Chapter 4
Secure Data Management Technology

Tomoaki Mimoto, Shinsaku Kiyomoto, and Atsuko Miyaji

Abstract In this chapter, we introduce data anonymization techniques for several types of datasets. Data anonymity of anonymized datasets is an index for estimating the (maximum) reidentification risk from anonymized datasets and is generally defined as a quantitative index based on adversary models. The adversary models are implicitly defined according to the attributes in the datasets, use cases, and anonymization techniques. We first review existing anonymization techniques and the adversary models behind the data anonymity definitions for anonymization techniques; then, we propose a common anonymity definition and its adversary model, which is applicable to several types of anonymization techniques. Furthermore, some extensions of the definition, which is optimized for specific types of datasets, are presented in the chapter.

4.1 Introduction

Secure data management is a key issue in personal data distribution and analysis. Anonymization techniques have been used to harmonize the utility of data and their privacy risks. These techniques transform personal data into anonymized data to reduce the success probability of reidentification of data principals from the data. If the data are well anonymized, they cannot be connected to a person; thus, the privacy of the person is protected by anonymization techniques.

Secure computation is sometimes a realistic solution for commercial services due to its cost for data of very large size. Some anonymization techniques work

T. Mimoto (✉) · S. Kiyomoto
KDDI Research, Inc., 2-1-15 Ohara, 356-8502 Fujimino-shi, Saitama, Japan
e-mail: to-mimoto@kddi-research.jp

S. Kiyomoto
e-mail: kiyomoto@kddi-research.jp

A. Miyaji
Osaka University, Suita, Japan
e-mail: miyaji@comm.eng.osaka-u.ac.jp

A. Miyaji and T. Mimoto (eds.), *Security Infrastructure Technology for Integrated Utilization of Big Data*,
https://doi.org/10.1007/978-981-15-3654-0_4

on commercial services as a "practical" solution, even though the size of the data is very large. Thus, anonymization techniques have been applied for personal data distribution and data analysis. For example, k-anonymization was first proposed as a practical solution to reduce the reidentification risks of public data; since then, it has been considered to be able to be used for the secure management of personal data.

Quantitative measures for anonymity are required for estimating privacy risks and assessing the feasibility of privacy requirements. In several studies on anonymization, privacy notions providing quantitative measures for anonymity have been defined for each anonymization technique; however, no common notion for all anonymization techniques has been presented to date, which means that each privacy notion is not universal but is localized, and heuristic approaches are still used to harmonize the usability of data and privacy risks through whole processes or services. A common notion is required for consistent secure data management for the whole process.

In this chapter,[1] we discuss a new common privacy notion based on an adversary model, which is applicable to several anonymization techniques, and introduce a novel anonymization technique and implementation of the technique. In Sect. 4.2, we revisit adversary models on several anonymization techniques and review anonymization techniques. We propose a common adversary model and quantitative measures using the adversary model are presented in Sect. 4.3. An extension is discussed in Sect. 4.4. Our implementation of an anonymization tool is introduced in Sect. 4.5. We conclude this chapter in Sect. 4.6.

4.2 Anonymization Techniques and Adversary Models, Revisited

The related work presented below is grouped under k-anonymization and noise addition as anonymization methods.

4.2.1 k-Anonymization

k-anonymity [4–6] is a well-known privacy model. The property of k-anonymity is that each published record is such that every combination of values of quasi-identifiers can be matched to at least k respondents.

[1] This chapter is reprinted from [1–3].

4.2.1.1 Adversary Model

k-anonymized datasets are assumed to be in public domains. An adversary can obtain all the attribute values in a dataset and execute arbitrary operations on the attribute values.

There are few formal definitions or models for the adversary that aim to identify the attributes of a certain individual in a k-anonymized dataset. Kiyomoto and Martin modeled an adversary [7] for k-anonymized datasets based on two query functions as follows:

Let d be an index of the dth record, q_x be a set of m attribute values in T^{q*}, and s be a value for the sensitive attribute. The two query functions are defined as:

- **read.** For the input of an index value d, the function outputs the dth record. That is, $f(T^*, query = \{\textbf{read}, d\}) \rightarrow \{d, q_x^d, s^d\}$, where q_x^d and s^d are values of the quasi-identifier and the sensitive attribute in the dth record, respectively. If the dth record does not exist, then the function outputs $failed$.
- **search.** For input q_x and/or s, the function outputs the number u of records and index values that have a quasi-identifier q_x and/or sensitive attribute s. That is, $f(T^*, query = \{\textbf{search}, q_x, s\}) \rightarrow u, D$, where u and D are the number of records and a sequence of index values that have the same quasi-identifier and/or sensitive attribute, respectively. If s or q_x do not exist, then the function outputs $failed$.

4.2.1.2 *k*-Anonymization Algorithm

This idea is easy to understand, and many types of k-anonymization algorithms have been proposed. The Incognito algorithm [8] generalizes the attributes using taxonomy trees, and the Mondrian algorithm [9] averages or replaces the original data with representative values and achieves k-anonymization. In this paper, we use a k-anonymization algorithm based on clustering and denote $A_k(D)$ as k-anonymization for dataset D. The algorithm finds close records and creates clusters such that each partition contains at least k records. For details of the algorithm, see [10].

4.2.2 Noise Addition

Noise addition works by adding or multiplying stochastic or randomized numbers to confidential data [11]. The idea is simple and is also well known to be an anonymization technique.

4.2.2.1 Adversary Model

One objective of an adversary against noise-added datasets is to remove the noise or estimate the original values from the noise-added attribute values. One potential scenario is a probabilistic approach in which an adversary estimates the distribution of noise and chooses an attribute value with high probability. There is no formal adversary model on static noise-added datasets, but *Differential Privacy* settings assume data include dynamically added noise, and their adversary simulations are defined as query-based.

4.2.2.2 Anonymization Algorithm by Noise Addition

The first work on noise addition was proposed by Kim [12], and the idea was to add noise ϵ with a distribution $\epsilon \sim N(0, \sigma^2)$ to the original data. Additive noise is uncorrelated noise and preserves the mean and covariance of the original data, but the correlation coefficients and variance are not retained. Another variation of additive noise is correlated additive noise, which keeps the mean and allows the correlation coefficients in the original data to be retained [13]. Differential privacy is a state-of-the-art privacy model that is based on the statistical distance between two database tables differing by at most one record. The basic idea is that, regardless of background knowledge, an adversary with access to the dataset draws the same conclusions, irrespective of whether a person's data are included in the dataset. Differential privacy is mainly studied in relation to perturbation methods in an interactive setting, although it is applicable to certain generalization methods.

In this paper, we use Laplace noise as a noise addition and add noise $\epsilon \sim Lap(0, 2\phi^2)$ to each attribute. We denote $A_\phi(D)$ as noise addition for dataset D.

4.2.3 K-Anonymization for Combined Datasets

We introduce an adversary model for a combined dataset from datasets produced by two service providers and anonymization methods [14].

4.2.3.1 Adversary Model

If we consider the existing adversary model and assume that the anonymization tables produced by the service providers satisfy k-anonymity, the combined table also satisfies k-anonymity. However, we have to consider another type of adversary in our new service model. In our service model, the combined table includes many sensitive attributes; thus, the adversary can distinguish a data owner using background knowledge of combinations of sensitive attribute values of the data owner. If the adversary finds a combination of known sensitive attributes on only one record, the

adversary can obtain information; the record is a data owner that the adversary knows, and the adversary also knows the remaining sensitive attributes of the data owner. We model the above type of new adversary as follows:

π-knowledge Adversary Model. An adversary knows certain π sensitive attributes $\{s^i_1, ..., s^i_j, ..., s^i_\pi\}$ of a victim i. Thus, the adversary can distinguish the victim with an anonymization table in which only one record has any combinations (maximum π-tuple) of the attributes $\{s^i_1, ..., s^i_j, ..., s^i_\pi\}$.

4.2.3.2 Modification of Quasi-identifiers

The first strategy is to modify the quasi-identifiers of the combined table. The data user generates a merged table from two anonymization tables as follows: First, the data user simply merges the records in the two tables as $|q^g_C|s^h_{AB}|s^i_A|s^j_B|$. Then, the data user modifies q^q_C to satisfy the following condition, where θ is the total number of sensitive attributes in the merged table.

4.2.3.3 Modification of Sensitive Attributes

The second approach is to modify the sensitive attributes in the combined table for the condition. If a subtable $|s^h_{AB}|s^i_A|s^j_B|$ that consists of sensitive attributes is required to satisfy k-anonymity, some sensitive attribute values are removed from the table and are changed to $*$ to satisfy k-anonymity. Note that we do not accept that all sensitive attributes are $*$ due to having no information record.

4.2.3.4 Algorithm for Modification

One algorithm that finds a k-anonymized combined dataset is executed as follows:

1. The algorithm generalizes quasi-identifiers to satisfy the condition that each group of the same quasi-identifiers has at least $\pi \times k$ records.
2. The algorithm generates all the tuples of π sensitive attributes in the table.
3. For each tuple, the algorithm finds all the records that have the same sensitive attributes as the tuple or has $*$ for sensitive attributes and makes them a group. We define the number of sensitive attributes in the group which is θ. The algorithm generates a partial table that consists of $\theta - \pi$ sensitive attributes and checks whether the partial table has at least k different combinations of sensitive attributes.
4. If the partial table does not satisfy the above condition, the algorithm chooses a record from other groups that have different tuples of π sensitive attributes and changes the π sensitive attributes to $*$. The algorithm executes this step until the partial table has up to π different combinations of sensitive attributes.

5. The algorithm executes step 3 and step 4 for all the tuples of π sensitive attributes in the table.

4.2.4 Matrix Factorization for Time-Sequence Data

Some studies have used matrices for time-sequence datasets. Zheng et al. [15, 16] proposed predicting a user's interests in an unvisited location. They assumed users' GPS trajectory as a user-location matrix where each value of the matrix indicates the number of visits of a user to a location. The matrix is very sparse because each user visits only a handful of locations, so a collaborative filtering model is applied to the prediction. Zheng et al. [17] built a location-activity matrix, M, which has missing values. M is decomposed into the two low-rank matrices U and V. The missing values can be filled by $X = UV^{\mathsf{T}} \simeq M$, and locations can be recommended when some activities are given. Chawla et al. [18] constructed a graph from the trajectories of taxis and transformed the graph into matrices. The authors of [19] proposed a method of identifying traffic flows that cause an anomaly between two regions.

4.2.5 Anonymization Techniques for User History Graphs

In this subsection, we introduce two anonymization techniques for user history graphs, which are proposed in [1].

4.2.5.1 Adversary Model

Privacy leakage from a merged history graph is the disclosure of the actions of a particular person from the graph. Attacks against user history graphs are intended to obtain the private information of a particular user from the graph. We assume that the merging process is executed on a trusted domain and that only the merged history graph is published; thus, the adversary can only obtain the merged graph. Furthermore, we assume that the adversary has the following knowledge about the user: The history of the user is included in the merged graph and the user performs an action t. The adversary tries to discover other actions of the user to be able to guess which edges connecting to node t can be assigned to the user.

We summarize the adversary model as follows:

Adversary against a Merged History Graph. It is assumed that an adversary knows that a victim A executed an action t. The objective of the adversary is to obtain the actions that A executed before or after the action t. Thus, the adversary searches the merged history graph, which includes actions of other people and finds the actions of A using the knowledge that action t was executed.

We define privacy notions to use with the above adversary model in a later subsection.

4.2.5.2 Notions for the Untraceability of a Graph

We consider two levels of privacy notions: partial k-untraceability and complete k-untraceability. Partial k-untraceability accepts the leakage of some partial actions of a user but prevents all the actions of the user from being revealed. The definition of complete k-untraceability involves meeting the requirement that no action of the user is leaked. The symbol $Act^A_{N_{x\to y}}$ for user A denotes the sequence of all the actions of user A from action x to action y. For example, the sequence of actions from the first action to action x and the sequence of actions from action x to the final action are denoted as $Act^A_{\mathcal{N}_{start\to x}}$ and $Act^A_{\mathcal{N}_{x\to end}}$, respectively.

Definition 4.1 (*Partial k-untraceability*) We assume that an adversary knows an action t of a user A, and we consider all the possible adversaries defined for any action t of the user in the merged graph. If at least k sequences of actions are potentially associated with user A and $k-1$, other users exist as candidates for all actions $Act^A_{\mathcal{N}_{start\to t}}$ and $Act^A_{\mathcal{N}_{t\to end}}$, the digraph satisfies k-untraceability for A. If the digraph satisfies the above condition for all users, then the digraph is said to satisfy partial k-untraceability.

Definition 4.2 (*Complete k-untraceability*) We assume that an adversary knows an action t of a user A and we consider all the possible adversaries defined for any action t of the user in the merged graph. If at least k actions are potentially associated with user A and $k-1$ other users exist as candidates for each action in $Act^A_{\mathcal{N}_{start\to t}}$ and $Act^A_{\mathcal{N}_{t\to end}}$, the digraph satisfies k-untraceability for A. If the digraph satisfies the above condition for all users, the digraph satisfies complete k-untraceability.

Generally, many trivial actions are performed by many users. It is not important for privacy purposes where we keep the information about such actions. Thus, we relax the above definitions to produce an anonymized graph that includes much of the information needed to analyze a user's history. Let v be the threshold value for the number of performing users that establishes that an action is trivial; that is, we judge the actions $x \to y$ to be trivial if the label $L(x \to y) \geq v$. Both definitions are modified as follows:

Definition 4.3 (*Partial (k, v)-untraceability*) We assume that an adversary knows an action t of a user A, and we consider all the possible adversaries defined for any t in the merged graph. If at least k sequences of actions are potentially associated with user A and $k-1$ other users exist as candidates for all actions $Act^A_{\mathcal{N}_{start\to t}}$ and $Act^A_{\mathcal{N}_{t\to end}}$ except trivial actions $x \to y$ that have a label $L(x \to y) \geq v$, then the digraph satisfies partial (k, v)-untraceability for A. If the digraph satisfies the above condition for all users, then the digraph satisfies partial (k, v)-untraceability.

Definition 4.4 (*Complete (k, v)-untraceability*) We assume that an adversary knows an action t of a user A, and we consider all the possible adversaries defined for any t in the merged graph. If at least k actions are potentially associated with user A and $k-1$ other users exist as candidates for each action in $Act^{A}_{\mathcal{N}_{start \to t}}$ and $Act^{A}_{\mathcal{N}_{t \to end}}$ except trivial actions $x \to y$ that have a label $L(x \to y) \geq v$, then the digraph satisfies complete (k, v)-untraceability for A. If the digraph satisfies the above condition for all users, then the digraph satisfies complete (k, v)-untraceability.

In a complete (k, v)-untraceable graph, each action t except trivial actions has k outgoing edges and incoming edges; thus, an action of user A that connects to action t cannot be identified from k candidates. Thus, the graph satisfies untraceability for an adversary who knows action t of the user. It is trivial that a complete (k, v)-untraceable graph satisfies partial (k, v)-untraceability; all actions except trivial actions are connected to k potential actions in a complete (k, v)-untraceable graph. A graph that satisfies partial (k, v)-untraceability generally produces much more information than a complete (k, v)-untraceable graph, where the partial (k, v)-untraceable graph and the complete (k, v)-untraceable graph are generated from a user history graph. However, the (k, v)-untraceable graph may reveal partial actions of users due to the relaxed definition of the privacy notion; an attack is successful when an adversary obtains all the actions of a user. To trace all the actions of the user, the adversary has to select a sequence of actions from k sequences of actions; thus, all the actions of the user are untraceable, even though some actions are traceable by the adversary. The parameter k means that an action (or a sequence of actions) is potentially associated with a user and $k-1$ other users in the untraceable graph, and the parameter v means that v users perform the same action in the graph. Generally, we should select the parameter $v = k$ with regard to the privacy requirement for a merged graph. The actions of a user are hidden in the actions of a group that consists of k members including the user. A privacy notion for the graph should be selected from the above two notions according to a use case of the graph and its privacy requirements.

4.2.5.3 Algorithm Generating a Partial (k, V)-Untraceable History Graph

The details of the algorithm are denoted as **Algorithm 4.1**, where oe_t and ie_t are defined as the number of outgoing edges and incoming edges of a node t, respectively. The algorithm for generating a partial (k, v)-untraceable history graph is as follows:

1. This step consists of a part of the detailed algorithm, from line 1 to line 3. For the input of a user history graph **G**, the algorithm adds a virtual incoming edge $(s_r \to r)$ to each node $r \in start$ until the number of incoming edges is the same as the number of outgoing edges. Then, the algorithm adds a virtual outgoing edge $(q \to u_q)$ to each node $q \in end$ until the number of outgoing edges is the same as the number of incoming edges. A label of a virtual incoming edge $L(s_x \to x)$ denotes the number of users who first perform the action, and a label of a virtual

outgoing edge $L(y \rightarrow u_y)$ denotes the number of users who perform the action at the end.

2. This step consists of a part of the detailed algorithm, from line 4 to line 12. The algorithm searches for a node t that has fewer outgoing edges than k and for which all its lower nodes $\mathcal{N}_{t \rightarrow end \backslash t}$ have fewer outgoing edges than k. Then, the algorithm removes all the outgoing edges $(t \rightarrow *)$ that satisfy $L(t \rightarrow *) < v$. Next, the algorithm searches for a node t' that receives incoming edges numbering less than k and all upper nodes $\mathcal{N}_{start \rightarrow t' \backslash t'}$ that receive fewer incoming edges than k. Then, the algorithm removes all the incoming edges $(* \rightarrow t')$ that satisfy $L(* \rightarrow t') < v$. The algorithm repeats this step until no node that meets the conditions is found.
3. This step is the same as line 13, line 14 and line 15 in the detailed algorithm. The algorithm removes virtual incoming and outgoing edges, removes nodes that have no edges, and outputs the modified graph.

Algorithm 4.1 Generation of a Partial (k, v)-Untraceable History Graph

Input: User History Graph G, parameters k and v
Output: Anonymized Graph $G^{\alpha}(G, k, v)$
1: $G^{\alpha}(G, k, v) \leftarrow G$
2: Add virtual incoming edges to *start* nodes
3: Add virtual outgoing edges to *end* nodes.
4: $T \leftarrow$ all nodes t, where $oe_{\mathcal{N}_{t \rightarrow end}} < k$ and all of its edges do not have $L(t_i \rightarrow *) \geq v$
5: $T' \leftarrow$ all nodes t', where $ie_{\mathcal{N}_{start \rightarrow t'}} < k$ and all of its edges do not have $L(* \rightarrow t'_j) \geq v$
6: **while** $T \neq \emptyset$ or $T' \neq \emptyset$ **do**
7: Choose t_i from T
8: Remove all outgoing edges of t_i where $L(t_i \rightarrow *) < v$ from $G^{\alpha}(G, k, v)$
9: Choose t'_j from T'
10: Remove all incoming edges of t'_j where $L(* \rightarrow t'_j) < v$ from $G^{\alpha}(G, k, v)$
11: Update T and T'
12: **end while**
13: Remove virtual edges
14: Remove all nodes t'' where $oe_{t''} = 0$ and $ie_{t''} = 0$ from $G^{\alpha}(G, k, v)$
15: **return** $G^{\alpha}(G, k, v)$

4.2.5.4 Algorithm Generating a Complete (k, V)-Untraceable History Graph

The details of the algorithm are denoted as **Algorithm 4.2**. The algorithm for generating a complete (k, v)-untraceable history graph is as follows:

1. The algorithm first executes **Algorithm 4.1** except line 13 and line 15.
2. This step consists of a part of the detailed algorithm, from line 3 to line 11. The algorithm searches for a node t that has fewer outgoing edges than k and removes

all the outgoing edges $(t \rightarrow *)$ that satisfy $L(t \rightarrow *) < v$, until no node is found. Then, the algorithm searches for a node t' that receives fewer incoming edges than k and removes all the edges $(* \rightarrow t')$ that satisfy $L(* \rightarrow t') < v$. The algorithm repeats this step until no node that meets the conditions is found.

3. This step consists of line 12, line 13, and line 14 in the detailed program. The algorithm removes virtual edges, removes nodes to which no edge is connected, and outputs the modified graph.

4.2.6 Other Notions

Differential Privacy [20, 21] is a notion of privacy for perturbative methods based on the statistical distance between two database tables differing by, at most, one element. The basic idea is that, regardless of background knowledge, an adversary with access to the dataset draws the same conclusions whether a person's data are included in the dataset. That is, a person's data have an insignificant effect on the processing of a query. Differential privacy is mainly studied in relation to perturbation methods [22–24] in an interactive setting. Attempts to apply differential privacy to search queries have been discussed in [25]. Li et al. proposed a matrix mechanism [26] applicable to predicate counting queries under a differential privacy setting. Computational relaxations of differential privacy were discussed in [27–29]. Another approach for quantifying privacy leakage is an information-theoretic definition proposed by Clarkson and Schneider [30]. They modeled an anonymizer as a program that receives two inputs: a user's query and a database response to the query. The program acted as a noisy communication channel and produced an anonymized response as the output. Hsu et al. provides a generalized notion [31] in decision theory for making a model of the value of personal information. An alternative model for the quantification of personal information is proposed in [32]. In the model, the value of personal information is estimated by the expected cost that the user has to pay for obtaining perfect knowledge from given privacy information. Furthermore, the sensitivity of different attribute values is taken into account in the average benefit and cost models proposed by Chiang et al. [33]. Krause and Horvitz presented utility-privacy tradeoffs in online services [34, 35].

4.2.7 Combination of Anonymization Techniques

A combination of anonymization methods leads to the construction of datasets that are useful and that preserve privacy. Some countries publish census data, and they combine several anonymization methods, such as generalization, noise addition, and sampling [36, 37]. However, some problems remain. One problem is that it is difficult to evaluate the privacy risks of anonymized datasets when anonymization methods are combined. Some research is available about the relationships among anonymization

methods. Chaudhuri et al. proposed (c, ϵ, δ)-privacy [38] and studied the relationship among sampling and differential privacy [39]. Li et al. proposed $(\beta, \epsilon, \delta)$-differential privacy and studied the relationship among sampling, differential privacy, and k-anonymity. Soria-Comas et al. proposed a k-anonymized algorithm for differential privacy using an insensitive algorithm [40].

4.3 (p, N)-Identifiability

4.3.1 Common Adversary Model

Existing privacy measures are supposed to protect against idealized attackers, and it is difficult to maintain their utility and assess their reidentification risk. We designed adversary models to describe more realistic attackers by structuring a real setting for the attackers. In the case of exchanging anonymized datasets between companies, for instance, a data-providing company first anonymizes and encrypts datasets for transmission to a receiver company via a secure channel. The receiver company locates the dataset in a secure room and allows only authorized employees to access the anonymized dataset. This process can reduce the reidentification risk in the anonymized dataset, and it specifies the attacker and limits the ability to access datasets so that the attacker must know the quasi-identifiers of the neighbors or acquaintances. For example, it seems to be quite rare for an attacker to know all the quasi-identifiers of a target because the target is a neighbor of the attacker. Thus, a more stringent analysis of the reidentification risk can be achieved when we assume a more realistic situation, such as that the attacker has only limited knowledge of the victim.

Access rights to an anonymized dataset may be given to attackers, and attackers may acquire some information about the original dataset or obtain the anonymization algorithm used to generate the anonymized dataset. Information about the original dataset is categorized into three parts as follows: information on a specified record such as a neighbor; the original dataset; and any other information except the target information that the attacker is seeking. The case of William Weld, who was governor of Massachusetts [41], is a typical example of reidentification, and an attack on the Netflix Prize dataset was carried out by a strong attacker who gained access to the Internet Movie Database [42].

We can consider the abilities of an attacker in two areas: knowledge about the dataset and the ability to simulate anonymization algorithms. Many previous studies such as [43, 44] assumed that an attacker has all the information required except knowledge of the target of the attack. In this paper, we consider an attacker who has knowledge of only the target record and can simulate anonymization algorithms to obtain anonymized records that may correspond to the target record.

4.3.1.1 Definitions of Actual Attackers

Generally, when an anonymized dataset is published on the Web, anyone who can access the dataset is a potential attacker; thus, the adversary model should be ideal because we cannot assume there is only a limited-knowledge adversary, and we have to assume all possible adversaries are present. On the other hand, when the dataset is managed under strict controls, the model adversary is not considered to be an unlimited-knowledge adversary. We design two realistic adversary models under the assumption that the dataset is managed in a restricted area (not public) and only a limited set of attackers can access the dataset; and then, we propose a privacy metric for privacy risk analysis.

Definition 4.5 (*Anonymization Simulator* f_{sim}) Let D_0 with n_0 records, D_1 with n_1 records, $r_i^x[QI]$, and $r_i^x[SI]$ be an original dataset, an anonymized dataset generated from the original dataset, the quasi-identifiers of a record $r_i^x \in D_x$, and sensitive information from the record $r_i^x \in D_x$, respectively. An anonymization simulator f_{sim} simulates an anonymization algorithm used to generate an anonymized dataset as an oracle and outputs $r_i^1[QI] \in D_1$ for the input $r_i^0[QI] \in D_0$. That is, $f_{sim} : r_j^0[QI] \rightarrow \left\{\mathbf{r}^1[QI], \perp\right\}$, where $\mathbf{r}^1[QI]$ is a set of $r_i^1[QI]$ and no output is produced in the case of $\perp$.

The simulator is a deterministic process for deterministic anonymization, such as top-coding and bottom-coding, and a probabilistic process for probabilistic anonymization, such as random sampling. The simulator can provide access to D_0 to simulate the anonymization algorithm, even though no adversary can access D_0. Next, we define two adversary models.

Definition 4.6 (*Deanonymizer for Anonymized Datasets,* $\mathcal{DA}$) When $\exists_1 r_j^0[QI] \in D_0$, $\forall r_i^1[QI||SI] \in D_1$ and f_{sim} are given, a deanonymizer $\mathcal{DA}$ lines up potential candidates r_i^1 corresponding to r_j^0 by executing the simulator f_{sim}; then, the deanonymizer $\mathcal{DA}$ outputs a list of candidates $r_i^1[QI||SI]$ for r_j^0, where the number of records in the list is n_q, the number of sensitive information items in the list is n_s and $0 \leq n_s \leq n_q \leq n_0$.

If an attacker knows the actual anonymization function f, the attacker can use f as f_{sim}, and the evaluation result should be more credible.

Definition 4.7 (*Reidentifying Adversary versus Anonymized Datasets*) When $\exists_1 r_j^0[QI] \in D_0$, $\forall r_i^1[QI||SI] \in D_1$ and f_{sim} are given, a reidentifying adversary executes the deanonymizer $\mathcal{DA}$ and can identify r_i^1, which is a record of the same person in the record r_j^0, from the records in a dataset D_0, where $r_j^0 \in D_0$ is given. The success probability of the attack is calculated as $1/n_q$ when r_j^1 is included in the output by $\mathcal{DA}$; otherwise, it is 0.

Assuming an attacker who has $\exists_1 r_j^0[QI] \in D_0$ is the same as assuming $|D_0|$ attackers who have $r_j^0(j = 1, ..., |D_0|) \in D_0$.

Definition 4.8 (*Revealing Adversary versus Anonymized Datasets*) When $\exists_1 r_j^0[QI] \in D_0$, $\forall r_i^1[QI||SI] \in D_1$ and f_{sim} are given, a revealing adversary executes the deanonymizer $\mathcal{DA}$ and finds a $r_j^0[SI]$ from $r_i^1[SI]$ such that r_i^1 is a record of the same person as the record r_j^0. The success probability of the attack is calculated as $1/n_s$ when r_j^1 is included in the output of $\mathcal{DA}$; otherwise, it is zero.

A revealing adversary does not try to identify the record but tries to access sensitive information. In other words, the attacker seeks only to obtain sensitive information from the record in question. More precisely, the success probability of the *revealing adversary* can be calculated as $[n_s]/n_q$, where the correct number of sensitive items in the list is $[n_s]$, but the probability itself may be uncertain. Assume that when the probability is 0.99, some attackers are convinced that the target should be the majority. Furthermore, in the case that the deanonymizer $\mathcal{DA}$ is leaked and the f_{sim} used in the deanonymizer is a deterministic process, an attacker can infer the sensitive information of r_j^0. On the other hand, when the f_{sim} used in the deanonymizer is a probabilistic process, even if $\mathcal{DA}$ is leaked, outputting the result should not involve uncertainty.

4.3.1.2 (p, N)-Identifiability

Here, we assume that anonymized datasets are strictly controlled and that the attacker has knowledge of a specific record and the anonymization algorithms. We assume that the attacker is the strongest type of attacker and has knowledge of the most characteristic record. Nevertheless, it is difficult to quantify this characteristic, so we assume that each attacker has an original record. In other words, we assume there are as many attackers as there are original records.

Definition 4.9 ((p, N)-*identifiability*) Let p be the success probability for an adversary who has $\exists_1 r^0[QI] \in D_0$, $\forall r_i^1[QI||SI] \in D_1$ and f_{sim}, and N be the number of adversaries whose attack success probability is p.

The probability p is the conditional probability that the adversary can select the correct record from the list produced by the deanonymizer $\mathcal{DA}$ when the collected record is included in the list. The probability that the deanonymizer successfully produces the list, including the correct record, depends on the anonymization algorithms.

Our model can extend to an adversary who has knowledge of two or more records. For simplicity, we use an adversary model that knows a single record and consider N single knowledge adversaries in our risk analysis. The idea of (p, N)-identifiability is studied in [2].

4.3.2 Success Probability Analysis Based on the Common Adversary Model

In this section, we assume the attackers described in the previous section and explain the calculation to obtain the success probability of attacks on representative anonymization methods: generalization, noise addition, and sampling. We consider that f_{sim} is constructed as a typical combined algorithm selected from three anonymization algorithms, $f_{generalization}$, $f_{sampling}$ and f_{noise}. We explain the above three anonymization algorithms and show combined anonymization using an example dataset.

4.3.2.1 Generalization

We include deletion of records or cells and top- or bottom-coding as steps in generalization. One step of $f_{generalization}$ is similar to k-anonymity in checking the number of identical combinations of quasi-identifiers. When an anonymized dataset has k-anonymity, p equals $1/k$. k-anonymity is an intuitive privacy metric, but the greater the number of attributes, the more difficult it is for the datasets to achieve k-anonymity. If an attacker has generalization trees for each attribute, the attacker adds records which satisfy the requirements of the trees of the list of candidates. When there is a record whose address attribute is Tokyo, for instance, an attacker who has the generalization tree adds records whose addresses are in the Kanto region as well as records whose addresses are in Eastern Japan to the list of candidates. It is appropriate that an attacker can infer the generalization tree and in our experiment, f_{sim} can be considered capable of accessing the generalization trees of each attribute.

4.3.2.2 Random Sampling

When an attacker who has one original record is assumed, the privacy risk differs greatly among the original datasets. Consider an original dataset with many unique records, and assume that random sampling is implemented. Let M be the number of unique records and α be the sampling rate. The probability that unique records will not appear is $(1-\alpha)^M$. Even when $\alpha = 0.1$ and $M = 44$, the probability is less than 0.1%. When a large dataset is anonymized, it is possible that there will be more than 44 unique records, which shows that if sampling is implemented, a characteristic record may be identified or suspected.

We evaluate sampling as follows: For simplicity, we consider the case where the anonymization method is only random sampling. When a unique record is sampled, an attacker who knows the person is certain that the record is for that person. Thus, the probability p does not change. On the other hand, sampling reduces the number of unique records, and N decreases accordingly. When unique records are very few

and do not appear in an anonymized dataset, p decreases. We apply this approach to the case of combining different anonymization methods.

The approaches to sampling vary, and we can also consider $f_{sampling}$ in various ways. For instance, the probability of disclosing the identity of any individual is evaluated by using the posterior probability of population uniqueness [45].

4.3.2.3 Noise Addition

There are two cases of noise addition: One is adding noise to the numerical data itself, and the other is adding noise to its quantity. In the former case, the data consist of original numerical data or data anonymized by a process, such as microaggregation, and in the latter case, the data are original quantity data or anonymized data, such as 11–20 in the age attribute.

In the former case, we can consider f_{noise} as follows. Noise is added based on a probability distribution, such as normal, Laplace, and exponential distributions. In particular, it has been mathematically proven that adding Laplace noise to the output of some queries achieves differential privacy [39], so this type of noise is widely used. Therefore, when an anonymized record is included in the 90 or 95% confidence interval, the record is added to the list of candidates. More simply, when original data and anonymized data have small differences such as 10 or 20% for each attribute, the attacker may consider the possibility that they are the same.

In the latter case, we cannot use the same method. When a record has 72 and is anonymized to 95, for instance, the attacker whose target is a specific person may not regard the target to be that person. However, the attacker can link them after the top-coding is executed and change the value to 70-. On the other hand, when a record is 19, is anonymized to 20 and is generalized to 20–29, the attacker may not link them. One of the ideas of f_{noise} is that a group with each attribute can be changed to next group and such records are output as candidates. As in the generalization step, an attacker can infer the next group for each group and f_{noise} can be thought of as defining the distance of each classification.

The description above shows that when the order of anonymization is changed, f_{sim} will also be changed.

4.3.2.4 Combination of Anonymization Methods

The principles of each anonymization can be combined by evaluating each anonymization step by step. Stated differently, an attacker has $f_{generalization}$, $f_{sampling}$, and f_{noise} as f_{sim}. We show examples of combined cases by using a sample dataset (Fig. 4.1). An attacker should change his or her approach when the order of anonymization is changed if he or she knows this fact. We assume five attacker models, A_1 to A_5, in the following example, and the candidates of each attacker model are represented as C_1 to C_5. We denote C_i of r_j in the following figures as the candidates of an attacker A_i who has r_j as a target. The adversary model for A_1 to

Fig. 4.1 Sample dataset

record	$ATTR_1$	$ATTR_2$	$ATTR_S$
r_1	28	178	Hospital
r_2	31	179	Office
r_3	38	165	Office
r_4	30	180	Shop
r_5	27	167	Hospital
r_6	29	171	Shop
r_7	33	173	Hospital

A_4 is the *reidentifying adversary* defined in Definition 4.3, and the adversary model in Fig. 4.4 is the *revealing adversary* defined in Definition 4.4.

Let the conditions of attackers be as follows: A_1 and A_3 do not consider noise-adding and generalization but simply compare $r_i^1 \in D_1$ with $r_j^0 \in D_0$. This is one approach to f_{noise} and $f_{generalization}$. On the other hand, A_2, A_4, and A_5 do consider the added noise and generalization. We define the noise addition shown in Fig. 4.2 as follows: the classifications of each attribute change to the next classification with a certain probability. We assume A_2 knows the rule of noise addition and that f_{noise} of A_2 outputs candidates that have a different classification in one attribute from an original record. On the other hand, let a small amount of noise be added in step (a) of Figs. 4.3 and 4.4. We assume the attackers A_4 and A_5 know the rule and that f_{noise} of A_4 and A_5 outputs candidates whose values of $ATTR_1$ are different but within 2 from the original record and whose values of $ATTR_2$ are different but within 4 from the original record. In the figures, the boldface sections show that the classifications are not correct but are within the permissible range for f_{noise} of A_2, A_4, and A_5: The red boldface sections show that there are substantial distances from the original values and that attackers who have the record cannot link them.

4.3.2.5 Examples of Analyses

The Case of A_1

Record	$ATTR_1$	$ATTR_2$	C_1	C_2
r_1	-30	175-	1,4	-
r_2	31-	175-	2	-
r_3	31-	-174	3,7	-
r_4	-30	175-	1,4	-
r_5	-30	-174	5,6	-
r_6	-30	-174	5,6	-
r_7	31-	-174	3,7	-

(a) Generalization

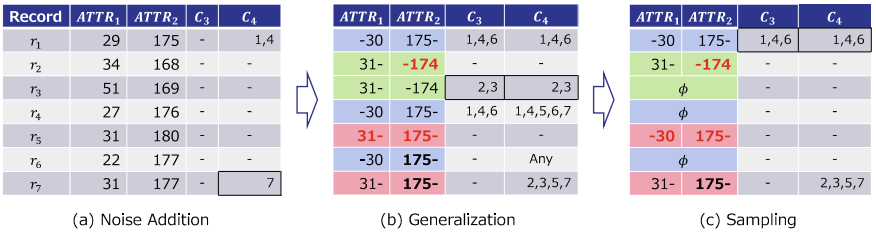
$ATTR_1$	$ATTR_2$	C_1	C_2
31-	175-	-	1,2,5,6
31-	175-	1,2	1,2,3,4,6,7
31-	-174	3,4,7	1,2,3,4,5,7
31-	**-174**	-	-
-30	-174	5	1,2,5,6
-30	**175-**	-	3,4,5,6,7
31-	-174	3,4,7	1,2,3,4,5,7

(b) Noise Addition

$ATTR_1$	$ATTR_2$	C_1	C_2
31-	175-	-	1,2,5,6
ϕ		-	-
31-	-174	3,7	1,2,3,4,5,7
ϕ		-	-
ϕ		-	-
-30	**175-**	-	3,4,5,6,7
31-	-174	3,7	1,2,3,4,5,7

(c) Sampling

Fig. 4.2 Sample anonymization and the result of simulation attack 1



Record	$ATTR_1$	$ATTR_2$	C_3	C_4
r_1	29	175	-	1,4
r_2	34	168	-	-
r_3	51	169	-	-
r_4	27	176	-	-
r_5	31	180	-	-
r_6	22	177	-	-
r_7	31	177	-	7

(a) Noise Addition

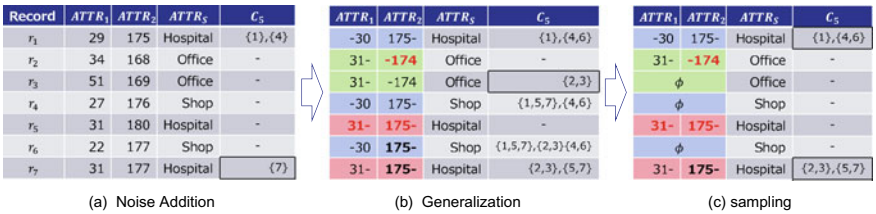
$ATTR_1$	$ATTR_2$	C_3	C_4
-30	175-	1,4,6	1,4,6
31-	-174	-	-
31-	-174	2,3	2,3
-30	175-	1,4,6	1,4,5,6,7
31-	175-	-	-
-30	175-	-	Any
31-	175-	-	2,3,5,7

(b) Generalization

$ATTR_1$	$ATTR_2$	C_3	C_4
-30	175-	1,4,6	1,4,6
31-	-174	-	-
ϕ		-	-
ϕ		-	-
-30	175-	-	-
ϕ		-	-
31-	175-	-	2,3,5,7

(c) Sampling

Fig. 4.3 Sample anonymization and the result of simulation attack 2



Record	$ATTR_1$	$ATTR_2$	$ATTR_S$	C_5
r_1	29	175	Hospital	{1},{4}
r_2	34	168	Office	-
r_3	51	169	Office	-
r_4	27	176	Shop	-
r_5	31	180	Hospital	-
r_6	22	177	Shop	-
r_7	31	177	Hospital	{7}

(a) Noise Addition

$ATTR_1$	$ATTR_2$	$ATTR_S$	C_5
-30	175-	Hospital	{1},{4,6}
31-	-174	Office	-
31-	-174	Office	{2,3}
-30	175-	Shop	{1,5,7},{4,6}
31-	175-	Hospital	-
-30	175-	Shop	{1,5,7},{2,3}{4,6}
31-	175-	Hospital	{2,3},{5,7}

(b) Generalization

$ATTR_1$	$ATTR_2$	$ATTR_S$	C_5
-30	175-	Hospital	{1},{4,6}
31-	-174	Office	-
ϕ		Office	-
ϕ		Shop	-
31-	175-	Hospital	-
ϕ		Shop	-
31-	175-	Hospital	{2,3},{5,7}

(c) sampling

Fig. 4.4 Sample anonymization and the result of simulation attack 3

Generalization, noise addition, and sampling are executed as anonymizing methods in Fig. 4.2. In the generalization step (a), all records are generalized to be divisible into equal parts. As a result, only r_2 is unique, and this dataset has (1, 1)-identifiability.

In step (b), r_1, r_4, and r_6 are changed by the addition of noise. As a result, r_1 and r_2 are indistinguishable. r_3, r_4, and r_7 are also indistinguishable, but r_5 and r_6 become unique. We define A_1 as not considering the addition of noise, so that an attacker who has r_6 cannot link the original record but an attacker who has r_5 can. Therefore, identifiability becomes (1, 1)-identifiability.

After sampling, in step (c), r_2, r_4, and r_5 do not appear. Then, r_3 and r_7 become the focus are focused and identifiability becomes $(1/2, 2)$-identifiability. This attacker simply checks how many of the same records there are in the dataset. Even if various anonymization methods are implemented, some records may not be affected. Therefore, it is important to assume such attackers. When we can say that a dataset has a certain level of privacy from such attackers, it means that an attacker cannot link the target with the original record by accident.

The Case of A_2

We omit the explanation of step (a) because noise is not added. In step (b), the attacker with r_1, for example, chooses r_1, r_2, r_5, and r_6 as candidates because one or more of their attributes match $r_1 = \{\text{-30, 175-}\}$. On the other hand, an attacker with r_4 cannot output candidates because both attributes of r_4 are changed. Hence, identifiability is $(1/4, 2)$-identifiability. In step (c), r_5 does not appear, and identifiability becomes $(1/4, 1)$-identifiability.

The Case of A_3

In Fig. 4.3, the dataset is anonymized by the addition of noise, generalization, and sampling.

In the case of A_3, the dataset with added noise is safe enough from attackers who do not consider the added noise and we omit this case; however, this does not mean that noise addition is safe, and when another attacker, such as A_4, is considered, the result should be different. In step (b), we focus on the attacker with r_3. This is the strongest attacker, and this attacker suspects that r_2 and r_3 are the candidates. More specifically, the scope is $r_3 = \{38, 165\} = \{31\text{-}, \text{-}174\}$ and r_2, r_3 meet the requirement. The attacker with r_2 seems to have the same risk but cannot identify the actual target r_2 is a possible candidate because the noise of $ATTR_2$ is great enough. Hence, the identifiability becomes $(1/2, 1)$-identifiability. In step (c), r_3 does not appear, and the privacy risk is $(1/3, 1)$-identifiability.

The Case of A_4

Next, we show the case of A_4. In step (a), every record but r_1 and r_7 has enough added noise, and attackers cannot infer which is the correct record. The attacker with r_7 regards the records within $\{33 \pm 2, 173 \pm 4\}$ as candidates. Only r_7 satisfies the condition, and the privacy risk is $(1, 1)$-identifiability. In step (b), the effect of noise addition becomes weak, and the number of attackers who should be considered increases. The attacker with r_6, for instance, regards the records within $\{29 \pm 2, 171 \pm 4\} = \{(\text{-}30, 31\text{-}), (\text{-}174, 175\text{-})\}$, namely, all records, as candidates. The privacy risk becomes $(1/2, 1)$-identifiability after generalization is finished. In step (c), similar to the previous steps, the privacy risk becomes $(1/3, 1)$-identifiability.

The Case of A_5

Finally, we show an example of a *revealing adversary*.

An attacker can claim to succeed when the sensitive information $ATTR_S$ of the target can be correctly identified. Step (a) is similar to that of the case of A_4. In step (b), the attacker with r_3 suspects r_2 and r_3 are the candidates. Their $ATTR_S$ are, however, "Office" and the attacker claims to identify the person. Thus, the privacy risk is $(2/2 = 1, 1)$-identifiability, which is similar to l-diversity. In step (c), the attacker with r_1 suspects r_1, r_4 and r_6 are the candidates; the $ATTR_S$ of r_1 is "Hospital," and that of the others is "Shop." Therefore, the probability of reidentification is 1/2. More precisely, the probability is 1/3 because there are three candidates and one is correct, but the probability may be important information for the attacker with r_1. The same can be said of the attacker with r_7; therefore, the risk according to our definition is $(1/2, 2)$-identifiability.

As described above, when the adversary model is different, the result of the risk is also different. Assuming attackers who disregard noise, we consider the risk to the records whose fluctuations are due to anonymization to be small. On the other hand, assuming attackers who do consider the actual added noise, we consider the risk to the dataset as a whole. Moreover, strong attackers can be assumed to use the inverse function of the actual noise or anonymization method. In the case that noise based on a normal distribution is added, for instance, an optimal distance-based record linkage can be performed [46].

It is important to consider the various types of attackers in this way, because the most important factor of privacy is the inability to definitely link an anonymized record X' and original record X. Our metrics ensure that the attackers considered can neither identify a record nor make an identification by chance, by considering many attackers.

4.3.2.6 Implementation of the Analysis Algorithm

Processing time is a problem when our metric is applied to a large dataset. In this section, we discuss this problem.

First, we have to evaluate the risk from attackers with each record, and when sampling is implemented, the candidates in each record need to be preserved across the sampling. However, we do not need to store the candidates for every record or the records that have certain risks because the metric does not consider attackers who have knowledge of a record that does not have the highest risk. Moreover, when anonymization and evaluation are performed repeatedly, it takes a long time to evaluate the risk because the same number of attackers as the number of records are assumed. Thus, a threshold risk can be introduced to resolve the problem. When the risk of an attack does not exceed the threshold, attackers do not need to be evaluated. It is possible, however, that the risk may increase depending on the situation (see r_5, r_6 in Fig. 4.2). Therefore, when a threshold is introduced, the accuracy of the privacy risk may worsen. We describe the pseudocode of risk analysis as follows:

Algorithm 4.5 (D_0, D_1, A, f_{sim}): Risk analysis.

Input: Original dataset D_0, Anonymized dataset D_1, Adversary model A, and attack simulator f_{sim}

1: **while** $\forall r_i^0 \in D_0$ **do**
2: $p_i \leftarrow$ simulation attack(r_i^0, D_1, A, f_{sim})
3: **end while**
4: $p \leftarrow \max(p_i)$
5: $N \leftarrow$ count$(\max(p_i))$
6: **return** p, N

Second, the attackers do not have to compare their records with every record because the method of evaluation is similar to that of k-anonymity, and the attackers only need to compare a representative of each group. The attackers need to compare their records with {-30, 175-}, {31-,-174}, and {31-, 175-} in (b) of Fig. 4.3, for instance. However, when the levels of generalization are different, such methods cannot be applied, and every record should be checked. To solve the problem, we first count the number of values of each attribute and then compare each attribute of r_j^0 with that of each record of D_1 in accordance with the large number of varieties.

Finally, when the procedure for anonymization is known in advance, it is possible to perform the evaluation more quickly by considering the effect of the initial part of

the anonymization. For instance, in Fig. 4.3a, we only have to consider cells whose values do not exceed 30 in $ATTR_1$ or fall short of 174 in $ATTR_2$.

4.3.3 Experiment

4.3.3.1 Experimental Environments

We conducted experiments to evaluate the validity of the proposed metrics. We measured the time to output the risk and confirmed that the privacy metric was appropriate. We used three parameters, k, β, ϵ, for comparison and verified the relationships among k-anonymity, sampling, and noise addition. We implemented our risk analysis method on a PC with an Intel Core i7-4790 3.6-GHz CPU and a 16.0-GB memory.

4.3.3.2 Dataset and Adversary Model

We used a pseudomedical dataset based on an actual medical dataset. The dataset had 10,000 records and two attributes, total cholesterol (TC) and HbA1c, and the

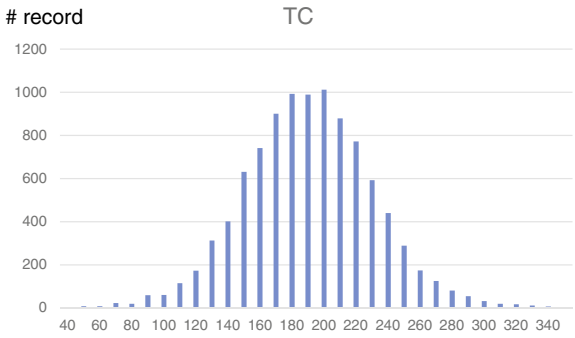


Fig. 4.5 Distribution of TC

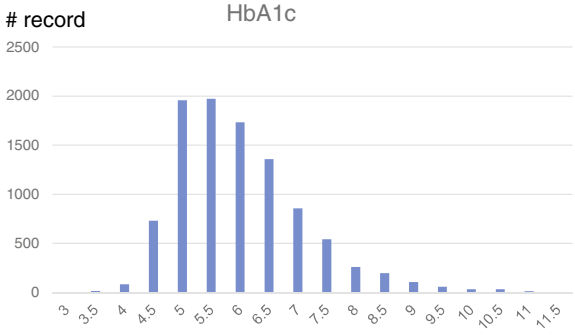


Fig. 4.6 Distribution of HbA1c

distribution of each attribute is shown in Figs. 4.5 and 4.6. We first measured the computation time while changing the number of records and then evaluated the validity of our metrics while changing the parameters of each anonymization method. Noise addition, generalization, and sampling were used as representative anonymization methods, and we adopted the Mondrian algorithm [9] for k-anonymization, Laplace noise for noise addition, and random sampling for sampling. We assumed *reidentifying adversary* A_1 to A_4. The conditions of the attacker models are the same as those of Sect. 4.3.2.4 except for noise addition. We define the f_{noise} of the A_2 and A_4 output records, whose value for each attribute differed by 5% from the original value, to be candidates.

4.3.4 Results

4.3.4.1 Computational Complexity

Our proposed privacy metrics are intended to be able to applied to large datasets. We measured the execution time by changing the number of records (Table 4.1) and parameters (Table 4.2, 4.3 and 4.4).

It takes little time to evaluate the risk when simple attackers, such as A_1 and A_3, are considered. On the other hand, when reflective attackers are assumed, the number of calculations increases and more time is required for evaluation. However, some of the processing described above reduces the time. For instance, the number of combinations of attributes increases with increasing numbers of records, and once an attacker has checked the risk of a record, that attacker does not have to calculate the risk of other records that have the same values. Therefore, the analysis algorithm is appropriate for large datasets.

Table 4.1 Execution time

# of records	A_1 (ms)	A_2 (ms)	A_3 (ms)	A_4 (ms)
1000	1.8	699.6	131.8	569.0
5000	2.6	17,005.6	751.2	8,920.8
10000	4.7	32,764.2	1,361.6	12,925.5

Table 4.2 The case of $\epsilon = 0.5, k = 2$

β	A_1 (ms)	A_2 (ms)	A_3 (ms)	A_4 (ms)
0.05	2.6	17,005.6	751.2	8,920.8
0.10	1.2	18,950.8	512.8	5,084.8
0.30	2.0	26,715.4	139.2	8,285.4

Table 4.3 The case of $\beta = 0.05, k = 2$

ϵ	A_1 (ms)	A_2 (ms)	A_3 (ms)	A_4 (ms)
0.5	2.6	17,005.6	751.2	8,920.8
1.0	1.4	17,002.4	628.6	9,256.4
3.0	1.6	16,894.8	945.0	8,968.2

Table 4.4 The case of $\beta = 0.05, \epsilon = 0.5$

k	A_1 (ms)	A_2 (ms)	A_3 (ms)	A_4 (ms)
2	2.6	17,005.6	751.2	8,920.8
3	2.9	16,828.6	744.2	8,788.4
4	2.8	17,211.9	755.8	9,013.1

Table 4.5 Relationship among parameters and our metrics (p, N)

$k = 2$		β		
		0.05	0.1	0.3
ϵ	0.1	(0.0196, 1)	(0.0303, 2)	(0.0909, 1)
	0.5	(0.0204, 1)	(0.0250, 1)	(0.1000, 1)
	1.0	(0.0208, 1)	(0.0278, 1)	(0.1000, 1)

When the sampling rate is changed, the computation time differs depending on the attacker. This is because there are two loop processes, one for sampled records and one for nonsampled records, and the calculation methods of each process differ depending on the attacker.

The effect of noise addition on computation time is not different in this experiment, but when a very large amount of noise is added, the distribution of the records is uniform and the different kinds of records increase; as a result, the computation time may increase.

The effect of k-anonymity also seems minimal, but when k is large the number of different types of records decreases and the computation time may decrease.

Validation

We observed p and N by changing the sampling rate β and the noise parameter ϵ to verify the validity of our metrics. We evaluated the attacker model A_4 while changing the parameters k, β, and ϵ. The evaluation result is shown below (Table 4.5, 4.6).

The risk to privacy decreases as k increases and as β and ϵ decrease, and the risk is a valid privacy metric. Sampling rates are the key factor that reduces the risk in this experiment. There are some outliers in the datasets, and they are the cause of the risk. In fact, if such records are not sampled, the privacy risk decreases. We conducted this experiment multiple times, and the result was different each time. Table 4.7 presents a sample of the evaluation results. Some outliers were included in

Table 4.6 Relationship among parameters and our metrics (p, N)

$k = 4$		β		
		0.05	0.1	0.3
ϵ	0.1	(0.0154, 1)	(0.0270, 2)	(0.0667, 2)
	0.5	(0.0192, 1)	(0.0227, 2)	(0.0625, 3)
	1.0	(0.0200, 1)	(0.0238, 2)	(0.0625, 1)

Table 4.7 Case of $\beta = 0.05, \epsilon = 1.0$

Times	A_1	A_2	A_3	A_4
1	(1.0000,3)	(0.0035,1)	(0.0083,1)	(0.0049,1)
2	(1.0000,2)	(0.0013,4)	(0.0108,1)	(0.0035,1)
3	(1.0000,4)	(0.0217,1)	(0.1667,1)	(0.0204,1)
4	(0.5000,5)	(0.0030,1)	(0.0667,1)	(0.0050,1)
5	(1.0000,5)	(0.0032,1)	(0.0294,1)	(0.0051,1)

the third operation, and the risk was higher than that of other operations. Therefore, the key factor may change when outliers are removed in advance.

4.4 Extension to Time-Sequence Data

4.4.1 Privacy Definition

We define two types of attack models for time-sequence datasets. The first, a reidentification attack, is a general attack model where an attacker has information on the original dataset M and tries to reidentify it in an anonymized dataset $A(M)$. This model assumes that an attacker has maximal information about the original dataset. This model is the same as that of k-anonymization, where even if an attacker has an original dataset, the probability of the reidentification of a k-anonymized dataset is $1/k$.

Definition 4.10 (*Reidentification attack*) Let an attacker have a matrix $M_{t_1} \in \mathbb{R}^{n \times m}$ and an anonymized matrix $A(M_{t_1}) \in \mathbb{R}^{n \times m}$. A reidentification attack against a record r_i succeeds if record $r_i \in M_{t_1}$ is linked to record $r'_j \in A(M_{t_1})$, where r_i and r'_j are the same user.

A linkage attack, which is an attack on a valid user, is one in which an attacker tries to obtain information from the given datasets $A(M_{t_1})$ and $A(M_{t_2})$. $A(M_{t_1})$ and $A(M_{t_2})$ are assumed to include the same users, but the primary keys are different. An attacker in this model has only anonymized datasets, so a valid user is assumed

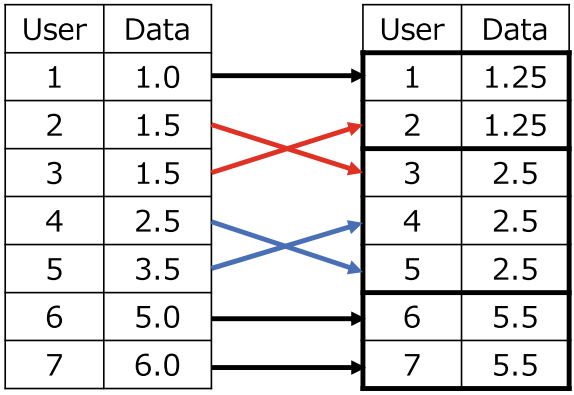


Fig. 4.7 Example of a risk evaluation

to be an attacker in this model. There are few studies concerning this problem, and we evaluate the risk using actual datasets in this paper.

Definition 4.11 (*Linkage attack*) Let an attacker have two anonymized matrices, $A(M_{t_1}) \in \mathbb{R}^{n\times m}$ and $A(M_{t_2}) \in \mathbb{R}^{n\times m}$. M_{t_1} and M_{t_2} include the same users and items, where each user and item of M_{t_2} are the same as those of M_{t_1}. A linkage attack against a record r_i succeeds if record $r'_i \in A(M_{t_1})$ is linked to record $r''_j \in A(M_{t_2})$, where r'_i and r''_j are the same user.

We next define the privacy metric as follows:

Definition 4.129 (*Privacy metric*) Let n be the total number of users of a dataset M and n' be the number of users that are successfully attacked. The privacy risk of M is defined as $\frac{n'}{n}$.

We consider the attacks to be the same as the previous ones to solve an assignment problem. An assignment problem is to find an appropriate task assignment when there are n users and tasks, and the Hungarian algorithm [47] solves the assignment problem in such a way that the entire cost is minimal.

We apply the same algorithm as used for reidentification and linkage attacks and assume that when an attacker assigns a record to the correct user, the attack succeeds. When a dataset is k-anonymized, there are at least $k-1$ of the same records. Hence, when a record is assigned to the cluster to which the correct record belongs to, we regard the record as being assigned correctly even if the assigned record is not actually correct. Furthermore, we define the privacy metric as the result obtained by multiplying the probability, and we define $1/k$ because the probability is the ratio of correctly assigned clusters (Fig. 4.7).

Figure 4.1 shows an example of a risk evaluation. The dataset on the left is the original dataset and that on the right is the anonymized dataset. The arrows indicate the assignment result. User 2 of the original dataset, for instance, is assigned to user 3 of the anonymized dataset, so the attack on user 2 fails. When noise addition is used as the anonymization method, users 2, 3, 4, and 5 are assigned to the wrong

users and the privacy risk is $3/7$. On the other hand, when k-anonymization is used, in this case, $k = 2$, users 4 and 5 are assigned to the wrong users (blue arrows) but are assigned to the clusters that are the same as those of the correct users. Therefore, we consider the attacks on users 4 and 5 to be successful. The failed attacks are only for users 2 and 3 (red arrows), and the privacy risk is $5/7 \times 1/2 = 5/14$.

4.4.2 Utility Definition

We define the utility metric here. In previous research, most utility metrics are based on either the distance between the original dataset and the anonymized dataset, or the amount of information loss [48, 49]. However, the utility depends on the situation (i.e., context and use case), and these metrics do not necessarily match the actual utility. Therefore, we consider a use case scenario and present a utility definition that matches the scenario. Specifically, we consider a use case in which an anonymized dataset is used as training data for a machine learning algorithm. In the case of a Web access log dataset, for example, a client, who is a developer of an anti-virus software, may generate a machine learning model from an anonymized dataset and predict whether their user will access a phishing Web site.

Definition 4.13 (*Utility metric*) Let $F(M, E)$ be the F-measure of a machine learning model, where the training data are M and the test data are E. The utility metric is defined as follows:

$$Uti(A(M)) = \frac{F(A(M), E)}{F(M, E)}. \tag{4.1}$$

Figure 4.8 gives an overview of the utility evaluation. We first generate two machine learning models: One is from an original dataset, and the other is from its anonymized dataset. An item is randomly chosen as an objective variable, and the remaining items are explanation variables. Then, we use these models and predict an attribute of each record of an evaluation dataset that has the same attributes as those of the original dataset. This operation is performed several times while an objective variable is changed. The utility is defined as the average of the ratio of the F-measure of a model of the anonymized dataset to that of a model of the corresponding original dataset. In this paper, we apply logistic regression as the machine learning algorithm and predict fifty attributes.

4.4.3 Matrix Factorization

Matrix factorization is a fundamental task in data analysis, and the technique is used in various scenarios, such as text data mining, acoustic analysis, and product recom-

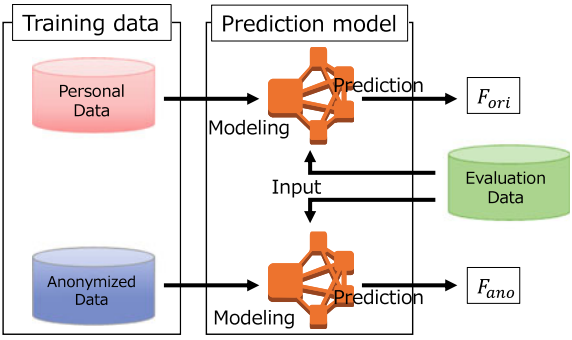


Fig. 4.8 Overview of utility evaluation

mendation by collaborative filtering. We use matrix factorization as an anonymization technique, so we present an overview of matrix factorization in this section.

4.4.3.1 SGD Matrix Factorization

We consider an unknown rank-r matrix $M \in \mathbb{R}^{n \times m}$ and assume that we know a set of elements $\Omega \subset [n] \times [m]$. $P_\Omega(M) \in \mathbb{R}^{n \times m}$ is defined as:

$$P_\Omega(M) = \begin{cases} M_{ij} & \text{if}(i, j) \in \Omega, \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

The goal of matrix factorization is to find two matrices $U \in \mathbb{R}^{r \times n}$ and $V \in \mathbb{R}^{r \times m}$ which approximate the original matrix $M_{ij} \approx X_{ij}$ s.t. $\forall M_{ij} \in \Omega(M)$ with lower dimensionality $r << min(n, m)$. Here, $X = U^\top V$.

This problem is defined to solve the following optimization problem:

$$\min_{u^*, v^*} \sum_{(i,j) \in P_\Omega(M)} (M_{ij} - u_i^\top v_j)^2 + \lambda(||u_i||^2 + ||v_j||^2), \tag{4.3}$$

where u_i is a vector of user factors and v_j is a vector of item factors. When u_i and v_j are variables, this function is not a convex set, so the problem described above cannot be solved. Some techniques are proposed to solve the problem, and gradient descent [50], for example, is a fundamental technique to find a local minimum value. However, gradient descent needs to update vectors iteratively to obtain an optimal solution and using gradient descent is computationally expensive, so stochastic gradient descent (SGD) is widely used, for example, in the KDD Cup 2011 [51] and the Netflix Prize [52].

There has been some research to speed up SGD-based matrix factorization, such as [53–56], and each algorithm updates the matrices in parallel or in a distributed manner.

In this paper, we apply a simple SGD technique to optimize formula (2) and denote $Update(A)$ as the update of a matrix A using the SGD technique.

4.4.4 Anonymization Using Matrix Factorization

We consider matrix factorization to be an anonymization method, and rank r contributes to the accuracy of the matrix approximation. Moreover, we propose combining matrix factorization with another anonymization method ano, such as k-anonymization or noise addition. We denote p as a parameter of the anonymization method, and p is k or ϕ in this paper. A basis matrix U and weighting matrix V can be assumed to be the characteristics of the rows and columns, respectively, and U is a characteristic matrix of users in our dataset. Therefore, we propose to anonymize U and maintain V so that the characteristics of the domain are preserved. In our algorithm, we first divide the dataset M into U and V, and anonymize U. Then, we optimize V once and recombine it with the anonymized U. The algorithm is described below.

We indicate that $A_r(D)$ applies matrix factorization to matrix D and that $A_{(ano,r)}(D)$ combines matrix factorization and the anonymization method ano by:

$$A_{(ano,r)}(D) = (A_{(ano)}(U))^{\mathsf{T}} V, \text{where } U \in \mathbb{R}^{r \times n},\ V \in \mathbb{R}^{r \times m}. \tag{4.4}$$

Algorithm 4.6 (M, r, I, ano, p) : Anonymization using Matrix Factorization

Input: Original dataset M, rank r, and the number of iterations I.

1: $t = 0$
2: Construct $U_t \in [0, 1]^{n \times r}$ and $V_t \in [0, 1]^{m \times r}$ randomly
3: **while** $t < I$ **do**
4: $\quad U_{t+1} = Update(U_t)$
5: $\quad V_{t+1} = Update(V_t)$
6: $\quad t = t + 1$
7: **end while**
8: $U'_{t+1} = A_{(ano)}(U_{t+1})$
9: **return** $X = U'^{\mathsf{T}}_{t+1} V_{t+1}$

Table 4.8 Dataset format

ID (= i)	Date	URL (= j)
x_{t_1} (= 1)	2016-12-01 16:13:48	www.google.com (= 1)
y_{t_1} (= 2)	2016-12-01 16:15:14	www.mail.google.com (= 2)
x_{t_1}	2016-12-01 16:17:13	www.youtube.com (= 3)
z_{t_1} (= 3)	2016-12-01 16:19:01	www.facebook.com (= 4)
x_{t_2} (= 1)	2016-12-01 16:21:15	www.youtube.com
x_{t_2}	2016-12-01 16:22:42	www.google.com
z_{t_2} (= 3)	2016-12-01 16:25:01	www.youtube.com

4.4.5 Experiment

4.4.5.1 Dataset

We use an actual Web access log dataset as a time-sequence dataset. The dataset consists of an ID, a time stamp, and the access domain, as shown in Table 4.8. We convert the dataset into a matrix as follows:

$$M_T = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nm} \end{bmatrix} \tag{4.5}$$

Here, T is the observation time.

We say $r_{ij} = 1$ if a user whose ID is i accesses domain j during time T, and otherwise, $r_{ij} = 0$. For example, we construct the datasets in Table 4.8 as follows:

$$M_{t_1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.6}$$

$$M_{t_2} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.7}$$

Here, t_1 is the 10-min span between 2016-12-01 16:10:00 and 2016-12-01 16:19:59, and t_2 is the similar 10-min span between 2016-12-01 16:20:00 and 2016-12-01 16:29:59. The IDs are different between t_1 and t_2, but x_{t_1} and x_{t_2}, and z_{t_1} and z_{t_2} represent the same users.

Table 4.9 Linkage attack against a non-anonymized dataset

Observation time (h)	Linkage attack probability
2	0.51
4	0.64
8	0.80

In the following experiments, we chose randomly 200 users and 1,000 domains from an actual Web access log and let the pseudonymous ID be changed at each designated time T.

4.4.5.2 The Privacy Risk Against a Linkage Attack

First, we evaluate whether a linkage attack is possible. We set the observation time t_1 as 2, 4, and 8 h from 16:00 on a weekday and the observation time t_2 as the same time on another weekday. The probability of a linkage attack between M_{t_1} and M_{t_2} is shown in Table 4.9.

The matrix only includes information on whether a domain has been accessed, and even if the observation time is 2 h, the linkage attack probability, i.e., risk, is very high (over 50%). Moreover, the risk increases as the observation time increases because when the observation time increases, the trend of a user becomes noticeable. The result shows that the pattern of Web access for people has consistent characteristics. Hence, we need to consider not only reidentification attacks but also linkage attacks to avoid privacy leakages.

4.4.5.3 Effects of Matrix Factorization

Observation times t_1 and t_2 are fixed as 8 h from 16:00 h on a weekday in the following experiments. The inputs of matrix factorization are the original dataset M, the number of iterations I, and the rank r. Furthermore, λ and γ and are the hyperparameters. We fix $I = 100$, which is enough to converge, $\gamma = 0.05$, and $\lambda = 0.01$. The convergence result is shown in Fig. 4.9. The rank r can be treated as the parameter of anonymization by matrix factorization because the accuracy of dataset $X = UV^{\mathsf{T}}$ depends on the rank r, so r is the parameter of our algorithm; we set $r = 10, 20, 30, 40$. We set larger values in the experiments in [3], but the results of the case $r > 40$ are saturated. The probabilities of reidentification and linkage attacks are shown in Table 4.10.

The results show that matrix factorization itself does not have much effect on reidentification attacks. Note that matrix factorization can preserve the relative positional relationship among the records so that the privacy risk of the reidentification attack does not decrease much by using a matching algorithm. When the rank is

Fig. 4.9 Convergence result

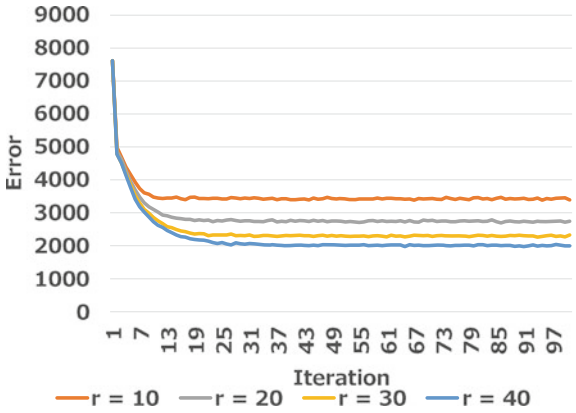
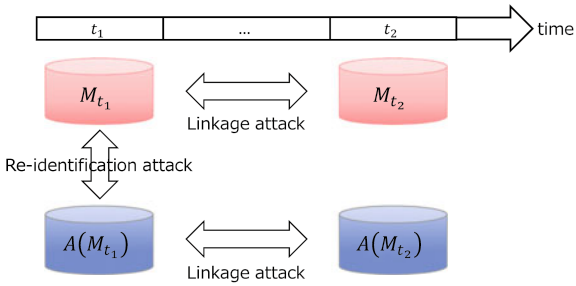


Table 4.10 Attacks against matrix factorization

Rank	Reidentification attack	Linkage attack
10	0.98	0.31
20	1.00	0.45
30	1.00	0.54
40	1.00	0.58

Fig. 4.10 Overview of the experiment



small enough, $r = 10$, the positional relationship is broken, and the privacy risk is lowered.

On the other hand, compared with the reidentification attack presented in Table 4.9, the linkage attack probability between $A_r(M_{t_1})$ and $A_r(M_{t_2})$ is better. This is because the relationship between the records of M_{t_1} and M_{t_2} is weaker than that between M_{t_1} and $A_r(M_{t_1})$. In our experiment, the dataset of the observation time is 8 h and $r = 30$ has almost the same privacy level as when the observation time is 2 h (Fig. 4.10).

Table 4.11 Experiment 1

k	Reidentification attack	Linkage attack
2	0.500	0.185
4	0.250	0.050
6	0.167	0.038
8	0.125	0.027
10	0.098	0.023

4.4.6 Results

4.4.6.1 Risk Evaluation

We evaluate our anonymization method, Algorithm 4.1, in the following experiments. We apply the method described in [10] as k-anonymization and Laplace noise as the noise addition. When noise addition is applied, noise $\epsilon \sim Lap(0, 2\phi^2)$ is added to each element, and the parameter is ϕ.

1. Evaluate the privacy risk of a reidentification attack between $A_k(M_{t_1})$ and M_{t_1} and a linkage attack between $A_k(M_{t_1})$ and $A_k(M_{t_2})$.
2. Evaluate the privacy risk of a reidentification attack between $A_\phi(M_{t_1})$ and M_{t_1} and a linkage attack between $A_\phi(M_{t_1})$ and $A_\phi(M_{t_2})$.
3. Evaluate the privacy risk of reidentification attacks between $A_k(U_{t_1})^\mathsf{T} V$ and M_{t_1} and linkage attacks between $A_k(U_{t_1})^\mathsf{T} V$ and $A_k(U_{t_2})^\mathsf{T} V$.
4. Evaluate the privacy risk of reidentification attacks between $A_\phi(U_{t_1})^\mathsf{T} V$ and M_{t_1} and linkage attacks between $A_\phi(U_{t_1})^\mathsf{T} V$ and $A_\phi(U_{t_2})^\mathsf{T} V$.

The evaluations of the reidentification attacks in experiments 1 and 2 are almost the same as those conducted in many previous studies. The difference is the privacy metric (see 4.4.1), and these results are used for comparison with experiments 3 and 4, which are evaluations of our algorithm. There are few studies on linkage attacks, and evaluations of this type of attack are one of our contributions.

The evaluation of the reidentification attack in experiment 1 (Table 4.11) is simple, and the result is almost the same as for k-anonymization. However, our privacy metric is slightly different from that for k-anonymity, so the result is also slightly different from $1/k$. The result of the linkage attack also shows that k-anonymization can greatly improve the privacy of linkage attacks and that 2-anonymization can reduce the privacy risk by 77%($0.8 \rightarrow 0.185$).

The evaluations of experiment 2 are shown in Table 4.12. The privacy of the reidentification attack is improved from $\phi \geq 0.9$, and when ϕ is large, for example, $\phi = 1.5$, the score appears to be good. However, almost half of the records are changed by more than 1 by the added noise, and each original value of M is 0 or 1, namely, $M_{ij} \in \{0, 1\}$, so that the noise is too large to preserve utility. Therefore, we conclude that simple noise addition is not good, in terms of utility preservation, as an

Table 4.12 Experiment 2

ϕ	Reidentification attack	Linkage attack
0.3	1.00	0.33
0.6	1.00	0.10
0.9	0.95	0.01
1.2	0.81	0.03
1.5	0.62	0.00

Table 4.13 Experiment 3: reidentification attack

k	$r = 10$	$r = 20$	$r = 30$	$r = 40$
2	0.44	0.50	0.50	0.50
4	0.21	0.24	0.25	0.25
6	0.12	0.14	0.15	0.16
8	0.10	0.11	0.11	0.12
10	0.08	0.08	0.08	0.08

Table 4.14 Experiment 3: linkage attack

k	$r = 10$	$r = 20$	$r = 30$	$r = 40$
2	0.11	0.15	0.15	0.15
4	0.05	0.07	0.08	0.07
6	0.04	0.03	0.03	0.04
8	0.03	0.03	0.03	0.03
10	0.02	0.02	0.02	0.02

anonymization method. On the other hand, we obtain an interesting result for linkage attacks. The privacy for linkage attacks is improved even if the noise is very small and adding even a small amount of noise is an effective countermeasure against a linkage attack.

In experiment 3, we evaluate the effect of our proposed algorithm, which is a combination of matrix factorization and k-anonymization. Table 4.13 presents the result of the reidentification attack. In the experiment, we cannot find the effect of the matrix factorization very well, but the privacy slightly improves as r increases. This is because k-anonymization has a large effect on the reidentification risk, and the effect of the matrix factorization does not appear.

The results of the linkage attack in experiment 3 are shown in Table 4.14. In the experiment, we cannot obtain new knowledge about the effect of matrix factorization. When the datasets, which are observed at different time periods, are sufficiently anonymized by k-anonymization, there is no relationship among the same users of each dataset and only outliers can be linked.

Table 4.15 Experiment 4: reidentification attack

ϕ	$r = 10$	$r = 20$	$r = 30$	$r = 40$
0.05	0.75	0.95	0.97	1.00
0.10	0.42	0.72	0.85	0.86
0.15	0.25	0.50	0.61	0.70
0.20	0.18	0.28	0.40	0.49

Table 4.16 Experiment 4: linkage attack

ϕ	$r = 10$	$r = 20$	$r = 30$	$r = 40$
0.05	0.21	0.34	0.34	0.50
0.10	0.12	0.15	0.14	0.20
0.15	0.07	0.11	0.09	0.10
0.20	0.03	0.03	0.03	0.02

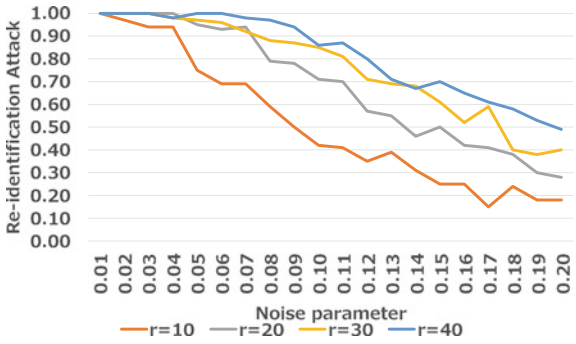


Fig. 4.11 Reidentification risk of the combination of matrix factorization and noise addition

In experiment 4, we evaluate the impact of our method, which is a combination of matrix factorization and noise addition. The evaluation results of the reidentification attack are presented in Table 4.15. Noise is added to U, which is the user's characteristics, and then, U^{T} is multiplied by V. Therefore, we cannot simply compare the results with those of experiment 2, but the impact of the matrix factorization is high. This result shows that using matrix factorization can help to construct anonymized datasets flexibly from the viewpoint of privacy. For example, the privacy risk of $A_{(\phi=0.15,r=20)}(M_{t_1})$ and $A_{(\phi=0.20,r=40)}(M_{t_1})$ is almost the same as that of $A_{(k=2)}(M_{t_1})$ and $A_{(\phi=1.5)}(M_{t_1})$.

The results of the linkage attack in experiment 4 are presented in Table 4.16. The trend is the same as that of the reidentification attack, and the matrix factorization is compatible with noise addition. We present the details of the results of the reidentification attack and the linkage attack in Figs. 4.11 and 4.12.

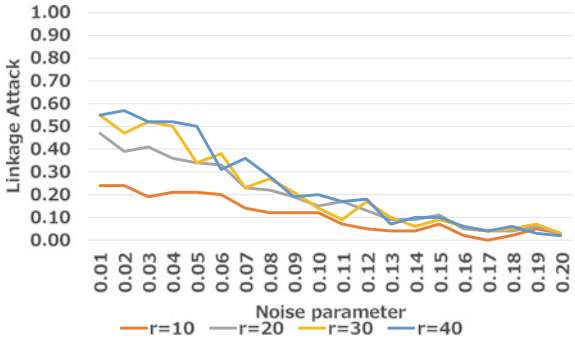


Fig. 4.12 Linkage risk of the combination of matrix factorization and noise addition

Table 4.17 Utility evaluation 1

Dataset D	Precision	Recall	F-measure	$Uti(D)$
$A_{(k=2)}(M_{t_1})$	0.780	0.720	0.749	0.981
$A_{(k=4)}(M_{t_1})$	0.741	0.688	0.714	0.936
$A_{(k=6)}(M_{t_1})$	0.755	0.691	0.721	0.946
$A_{(k=8)}(M_{t_1})$	0.737	0.659	0.696	0.913
$A_{(k=10)}(M_{t_1})$	0.748	0.677	0.711	0.932

4.4.6.2 Utility Evaluation

We next evaluate the utility of anonymized datasets. We evaluate the utility of datasets by applying a machine learning algorithm. Logistic regression (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html) is applied in the following experiment, and the parameters are those of the default setting. One of the applications of an access log dataset is to predict a malicious site and inform the web browser's users. Therefore, we use a machine learning algorithm and predict whether each user will access a malicious site. We generate learning models using the original (non-anonymized) dataset and the anonymized datasets and input the test dataset to these models. The utility score is defined in Definition 4.13, and the F-measure of the model of the original dataset is 0.763. Each result of the evaluation is shown in Tables. 4.17, 4.18, 4.19, and 4.20.

1. Evaluate the utility of $A_{(k)}(M_{t_1})$ for $k = 2, 4, 6, 8$, and 10.
2. Evaluate the utility of $A_{(\phi)}(M_{t_1})$ for $\phi = 0.3, 0.6, 0.9, 1.2$, and 1.5.
3. Evaluate the utility of $A_{(k=2,r)}(M_{t_1})$ for $r = 10, 20, 30$, and 40.
4. Evaluate the utility of $A_{(\phi,r)}(M_{t_1})$ for $\phi = 0.1$ and 0.15 and $r = 10, 20, 30$, and 40.

In experiment 1, each element is $M_{ij} \in \{0, 1\}$ and the matrix is sparse, even when k-anonymization is effective. However, when the dataset is more complex, the utility of k-anonymization will decrease; this is widely known as the curse of dimensionality.

Table 4.18 Utility evaluation 2

Dataset D	Precision	Recall	F-measure	$Uti(D)$
$A_{(\phi=0.3)}(M_{t_1})$	0.780	0.664	0.717	0.941
$A_{(\phi=0.6)}(M_{t_1})$	0.738	0.610	0.668	0.876
$A_{(\phi=0.9)}(M_{t_1})$	0.719	0.541	0.618	0.810
$A_{(\phi=1.2)}(M_{t_1})$	0.652	0.507	0.571	0.748
$A_{(\phi=1.5)}(M_{t_1})$	0.625	0.520	0.567	0.744

Table 4.19 Utility evaluation 3

Dataset D	Precision	Recall	F-measure	$Uti(D)$
$A_{(k=2,r=10)}(M_{t_1})$	0.686	0.735	0.710	0.930
$A_{(k=2,r=20)}(M_{t_1})$	0.699	0.767	0.731	0.959
$A_{(k=2,r=30)}(M_{t_1})$	0.695	0.773	0.732	0.960
$A_{(k=2,r=40)}(M_{t_1})$	0.712	0.786	0.747	0.980

Table 4.20 Utility evaluation 4

Dataset D	Precision	Recall	F-measure	$Uti(D)$
$A_{(\phi=0.10,r=10)}(M_{t_1})$	0.742	0.650	0.693	0.909
$A_{(\phi=0.10,r=20)}(M_{t_1})$	0.752	0.688	0.719	0.943
$A_{(\phi=0.10,r=30)}(M_{t_1})$	0.736	0.703	0.719	0.943
$A_{(\phi=0.10,r=40)}(M_{t_1})$	0.737	0.735	0.736	0.965

Table 4.21 Utility evaluation 5

Dataset D	Precision	Recall	F-measure	$Uti(D)$
$A_{(\phi=0.15,r=10)}(M_{t_1})$	0.718	0.614	0.662	0.868
$A_{(\phi=0.15,r=20)}(M_{t_1})$	0.748	0.655	0.698	0.915
$A_{(\phi=0.15,r=30)}(M_{t_1})$	0.704	0.680	0.692	0.907
$A_{(\phi=0.15,r=40)}(M_{t_1})$	0.716	0.711	0.713	0.935

The results of experiment 2 show that the utility of the dataset decreases as noise increases. As stated in the risk evaluation section, each element of the original dataset is 0 or 1, and the utility drastically worsens when the noise parameter is large, such as $\phi = 1.5$.

When k-anonymization and matrix factorization are combined, the effect of matrix factorization is small, as is the case for the privacy risk. In this experiment, the effect of k-anonymization is large, and the effect of matrix factorization is relatively small.

The evaluation results of the combination of noise addition and matrix factorization show a good performance (Tables 4.20 and 4.21). A dataset generated by combining matrix factorization and noise addition has more utility than a dataset generated by noise addition when each dataset has the same privacy level.

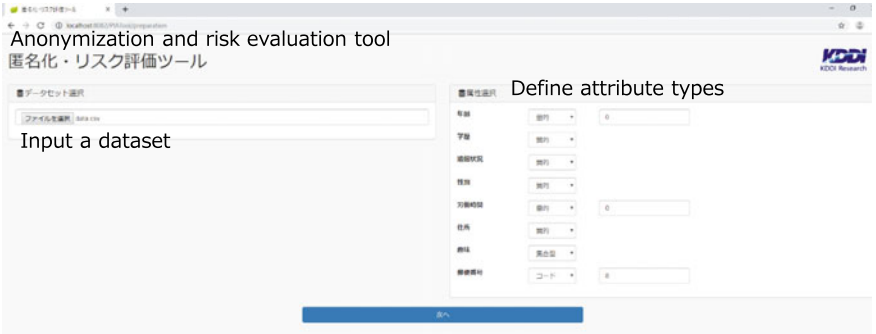


Fig. 4.13 Anonymization and privacy risk evaluation tool 1

4.5 Anonymization and Privacy Risk Evaluation Tool

In this section, we introduce an anonymization and privacy risk evaluation tool. So far, we have shown how to evaluate the privacy and utility of several datasets. We focus on static datasets and apply the theory we have described in the tool. First, we explain the outline of the tool. The tool requires a dataset that is the target of anonymization and privacy risk evaluation. At this time, the data type is defined for each attribute (see Fig. 4.13). Numerical, qualitative, set, code, and sensitive types can be defined. Age, height, and weight are defined as numerical types, and a user can assign a range of values. For instance, a user may want to divide age into groups of two years or five years depending on the situation. Qualitative-type records have nonnumerical value, such as gender and occupation. The set type is an extended numerical or qualitative type, and attributes that include multiple data correspond to this type. The code type is defined when every value is the same digit, such as a postcode. The sensitive type corresponds to sensitive information. The privacy risk is evaluated using quasi-identifiers in our tool, and the attributes that are sensitive do not effect the privacy risk. However, it is known that sensitive information may cause privacy leakages, and the tool can cover the risk for sensitive information such as l-diversity.

After the type of each attribute is decided, a user defines the noise and sampling parameters. Our tool can evaluate datasets that are anonymized by the combined method. Then, the user generates a hierarchical tree for each attribute, and the tool anonymizes the values in accordance with the tree. The user can generate and change the construction of hierarchical trees by using a UI (see Fig. 4.14.).

After these preparations are finished, the user can define the conditions and generate a dataset flexibly. A sample operation screen is shown in Fig. 4.15. Let us introduce a method commonly used as an example. First, a user searches records that do not achieve k-anonymity. Namely, the user searches records that do not include more than k copies of the same record, and then the user changes the level of an attribute of the records. The records that are secure enough are not processed, so the

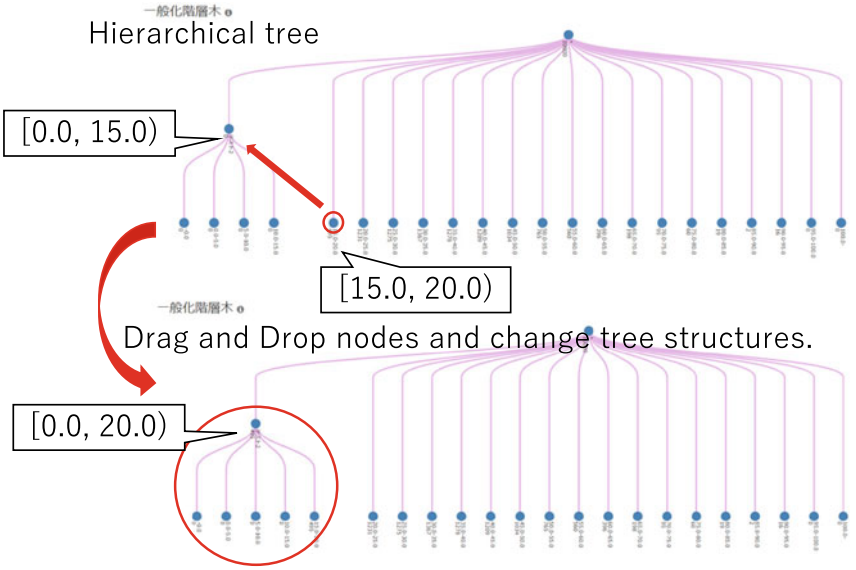


Fig. 4.14 Anonymization and privacy risk evaluation tool 2

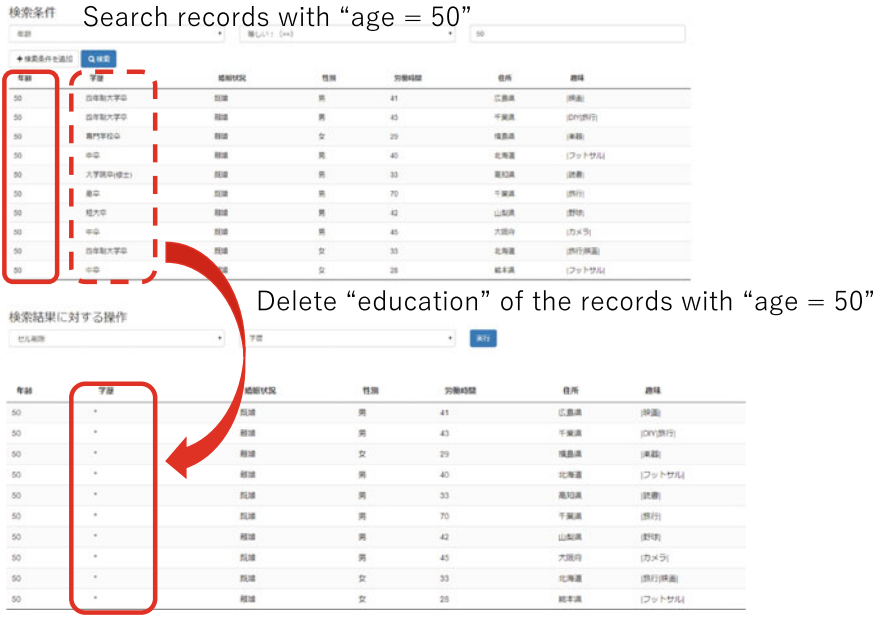


Fig. 4.15 Anonymization and privacy risk evaluation tool 3

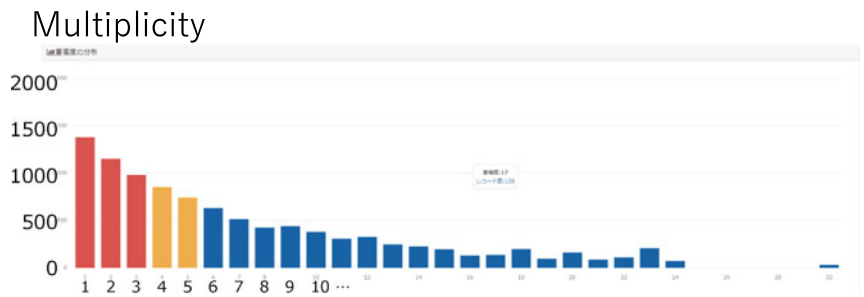


Fig. 4.16 Anonymization and privacy risk evaluation tool 4

utility of the dataset can be maintained. The conditions can be more complex. For example, the records that have a value of "age" over 80 and a value of "occupation" that is not "self-employed" are identified and anonymized. The ranks of the records are "balanced" according to the hierarchical tree. The privacy risk can be seen in real time (in Fig. 4.16), and the user can anonymize a dataset by trial and error. The operation procedure can be output as a setting file, and once the operation is decided, the procedure can be performed automatically, such as in batch processing.

4.6 Conclusion

In this chapter, we considered the importance of data and privacy. Several anonymization techniques, including k-anonymization, are introduced in Sect. 4.2, and the privacy and adversary model for static data are shown in Sect. 4.3. We focused on static data and time-sequence data in this project, and we discuss time-sequence data in Sect. 4.4. Finally, in Sect. 4.5, we introduce an anonymization and privacy risk evaluation tool. The tool is partly developed in this project, and we are proactive in using it commercially.

References

1. S. Kiyomoto, K. Fukushima, Y. Miyake, Privacy preservation of user history graph, in *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*, ed. by I. Askoxylakis, H.C. Pöhls, J. Posegga (Springer, Berlin, 2012), pp. 87–96
2. T. Mimoto, S. Kiyomoto, K. Tanaka, A. Miyaji, (p, n)-identifiability: anonymity under practical adversaries, in *2017 IEEE Trustcom/BigDataSE/ICESS* (IEEE, 2017), pp. 996–1003
3. T. Mimoto, S. Kiyomoto, S. Hidano, A. Basu, A. Miyaji, The possibility of matrix decomposition as anonymization and evaluation for time-sequence data, in *2018 16th Annual Conference on Privacy, Security and Trust (PST)* (IEEE, 2018), pp. 1–7

4. P. Samarati, L. Sweeney, Generalizing data to provide anonymity when disclosing information, in *Proceedings of PODS 1998* (1998), p. 188
5. P. Samarati, Protecting respondents' identities in microdata release. IEEE Trans. Knowl. Data Eng. **13**(6), 1010–1027 (2001)
6. L. Sweeney, Achieving *k*-anonymity privacy protection using generalization and suppression. J. Uncert. Fuzziness Knowl.-Base Syst. **10**(5), 571–588 (2002)
7. S. Kiyomoto, K.M. Martin, Towards a common notion of privacy leakage on public database, in *2010 International Conference on Broadband, Wireless Computing, Communication and Applications* (2010), pp. 186–191
8. K. LeFevre, D.J. DeWitt, R. Ramakrishnan, Incognito: efficient full-domain *k*-anonymity. Proceedings of SIGMOD **2005**, 49–60 (2005)
9. K. LeFevre, D.J. DeWitt, R. Ramakrishnan, Mondrian multidimensional k-anonymity, in *Proceedings of the 22nd International Conference on Data Engineering (ICDE '06)* (IEEE, 2006), pp. 25–35
10. J.-W. Byun, A. Kamra, E. Bertino, N. Li, Efficient k-anonymization using clustering techniques, in *International Conference on Database Systems for Advanced Applications* (Springer, 2007), pp. 188–200
11. K. Mivule, Utilizing noise addition for data privacy, an overview (2013). arXiv:1309.3958
12. J.J. Kim, A method for limiting disclosure in microdata based on random noise and transformation, in *Proceedings of the Section on Survey Research Methods* (American Statistical Association, 1986), pp. 303–308
13. T. Yu, S. Jajodia, *Secure Data Management in Decentralized Systems*, vol. 33 (Springer Science & Business Media, Berlin, 2007)
14. S. Kiyomoto, K. Fukushima, Y. Miyake, Data anonymity in multi-party service model, in *Security Technology*, ed. by T.-H. Kim, H. Adeli, W.-C. Fang, J.G. Villalba, K.P. Arnett, M.K. Khan (Springer, Berlin, 2011), pp. 21–30
15. Y. Zheng, L. Zhang, Z. Ma, X. Xie, W.-Y. Ma, Recommending friends and locations based on individual location history. ACM Trans. Web (TWEB) **5**(1), 5 (2011)
16. Y. Zheng, L. Zhang, X. Xie, W.-Y. Ma, Mining interesting locations and travel sequences from gps trajectories, in *Proceedings of the 18th International Conference on World Wide Web* (ACM, 2009), pp. 791–800
17. V.W. Zheng, Y. Zheng, X. Xie, Q. Yang, Collaborative location and activity recommendations with gps history data, in *Proceedings of the 19th International Conference on World Wide Web* (ACM, 2010), pp. 1029–1038
18. S. Chawla, Y. Zheng, J. Hu, Inferring the root cause in road traffic anomalies, in *2012 IEEE 12th International Conference on Data Mining (ICDM)* (IEEE, 2012), pp. 141–150
19. Y. Zheng, Trajectory data mining: an overview. ACM Trans. Intell. Syst. Technol. (TIST) **6**(3), 29 (2015)
20. C. Dwork, Differential privacy, in *Proceedings of ICALP 2006*. LNCS, vol. 4052 (2006), pp. 1–12
21. C. Dwork, Differential privacy: a survey of results, in *Proceedings of TAMC 2008*. LNCS, vol. 4978 (2008), pp. 1–19
22. C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, M. Naor, Our data, ourselves: privacy via distributed noise generation, in *Proceedings of Eurocrypt 2006*. LNCS, vol. 4004 (2006), pp. 486–503
23. C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in *Proceedings of TCC 2006*. LNCS, vol. 3876 (2006), pp. 265–284
24. C. Dwork, G.N. Rothblum, S. Vadhan, Boosting and differential privacy. Proceedings of IEEE FOCS **2010**, 51–60 (2010)
25. P. Kodeswaran, E. Viegas, Applying differential privacy to search queries in a policy based interactive framework, in *Proceedings of PAVLAD '09* (ACM, 2009), pp. 25–32
26. C. Li, M. Hay, V. Rastogi, G. Miklau, A. McGregor, Optimizing linear counting queries under differential privacy, in *Proceedings of PODS '10* (ACM, 2010), pp. 123–134

27. I. Mironov, O. Pandey, O. Reingold, S. Vadhan, Computational differential privacy, in *Proceedings of CRYPTO 2009, LNCS*, vol. 5677 (2009), pp. 126–142
28. A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, S. Vadhan, The limits of two-party differential privacy. Proceedings of IEEE FOCS **2010**, 81–90 (2010)
29. A. Groce, J. Katz, A. Yerukhimovich, Limits of computational differential privacy in the client/server setting, in *Proceedings of TCC 2011*, to appear. LNCS (2011)
30. M.R. Clarkson, F.B. Schneider, Quantification of integrity, in *Proceedings of 23rd IEEE Computer Security Foundations Symposium* (IEEE, 2010), pp. 28–43
31. T.-S. Hsu, C.-J. Liau, D.-W. Wang, J.K.-P. Chen, Quantifying privacy leakage through answering database queries, in *Proceedings of ISC '02*. LNCS, vol. 2433 (2002), pp. 162–176
32. Y.C. Chiang, T.-S. Hsu, S. Kuo, D.-W. Wang, Preserving confidentially when sharing medical data, in *Proceedings of Asia Pacific Medical Information Conference* (2000)
33. Y.T. Chiang, Y.C. Chiang, T.-S. Hsu, C.-J. Liau, D.-W. Wang, How much privacy? - a system to safe guard personal privacy while releasing database, in *Proceedings of 3rd International Conference on Rough Sets and Current Trends in Computing, LNCS*, vol. 2475 (2002), pp. 226–233
34. A. Krause, E. Horvitz, A utility-theoretic approach to privacy and personalization, in *Proceedings of AAAI'08*, vol. 2 (2008), pp. 1181–1188
35. A. Krause, E. Horvitz, A utility-theoretic approach to privacy in online services. J. Artif. Intell. Res. **39**, 633–662 (2010)
36. L. Zayatz, Disclosure avoidance practices and research at the us census bureau: an update. J. Offic. Stat. **23**(2), 253 (2007)
37. M. Freiman, J. Lucero, L. Singh, J. You, M. DePersio, L. Zayatz, The microdata analysis system at the us census bureau, in *Proceedings of the Joint Statistical Meetings, American Statistical Association Section on Survey Research Methods* (2011)
38. K. Chaudhuri, N. Mishra, When random sampling preserves privacy, in *Annual International Cryptology Conference* (Springer, 2006), pp. 198–213
39. C. Dwork, A. Roth et al., The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. **9**(3–4), 211–407 (2014)
40. J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, S. Martínez, Enhancing data utility in differential privacy via microaggregation-based k-anonymity. The VLDB J. **23**(5), 771–794 (2014)
41. L. Sweeney, k-anonymity: a model for protecting privacy. Int. J. Uncert. Fuzziness Knowl.-Based Syst. **10**(5), 557–570 (2002)
42. A. Narayanan, V. Shmatikov, How to break anonymity of the netflix prize dataset (2006), arXiv:cs/0610105
43. A. Machanavajjhala, J. Gehrke, D. Kifer, *t*-closeness: privacy beyond *k*-anonymity and *l*-diversity, in *Proceedings of ICDE'07* (2007), pp. 106–115
44. C. Dwork, Differential privacy: a survey of results. Proceedings of TAMC **4978**(2008), 1–19 (2008)
45. Y. Omori, Posterior probability of population uniqueness in microdata. Proc. Inst. Stat. Math. **51**(2), 223–239 (2003)
46. R.J.A. Little, Statistical analysis of masked data. J. Offic. Stat. **9**(2), 407 (1993)
47. H.W. Kuhn, The hungarian method for the assignment problem. Naval Res. Logist. (NRL) **2**(1–2), 83–97 (1955)
48. J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, A.W.-C. Fu, Utility-based anonymization for privacy preservation with less information loss. SIGKDD Explor. Newsl. **8**(2), 21–30 (2006)
49. J.-W. Byun, A. Kamra, E. Bertino, N. Li, Efficient k-anonymity using clustering technique, in *Proceedings of the International Conference on Database Systems for Advanced Applications* (2007), pp. 188–200
50. J. Nocedal, S. Wright, *Numerical Optimization* (Springer Science & Business Media, Berlin, 2006)
51. G. Dror, N. Koenigstein, Y. Koren, M. Weimer, The yahoo! music dataset and kdd-cup'11, in *Proceedings of the 2011 International Conference on KDD Cup 2011*, vol. 18 (JMLR.org, 2011), pp. 3–18

52. R.M. Bell, Y. Koren, Lessons from the netflix prize challenge. SiGKDD Explor. **9**(2), 75–79 (2007)
53. B. Recht, C. Re, S. Wright, F. Niu, Hogwild: a lock-free approach to parallelizing stochastic gradient descent, in *Advances in Neural Information Processing Systems* (2011), pp. 693–701
54. R. Gemulla, E. Nijkamp, P.J. Haas, Y. Sismanis, Large-scale matrix factorization with distributed stochastic gradient descent, in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2011), pp. 69–77
55. Y. Zhuang, W.-S. Chin, Y.-C. Juan, C.-J. Lin, A fast parallel sgd for matrix factorization in shared memory systems, in *Proceedings of the 7th ACM Conference on Recommender Systems* (ACM, 2013), pp. 249–256
56. J. Oh, W.-S. Han, H. Yu, X. Jiang, Fast and robust parallel sgd matrix factorization, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2015), pp. 865–874