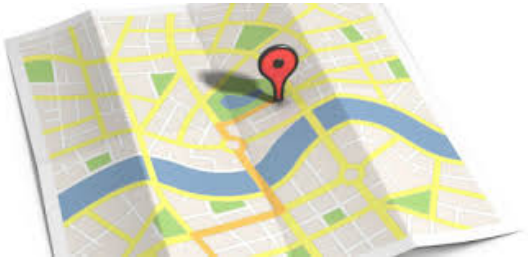
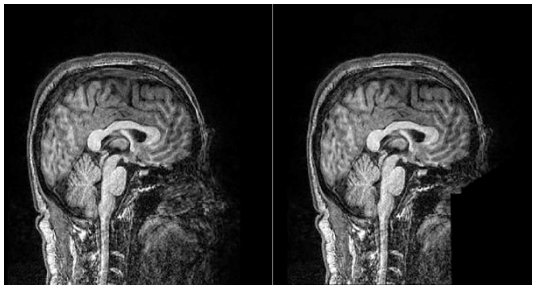
CS422
Database systems

Compression & Privacy

Data-Intensive Applications and Systems (DIAS) Laboratory
École Polytechnique Fédérale de Lausanne

The growth of sensitive data

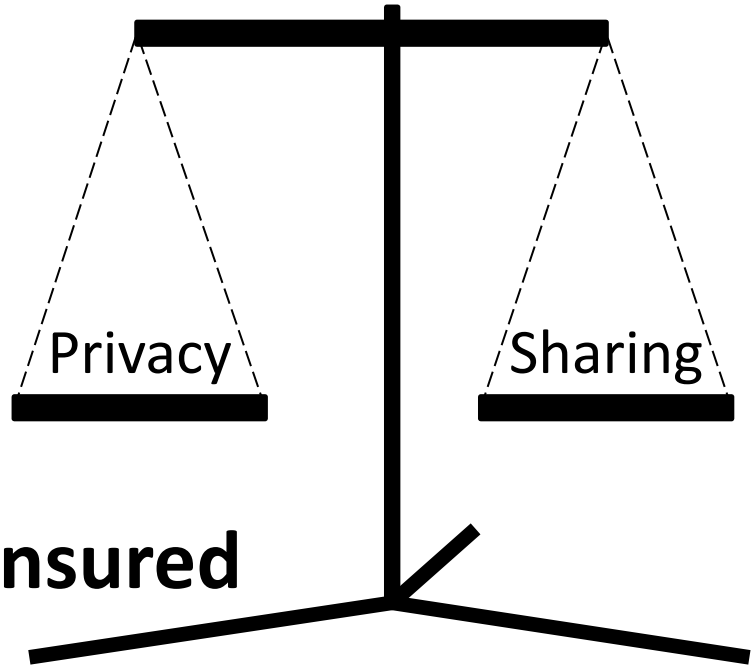




- Data volume and variety is growing
- Some types of data are “sensitive”

To share or not to share?

- We need to protect sensitive data
- Information sharing is valuable
  - Enterprise co-operation
  - Research
  - …



Share as long as privacy is ensured

Privacy-preserving data management

A small note: security is not privacy

Security → who accesses information

$\neq$

Privacy → what information to release

- Related areas but different objectives
- Security protects from direct disclosure

Security vs Privacy

Security

Privacy

tellers can access bank account details – customers cannot spy on other customers

yearly statistics leak specific customer's credit status

You can have security without privacy, but you can't have privacy without security.

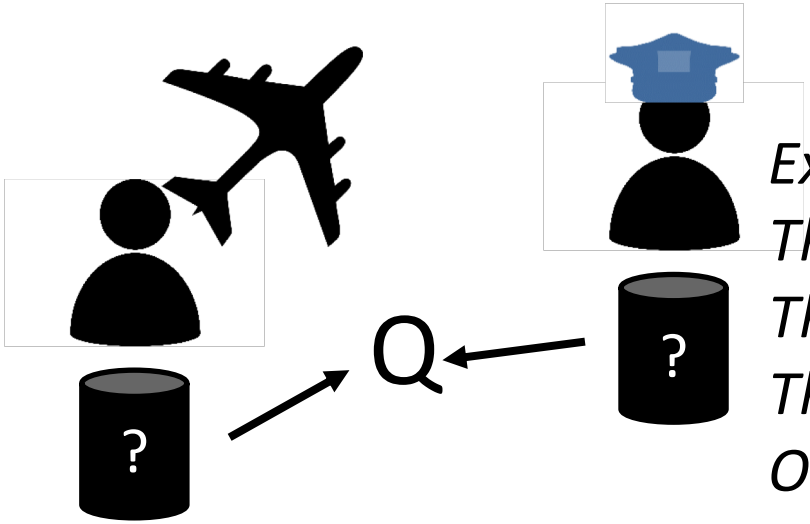
Privacy-preserving data management

- Operations across private databases
- Data Perturbation
- k-Anonymity
- Differential Privacy

Privacy-preserving data management

- **Operations across private databases**
- Data Perturbation
- k-Anonymity
- Differential Privacy

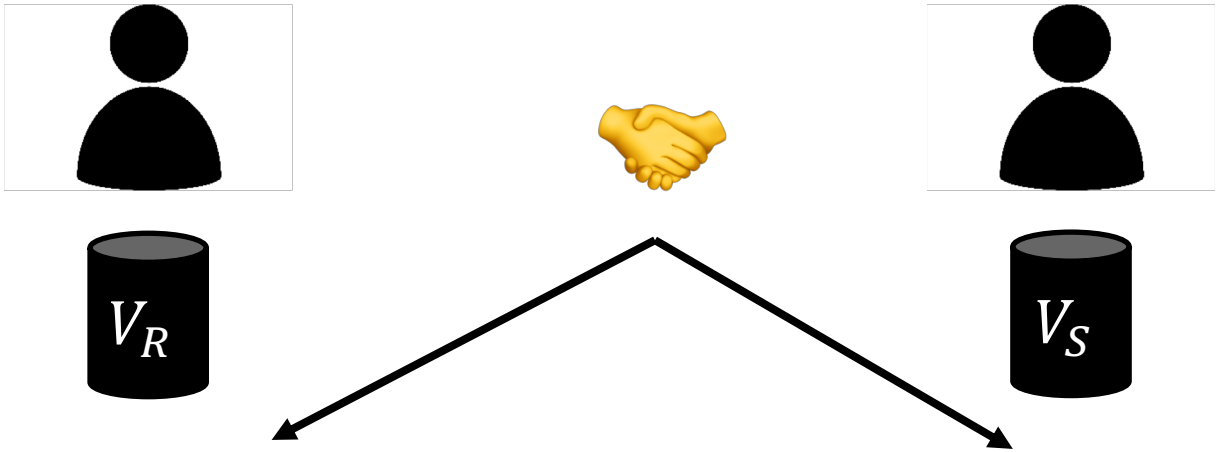
Information sharing across private DBs



Example:
The police checks for fugitive passengers
The police should not know all passengers
The company should not know fugitives
Other actors should learn neither list

- Multiple actors, multiple private DBs
- Share information relevant to operation
- Avoid disclosing any other information
- No trusted third-parties

Model: minimal sharing



R learns result, size of V_S

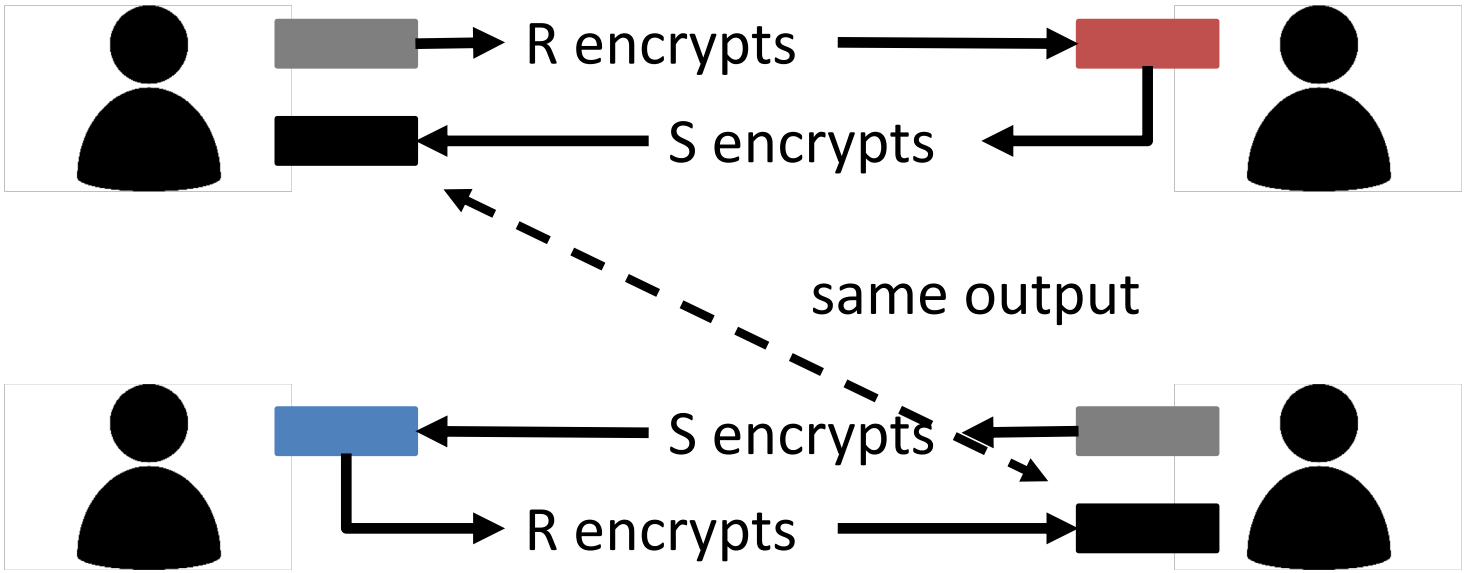
S learns V_R

- Co-operatively evaluate query across DBs
- Enforce privacy with protocols
- Reveal query result + delta (e.g. cardinalities)

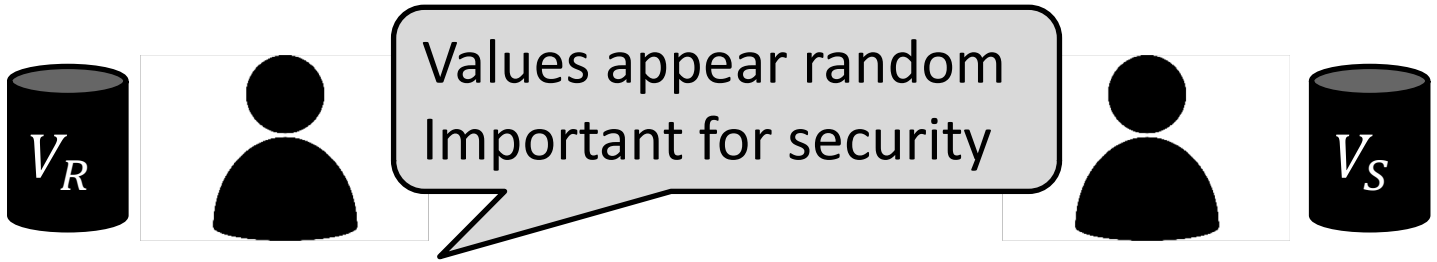
We will see the example of an intersection

A note on commutative encryption

- Apply encryption functions at any order
  - End-result will be the same
- Cornerstone of the protocols



Intersection protocol



Hash $X_R = h(V_R)$

Encrypt $Y_R = f_{e_R}(X_R)$

Send Y_R (re-ordered)

Learn $|V_S|$

Encrypt $Z_S = f_{e_R}(Y_S)$

Hash $X_S = h(V_S)$

Encrypt $Y_S = f_{e_s}(X_S)$

Send Y_S (re-ordered)

Learn $|V_R|$

Encrypt&send $\left(y, f_{e_s}(y)\right), y \in Y_R$

Return tuples with $f_{e_s}(y) \in Z_S$

Replace y with v with $y = f_{e_R}(\mathrm{h(v)})$

Values only matched, never decrypted

Intersection result

Takeaway: Queries across private DBs

- Guarantees limited disclosure
- Different protocol per operation (join, etc)
- Requires honest actors
- Inference from consecutive queries
- Heterogeneity and data integration

Privacy-preserving data management

- Operations across private databases
- **Data Perturbation**
- k-Anonymity
- Differential Privacy

Randomized databases

- Publish data after introducing noise
- Randomization protects privacy
  - Records not factual, adversary cannot suppress noise
  - Options: additive noise (e.g. Gaussian), swap across rows, retention-replacement

id	salary
123	10500
151	3800
199	6900
103	5400

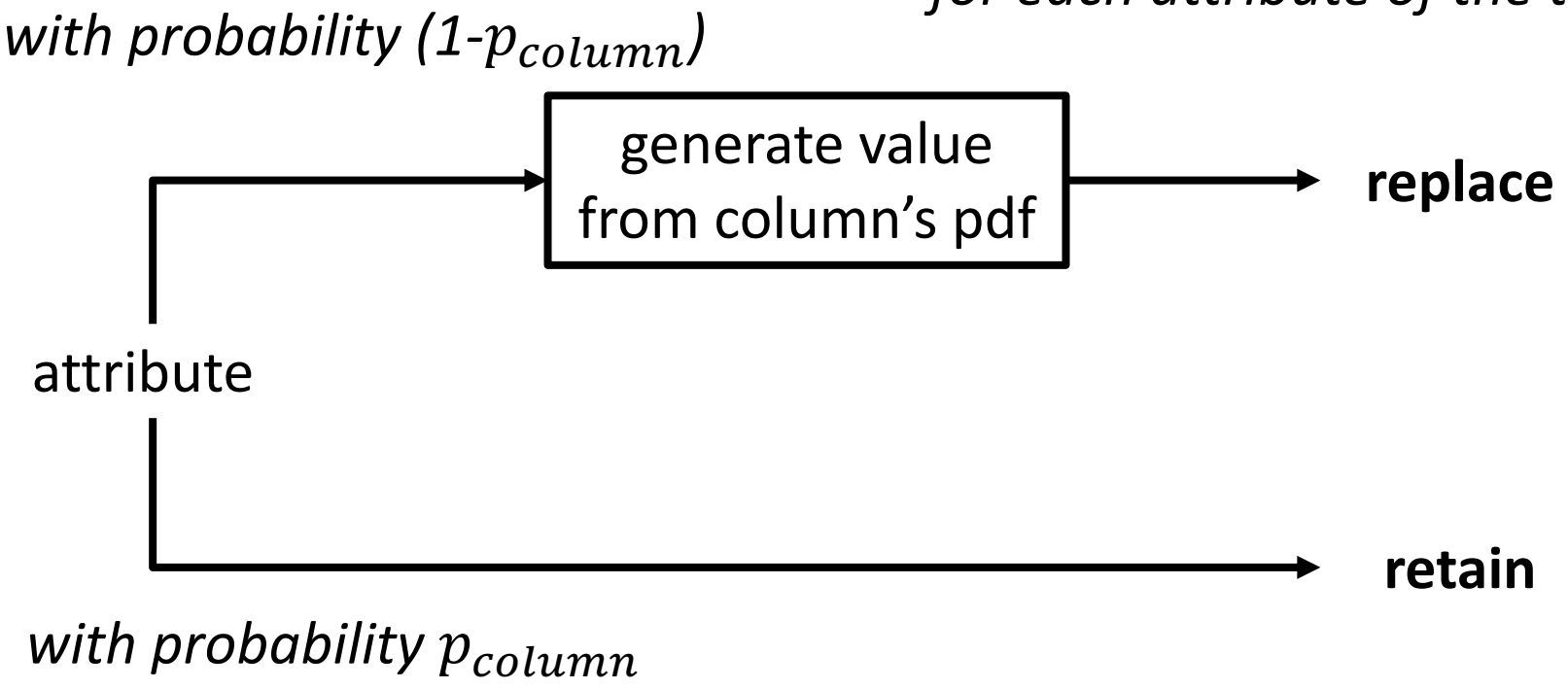
Perturbation Algorithm

id	salary
187	10500
151	7500
133	6900
123	4000

focus on retention-replacement perturbations

Retention-Replacement mechanism

**for each attribute of the table*



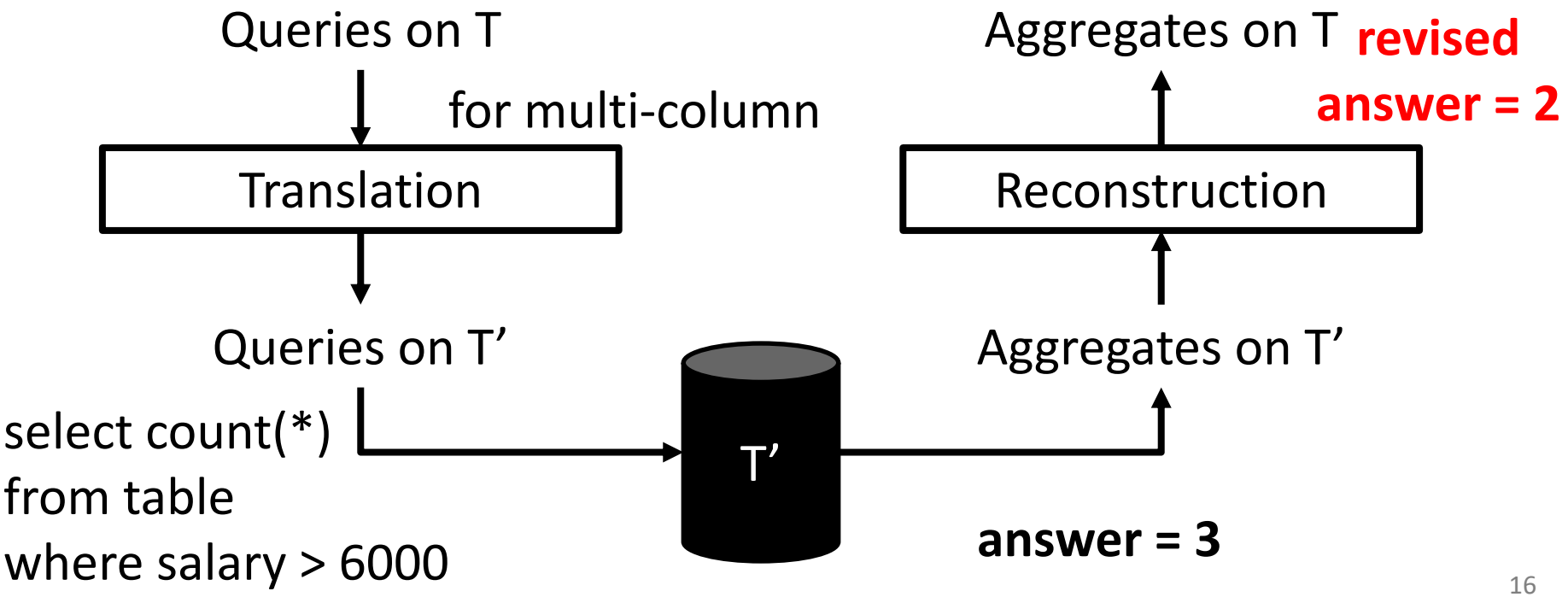
- Adversary cannot differentiate replaced values
- Replacement increases privacy but adds error

Retain some attributes, replace others

Aggregate reconstruction

- Aggregates on perturbed data suffer from error
- Idea: sample of retained data&from replacing pdf
  - remove estimated effect of replacement
  - estimate using retained sample

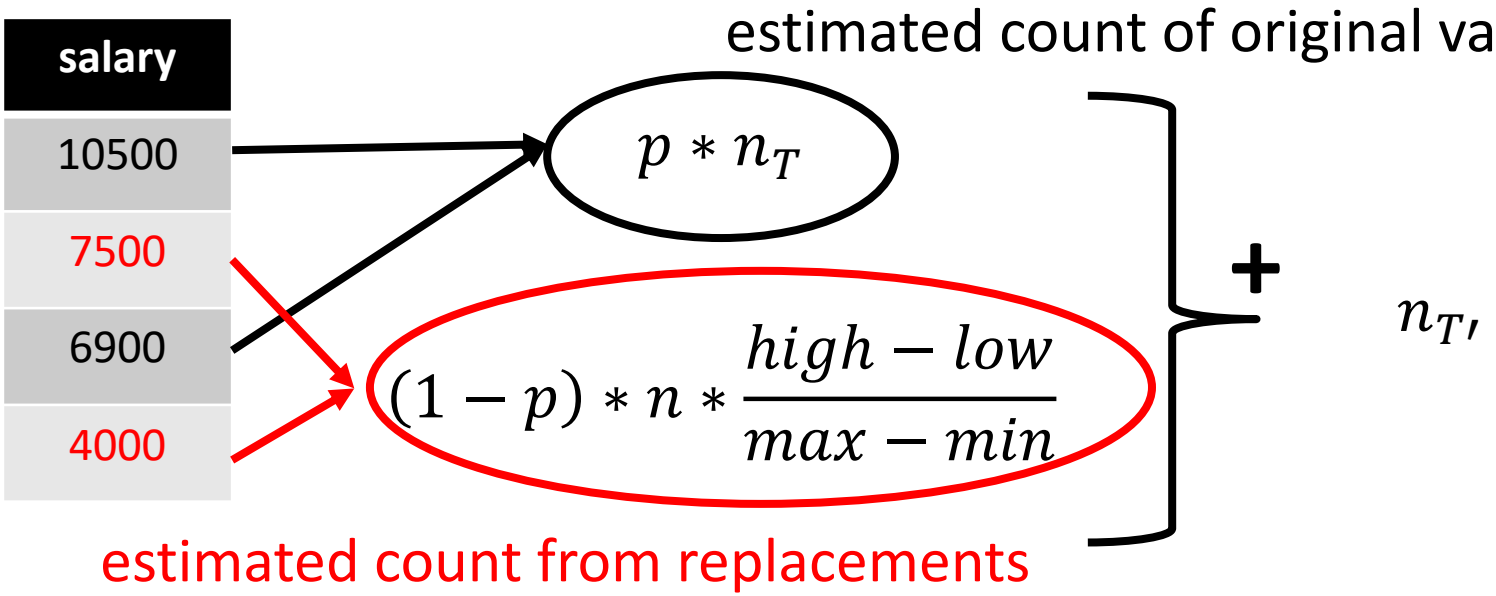
Reconstruction



Reconstructing single-column count

Assuming uniform replacing pdf in [min,max]

select count(*) from table where low<salary<high



$$\text{Estimate } n_T = \frac{n_T' - (1-p)*n*\frac{high-low}{max-min}}{p}$$

Maximum Likelihood, accurate for large n

Reconstructing multi-column count

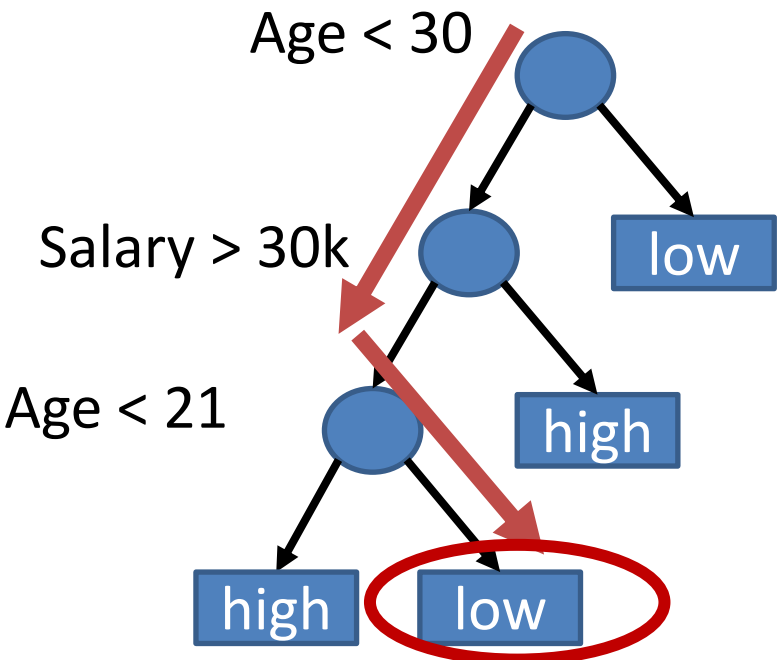
- Significantly more complex
- Requires computing 2^k queries
- Requires on transition matrix A

Query	Aggregates(T)	Aggregates(T')

- Compute **y**
- Compute transition matrix
- Estimate **x**
  - Inverted A
  - Bayesian inference

Applications in data mining

- Data mining (a.k.a. ML) critical for applications
  - regulation restricts use on private data
- Use randomized data to train models
  - error when training on randomized data due to noise
  - **Train model based on reconstructed queries**



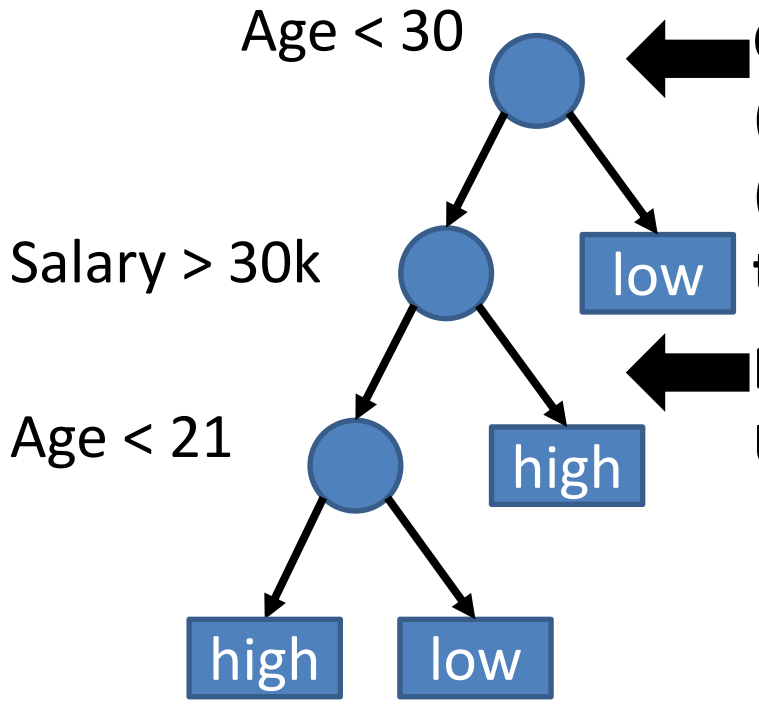
Decision tree for credit risk based on personal info

(115, John, 35k, 24)

Applications in data mining

- Build models i.e. decision trees with **counts**
- Consider class variable as another column

Decision tree for credit risk based on personal info



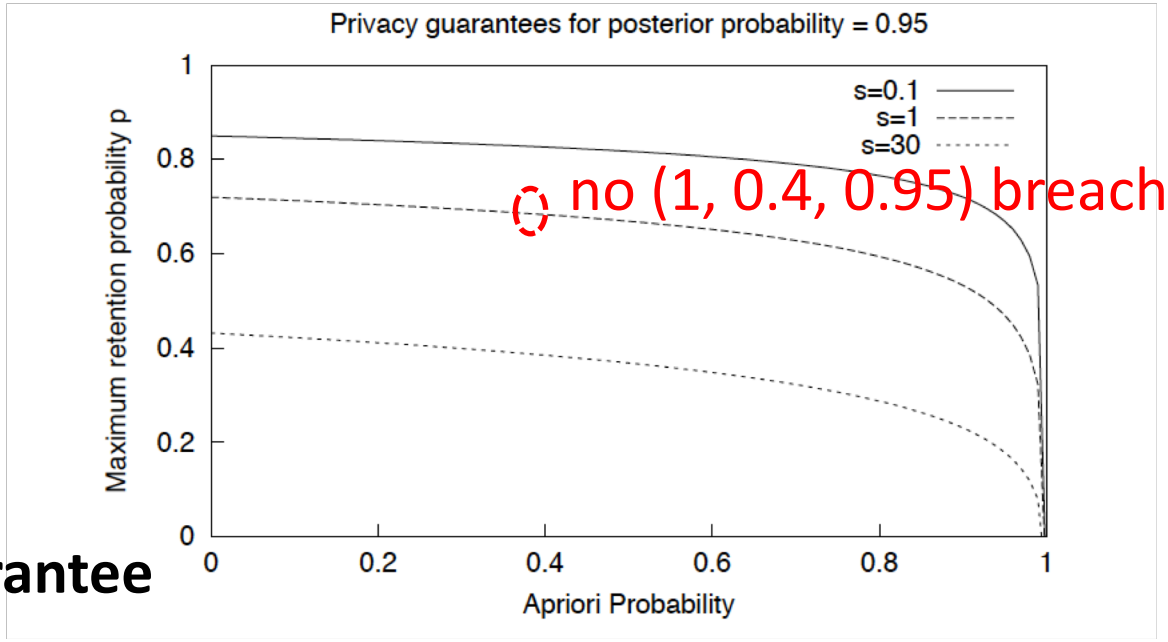
Compute *gini index* with
(Age ≥ 30)∧Low, (Age ≥ 30)∧High,
(Age < 30)∧Low, (Age < 30)∧High
to decide splits

Do not partition perturbed data
Use all predicates in the path

Randomization guarantees

- We use the notion of (s, p_1, p_2) privacy breach
  - Observing an event with an a priori probability of at most p_1 entails an a posteriori probability of at least p_2
  - An event is at most a factor s more common for unperturbed data compared to replacing distribution

query with predicate on 3-columns



Set *p* for the desired guarantee

Privacy-error tradeoff

Take-away: Data Perturbation

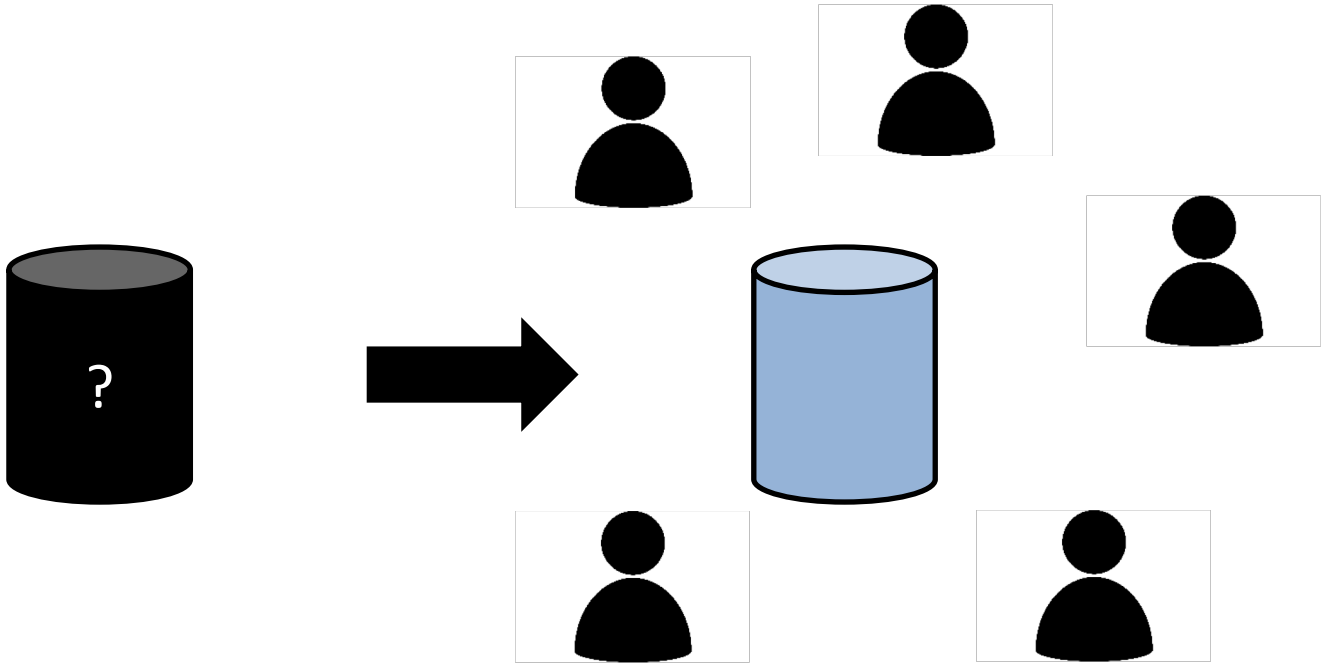
- Publish a randomized dataset
- Simpler to query
- Compose complex procedures
- Other retention replacement schemes
- Extensible for categorical data
- Breaks data integrity

Privacy-preserving data management

- Operations across private databases
- Data Perturbation
- **k-Anonymity**
- Differential Privacy

Publishing sensitive data

- Publish factual data useful for analysis
- Avoid identification of people with records
- Cannot we just remove candidate keys?



Suppressing identifiers

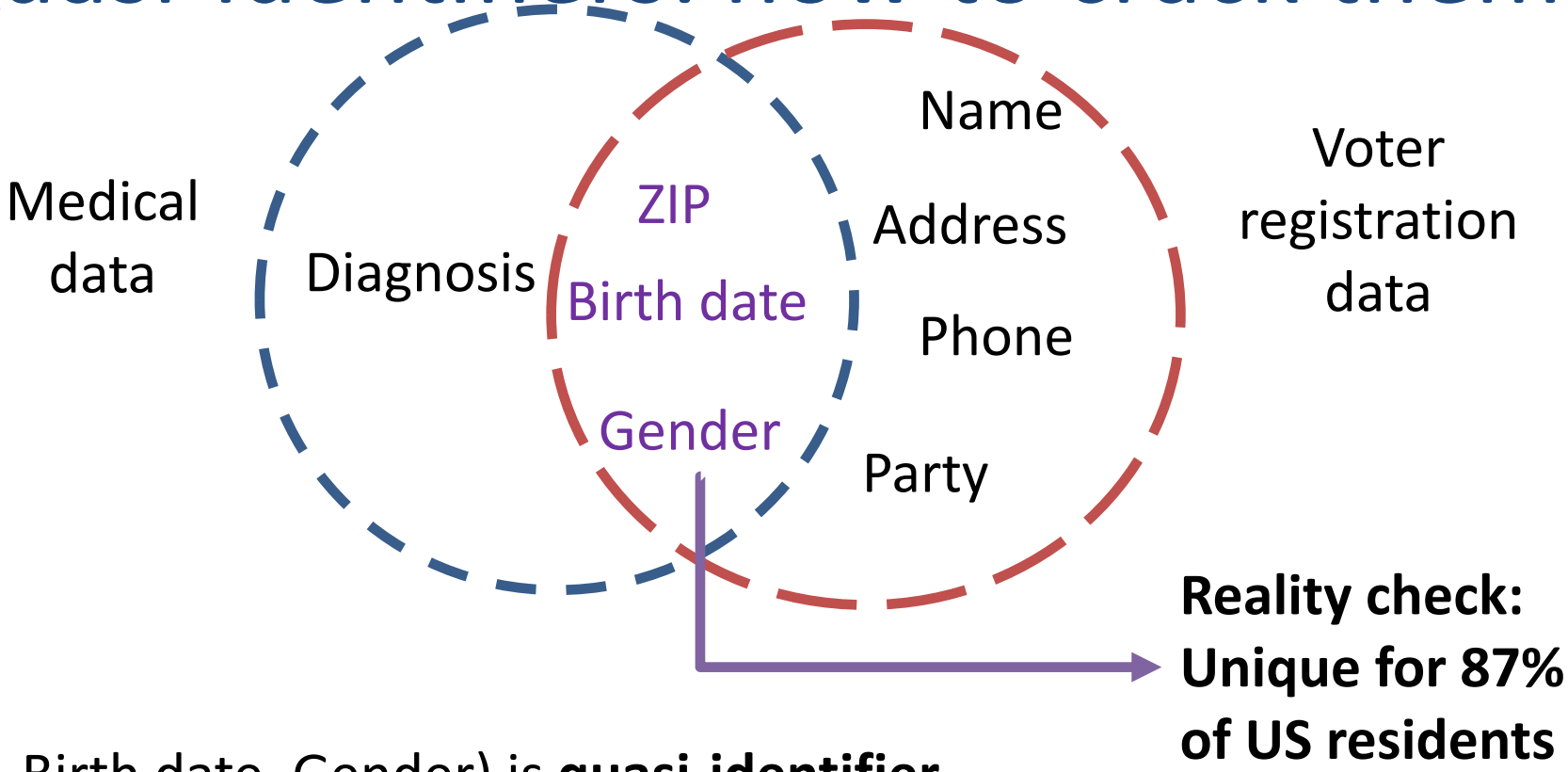
Common approach: remove candidate keys from the relations

Name	Address	Phone#	Birth date	ZIP	Gender	Diagnosis
Alice	..	..	01/02/98	1006	F	Flu
Bob	..	..	03/08/99	1005	M	Fracture
John	..	..	07/12/73	1004	M	Cancer
Mary	..	..	01/08/85	1006	F	Hepatitis

Birth date	ZIP	Gender	Diagnosis
01/02/98	1006	F	Flu
03/08/99	1005	M	Fracture
07/12/73	1004	M	Cancer
01/08/85	1006	F	Hepatitis

Is this approach sufficient?

Quasi-Identifiers: how to crack them



(ZIP, Birth date, Gender) is **quasi-identifier**

"William Weld was governor of Massachusetts at that time and his medical records were in the GIC data. Governor Weld lived in Cambridge Massachusetts. According to the Cambridge Voter list, six people had his particular birth date; only three of them were men; and, he was the only one in his 5-digit ZIP code."

Linking to other data breaks anonymity

k-Anonymity

- At least k occurrences for quasi-identifier tuple
- Linking attacks: at least k candidates ($\frac{1}{k}$ probability)
- Achieve with suppression and generalization

Nationality	Birth year	Gender	Diagnosis
Spanish	1965	m	Short breath
Spanish	1965	m	Chest pain
French	1975	f	Hypertension
French	1975	f	Hypertension
Swiss	1964	m	Obesity
Swiss	1964	m	Chest pain
Swiss	1964	m	Short breath

equivalence class

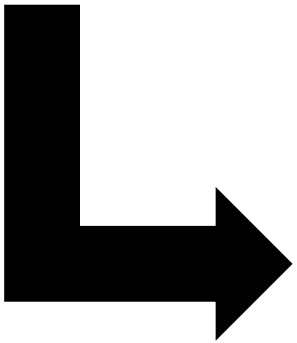
2-anonymity example

Protect from linking attacks

Generalizing attributes

Birth date	ZIP	Gender	Diagnosis
01/02/98	1006	F	Flu
03/08/99	1005	M	Fracture
07/12/73	1004	M	Cancer
01/08/85	1006	F	Hepatitis

- Generalize to value range
- Attribute still usable
- Achieve k-anonymity



Birth date	ZIP	Gender	Diagnosis
01/**/**	1006	F	Flu
//**	[1000-1005]	M	Fracture
//**	[1000-1005]	M	Cancer
01/**/**	1006	F	Hepatitis

More fine-grained than suppression

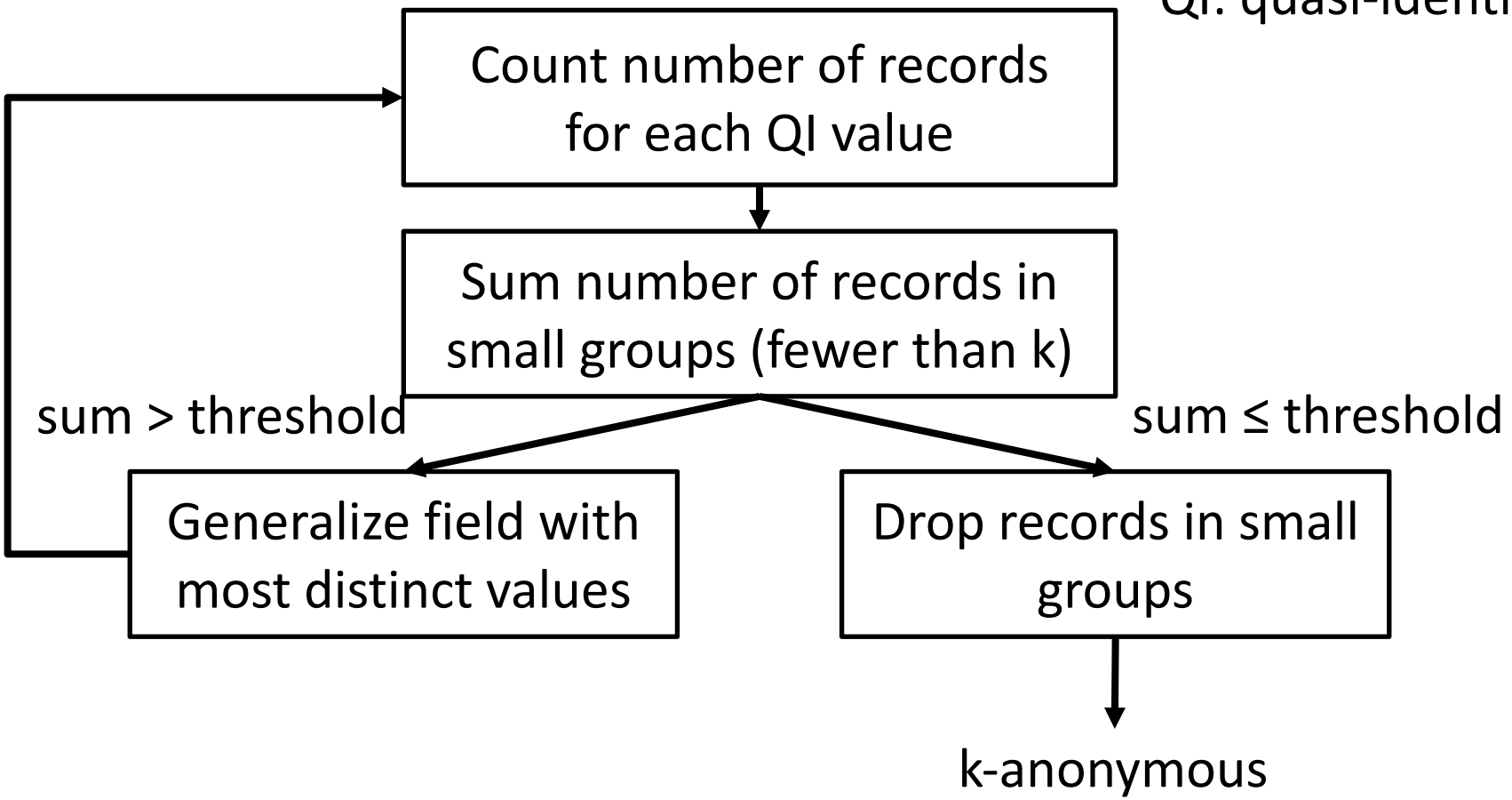
How to achieve k-anonymity

- Minimal k-anonymization is most useful
- Minimal k-anonymization is NP-hard
  - Optimal heuristic algorithms e.g. k-Optimize
  - Suboptimal but practical algorithms e.g. **Datafly**

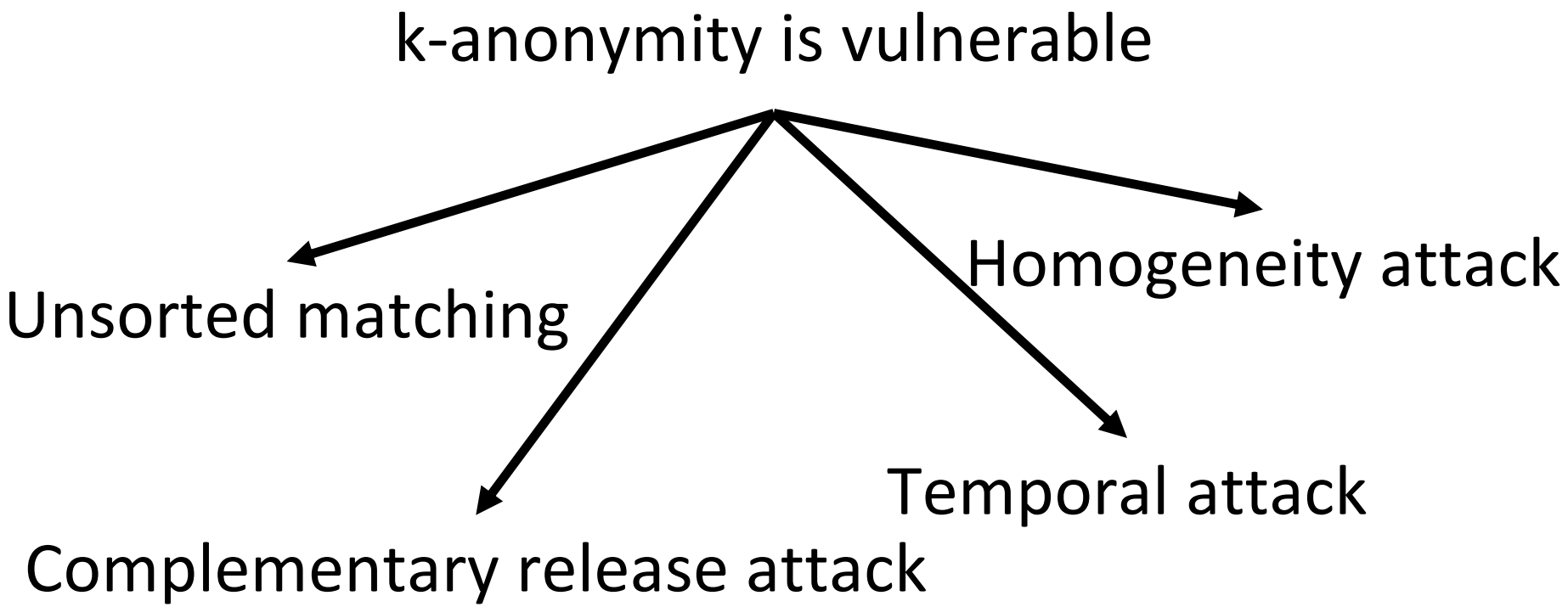
Datafly: An Overview

- Generalize infrequent QI values
- Drop when outliers are few

QI: quasi-identifier



Attacks on k-anonymity



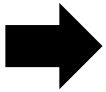
Unsorted matching

- In relational model, we cannot assume order
  - In practice, for DBMSs, we can

Nationality	Birth year
Spanish	196*
Spanish	196*
French	197*
French	197*
Swiss	196*
Swiss	196*
Swiss	196*

+

Nationality	Birth year
W. Europe	19*5
W. Europe	19*6
W. Europe	19*5
W. Europe	19*6
C. Europe	196*
C. Europe	196*
C. Europe	196*



Nationality	Birth year
Spanish	1965
Spanish	1966
French	1965
French	1976
Swiss	196*
Swiss	196*
Swiss	196*

Tuple re-ordering protects from matching

Complementary release attack

- Different quasi-identifiers across releases
- Other releases are also external sources

Nationality	Birth year	Gender	Diagnosis
Spanish	196*	m	Hypertension
Spanish	196*	m	Chest pain
Swiss	197*	f	Fever
Swiss	197*	f	Vomitting

Linking on diagnosis reverses generalization of values

Birth year	Gender	Diagnosis
1965	m	Hypertension
1965	m	Chest pain
1974	f	Vomiting
1974	f	Fever

Include sensitive attributes in quasi-identifier

Temporal attack

- Different k-anonymity scheme for updated table

Nationality	Birth year	Gender	Diagnosis
Spanish	196*	m	Hypertension
Spanish	196*	m	Chest pain
Swiss	197*	f	Fever
Swiss	197*	f	Vomitting

Anonymize released tables
+ updated records
-cannot leak extra info

Birth year	Gender	Diagnosis
1965	m	Hypertension
1965	m	Chest pain
1974	f	Fever
1974	f	Vomiting
19*3	f	Hypertension
19*3	f	Headache

Make new release compatible with older ones

Homogeneity attack

Nationality	Birth year	Gender	Diagnosis
Spanish	1965	m	Cancer
Spanish	1965	m	Cancer
French	1975	f	Hypertension
French	1975	f	Hypertension
Swiss	1964	m	Obesity
Swiss	1964	m	Chest pain
Swiss	1964	m	Chest pain

- Sensitive values lack diversity ⇒ inference
- Require multiple well-represented values per class

  ⇒ probabilistic inference
- Require evenly distributed values

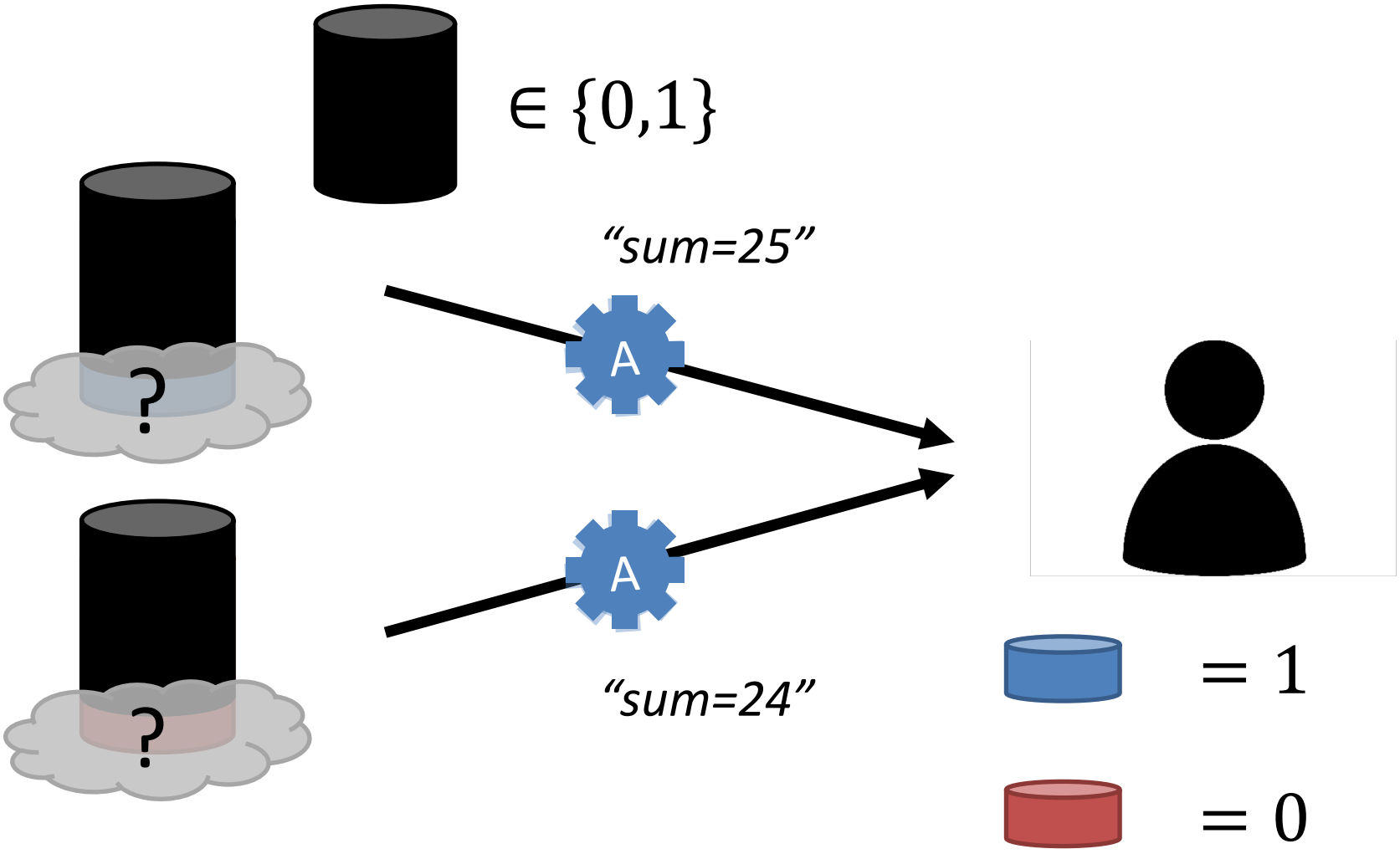
Take-away: k-anonymity

- Robust privacy guarantees
- Complexity of minimal k-anonymity
- Vulnerable to range of attacks

Privacy-preserving data management

- Operations across private databases
- Data Perturbation
- k-Anonymity
- **Differential Privacy**

Differential privacy

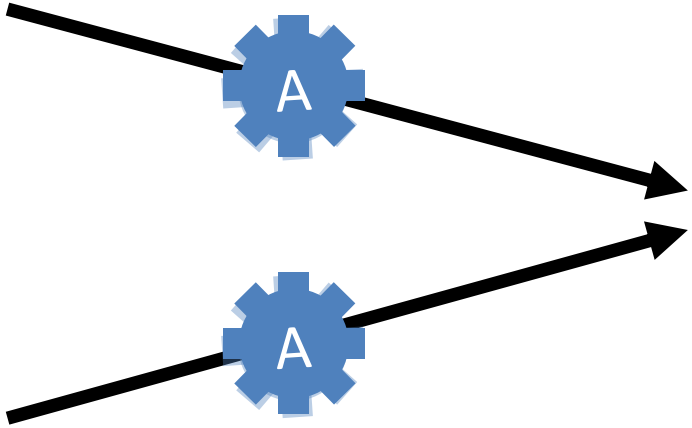


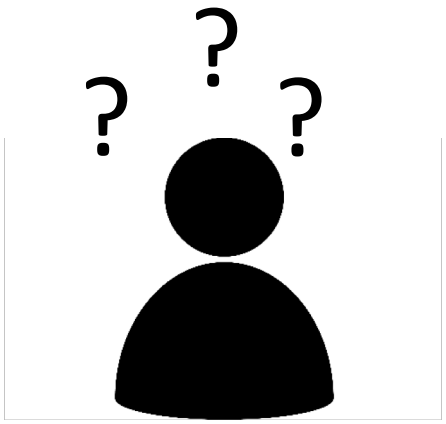
Privacy breach

Differential privacy

A randomized algorithm is differentially private if the difference in the results is statistically insignificant for neighboring databases

Neighboring=differ in at most one element



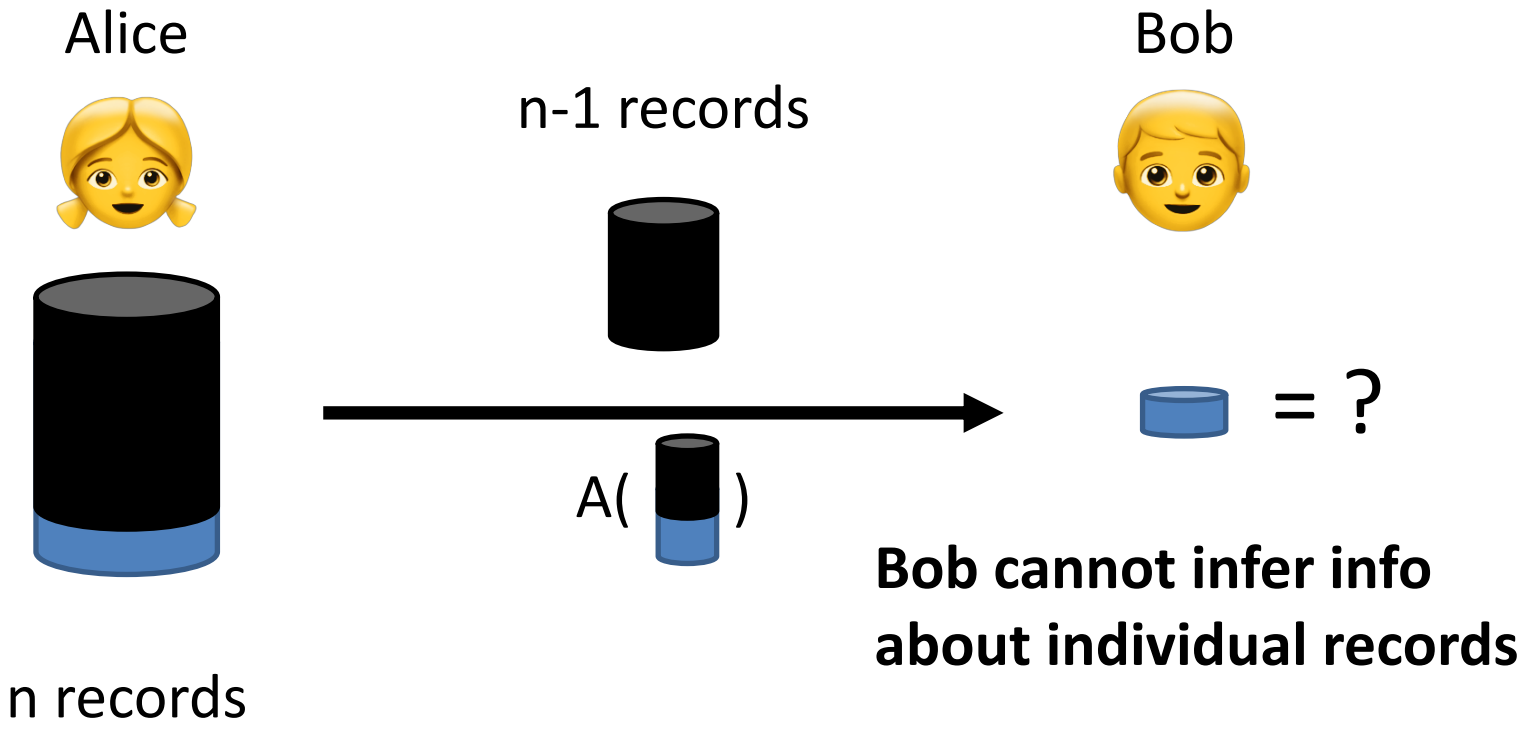


A is ε-differentially private iff

$$\frac{P_r\left(A(D_1)\in Y\right)}{P_r\left(A(D_2)\in Y\right)} \leq e^{\varepsilon}$$

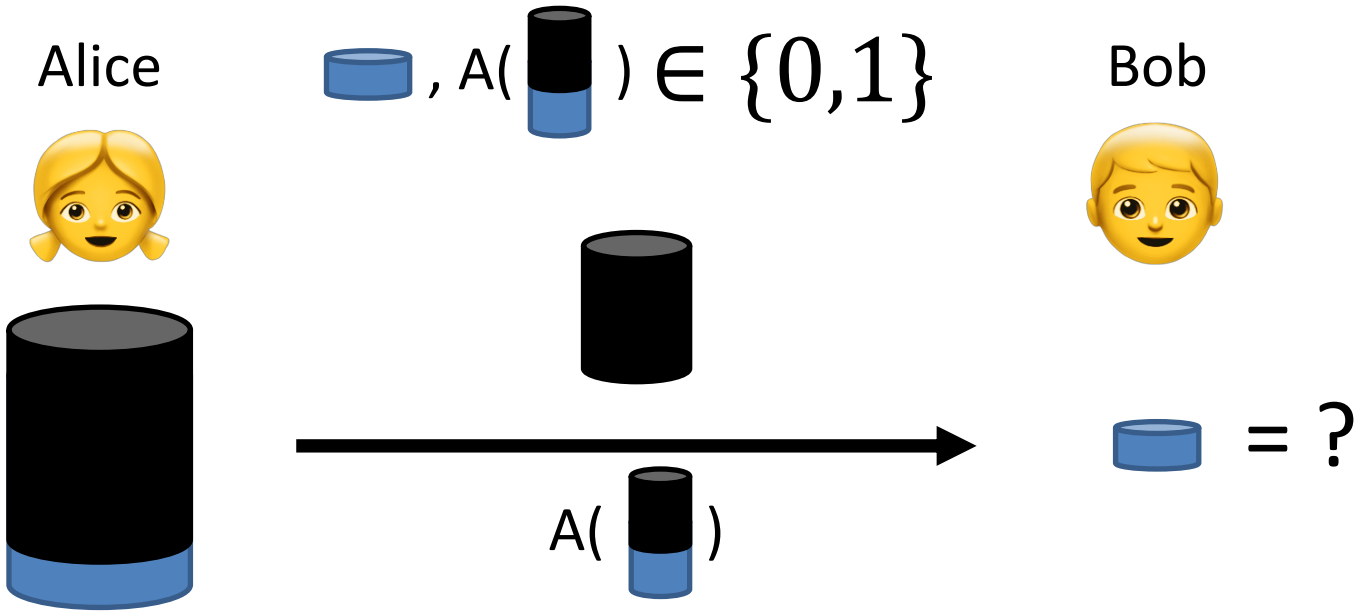
More rigorous privacy guarantees

The strength of differential privacy



Extrapolation as accurate as random guess

Differential Privacy: 0-1 example



With differential privacy: $P_r(Bob\ correct) \leq \frac{1}{2} + \frac{\varepsilon}{2}$

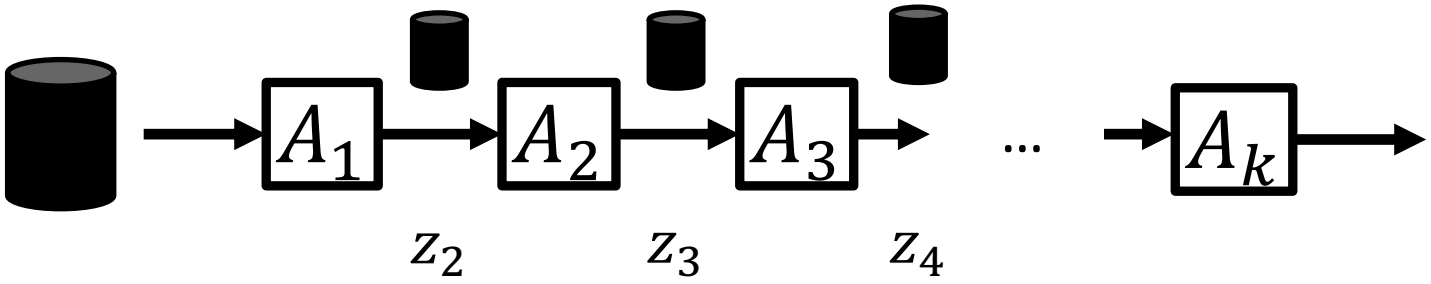
Without differential privacy: $P_r(Bob\ correct) = 1$

under some circumstances…

Extrapolation as accurate as random guess

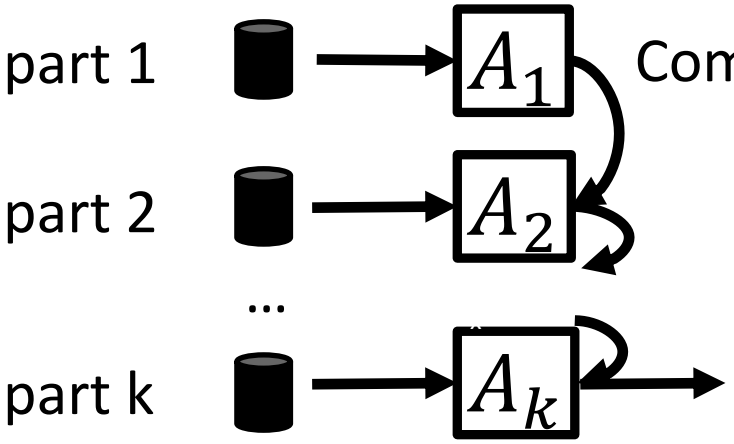
Composability

- Sequential composability



Composition result is kε-differentially private

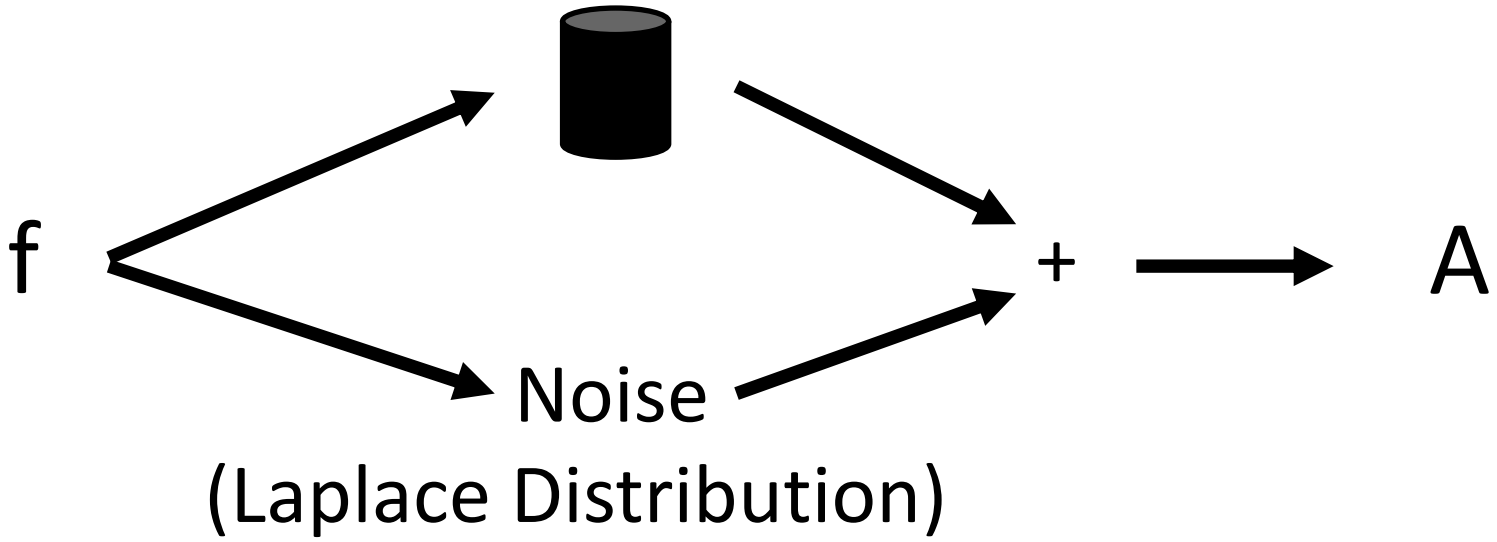
- Parallel composability



Composition result is ε-differentially private

How to achieve differential privacy

- Differential privacy has desired properties
  - But how do we design such algorithms?
- We can use the Laplace Mechanism



Randomized algorithm is differentially private

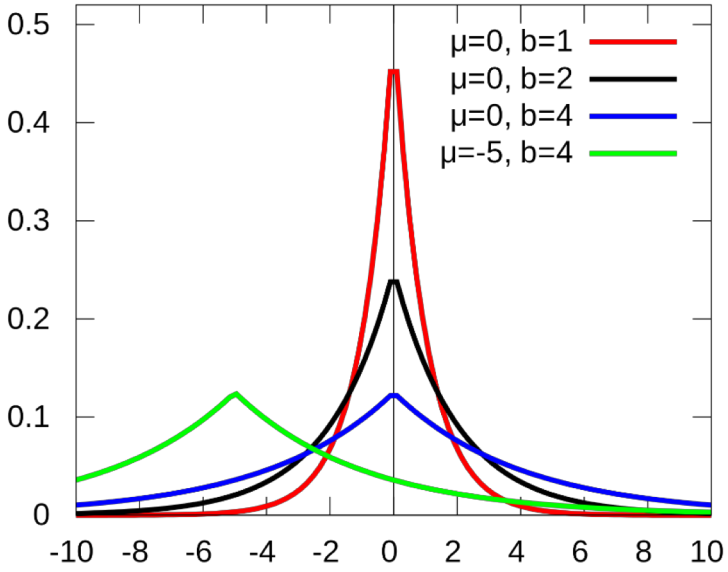
Global noise sensitivity

- Required noise depends on function properties
- *Global noise sensitivity*: distance of neighbors
- Produce noise with parameterized distribution

$$\boldsymbol{GS(f) = sup_{D_1, D_2 \in D^n} ||f(D_1) - f(D_2)||_1}$$

for neighboring D_1, D_2, then

$$\boldsymbol{A(D) = f(D) + Lap(\frac{GS(f)}{\varepsilon})}$$



Laplace mechanism-based algorithms

- Randomized algorithms
  - Noisy summation
  - Noisy average
  - Noisy linear regression

  …

sum ✗

sum + noise ✓

- Laplace mechanism is pessimistic
  - Perturb objective function
  - Use Euclidean norm for vector-based

Take-away: differential privacy

- Strongest guarantees for the worst-case
- Privacy-error tradeoff
- Pessimistic in nature
- Potential for relaxed definitions
- Potential for specialized algorithms

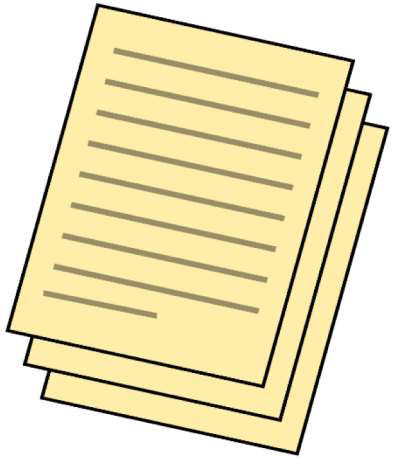
Readings in Privacy

- R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *SIGMOD*, 2003.
- R. Agrawal, R. Srikant, and D. Thomas. Privacy preserving OLAP. In *SIGMOD*, pages 251–262, 2005.
- L. Sweeney. K-Anonymity: a model for protecting privacy. Intl. Journal on Uncertainty, Fuzziness and Knowledge- based Systems, 10(5), 2002.
- L. Sweeney. Datafly: A system for providing anonymity in medical data. In Proceedings of Eleventh International Conference on Database Security, pages 356-381. Database Security XI: Status and Prospects, 1998.
- C. Dwork et al, Calibrating Noise to Sensitivity in Private Data Analysis, In Theory of cryptography conference, 2016.

OTHERSIDE

RED HOT CHILI PEPPERS

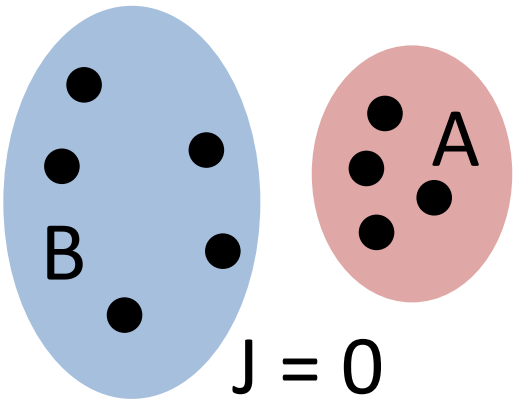
Finding similar items

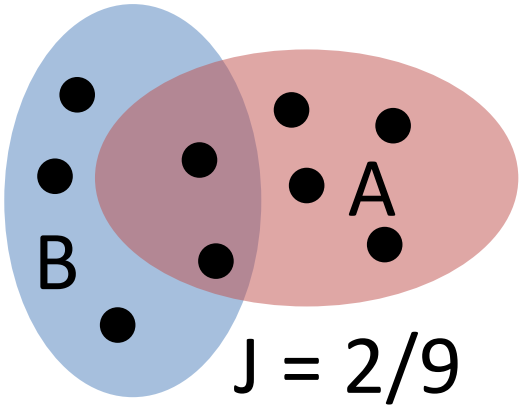


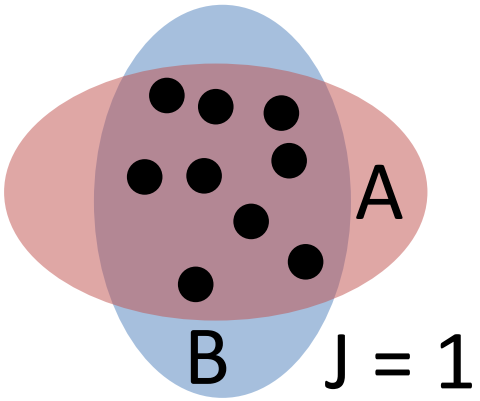
Jaccard similarity

- Similarity measure between two **sets**
- Usage: “Find similar (word-wise) articles”
- Ratio of common items over union of items

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$







Sets to Bitvectors

- Logically, represent sets as bitvectors
- Each bit corresponds to a possible element
- Bit is set iff element is contained in the set

A = {a, b, c}

B = {b, c, f}

	A	B
a	1	0
b	1	1
c	1	1
d	0	0
e	0	0
f	0	1

J(A, B) = 2/4

- J(A, B) = popcnt(A & B)/popcnt(A | B)

From a bitvector to summarization

- h(C) = index of first row where C is 1

A = {a, b, c}

B = {b, c, f}

	A	B
a	**1**	0
b	1	**1**
c	1	1
d	0	0
e	0	0
f	0	1

h(A) = 0 (a)

h(B) = 1 (b)

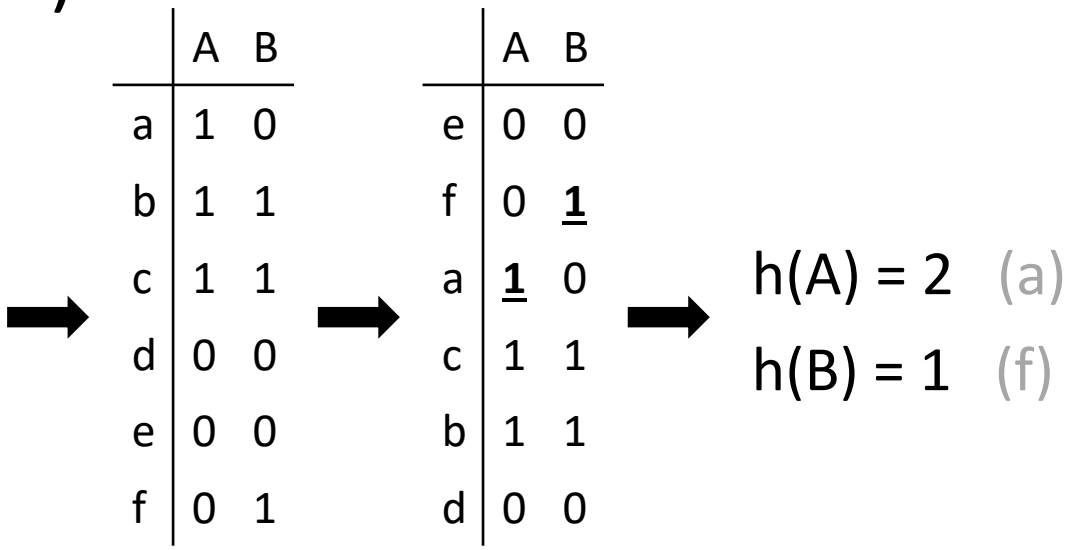
J(A, B) = 2/4

- ceil(log(N)) space per hash, N = |Ω|

Minhashing

- **Randomly** permute rows
- Minhash: h(C) = index of first row where C is 1

A = {a, b, c}
B = {b, c, f}



	A	B
a	1	0
b	1	1
c	1	1
d	0	0
e	0	0
f	0	1

	A	B
e	0	0
f	0	**1**
a	**1**	0
c	1	1
b	1	1
d	0	0

h(A) = 2 (a)
h(B) = 1 (f)

- Considering all permutations:

$$P(h(A) = h(B)) = J(A, B)$$

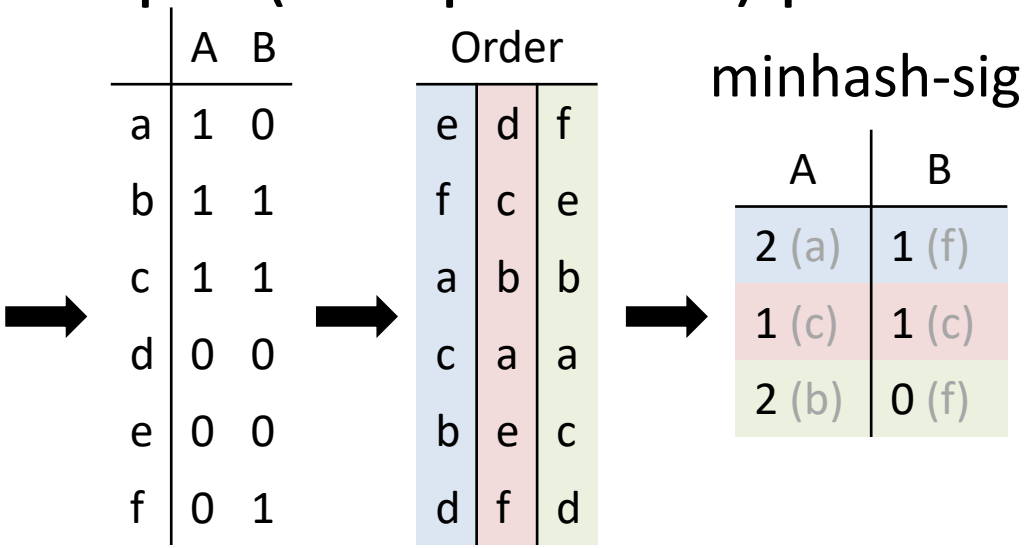
- ceil(log(N)) space per hash, N = $|\Omega|$

Minhash Signature

- Repeat for multiple (independent) permutations

A = {a, b, c}

B = {b, c, f}



	A	B
a	1	0
b	1	1
c	1	1
d	0	0
e	0	0
f	0	1

Order

e	d	f
f	c	e
a	b	b
c	a	a
b	e	c
d	f	d

minhash-sig

A	B
2 (a)	1 (f)
1 (c)	1 (c)
2 (b)	0 (f)

- Signature similarity = fraction of equal hashes

Sim(minhash-sig(A), minhash-sig(B)) = 1/3

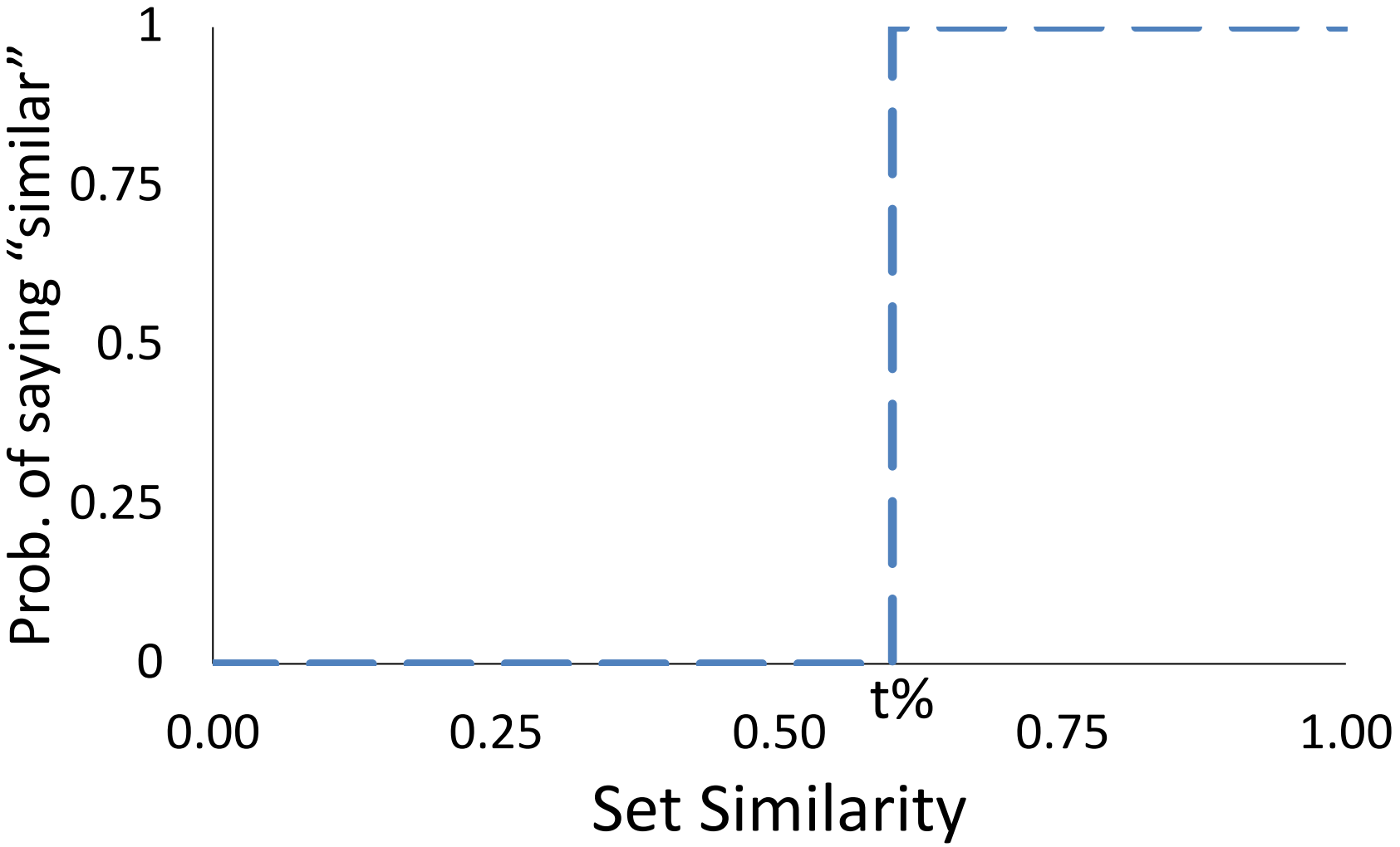
- Expected sig similarity = Jaccard similarity

E[Sim(minhash-sig(A), minhash-sig(B))] = J(A, B)

- Expected error decreases as sig size increases

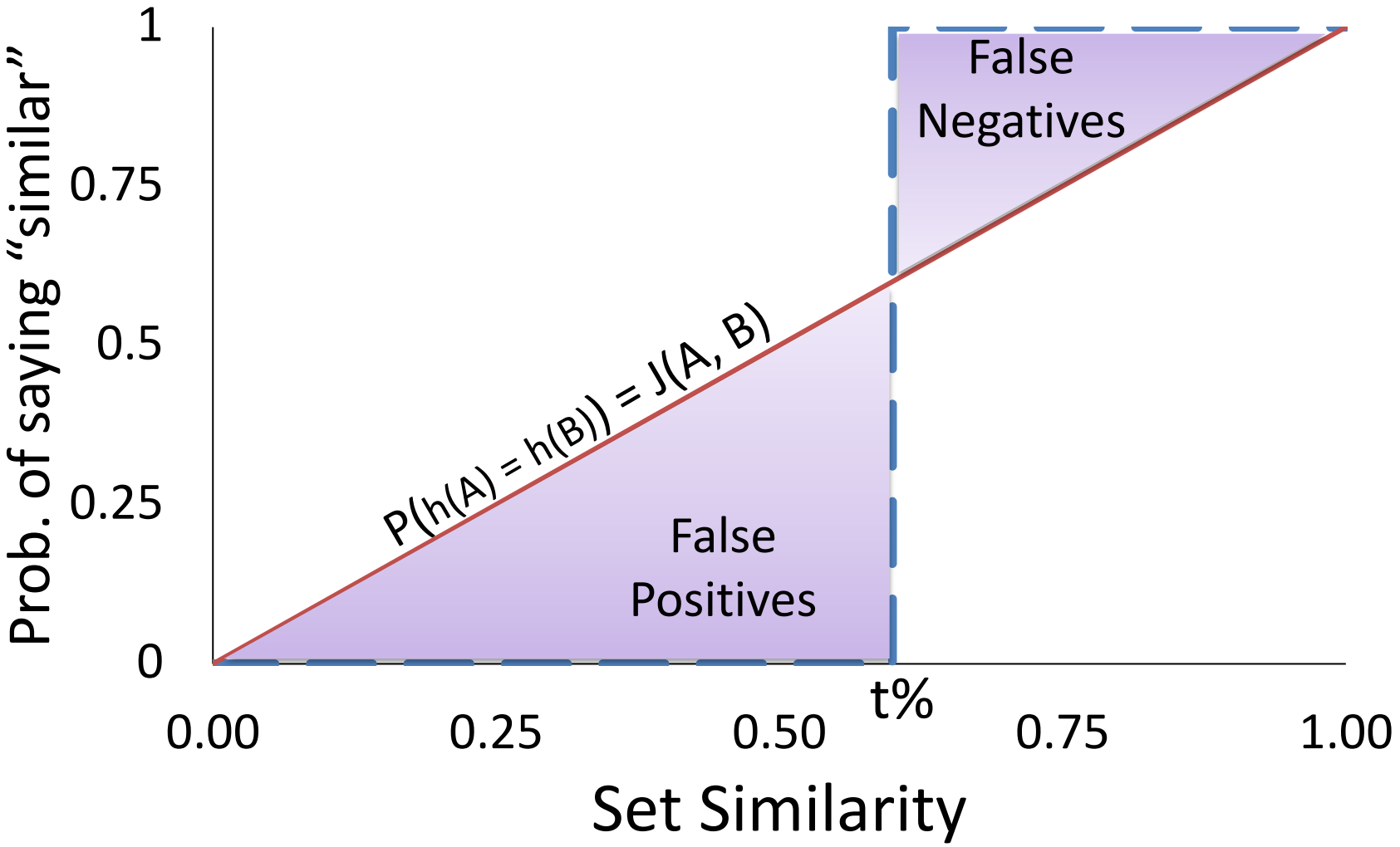
Finding similar pairs -- Ideally

- "Find pairs of sets with at least t% similarity"



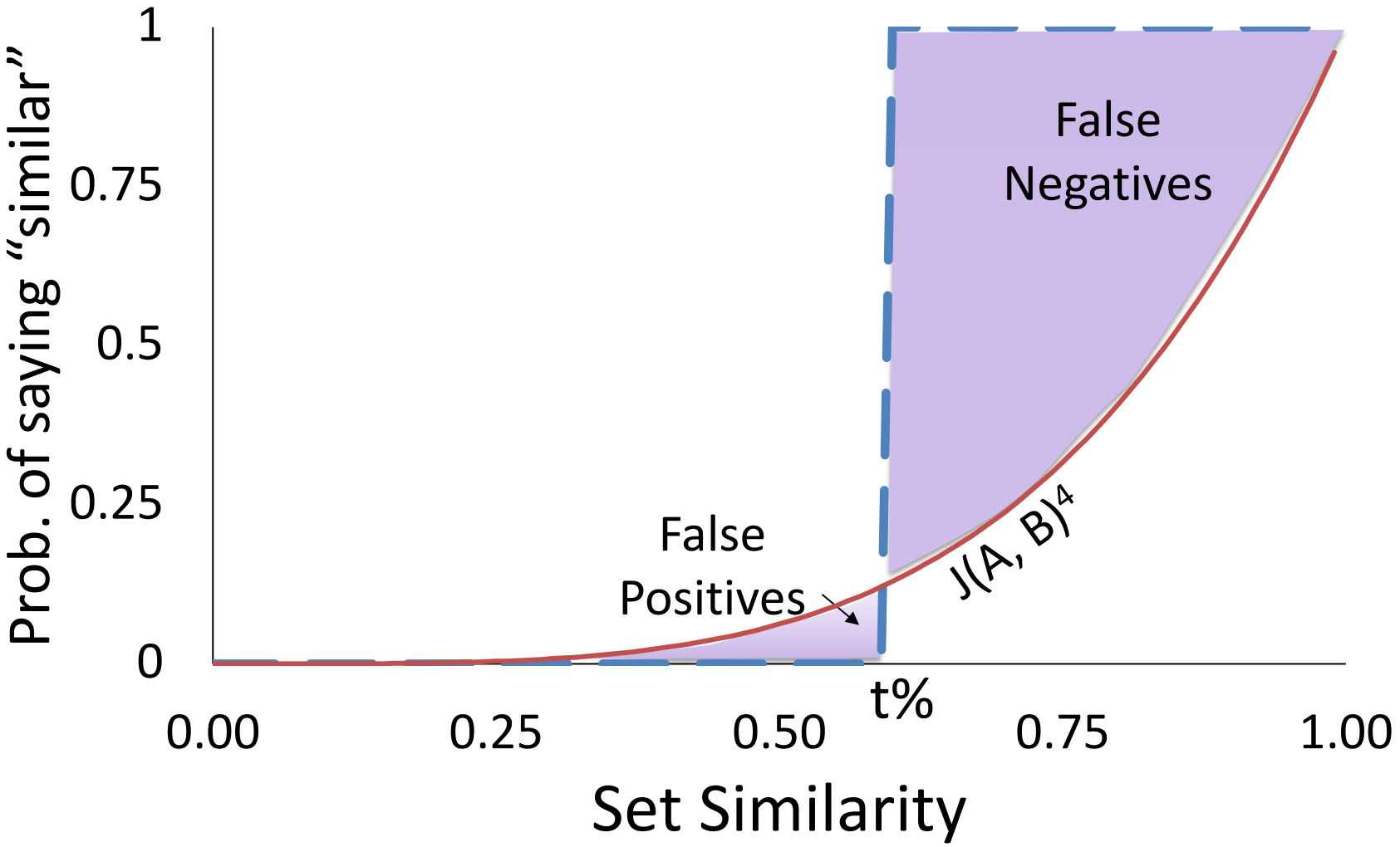
Finding similar pairs using Minhash

- Answer similar if minhashes match



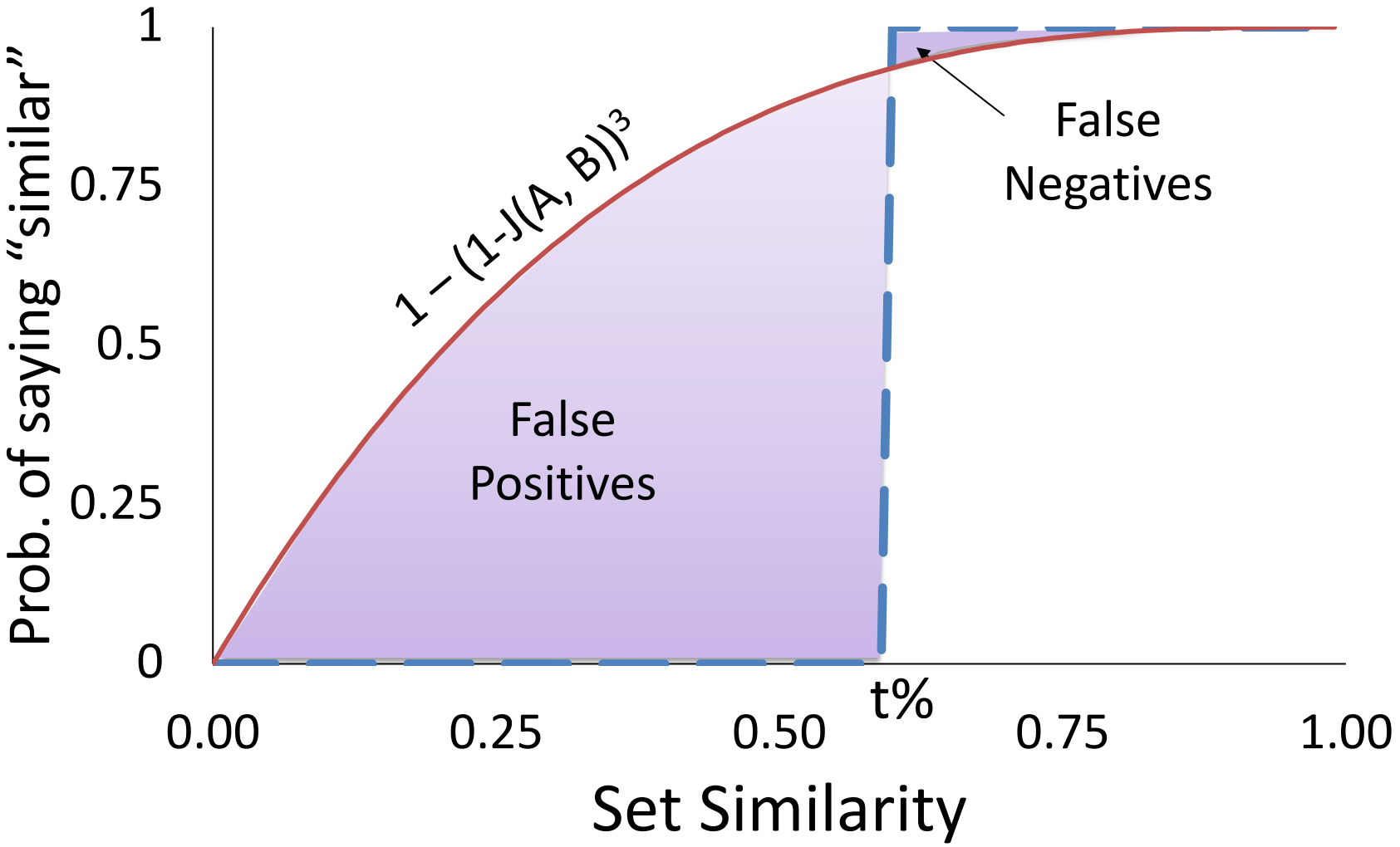
Reduce False Positives – "AND"

- Answer "similar" if all minhash match



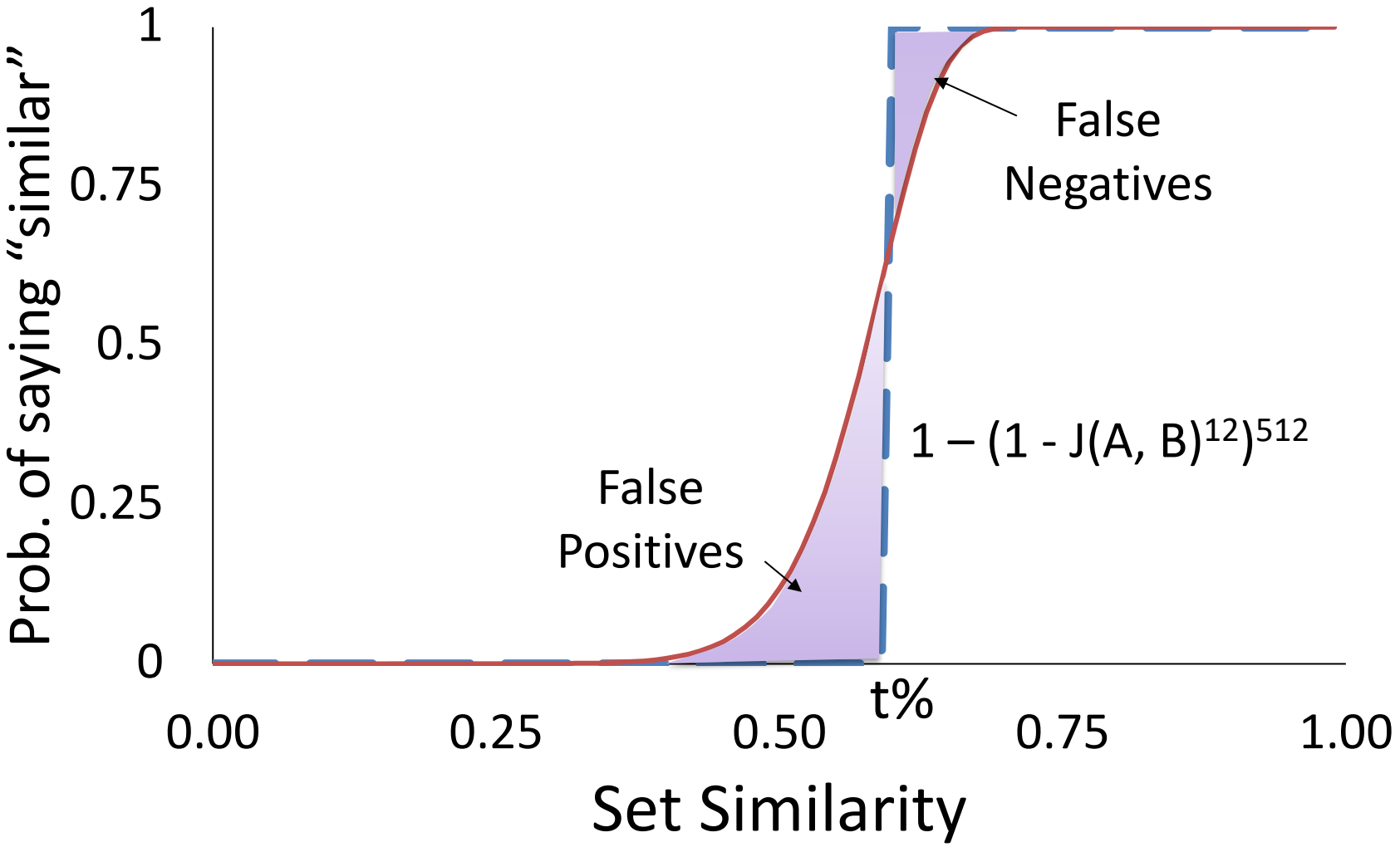
Reduce False Negatives – "OR"

- Answer "similar" if any minhash match



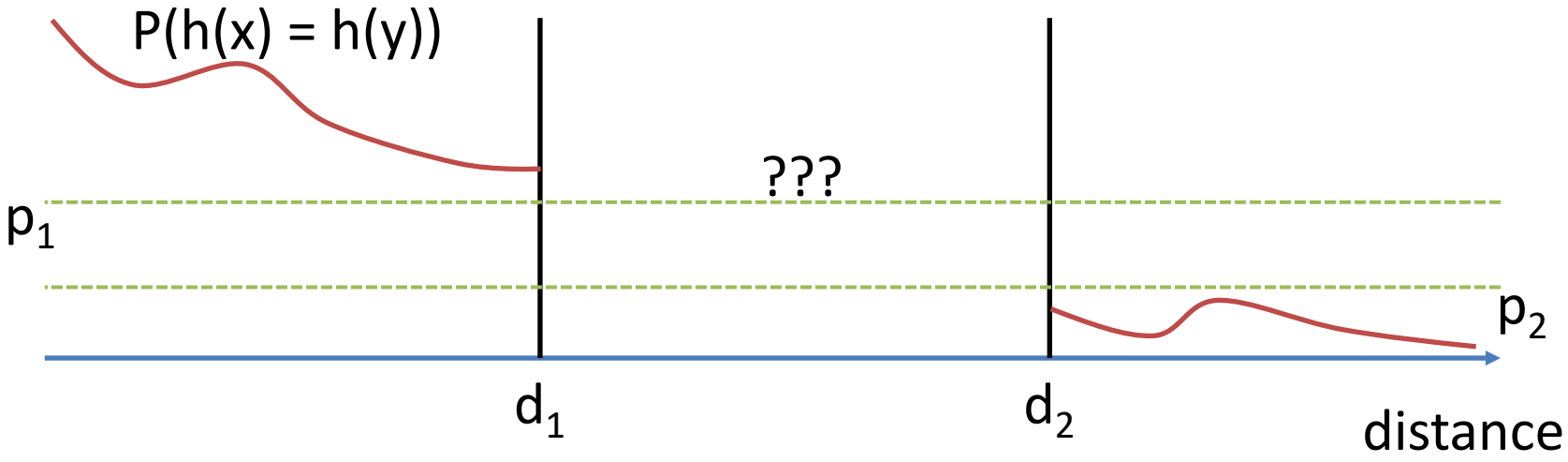
Reduce False Answers – AND+OR

- "similar" if any group matches in all minhashes



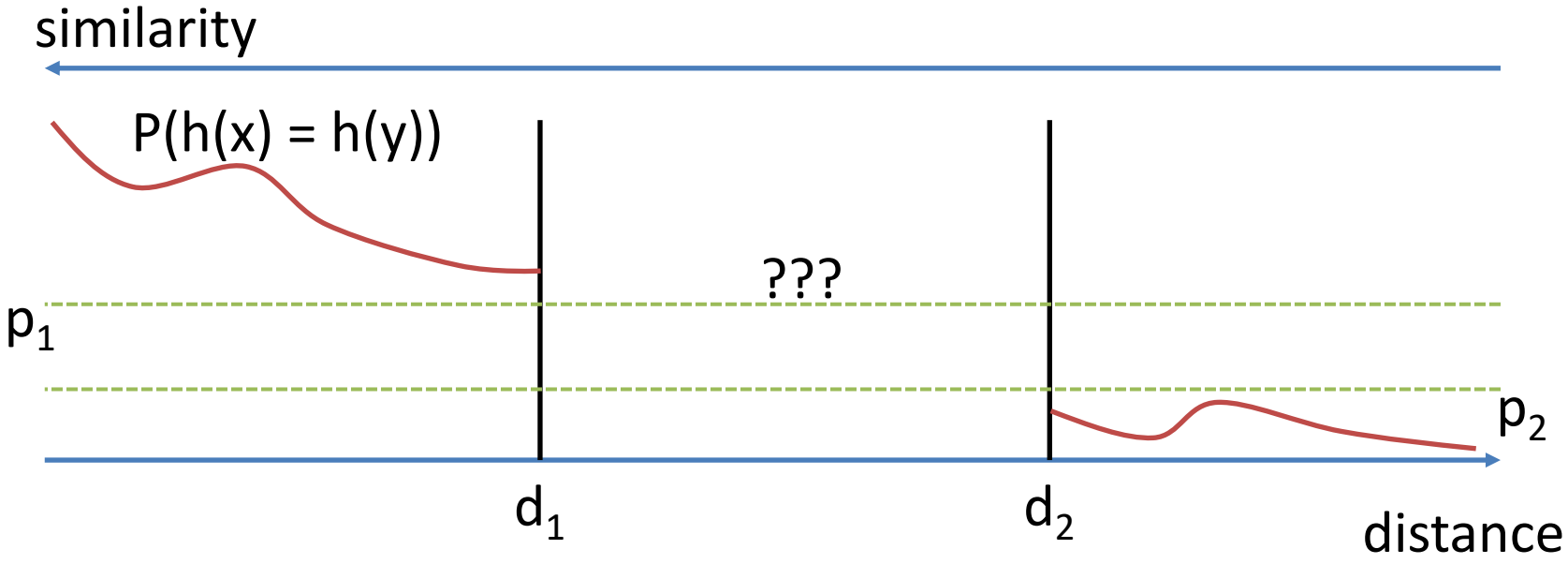
Locality-Sensitive Hash Functions

- Given a domain S and a **distance** measure d, a family H of hash functions is **(d_1, d_2, p_1, p_2)-sensitive** if for any x, y in S:
  - If $d(x, y) \leq d_1$, then $P(h(x) = h(y)) \geq p_1$, over all h in H
  - If $d(x, y) \geq d_2$, then $P(h(x) = h(y)) \leq p_2$, over all h in H



Jaccard Distance and LS Hash Functions

- For d = Jaccard distance = 1–Jaccard Similarity,
- (a, b, 1 - a, 1 - b)-sensitive, for any a < b
  - If d(x, y) ≤ a (J(x, y) ≥ 1 - a), then P(h(x) = h(y)) ≥ 1 - a
  - If d(x, y) ≥ b (J(x, y) ≤ 1 - b), then P(h(x) = h(y)) ≤ 1 - b



AND Construction

- Assume H is (d_1, d_2, p_1, p_2)-sensitive
- Construct H' with $h' = [h_1, h_2, \ldots, h_r]$, h_i in H
  - $h'(x) = h'(y)$ iff for ***all*** selected h_i, $h_i(x) = h_i(y)$
  - Requires r independent hash functions
- H' is $(d_1, d_2, (p_1)^r, (p_2)^r)$-sensitive
- Intuition & take aways:
  - Make all probabilities shrink
  - Use to reduce false positives
  - But also increases false negatives, tune r carefully

OR Construction

- Assume H is (d_1, d_2, p_1, p_2)-sensitive
- Construct H' with $h' = [h_1, h_2, \ldots, h_b]$, h_i in H
  - $h'(x) = h'(y)$ iff for ***any*** selected h_i, $h_i(x) = h_i(y)$
  - Requires b independent hash functions
- H' is $(d_1, d_2, 1 - (1 - p_1)^b, 1 - (1 - p_2)^b)$-sensitive
- Intuition:
  - Make all probabilities grow
  - Use to reduce false negatives
  - But also increases false positives, tune b carefully

Composing Constructions

- Assume H is (d_1, d_2, p_1, p_2)-sensitive
- AND-OR:

$(d_1, d_2, 1 - (1 - (p_1)^r)^b, 1 - (1 - (p_2)^r)^b)$-sensitive

  - OR-constructed family requires *b* AND-constructed functions
  - AND-constructed family requires *r* OR-constructed functions
  - Total: requires r * b independent hash functions

- OR-AND:

$(d_1, d_2, (1 - (1 - p_1)^b)^r, (1 - (1 - p_2)^b)^r)$-sensitive

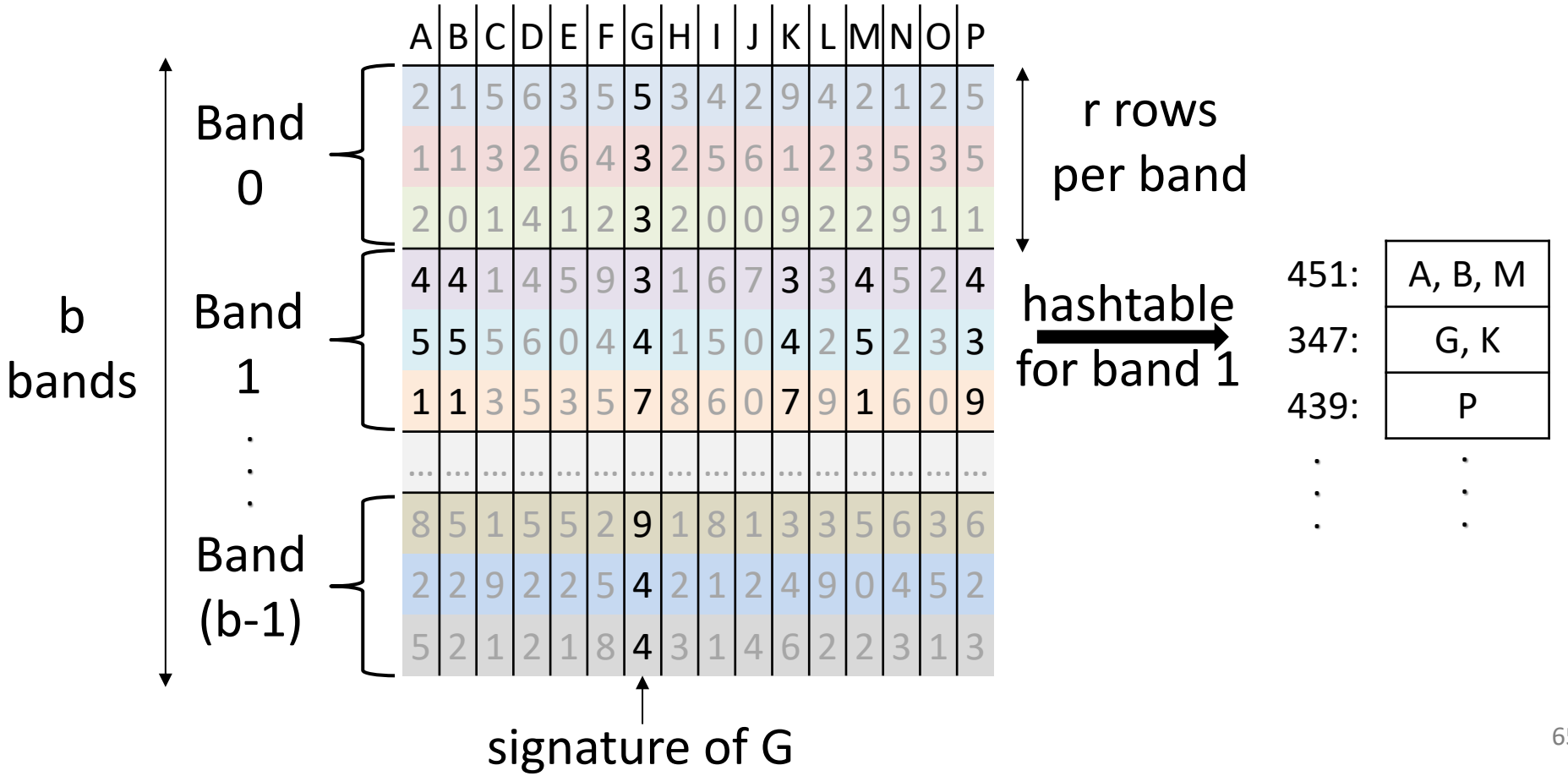
  - Similarly: requires r * b independent hash functions

- … Any sequence of AND/OR constructions

Select sequence and parameters very carefully!

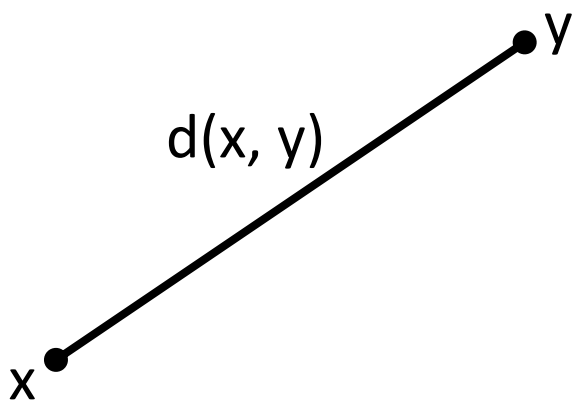
Locality-sensitive Hashing

- AND: Bucketizing
- OR: Signature partitioning



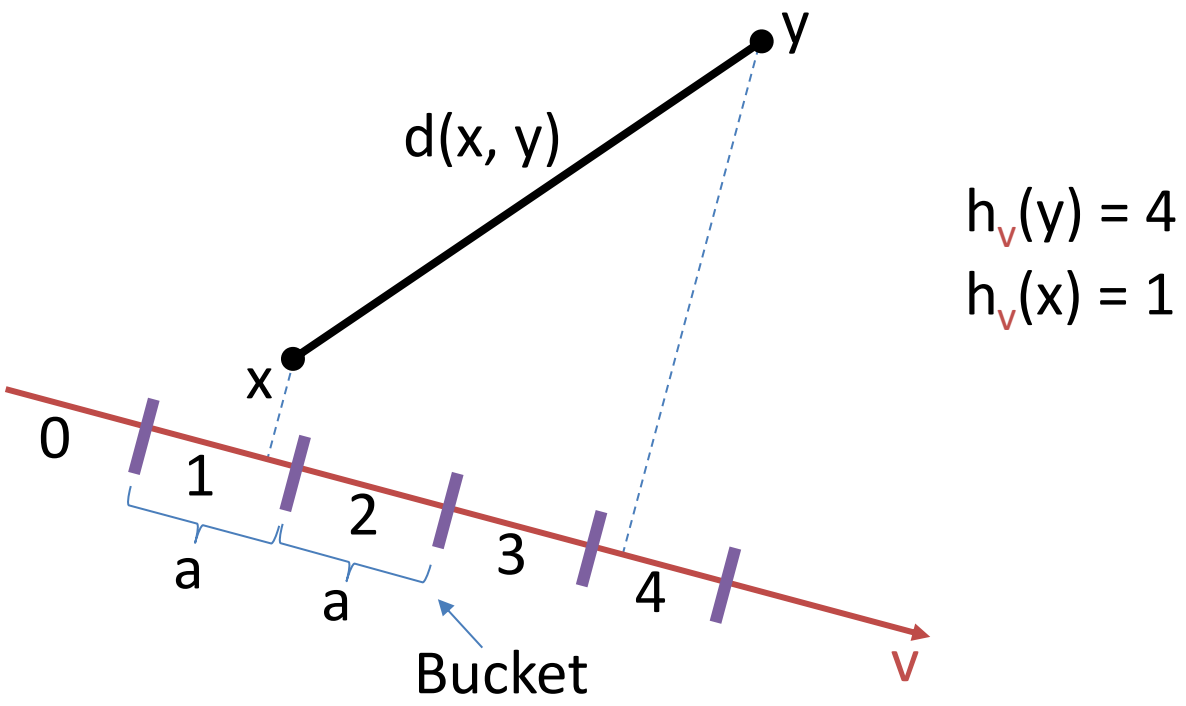
Euclidian Distance

- Definition: $d(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$
- Usage: “Find places that are close together (<5km)”



Approximate Euclidian Distance

- Project to random lines v
- Partition lines in buckets of size a
- $h_v(x)$ = bucket containing the projection of x
- (a/2, 2a, 1/2, 1/3)-sensitive

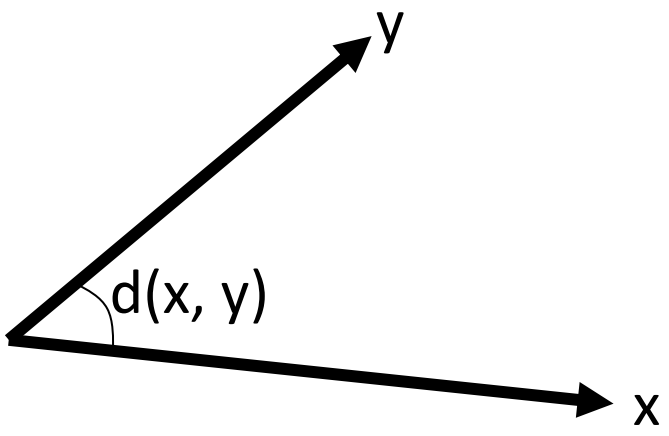


Cosine Distance

- Definition: Angle between two vectors

$$d(x,y) = \cos^{-1}\left(\frac{x^T y}{||x||\,||y||}\right) = \cos^{-1}\left(\frac{\sum_i x_i y_i}{\sqrt{\sum_i {x_i}^2}\sqrt{\sum_i {y_i}^2}}\right)$$

- Application: Word frequency



Approximate Cosine Distance

- Random hyperplanes
- Project on random vectors v with ±1 components
- $h_v(x) = 1$ if $\sum_i v_i x_i > 0$, else -1
- $P(h_v(x) = h_v(y)) = 1 - (\text{angle between x, y})/180$
- $(d_1, d_2, 1-{}^{d_1}/_{180}, 1-{}^{d_2}/_{180})$-sensitive for any d_1, d_2