every user

$$\mathrm{PI}(U;O) = \mathrm{H}[U] + \sum_{u \in \mathcal{U}} \Pr[u] \cdot \sum_{o \in \mathcal{O}} \underbrace{\Pr[o|u]}_{true} \cdot \log_2 \underbrace{\hat{\Pr}[u|o]}_{model}.$$

↓

$$\hat{\mathrm{PI}}(U;O) = \mathrm{H}[U] + \sum_{u \in \mathcal{U}} \Pr[u] \cdot \sum_{o \in \mathrm{DB}_1} \frac{1}{|\mathrm{DB}_1|} \cdot \log_2 \underbrace{\hat{\Pr}[u|o]}_{model}.$$

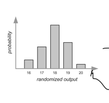
↓ for every u, it has Correctly this o

Trusted DB with sanitized answer

Sanitization:

↳ We will add some noise to the answers of the queries:

pseudo.	smoker	cancer
u1	Y	Y
u2	Y	N
u3	N	N
u4	N	N
u5	Y	Y
u6	N	N
u7	N	Y
...	...	...

Query "# of smoker" → Add Noise →  → Answer with one answer with probability α

Differential Privacy

→ Idea: " For two neighbor DB's (= 1 person has been added or removed) the results are indistinguishable "

→ Definition: ε-DP for Mechanism (= query) Π if for two neibor DB's D_1, D_2 and $c \in \mathrm{range}(\Pi)$ then:

$$\Pr[\Pi(D_1) = c] \leq (1+\varepsilon) \Pr[\Pi(D_2) = c] \quad \text{(are multi close)}$$

↓ (to deal when $\Pr = 0$)

$$(\varepsilon, \delta)\text{-DP} \quad \Pr[\Pi(D_1) = c] \leq (1+\varepsilon) \Pr[\Pi(D_2) = c] + \delta$$

→ Sensitivity: $\Delta f = \max_{D_1, D_2} |f(D_1) - f(D_2)|$

→ Laplacian noise: $b = \frac{\Delta f}{\varepsilon}$ achieves ε-DP

→ Note: • if f_1 - ε_1 DP followed by f_2 - ε_2 then $(\varepsilon_1 + \varepsilon_2)$ DP which grows linearly

• Often the noise needed for privacy will make the data not useful

Untrusted DB

- MPC is better (as seen in previous chapter)
- Fully Homorphic encryption:
  - idea: Dec(f on Enc(m)) = f on m
  - We need additive homorphism (easy)
  - ~~the following~~ multiplicative homorphism which has issues
    - Problem 1: (basic homomorphic) decryption has to know the coefficients of a quadratic polynomial in the variables of $\mathbf{x}$ ⇒ quadratic blowup of ciphertext size
    - Problem 2: If $\sigma^2(c) = \epsilon$, how much is $\sigma^2(c \cdot c')$? ⇒ ciphertexts are rapidly too noisy to be decrypted (*But are highly techical and so far extremely expensive*)