Service Oriented Computing and Applications (2018) 12:259–273
https://doi.org/10.1007/s11761-018-0248-y

SPECIAL ISSUE PAPER

Privacy preservation for recommendation databases

Surapon Riyana[1] · Juggapong Natwichai[1,2]

Received: 18 March 2018 / Revised: 10 October 2018 / Accepted: 15 October 2018 / Published online: 31 October 2018

Abstract
Since recommendation systems play an important role in the current situations where such digital transformation is highly demanded, the privacy of the individuals' collected data in the systems must be secured effectively. In this paper, the vulnerability of the existing query framework for the recommendation systems is identified. Thus, we propose to apply the well-known k-anonymity model to generalize the given recommendation databases to satisfy the privacy preservation constraint. We show that such data generalization problem which minimizes the impact on data utility is NP-hard. To tackle with such problem, an algorithm to preserve the privacy of the individuals in the recommendation databases is proposed. The idea is to avoid excessive generalizing on the databases by forming a group of similar tuples in the databases. Thus, the impact on the data utility of the generalizing such group can be minimized. Our work is evaluated by extensive experiments. From the results, it is found that our work is highly effective, i.e., the impact quantified by the data utility metrics and the errors of the query results are less than the compared algorithms, and also it is highly efficient, i.e., the execution time is less than the result of its effectiveness-comparable algorithm by more than three times.

Keywords Recommendation databases · Privacy preservation · k-Anonymity

1 Introduction

Recommendation systems for suggestion of suitable artifacts to the target user play an important role since their introduction in 1995 [1]. Due to its flexibility, the range of the applications of recommendation systems can be very wide, e.g., e-commerce, telecommunication, or even software development industry. On business side, in [2], it has been reported that the market share which utilizes the recommendation systems was at least $69.7 billion US dollars in 2016. Furthermore, it has been forecasted that the systems can contribute up to $188.9 billion US dollars in only one business sector in the near future. On the other end, for the software development industry, such example of the applications is recommendation on appropriate servers in the Infrastructure as a Service (IaaS) cloud services [3] or recommendation of web services in the discovery process [4].

One of the important components in recommendation systems is recommendation databases, where the historical data of users and their rating on artifacts are stored. They compose historical behavior of users in the system, i.e., how they rate the object of interests, or so-called artifacts, and the other personal information of the users, e.g., users' profile, or the location of the users. The structure of such databases can be depicted in Fig. 1. An example of the databases is shown in Table 1. From the table, it can be seen that each row represents one user. There are five artifacts, which are books, and five personal attributes in this example. The rating of a user against each artifact is represented by a number, in which the higher means the more preference on such artifact. For example, Tuple ID 10, a 29-year-old married person with doctoral degree in education who obtains $51,000 as salary and lives in Columbus city, prefers the book with titled Joy Ride more than Egomaniac and Warcross. Such kind of databases can be collected, and the recommendation engines [5,6] can utilize the data to recommend potential preferred artifacts to the users as a framework shown in Fig. 2.

The recommendation databases are not only utilized in conventional approach as mentioned, but they can also be

✉ Juggapong Natwichai
juggapong@eng.cmu.ac.th

Surapon Riyana
surapon.riyana@gmail.com

1 Data Engineering and Network Technology Laboratory, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand

2 Center of Data Analytics and Knowledge Synthesis for Healthcare, Chiang Mai University, Chiang Mai, Thailand

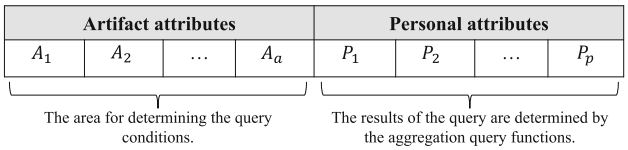

Fig. 1 Artifact rating and personal information in recommendation databases

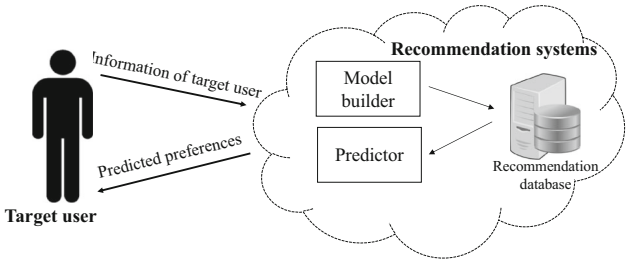

Fig. 2 A recommendation system framework [5]

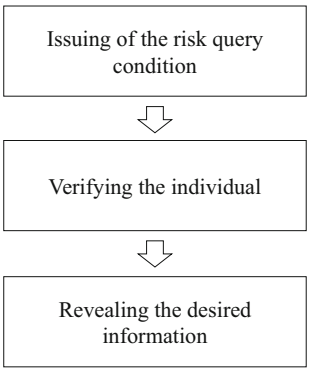

Fig. 3 The procedure for revealing the desired information in recommendation databases through the aggregate query functions

used for data analytics in the query-answering styles. In [7], the authors proposed a framework for such usages, in which it also provides a privacy protecting mechanism additionally. The approach allows queries to the databases for the rating score and also the personal information in "aggregate form" to protect the privacy. An example of the query to the framework to the data in table is illustrated in Query 1,

Query 1: *"SELECT AVERAGE(Age) FROM Table 1 WHERE Egomaniac = 5"*.

The framework subsequently returns 33 as the average age. Such information can be utilized in other analytics further. As the answer is in aggregate form, it seems that the re-identification, or privacy violation, is not feasible. However, in this paper, we will illustrate the vulnerability of this framework in the following example.

1.1 Motivation

Suppose that an adversary wants to violate the privacy of the **25**-year-old **married** person who lives in **New York** city and whose data shown in Table 1 by revealing the salary of such person. A revealing method as illustrated in Fig. 3 can be explained as follows. The adversary can firstly issue $COUNT$ queries to determine the risk query condition i.e., the arbitrary query condition which has a single tuple as the answer. Consecutively, the adversary's background knowledge and an appropriate aggregate function with such determined condition are utilized to verify the individuals. Finally, the adversary can issue the query on the individuals to reveal desired information.

In the example database in Table 1, the adversary can firstly issue $COUNT$ queries to determine the risk query condition of the target user. Suppose that, after the trial-and-error process, the adversary can eventually determine that the queries as shown in Queries 2 and 3 return a single-tuple result, i.e., Tuple ID 1 and 7, respectively.

Table 1 An example of recommendation databases

Tuple ID	Artifact attributes					Personal attributes				
	Warcross	Joy Ride	Egomaniac	Pachinko	Geekerella	Salary	Age	Marital status	City	Education
1	**4**		**4**	**1**	**3**	**$42,000**	**25**	**Married**	**New York**	Bachelor
2	5		4		3	$45,000	23	Married	Los Angeles	Bachelor
3	4		3		4	$50,000	27	Single	New York	Master
4		5	5	1	2	$35,000	31	Married	San Antonio	Bachelor
5		5	5	1	2	$40,000	35	Single	New York	Doctoral
6		5	4	2	1	$50,000	25	Single	Los Angeles	Master
7	**1**	**5**	**4**		**2**	$45,000	32	Married	Los Angeles	Bachelor
8	2	4	3			$47,000	29	Married	Los Angeles	Doctoral
9	2	4	3			$49,000	24	Single	Columbus	Master
10	2	4	3			$51,000	29	Married	Columbus	Doctoral

Bold values indicate the privacy breach according to the motivation example in Sect. 1.1

Query 2: *"SELECT COUNT(*) FROM Table 1 WHERE Warcross IS NOT NULL AND Egomaniac IS NOT NULL AND Pachinko IS NOT NULL AND Geekerella IS NOT NULL"*.

Query 3: *"SELECT COUNT(*) FROM Table 1 WHERE Warcross IS NOT NULL AND Joy Ride IS NOT NULL AND Egomaniac IS NOT NULL AND Geekerella IS NOT NULL"*.

Then, the adversary can start to verify the age of the target user by $AVERAGE$ function in conjunction with the determined risk query conditions, as shown in Query 4 and 5.

Query 4: *"SELECT AVERAGE(Age) FROM Table 1 WHERE Warcross IS NOT NULL AND Egomaniac IS NOT NULL AND Pachinko IS NOT NULL AND Geekerella IS NOT NULL"*.

Query 5: *"SELECT AVERAGE(Age) FROM Table 1 WHERE Warcross IS NOT NULL AND Joy Ride IS NOT NULL AND Egomaniac IS NOT NULL AND Geekerella IS NOT NULL"*.

The adversary can see that the age from query result of Query 4 is **"25,"** which matches with the adversary's background knowledge of the target user. Meanwhile, the result from Query 5 does not match with the adversary's background knowledge, so we can omit such risk query condition. Consecutively, the verification can be further conducted on the martial status by issuing MAX function queries[1] in conjunction with the remaining risk query conditions as shown in Query 6.

Query 6: *"SELECT MAX(Marital status) FROM Table 1 WHERE Warcross IS NOT NULL AND Egomaniac IS NOT NULL AND Pachinko IS NOT NULL AND Geekerella IS NOT NULL"*.

The query result of Query 6 is **"married"** which also matches with the adversary's background knowledge, and thus, the risk query condition can be further verified against the city attribute as in Query 7.

Query 7: *"SELECT MAX(City) FROM Table 1 WHERE Warcross IS NOT NULL AND Egomaniac IS NOT NULL AND Pachinko IS NOT NULL AND Geekerella IS NOT NULL"*.

It can be seen that the result of Query 7 is **"New York"**, and hence, the adversary can identify the target user with high confidence. Finally, the adversary utilizes MAX function in conjunction with the determined risk query condition to reveal the salary, **"$42,000"**, of the target user by Query 8.

[1] *MAX* or *MIN* functions only return one value as the query result such that *MAX* function that only returns a maximum value from a set of values is satisfied by the query condition, and *MIN* function that only returns a minimum value from a set of values is satisfied by the query condition. They are designed to support the various data domains such as *numeric*, *character*, *unique-identifier*, and *date-time*, as referenced from the website: https://msdn.microsoft.com/en-us/library/ms187751.aspx.

Query 8: *"SELECT AVERAGE(Salary) FROM Table 1 WHERE Warcross IS NOT NULL AND Egomaniac IS NOT NULL AND Pachinko IS NOT NULL AND Geekerella IS NOT NULL"*.

Obviously, although the framework which allows only aggregation queries, the results in such form can still lead to the privacy violation. Thus, in this paper, we propose an algorithm for preserving the privacy of individuals in recommendation databases. Our work is based on an assumption that the existing information must only be utilized through the query framework as in [7], i.e., query conditions can only be specified to the artifact attributes, and the results of the query can be determined only from the personal attributes as the aggregate query functions such as *COUNT*, *SUM*, *AVERAGE*, *MAX*, or *MIN*. To achieve the privacy preservation in the given database, we apply k-anonymity in conjunction with the local generalization hierarchy in the forms of *DGH* or *NDGH*. First, a proof that the problem is NP-Hard is presented. Then, a heuristic algorithm, $k\text{-}Likeness$, to preserve the privacy of the given recommendation database with the goal of maximize the data utility is proposed. The idea of algorithm is based on forming the k-size tuples, which are similar to each other, i.e., the range between artifact attribute values is minimized. Thus, the generalization can be performed with less impact on data utility. Our proposed algorithm is to be evaluated by experiments both in terms of effectiveness and efficiency, i.e., the data utility metrics and actual query results, and the execution time, respectively.

The organization of this paper is as follows. The literature related to this work is presented in the next section. In Section 3, a privacy preservation model for recommendation databases is proposed, and it consists of the basic concepts and the problem definition including an NP-hardness proof. Our algorithm for preserving the privacy is presented in Sect. 4. Subsequently, in Sect. 5, the experimental results of our proposed work in various circumstances are reported. Finally, Sect. 6 gives the conclusion and the outlook to our future work.

2 Background and related work

2.1 *k*-Anonymity

In recent decades, there have been various proposed privacy preservation models that can resolve the mentioned issue in Sect. 1. In such models, the datasets are often the collection of tuples that consist of the identifier attributes, the quasi-identifier attributes (QI), and the sensitive attribute. Firstly, the identifier attribute, e.g., citizen ID or names, is to be removed. QI are the attributes that can still lead to privacy violation, i.e., the adversary can cross-examine the QI attributes with related external datasets according to his/her

background knowledge to reveal the sensitive value of a targeted individual. Therefore, the privacy of individuals must be protected by preventing linkage through the QI attributes.

k-Anonymity [8] is one of the most well-known privacy preservation models. It has been extended and applied widely [9–12]. In brief, to protect the privacy, given a parameter k as the privacy preservation constraint, the at least-k-size partitions have to be determined firstly. Consecutively, the values in QI attributes of each partition are to be transformed into a less specific value in order to make them identical; each distorted partition forms an equivalence class (EC). In general, the relationship between values in QI attributes and their less specific values can be represented by a tree structure, e.g., domain generalization hierarchy (DGH) [13], or a directed graph structure, natural domain generalization hierarchy ($NDGH$) [14]. The distortion techniques based on such structures can be referred as generalization. Each EC with identical QI attributes can provide the protection of privacy since there are always at least $k-1$ tuples which are indistinguishable from the targeted individuals.

To achieve the privacy preservation constraint that is based on k-anonymity, there are two well-known generalization approaches as follows. The first approach, global generalization, does not allow the values in a specific QI attribute to be mapped to the different level of the domain structure [15,16]. Meanwhile, the second approach, local generalization, allows the values in a specific QI attribute to be mapped to the different levels [12,17]. In [18], the authors report that the local generalization often leads to the higher data utility than the global generalization.

Typically, the generalized datasets, which satisfy the k-anonymity constraint, can lose some "data utility," the ability to be analyzed and utilized by other data processing. Hence, another goal for the generalization is to preserve the data utility or minimize the impact to the datasets as much as possible. There are a few well-known metrics to measure data utility as follows.

Discernibility metric (DM) [10] is proposed to quantify the data utility of the given dataset by considering the number of ECs and the number of suppressed tuples, the tuples which are removed from the given dataset in order to preserve the privacy. To obtain more data utility, the output datasets should be distorted in order to have smaller size of ECs and less number of suppressed tuples. Another metric which is also based on the size of ECs is Average Equivalence Class Size Metric (C_{AVG}) [12]. With this metric, not only the size of the EC is considered, but also the k values which affect highly the data utility are also considered.

Generalized information loss for numeric ($GenILoss$) is one of well-known metrics that is widely applied [14,17,19]. It is proposed to measure the data utility of the generalized dataset for only numeric QI attribute domain. This metric assigns a penalty to each distorted value based on the range between the upper and lower bounds of the generalized values, e.g., generalizing age of 15 years into 15–18 affects less than generalizing it into 15–20. The penalty cost of $GenILoss$ for the whole dataset is the summation of penalty of each distorted value.

Aside from the traditional k-anonymity model, there are several well-known privacy preservation models which extend from k-anonymity in which they can also be applied to preserve the privacy of individuals in our focused problem, for example Mondrian multidimensional k-anonymity [12], (k, e)-Anonymous [20], or t-Closeness [21]. In this paper, we review a few works that are related to our focused problems as follows.

First, there is an extension to the k-anonymity model by adding the "confidence bounding" of re-identifying the specific sensitive values. In l-diversity model [22], the number of tuples and the distinct number of sensitive values in every EC must be equal to or greater than the value of l. Meanwhile, (α, k)-anonymity [23] incorporates both k-anonymity constraint and such confidence bounding concept. That is, each EC must contain at least k tuples, and the confidence of linkage between QI attributes and each specific sensitive value must be less than or equal to the threshold α for privacy data protection. With these two privacy preservation models, the confidence of re-identification of all sensitive values in the generalized datasets is at most $1/l$ for l-diversity and or α for (α, k)-anonymity models. In [21]; however, the authors show that l-diversity often leads to the larger size of ECs in the generalized datasets than k-anonymity. With this reason, the values in the generalized datasets could be excessively generalized; in the other words, they could have less data utility than k-anonymity.

k^m-anonymity [24] is another extension of k-anonymity model that is proposed to address the excessive generalization issue when the given dataset is in higher dimension with regard to the QI attributes. This model represents the maximum background knowledge of adversaries with regard to the target as m value. Therefore, only the values of m-QI attributes are distorted to be identical with other $k-1$ tuples for each case. For example, from the data in Table 1, suppose that the k value is set at 3 and the m value is set at 2. It is assumed that the adversary can know at most 2 preferences of books of the target user, such as the rate of (Warcross, Joy Ride), (Warcross, Egomaniac), (Warcross, Pachinko), or (Pachinko, Geekerella). Therefore, for preserving the privacy, ratings of each 2 books in the dataset are to be distorted to be identical to the other 2 tuples.

2.2 Recommendation systems

In general, recommendation systems are proposed to recommend or suggest suitable artifacts to the target user [25,26]. For computing the suitable artifacts, the systems

utilize the existing information in the given recommendation database in conjunction with recommendation algorithms, e.g., collaborative filtering [27], content-based filtering [28], knowledge-based recommendation [29], hybrid recommendation [30].

In [31,32], several issues in order to develop effective recommendation systems are raised, e.g., cold start problem [33], scalability [34], privacy concerns [5,7,35,36]. Clearly, privacy concern is one of the most important issues.

One of the notable privacy violation incidents related to recommendation systems is when Netflix, which it is one of well-known online DVD delivery services, decided to improve its recommendation system in 2006 [37,38] by releasing 100 million tuples contained movie ratings by 500,000 of its users to data analysts. To preserve the privacy of individuals in the released data, the users' identities of each tuple, e.g., name, or IP-Address, are removed. However, later on, it was reported that the individuals can be re-identified again by linking the ratings between the released data from Netflix and the rating data from Internet Movie Database (IMDb). This incident indicates that although all identifications of users in recommendation databases are removed, the re-identifying is still possible. However, prohibiting the data from being analyzed could mean losing of opportunity to improve and invent products and services for organizations as well as society.

3 Basic concepts and problem definition

In this section, we introduce the basic concepts for our focused problem as well as the problem definition.

3.1 Basic definitions

Definition 1 (*Rating Scores*) Let R be a set of possible non-negative integer rating scores, $R \subseteq \mathbb{N}$. Let r be an arbitrary rating score in R, and it represents a user's preference level toward an artifact such that a higher level means more preference to the target artifact.

Definition 2 (*Recommendation Databases*) Let database T be a collection of finite tuples. T consists of attributes $(A_1, A_2, \ldots, A_{a-1}, A_a, P_1, P_2, \ldots, P_{p-1}, P_p)$ such that $A_1, A_2, \ldots, A_{a-1}, A_a$ are the artifact attributes, and $P_1, P_2, \ldots, P_{p-1}, P_p$ are the personal attributes. Each tuple represents a user's data stored in the database. For each tuple $t_i \in T$, the value of an artifact attribute A_x and personal attribute P_y is denoted as $t_i[A_x]$ and $t_i[P_y]$, respectively. The projection of the artifact attributes and the personal attributes over tuple t_i is denoted as $t_i[A]$ and $t_i[P]$, respectively.

Definition 3 (*Possible Query Conditions*) Let PQC be a set of all possible query conditions which can be issued on the artifact attributes $A_1, A_2, \ldots, A_{a-1}, A_a$.

Definition 4 (*Risk Query Conditions*) Let RQC be a set of privacy violation risk query conditions, where $RQC \subseteq PQC$, such that every query condition in RQC returns a single tuple as the query result.

Definition 5 (*Background Knowledge*) Let user t be the target user of an adversary, and tuple t_t represents such user's data in T. Let BG_t be a set of adversary's background knowledge for user t that can be used to re-identify the target user by the adversary.

Basically, BG_t is a version of $t_t[P]$ which has values in some "known" attributes and some "unknown" attributes, represented by background knowledge and null values, respectively. From the running example in Sect. 1.1, the **25**-year-old **married** person who lives in **New York** city can be represented as $BG_t =< null, 25, Married, New\ York, null >$.

Definition 6 (*Privacy Violation*) The privacy violation to target user t occurs when the adversary can reveal personal attribute value(s) by issuing the risk query condition queries from RQC, in conjunction with the suitable aggregate query functions and adversary's background knowledge BG_t.

Before the k-anonymity is defined, the domain generalization hierarchy and the natural domain generalization of rating scores in recommendation databases are first introduced as follows.

Definition 7 (*Domain generalization hierarchy of rating scores*) Let $f_{DGH}(R_l) : R_l \rightarrow R_{l+1}$ be a generalized function of rating scores R from level l to level $l+1$ such that all values in level l must be more specific than level $l+1$. Note here that the rating scores of a specific level must not be overlapped, for any rating score r_l in level l, $\cap r_l = \emptyset$, and $\cup r_l = R$.

From such function, the domain generalization hierarchy of rating scores R, DGH_R can be defined as a generalization sequence from level 0 to level l, $R_0 \xrightarrow{f_{DGH}(R_0)} R_1 \xrightarrow{f_{DGH}(R_1)} \cdots \xrightarrow{f_{DGH}(R_{l-2})} R_{l-1} \xrightarrow{f_{DGH}(R_{l-1})} R_l$.

In Fig. 4, a few domain generalization hierarchies for $R = [0, 5]$ are presented. The 0-level DGH represents the original rating scores; meanwhile, the DGH level 2 represents the most general values.

Definition 8 (*Natural Domain Generalization Hierarchy of Rating Scores*) *NDGH* is defined by the "subset" relation, which is a partial order relation, over the set of rating scores R. The level of each subset S in the *NDGH* can be defined

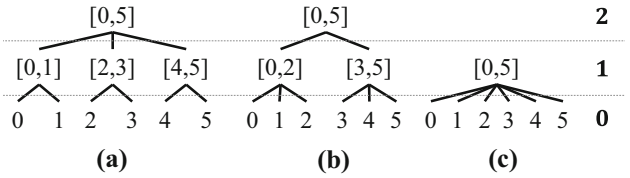

Fig. 4 The domain generalization hierarchies of $R = [0, 5]$

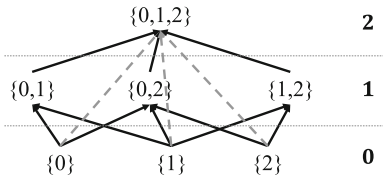

Fig. 5 The natural domain generalization hierarchy of $R = \{0, 1, 2\}$

as its cardinality $|S| - 1$. Thus, a generalization of the given rating score r can be performed by increasing the level in the *NDGH*. Comparing with the *DGH*, the rating scores of a specific level in *NDGH* can be overlapped; however, $\cup r_l = R$, where r_l is a value in level l.

An example of $NDGH_R$, where $R = \{0, 1, 2\}$, is shown in Fig. 5. It has three natural generalization levels. The cardinality of the elements in level 0 is 1; meanwhile, the most generalized value is {0, 1, 2} in level 2. It can be seen that, for a certain R, there could be many possible *DGH*s and *NDGH*s, and the effect of such variation is to be further observed in the experiment section.

Definition 9 (*k-Anonymity*) Let the positive integer k be the privacy preservation constraint. Let T' be the generalized database which is transformed from T by a generalized function $f_1 : T \rightarrow_{k,DGH_R} T'$ or $f_2 : T \rightarrow_{k,NDGH_R} T'$ such that the values in the artifact attributes $A_1, A_2, \ldots, A_{a-1}, A_a$ are generalized by a specific domain generalization, DGH_R or $NDGH_R$, to be indistinguishable by at least k tuples.

From the running example database in Table 1, it can be seen that the quasi-identifiers in this example are the rating scores, and the attributes $(Salary, Age, Marital\ status, City, Education)$ of an individual are the sensitive attributes. Suppose that the k value constraint is set at 3, by using the $DGH_{[0,5]}$ in Fig. 4b, the database can be generalized to Table 2. It can be seen that each EC has at least 3 tuples with identical generalized quasi-identifier values. In case that an $NDGH_{[0,5]}$ is used, an example of the generalized database is shown in Table 3.

As mentioned before, the generalized databases which satisfy the k- anonymity constraint can lose some data utility for other data analytics, and thus, the impacts on the utility, which are applied in this work, are defined as follows.

Definition 10 (*Discernibility Metric (DM)* [10]) Let T' be a generalized database that satisfies the k-anonymity. The DM of T', DM(T') is defined as Eq. 1.

$$DM(T') = \sum_{ec \in EC} |ec|^2 \tag{1}$$

Where

– EC is a set of equivalence classes of T'.

It can be seen that an important factor of the data utility is the size of equivalence classes in the generalized database. That is, the smaller the size of equivalence classes, the higher data utility or the lower the penalty cost of *DM*. In our running example database in Table 2, the *DM* cost of $3^2 + 4^2 + 3^2 = 34$.

Definition 11 (*Average Equivalence Class Size Metric* (C_{AVG}) [12]) Let T' be a generalized database that satisfies the k-anonymity, the C_{AVG} cost of T' is defined as Eq. 2.

$$C_{AVG}(T') = \frac{|T'|}{|EC| \cdot k} \tag{2}$$

Where

– $|EC|$ is the total number of equivalence classes of T'.

This metric also quantifies the impact on the data utility based on the size of equivalence classes and the value of k constraint. For this reason, the k-size-equivalence class is more preferred. In our running example database in Table 2, the C_{AVG} is 1.11, i.e., $\frac{10}{3 \cdot 3}$.

Definition 12 (*Generalized Information Loss for Numeric* (*GenILoss*) [19]) Let T' be a generalized database that satisfies the k-anonymity, and the $GenILoss(T')$ is defined as Eq. 3.

$$GenILoss(T') = \frac{\sum_{i=1}^{|T'|} \sum_{j=1}^{|A|} \frac{MAX(t'_i[A_j]) - MIN(t'_i[A_j])}{MAX(A_j) - MIN(A_j)}}{|T'| \cdot |A|} \tag{3}$$

Where

– A is a set of artifact attributes, the quasi-identifier attributes, of T',
– $MAX(t'_i[A_j])$ is the upper bound, the maximum value, of generalized value in a specific attribute A_j of a tuple t'_i,
– $MIN(t'_i[A_j])$ is the lower bound, the minimum value, of generalized value in a specific attribute A_j of a tuple t'_i,
– $MAX(A_j)$ is the upper bound, the maximum value, in the attribute A_j,

Table 2 An example of generalized recommendation database T' with $k = 3$ using $DGH_{[0,5]}$

Tuple ID	Artifact attributes					Personal attributes				
	Warcross	Joy Ride	Egomaniac	Pachinko	Geekerella	Salary	Age	Marital status	City	Education
1	[3,5]		[3,5]	[0,2]	[3,5]	$42,000	25	Married	New York	Bachelor
2	[3,5]		[3,5]	[0,2]	[3,5]	$45,000	23	Married	Los Angeles	Bachelor
3	[3,5]		[3,5]	[0,2]	[3,5]	$50,000	27	Single	New York	Master
4	[0,2]	5	[3,5]	[0,2]	[0,2]	$35,000	31	Married	San Antonio	Bachelor
5	[0,2]	5	[3,5]	[0,2]	[0,2]	$40,000	35	Single	New York	Doctoral
6	[0,2]	5	[3,5]	[0,2]	[0,2]	$50,000	25	Single	Los Angeles	Master
7	[0,2]	5	[3,5]	[0,2]	[0,2]	$45,000	32	Married	Los Angeles	Bachelor
8	2	4	3			$47,000	29	Married	Los Angeles	Doctoral
9	2	4	3			$49,000	24	Single	Columbus	Master
10	2	4	3			$51,000	29	Married	Columbus	Doctoral

Table 3 An example of generalized recommendation database T' with $k = 3$ using $NDGH_{[0,5]}$

Tuple ID	Artifact attributes					Personal attributes				
	Warcross	Joy ride	Egomaniac	Pachinko	Geekerella	Salary	Age	Marital status	City	Education
1	{4, 5}		{3, 4}	{0, 1}	{3, 4}	$42,000	25	Married	New York	Bachelor
2	{4, 5}		{3, 4}	{0, 1}	{3, 4}	$45,000	23	Married	Los Angeles	Bachelor
3	{4, 5}		{3, 4}	{0, 1}	{3, 4}	$50,000	27	Single	New York	Master
4	{0, 1}	5	{4, 5}	{0, 1, 2}	{1, 2}	$35,000	31	Married	San Antonio	Bachelor
5	{0, 1}	5	{4, 5}	{0, 1, 2}	{1, 2}	$40,000	35	Single	New York	Doctoral
6	{0, 1}	5	{4, 5}	{0, 1, 2}	{1, 2}	$50,000	25	Single	Los Angeles	Master
7	{0, 1}	5	{4, 5}	{0, 1, 2}	{1, 2}	$45,000	32	Married	Los Angeles	Bachelor
8	2	4	3			$47,000	29	Married	Los Angeles	Doctoral
9	2	4	3			$49,000	24	Single	Columbus	Master
10	2	4	3			$51,000	29	Married	Columbus	Doctoral

– $MIN(A_j)$ is the lower bound, the minimum value, the attribute A_j.

It is designed to measure the impact on the data utility when the domain of the quasi-identifier attributes is numeric. The penalty is quantified from the fraction of the domain values that have been generalized. From our running example database in Table 2, its $GenILoss$ is 0.348.

3.2 Problem statement

After the basic concepts, domain generalizations, and the impact on the data utility are introduced, our focused problem can be defined as follows.

Given a recommendation database T, there is a privacy preservation constraint k where $k \in \mathbb{Z}^+$ and $2 \leq k \leq |T|$. The problem is to transform T into T' such that the values in the artifact attributes are generalized by the domain generalization, DGH_R or $NDGH_R$, to be indistinguishable by at least k tuples; meanwhile, the impact on the data utility, DM, C_{AVG}, or $GenILoss$, is minimized.

3.3 NP-hard proof

Before our approach to solving the k-anonymity on recommendation databases is to be proposed, we show that such problem is an NP-Hard problem by the reductions from the Exact Cover by 3-Sets problem ($X3C$) [39]. First, let us consider the $X3C$ problem as follows.

Exact Cover by 3-Sets problem ($X3C$): Let U be a universal set such that the cardinality of U is $3q$, $|U| = 3q$, where $q \in \mathbb{Z}^+$. Let $S = \{s \mid s \subseteq U \wedge |s| = 3\}$ be the set of subset U. For solving the $X3C$ problem, one must determine a subcollection S' of S, $S' \subseteq S$, such that every element u_i, where $u_i \in U$, occurs exactly once in S', $\cup_{s' \in S'} s' = U$ and $\cap_{s' \in S'} s' = \emptyset$.

It can be seen that the $X3C$ problem can be represented by graph notations as follows. Let V be a set of vertices which represents the elements of U. Let E be a set of edges which

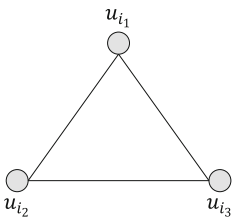

Fig. 6 A completed graph of 3 vertices

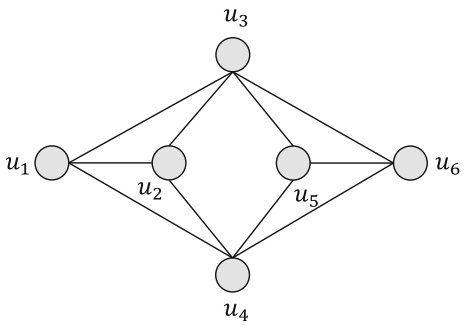

Fig. 7 An example graph of $X3C$ problem

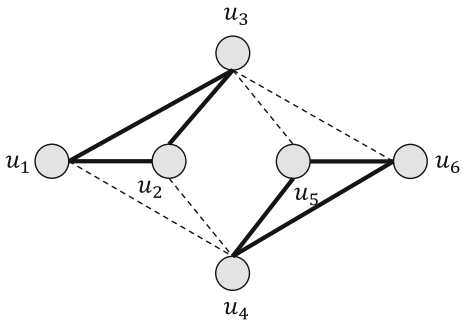

Fig. 8 The graph S'_1

represents the relationship of any paired elements in s of S. So, $G(V, E)$ can be a representative of an $X3C$ graph which is the simple graph, i.e., it is an undirected graph in which both multiple edges and loops are disallowed. Furthermore, let us represent each s of $X3C$ problem by the completed graph with 3 vertices, as shown in Fig. 6. For example, if $U = \{u_1, u_2, u_3, u_4, u_5, u_6\}$ and $S = \{\{u_1, u_2, u_3\}, \{u_1, u_2, u_4\}, \{u_3, u_5, u_6\}, \{u_4, u_5, u_6\}\}$, the $X3C$ graph is constructed from this instance as shown in Fig. 7.

To solve the $X3C$ problem, S' can be constructed from the $X3C$ graph, i.e., it can be constructed from the arbitrary 3-vertex subgraphs which are completed, disjointed, and covered all elements of U. From the input example as shown in Fig. 7, there are two S's which are satisfied by the $X3C$ constraint, i.e., $S'_1 = \{\{u_1, u_2, u_3\}, \{u_4, u_5, u_6\}\}$ and $S'_2 = \{\{u_1, u_2, u_4\}, \{u_3, u_5, u_6\}\}$, as shown in Figs. 8 and 9, respectively.

We then demonstrate that our problem is an NP-Hard problem which is based on the reductions from the $X3C$ problem. To improve the readability of our demonstrate, both

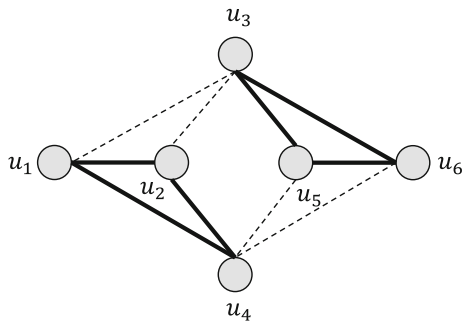

Fig. 9 The graph S'_2

Table 4 The recommendation database T, which is constructed from the graph in Fig. 7

	u_1	u_2	u_3	u_4	u_5	u_6
u_1	1	1	1	1	0	0
u_2	1	1	1	1	0	0
u_3	1	1	1	0	1	1
u_4	1	1	0	1	1	1
u_5	0	0	1	1	1	1
u_6	0	0	1	1	1	1

Table 5 The generalized database T'_1 satisfied by the k-anonymity constraint ($k = 3$), which is constructed from the graph in Fig. 8

	u_1	u_2	u_3	u_4	u_5	u_6
u_1	1	1	1	0	0	0
u_2	1	1	1	0	0	0
u_3	1	1	1	0	0	0
u_4	0	0	0	1	1	1
u_5	0	0	0	1	1	1
u_6	0	0	0	1	1	1

Table 6 The generalized database T'_2 satisfied by the k-anonymity constraint ($k = 3$), which is constructed from the graph in Fig. 9

	u_1	u_2	u_3	u_4	u_5	u_6
u_1	1	1	0	1	0	0
u_2	1	1	0	1	0	0
u_3	0	0	1	0	1	1
u_4	1	1	0	1	0	0
u_5	0	0	1	0	1	1
u_6	0	0	1	0	1	1

databases, T and T', of our problem consider only the artifact attributes. The recommendation database T of our problem is constructed from U and S. Suppose that the size of T is equal $3q \cdot 3q$ and the value of k is set at 3. Also, suppose that an arbitrary edge (u_i, u_j), where $(u_i, u_j) \in E$, associates with the cells $t_i[A_i]$, $t_j[A_j]$, $t_i[A_j]$, and $t_j[A_i]$ of T, such that the value is to be represented by 1, and the others values are to be represented by 0. So, the $X3C$ graph which $q = 2$, as shown in Fig. 7, can be constructed as an instance of the recommendation database as shown in Table 4. Subsequently, generalized database T' can be constructed from

the graph S' of the $X3C$ problem, i.e., database T_1' in Table 5 and T_2' in Table 6 can be constructed from the graph S_1' in Fig. 8 and graph S_2' in Fig. 9, respectively.

Theorem 1 *The privacy preservation for recommendation databases based on the k-anonymity constraint is NP-Hard problem when* $k \geq 3$.

Let us rewrite the $X3C$ problem as a new problem, so-called X_kC problem. Let k be a positive integer, where $k \in \mathbb{Z}^+$ and $k \geq 3$. Let $U = \{u_1, u_2, \ldots, u_{k \cdot q}\}$ be the universal set whose cardinality is equal to $k \cdot q$. $S = \{s \mid s \subseteq U \wedge |s| = k\}$ is a set of subset U. For solving X_kC, one must determine a subcollection S' of S, $S' \subseteq S$, such that each element u_i of U, where $1 \leq i \leq k \cdot q$, which occurs exactly once in S', i.e., $\cup_{s' \in S'} s' = U$ and $\cap_{s' \in S'} s' = \emptyset$.

Proof The reduction is from X_kC, as the X_kC problem is represented by the simple graph $G(V, E)$, so-called X_kC graph. Let V be a set of vertices which is represented by U in the X_kC problem. Let E be a set of edges which represents the relationships of elements in s of S. Is there S', $S' \subseteq S$, such that each element u_i of U occurs exactly once in S'?

Let each element s of S be a representative of the completed graph of k-vertices. For solving the X_kC problem, we can construct S' from the X_kC graph, i.e., it can be constructed from the arbitrary k-vertices subgraphs which are completed, disjointed, and covered all elements of U.

Also, we can construct the recommendation database T of our problem from the X_kC graph such that the size of T is $(k \cdot q) \cdot (k \cdot q)$. Values of each cell in T can be assigned as follows.

$$t_i[A_j], t_j[A_i], t_i[A_i], \text{ and } t_j[A_j] = \begin{cases} 1, \text{ if } u_i, u_j \in s, \\ 0, \text{ otherwise.} \end{cases} \quad (4)$$

Let the value of k equal the cardinality of s, where $s \in S$. The generalized database T' of our problem is constructed by S' of X_kC problem, i.e., values of each cell in T' can be assigned as follows.

$$t_i'[A_j'], t_j'[A_i'], t_i'[A_i'], \text{ and } t_j'[A_j'] = \begin{cases} 1, \text{ if } u_i', u_j' \in s', \\ 0, \text{ otherwise.} \end{cases} \quad (5)$$

Therefore, the generalized database T' which solves the instance of our problem is equivalent to the S' which solves the instance of $X3C$ problem. □

4 The proposed algorithm

In this section, the algorithm for the focused problem is to be proposed. Firstly, let us elaborate on a few issues to be addressed to solve the problem effectively as follows.

4.1 Design issues

4.1.1 Partitioning technique

As the ECs with at least k tuples have to be formed during the generalization process to satisfy k-anonymity, a choice of partitioning techniques is an important issue. In this paper, we propose to apply ECs forming approach as in [12,17] to partition the databases into ECs. Since the objective of partitioning techniques is that the at least k tuples in the same EC should be more similar to each other than the tuples in the other ECs. This characteristics can decrease the number of values which are to be generalized; hence, high data utility can be achieved. Such partitioning technique composes three processes as follows.

- Assigning the nucleus of the EC. This process is to determine the nucleus of each EC which is an arbitrary tuple $t \in T$.
- Assigning the top $k-1$ likeness tuples. This process is to assign the other $k-1$ tuples for each EC such that they are more similar to the nucleus t than the other tuples in T. The similarity between t and the other $k-1$ tuples in T can be measured by *Distance* metric, as shown in Eq. 6. Here, the more similar means the less distance.

$$Distance(t_t, MST) = \sum_{i=1}^{|MST|} \sum_{j=1}^{|A|} | t_t[A_j] - t_i[A_j] | \quad (6)$$

  Where
  - t is the nucleus of the EC,
  - MST is the set of arbitrary tuples of T,
  - $|A|$ is the number of artifact attributes of T.
- Assigning the EC for the remaining tuples. As there can be some tuples that remained from the above described process, the EC with the least distance to such tuples is to be determined by Eq. 6.

4.1.2 Domain generalization

As described before, there are two well-known domain generalizations: *DGH* [13] and *NDGH* [14], which can be applied in our focused problem, i.e., how to generalize rating score attributes in order to satisfy the privacy preservation constraint. The *DGH* can be easier to be applied as the values will become the connected range of generalized values, e.g., from a value of '5' into a value of [0, 5]. Meanwhile, the *NDGH* can cause a less impact on the data utility. These will be investigated in our experiments in Sect. 5.

4.1.3 Generalization method

The two well-known generalization methods for k-anonymity are local generalization [12,17,23] and global generalization [15,16]. Typically, the local generalization often leads to a higher data utility of the generalized databases [18]. Though, it leads to the higher execution time of the privacy preservation algorithm. So, in this work, we apply the local generalization and investigate whether it takes too much execution time.

4.2 Proposed algorithm

In this section, an algorithm for preserving the privacy for the recommendation databases based on the k-anonymity constraint is presented, so-called k-$Likeness$ (Algorithm 1). In general, the algorithm transforms the given database T by partitioning the tuples into ECs such that each EC has the number of tuples to be at least k in which they are more similar than the other tuples in other ECs. Then, the values in the artifact attributes $A_1, A_2, \ldots, A_{a-1}, A_a$ of each EC are generalized by the specific domain generalization, DGH_R or $NDGH_R$, to be indistinguishable from the others.

The input of the algorithm is an positive integer k as the privacy preservation constraint, a recommendation database T, and a specific domain generalization $DomainGen$, DGH_R or $NDGH_R$. Meanwhile, the output of k-$Likeness$ is the generalized database T' of T such that the tuples of the attributes $A_1, A_2, \ldots, A_{a-1}, A_a$ are generalized by the function $Gen(\{t_1, t_2, \ldots\}, DomainGen)$ to be indistinguishable by at least k tuples.

In the first step, the algorithm evaluates the size of T. If the size of T is less than k, $|T| < k$, then the algorithm returns $failure$ because the database T cannot satisfy the k constraint. If the size of T is in the range of between k and $2k-1$, $k \leq |T| \leq 2k-1$, the tuples of T are generalized by $Gen(T, DomainGen)$ to become one-EC T'.

In the second step, the tuples in T are partitioned into k-size ECs as follows. Firstly, an arbitrary tuple t is assigned to be the nucleus of an EC. Then, the set of $k-1$ most similar tuples of t, MST, is determined by the $LikenessTuple$ algorithm (Algorithm 2). Subsequently, the tuples in $MST \cup t$ are removed from T and generalized by $Gen(MST \cup t, DomainGen)$ to be an EC in T'. The algorithm will then re-evaluate the remaining tuples in T. If the number of tuples in T is still equal or greater than k, the algorithm repeats this step; otherwise, it proceeds to the third step.

Optionally, the algorithm assigns the remaining tuples in T which cannot be assigned to any EC by the second step. For determining a suitable EC, the similarity between each remaining tuple of T and the existing ECs is to be evaluated by $LikenessEC$ algorithm (Algorithm 3). Then, the EC with most similarity, $SelEC$ is removed from T'. Such $SelEC \cup t$ is generalized by $Gen(SelEC \cup t, DomainGen)$ to be a new equivalence class and stored in T'.

Algorithm 1: k-$Likeness(T, k, DomainGen)$

Input : A positive integer k, a recommendation database T, and a specific generalization $DomainGen$ (DGH_R or $NDGH_R$);
Output: A generalized database T' which is satisfied by the k-anonymity constraint;

```
1  T' := ∅;
2  MST := ∅;
3  if |T| < k then
4  |   Return Failure;
5  end
6  else if |T| ≥ k and |T| ≤ 2k − 1 then
7  |   T' := Gen(T, DomainGen);
8  end
9  else
10 |   while |T| ≥ k do
11 |   |   t := a selected arbitrary tuple from T;
12 |   |   MST := LikenessTuple(T − t, t, k);
13 |   |   T := T − {MST ∪ t};
14 |   |   T' := T' ∪ Gen({MST ∪ t}, DomainGen);
15 |   end
16 |   while |T| ≥ 1 and |T| < k do
17 |   |   t := a selected arbitrary tuple from T;
18 |   |   T := T − t;
19 |   |   SelEC := LikenessEC(T', t);
20 |   |   T' := T' − SelEC;
21 |   |   T' := T' ∪ Gen(SelEC ∪ t, DomainGen);
22 |   end
23 end
24 Return T';
```

5 Experiment

After the algorithm, k-Likeness, is proposed, this section presents the experiments results to evaluate its effectiveness and the efficiency by comparing it with two well-known algorithms, i.e., Mondrian Multidimensional k-anonymity [12] and k-Member [17]. The first comparing algorithm selects an attribute one-by-one to partition the ECs. If the size of each EC is still higher than k, another attribute is selected to further partition the databases. Meanwhile, the k-Member algorithm works in different angle, and it selects an arbitrary tuple and then adds the other k-1 tuples which cause least range in such EC.

5.1 Experimental setup

The experiments are conducted on an Intel core i5-6400 2.71 GHz CPU PC with 16 GB DDR4 memory and 512 GB SSD running Microsoft Windows 10 (64 bit edition). All the implementation is built and executed on Microsoft Visual Studio 2015 Community Edition.

Algorithm 2: $LikenessTuple(T, t, k)$

Input : A recommendation database T, a tuple t of T, and a positive integer k;
Output: A set of $k-1$ tuples such that $Distance(t, \{t_{i_1}, \ldots, t_{i_{k-1}}\})$ is minimal;

Let $MST := \{\{t_i, Sim\} \mid t_i \in T, Sim = Distance(t, t_i)\}$ be a set of target tuples, where $|MST| = k-1$;
Let $MDT = \{\{t_x, Sim\} \mid \{t_x, Sim\} \in MST\}$, such that $Sim = Distance(t, t_x)$ is maximal;
$MST := \emptyset$;
$MDT := \emptyset$;
$SimScore := 1$;
for $i := 1$ *to* $|T|$ **do**
  $SimScore := Distance(t, t_i)$;
  if $MST = \emptyset$ *or* $|MST| < k$ **then**
    if $MDT = \emptyset$ **then**
      $MDT := \{t_i, SimScore\}$;
    end
    else
      if $SimScore > Sim$, *where* $Sim \in MDT$ **then**
        $MDT := \{t_i, SimScore\}$;
      end
    end
    $MST := MST \cup \{\{t_i, SimScore\}\}$;
  end
  else
    if $SimScore < Sim$, *where* $Sim \in MDT$ **then**
      $MST := MST - MDT$;
      $MST := MST \cup \{t_i, SimScore\}$;
      $MDT := \{t_i, Sim\}$, such that $MDT \in MST$ and Sim is maximal;
    end
  end
end
Return MST;

Algorithm 3: $LikenessEC(T', t)$

Input : A generalized database T' and a tuple t of T;
Output: $EC_{target} \in T'$ such that $Distance(t, EC_{target})$ is minimal;

Let ECs be a set of the ECs which are available in T';
Let EC_{target} be an EC;
Let Sim be the highest similarity score between the tuple t and an EC;
$Sim := INF$;
$EC_{target} := \emptyset$;
for $g := 1$ *to* $|ECs|$ **do**
  $SimScore_g := Distance(t, EC_g)$, where $EC_g \in ECs$;
  if $SimScore_g < Sim$ **then**
    $Sim := SimScore_g$;
    $EC_{target} := EC_g$;
  end
end
Return EC_{target};

Our work is evaluated on *"ml-10m"* real-life dataset which is proposed by the "$GroupLens''$ recommendation system [40]. The dataset has 100,000 rating scores in the range of [0, 5] by 943 users on 1,682 movies. Each movie is always rated by at least one user, and each user always rates at least 20 movies. In order to conduct the experiments effectively, only top ten movies with regard to the highest number of ratings are selected, though the trends in both effectiveness and efficiency can still be seen. Thus, the dataset contains 4,863 ratings by 927 users on 10 movies. The profile for each user is *"(User ID, Age, Gender, Occupation, Zip code)"*.

5.2 Experimental results and discussion

5.2.1 Effectiveness based on data utility

In this section, we first report the impact on the data utility in the generalized databases by k-likeness, Mondrian Multidimensional k-anonymity, and k-member algorithm. We also evaluate the effect of the domain generalizations, *DGH* and *NDGH*. As the rating scores are in [0, 5], the three *DGH* as shown in Fig. 4 are applied to evaluate the effect of its height and range. In these experiments, the values of k are varied from 1 to 20. Note here that, in practice, the k value is usually set at most 10 [41]. However, the range of the settings can show the effectiveness of our algorithm even in a very utmost case.

In Fig. 10, the impact quantified by *DM* is reported. From the results, first, it can be observed that the values of *DM* from the generalized databases are increased when the k value is increased. It is straightforward that the higher k value means the higher generalization and hence it means more higher impact. However, the generalized databases by Mondrian Multidimensional k-anonymity algorithm have the higher impact with regard to the *DM* than the other algorithms. Also, we can see that our proposed algorithm, k-likeness, with *NDGH* domain generalization has the lowest impact based on *DM*; such impact is comparable to the k-member algorithm. Meanwhile, the impact by the k-likeness algorithm with *DGH*, where its height of the hierarchy is less ($DGH(c)$), has a higher impact based on *DM*. Last, for the *DGH* whose range is larger ($DGH(b)$), the data utility is higher.

Secondly, the impact on the data utility quantified by C_{AVG} is reported in Fig. 11. From the results, obviously, the impact based on C_{AVG} is rather stable when the k value is increased. This is because all algorithms generate the higher number of ECs according to the increasing k values. Also, from the result, it can be seen that the impact of the databases generated from the k-Likeness and k-Member algorithms are very close compared with the output from Mondrian Multidimensional k-anonymity algorithm, which has rather high impact. Furthermore, we observe that the impact of both k-likeness, particularly with *NDGH* and $DGH(b)$, and k-member algorithms are close to the optimal point of C_{AVG} at "1." That is because the size of the almost all of the ECs constructed by both algorithms is equal to k.

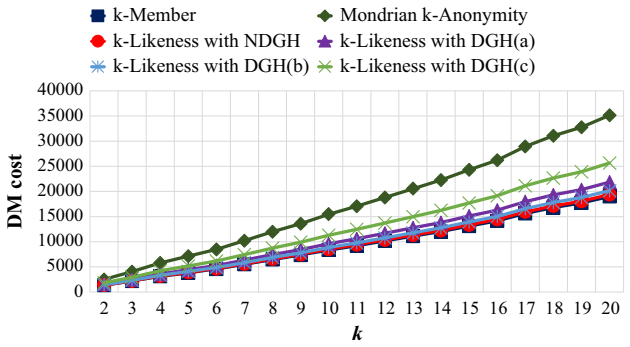

Fig. 10 Data utility based on discernibility metric (*DM*)

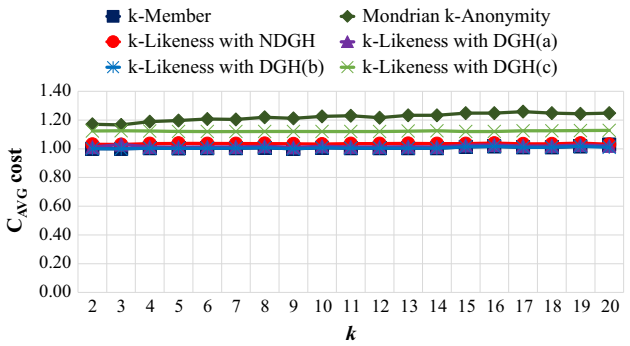

Fig. 11 Data utility based on average equivalence class size metric (C_{AVG})

The last experiment in this section is to present the impact based on the $GenILoss$ metric. Such metric quantifies the impact based on the range, i.e., maximal and minimal, of the generalized values in generalized databases. From the algorithmic point of view, such metric prefers smaller range in the generalized values. Such nature can suit with the k-member algorithm which generalizes the given databases based on the lower bound and the upper bound. The result is shown in Fig. 12. Obviously, when the k value is increased, the impact on the data utility from all algorithms is also increased. The rationale is that the higher k value leads to the higher range of the generalized values, i.e., higher level of *DGH* and larger size of values in *NDGH*. It can be seen that the result of the k-member algorithm has less impact than the other algorithms, followed by our k-likeness algorithm with *NDGH* domain generalization. At k value that is set at 2 to 7, the k-likeness algorithm can generalize the given databases with the same data utility of k-Member algorithm.

To summarize here, the results of the k-Likeness, our proposed algorithm, with *NDGH* generalization domain and the k-member algorithm are very effective when measured by the three metrics. For the *DGH* generalization domain, the less height and narrow range hierarchy can cause a higher impact on the data utility.

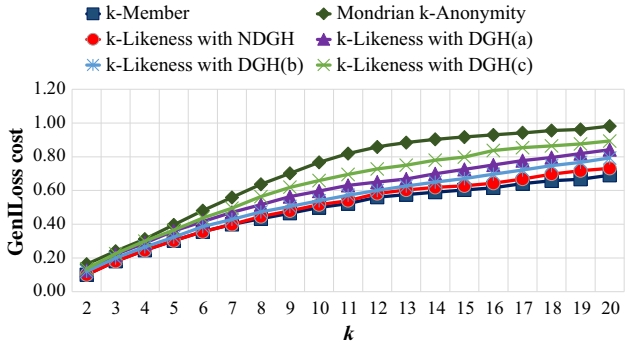

Fig. 12 Data utility based on $GenILoss$ metric

5.2.2 Effectiveness based on query results

Aside from the utility metrics, in this section, we report the effectiveness based on the errors of the query results. Firstly, each given recommendation database is generalized to satisfy the privacy preservation constraint by each comparing algorithms, in which the value of k is fixed at 2. Then, an input non-generalized recommendation database and its generalized databases are queried by the $AVERAGE$ aggregate query function over the Age attribute in conjunction with the OR, AND query operations, and the range of queries over the rating scores. The query conditions in these experiments are the combination from all possible conditions of the given recommendation database which does not return query result as *NULL* values. Finally, the error of query results is evaluated by the relative error [20], as shown in Eq. 7. Note here that the rest of the experiment section will show only the results of our proposed algorithm when the *NDGH* and the $DGH(b)$ domain generalization is applied, because of their higher performance.

$$\Delta_x = \frac{|x - x_0|}{x} \cdot 100 \tag{7}$$

Where

- x_0 is the actual value which is queried from the input recommendation database,
- x is the inferred value which is queried from the generalized database.

In Fig. 13, we report the errors of the query results with the effect by the OR operation over the rating scores. From the experimental results, we observe that the number of query condition attributes inversely influences the error of query results, i.e., the more attributes mean the less error of the query results. This is because the higher number of attributes means that the algorithms can have higher options to generalize the databases, and thus, the less errors can be obtained. In addition, it can be seen that our proposed one with *NDGH*

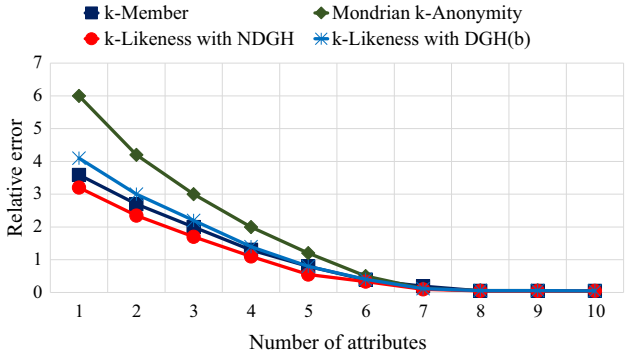

Fig. 13 Query errors with the effect of OR query operation

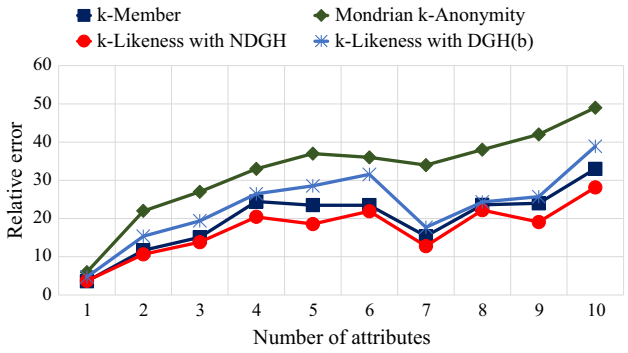

Fig. 14 Query errors with the effect of AND query operation

domain generalization outperforms the k-member algorithm, especially when the range of the attribute numbers is between 1 and 5. This is because our algorithm tends to produce smaller equivalence classes and hence less generalized the values. It can be seen that the relative errors by our proposed algorithm is very small, i.e., less than 4%.

In Fig. 14, the effect of query results from the AND query operation is reported. Obviously, when the number of involved attributes is increased, the relative errors are also increased. This is because of the limitation of the generalization options that the algorithms have. It is observed that when the number of attributes is very high, the values in the given input are very different. Thus, the generalization process has to move to the higher level in the hierarchy domain. However, our proposed k-likeness algorithm with *NDGH* domain generalization still outperforms the others.

The last experiment result in this section is shown in Fig. 15, where the effect of query ranges in the queried rating scores to the errors is reported. Note that when the range is set at 0, it means that the exact match on the rating score is applied. The results are similar to the result in Fig. 13 with the same reason, i.e., the larger range of query means the higher number of the options that the algorithm can obtain. Thus, it means the less error query results can be generated. Also, it can be observed that the results from the k-likeness with *NDGH* domain generalization have less error than other

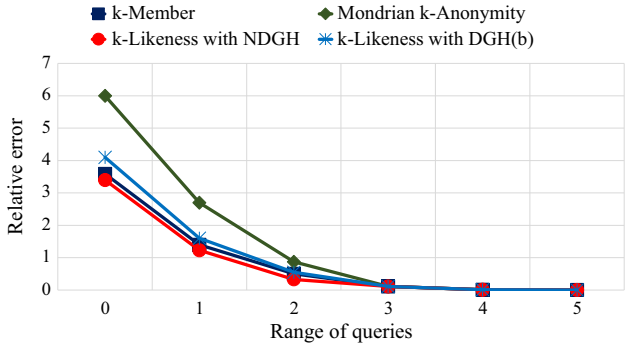

Fig. 15 Query errors with the effect of the query ranges

algorithms. When the range of rating score in the queries is set at greater than 1, the relative errors by the proposed algorithm are only less than 1.0%, which is very effective.

5.2.3 Efficiency

After the effectiveness of our proposed k-likeness algorithm is presented, in this section, the efficiency of our proposed algorithm with *NDGH* and $DGH(b)$ is to be evaluated based on the execution time of transforming the given recommendation databases to satisfy the privacy preservation constraint. The k values is varied from 1 to 20. From the experimental results as shown in Fig. 16, when the k value is increased, the execution time of both k-likeness and the Mondrian Multidimensional k-anonymity algorithms is decreased. This is because the higher k leads to the larger size of the ECs, and thus, the number of iteration in the outer loop of both algorithms to determine the suitable ECs for a tuple tends to be decreased. Unlike the k-member algorithm, its execution time is increased, since it determines the EC for one tuple at a time. From the result, it is obvious that the Mondrian Multidimensional k-anonymity algorithm is most efficient. However, its less-effectiveness that has been reported in the previous section can prevent it from being applied practically. For our proposed algorithm particularly with *NDGH* domain generalization, it can be seen that the execution is much less than the k-Member algorithm. To be specific on our proposed algorithm, the main portion of the execution time is from the $likenessEC$ module (69.23% of its execution time on average), i.e., determining an EC for the remaining tuples in which the algorithm has to scan all the ECs to find the highest score of similarity. The larger size of the remaining tuples means the higher execution time from this module; this is also a reason for the inconsistency of the execution time.

From the results from this section and the previous section, it can be concluded that our proposed k-likeness algorithm is both highly effective and efficient. That is, its effectiveness is comparable, if not *better*, to the k-Member algorithm

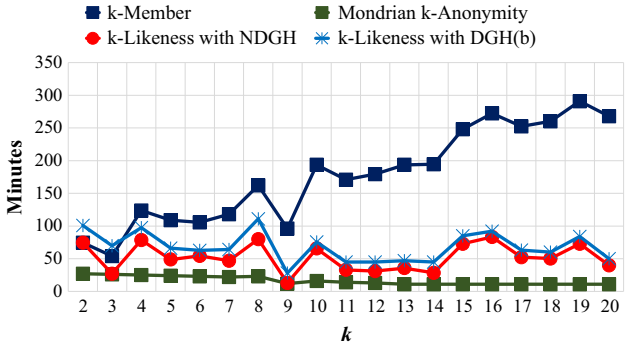

Fig. 16 Execution time

and much higher than the Mondrian Multidimensional k-anonymity algorithm as shown by both the data utility metrics and the error in query results. Also, its efficiency is also high and is in the same trends as the Mondrian Multidimensional k-anonymity algorithm.

6 Conclusion and future work

In this paper, we have addressed a problem of privacy preservation for recommendation services. The problem is based on the assumption that the databases must be able to utilize by other services, i.e., they can be queried against the artifact attributes, and the results of the query can be determined from the personal attributes as the aggregation such as $COUNT$, SUM, $AVERAGE$, MAX, and MIN as proposed in [7]. The problem has proven as NP-hard by the reductions from the X_3C problem. Subsequently, a local generalization algorithm is proposed. Its main approach is to determine the k-tuples ECs which are similar to each other in order to perform generalization as less as possible, i.e., maximize the data utility as much as possible. The proposed work has been evaluated by experiments in terms of both effectiveness and efficiency. The results have indicated that our proposed work is both highly effective and efficient. Also, it is found that if the $NDGH$ is acceptable for the query results, it should be applied since it can provide a higher data utility.

In our future work, we will focus on the data update issue since the movies, users, and rating can be appended all the time. If the generalized databases have to be computed for each update, the efficiency of the system can be degraded.

References

1. Shvarts M, Lobur M, Stekh Y (2017) Some trends in modern recommender systems. In: Proceedings of the 2017 XIIIth international conference on perspective technologies and methods in MEMS design. IEEE
2. The Statistics Portal (2016) Number of apps available in leading app stores as of June 2016. https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/. Retrieved 12 June 2016
3. Yamato Y (2017) Performance-aware server architecture recommendation and automatic performance verification technology on iaas cloud. Serv Oriented Comput Appl 11:121–135
4. Chan NN, Tata WG (2012) A recommender system based on historical usage data for web service discovery. Serv Oriented Comput Appl 6:51–63
5. Lam SKT, Frankowski D, Riedl J (2006) Do you trust your recommendations? An exploration of security and privacy issues in recommender systems. In: Proceedings of the 2006 international conference on emerging trends in information and communication security. ETRICS'06, Springer, pp 14–29
6. Beel J, Gipp B, Langer S, Breitinger C (2016) Research-paper recommender systems: a literature survey. Int J Digit Libr 17:305–338
7. Ramakrishnan N, Keller BJ, Mirza BJ, Grama AY, Karypis G (2001) Privacy risks in recommender systems. IEEE Internet Comput 5:54–62
8. Sweeney L (2002) K-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl Based Syst 10:557–570
9. Aggarwal CC (2005) On k-anonymity and the curse of dimensionality. In: Proceedings of the 31st international conference on very large data bases. VLDB '05, VLDB endowment, pp 901–909
10. Bayardo RJ, Agrawal R (2005) Data privacy through optimal k-anonymization. In: 21st International conference on data engineering (ICDE'05), pp 217–228
11. Fung BCM, Wang K, Wang L, Hung PCK (2009) Privacy-preserving data publishing for cluster analysis. Data Knowl Eng 68:552–575
12. LeFevre K, DeWitt DJ, Ramakrishnan R (2006) Mondrian multidimensional k-anonymity. In: 22nd International conference on data engineering (ICDE'06), pp 25–25
13. Sweeney L (2002) Achieving k-anonymity privacy protection using generalization and suppression. Int J Uncertain Fuzziness Knowl Based Syst 10:571–588
14. Nergiz ME, Clifton C (2007) Thoughts on k-anonymization. Data Knowl Eng 63:622–645
15. Fung BCM, Wang K, Yu PS (2005) Top-down specialization for information and privacy preservation. In: 21st international conference on data engineering (ICDE'05), pp 205–216
16. LeFevre K, DeWitt DJ, Ramakrishnan R (2005) Incognito: efficient full-domain k-anonymity. In: Proceedings of the 2005 ACM SIGMOD international conference on management of data. SIGMOD '05. ACM, pp 49–60
17. Byun JW, Kamra A, Bertino E, Li N (2007) Efficient k-anonymization using clustering techniques. In: Proceedings of the 12th International conference on database systems for advanced applications. DASFAA'07. Springer, Berlin, pp 188–200
18. Xu J, Wang W, Pei J, Wang X, Shi B, Fu AWC (2006) Utility-based anonymization using local recoding. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '06. ACM, New York, NY, pp 785–790
19. Iyengar VS (2002) Transforming data to satisfy privacy constraints. In: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining. KDD '02. ACM, pp 279–288
20. Zhang Q, Koudas N, Srivastava D, Yu T (2007) Aggregate query answering on anonymized tables. In: 2007 IEEE 23rd international conference on data engineering, pp 116–125
21. Li N, Li T, Venkatasubramanian S (2007) t-closeness: privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd international conference on data engineering, pp 106–115

22. Machanavajjhala A, Kifer D, Gehrke J, Venkitasubramaniam M (2007) *L*-diversity: privacy beyond *k*-anonymity. ACM Trans Knowl Discov Data. https://doi.org/10.1145/1217299.1217302
23. Wong RCW, Li J, Fu AWC, Wang K (2006) (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '06. ACM, New York, NY, pp 754–759
24. Terrovitis M, Mamoulis N, Kalnis P (2008) Privacy-preserving anonymization of set-valued data. Proc VLDB Endow 1:115–125
25. Verbert K, Manouselis N, Ochoa X, Wolpers M, Drachsler H, Bosnic I, Duval E (2012) Context-aware recommender systems for learning: a survey and future challenges. IEEE Trans Learn Technol 5:318–335
26. Beel J, Langer S, Genzmehr M, Gipp B, Breitinger C, Nürnberger A (2013) Research paper recommender system evaluation: a quantitative literature survey. In: Proceedings of the international workshop on reproducibility and replication in recommender systems evaluation. RepSys '13. ACM, pp 15–22
27. Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval. SIGIR '99. ACM, pp 230–237
28. Hijikata Y, Iwahama K, Nishida S (2006) Content-based music filtering system with editable user profile. In: Proceedings of the 2006 ACM symposium on applied computing, SAC '06. ACM, pp 1050–1057
29. Carrer-Neto W, Hernández-Alcaraz ML, Valencia-García R, García-Sánchez F (2012) Social knowledge-based recommender system. Application to the movies domain. Expert Syst Appl 39:10990–11000
30. Burke R (2002) Hybrid recommender systems: survey and experiments. User Model User-Adapt Interact 12:331–370
31. Isinkaye F, Folajimi Y, Ojokoh B (2015) Recommendation systems: principles, methods and evaluation. Egypt Inform J 16:261–273
32. Khusro S, Ali Z, Ullah I (2016) Recommender systems: issues, challenges, and research opportunities. In: Kim KJ, Joukov N (eds) Information science and applications (ICISA) 2016. Springer, Singapore, pp 1179–1189
33. Lam XN, Vu T, Le TD, Duong AD (2008) Addressing cold-start problem in recommendation systems. In: Proceedings of the 2nd international conference on ubiquitous information management and communication. ICUIMC '08. ACM, pp 208–211
34. Sarwar BM, Karypis G, Konstan J, Riedl J (2002) Recommender systems for large-scale e-commerce: scalable neighborhood formation using clustering. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.4.6985
35. Tsai WT, Xiao Wei YCRPJYC, Zhang D (2007) Data provenance in SOA: security, reliability, and integrity. Service Oriented Comput Appl 1:223–247
36. Calandrino JA, Kilzer A, Narayanan A, Felten EW, Shmatikov V (2011) "You might also like: " privacy risks of collaborative filtering. In: Proceedings of the 2011 IEEE symposium on security and privacy. SP '11, IEEE Computer Society, pp 231–246
37. Fung BCM, Wang K, Chen R, Yu PS (2010) Privacy-preserving data publishing: a survey of recent developments. ACM Comput Surv 42:14:1–14:53
38. di Vimercati SDC, Foresti S, Livraga G, Samarati P (2012) Data privacy: definitions and techniques. Int J Uncertain Fuzziness Knowl Based Syst 20(06):793–817
39. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-Completeness. W. H. Freeman & Co., New York
40. Harper FM, Konstan JA (2015) The movielens datasets: history and context. ACM Trans Interact Intell Syst 5:19:1–19:19
41. Office for Government Policy Coordination, RoK (2016) Guidelines for de-identification of personal data - guide for de-identification standards and support/management system. https://www.privacy.go.kr/cmm/fms/FileDown.do?atchFileId=FILE_000000000830764&fileSn=0