IT 11 023
Examensarbete 30 hp
Maj 2011

Privacy Preserving Data Publishing for Recommender System

Xiaoqiang Chen

Institutionen för informationsteknologi
Department of Information Technology

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Privacy Preserving Data Publishing for Recommender System

Xiaoqiang Chen

Driven by mutual benefits, exchange and publication of data among various parties is an inevitable trend. However, released data often contains sensitive information thus direct publication violates individual privacy. This undertaking is in the scope of privacy preserving data publishing (PPDP). Among many privacy models, K-anonymity framework is popular and well-studied, it protects data by constructing groups of anonymous records such that each record in the table released is covered by no fewer than k-1 other records. This thesis investigates different privacy models and focus on achieving k-anonymity for large scale and sparse databases, especially recommender systems. We present a general process for anonymization of large scale database. A preprocessing phase strategically extracts preference matrix from original data by Singular Value Decomposition (SVD) eliminates the high dimensionality and sparsity problem. A new clustering based k-anonymity heuristic named Bisecting K-Gather (BKG) is invented and proved to be efficient and accurate. To support customized user privacy assignments, we also proposed a new concept called customized k-anonymity along with a corresponding algorithm. Experiments on MovieLens database are assessed and also presented. The results show we can release anonymized data with low compromising privacy.

Key Words: Privacy Preserving Data Publishing, K-Anonymity, Customized K-Anonymity, Bisecting K-Gather, Singular Value Decomposition.

Handledare: Vincent Huang
Ämnesgranskare: Björn Victor
Examinator: Anders Jansson
IT 11 023
Sponsor: Jonas Björk, Simon Moritz

Tryckt av: Reprocentralen ITC

Preface

This Master's Thesis was carried out at Ericsson Research in Kista, Sweden from September of 2010 to March of 2011. The thesis is the final work for a Master's Degree in Computer Science at Uppsala University. The industrial supervisor at Ericsson was Vincent Huang and academic supervisor at Uppsala University was Björn Victor.

Acknowledgement

It has been a pleasure that doing this thesis in Ericsson Research for six mouths. First of all, I would like to thank my supervisor Vincent Huang and reviewer Prof. Björn Victor, always giving me advices, helps and good ideas. Also thanks to Jonas Björk, Simon Moritz who help me during the study. Without them, I could not finish the thesis alone. Finally, I would like to thank staff of Ericsson Research for their kindness and patience.

Contents

1 Background 9

1.1 Privacy Preserving Data Publishing 9

1.2 Recommender Systems 10

1.3 Motivation and Purpose 11

1.4 Organization of the Thesis 12

2 Privacy Models Investigation 13

2.1 Taxonomy 13

2.2 Perturbation Based Approach 14

2.2.1 Randomization 14

2.2.2 Rotation Perturbation 16

2.2.3 Geometric Perturbation 17

2.3 *K*-Anonymity 17

2.3.1 Generalization/Suppression Based *K*-Anonymity 18

2.3.2 Clustering Based $\boldsymbol{K}$-Anonymity ($\boldsymbol{K}$-Gather) 20

2.3.3 Challenges 22

2.3.4 L-Diversity and T-Closeness 24

3 General Solution 26

3.1 Preprocess (User Preference Matrix Extraction) 26

3.2 $\boldsymbol{K}$-Gather Clustering (Bisecting $\boldsymbol{K}$-Gather) 28

3.2.1 Algorithm Description 28

3.2.2 Time Complexity Analysis 30

3.3 Feature Extraction 32

4 Evaluation of the General Process 33

4.1 Data Set 33

4.2 Evaluation of BKG 33
4.2.1 Measure of Clustering Quality 33
4.2.2 Evaluation Method 33
4.2.3 Results 34
4.3 Evaluation of Utility Loss 35
4.3.1 Measure of Prediction Accuracy 35
4.3.2 Evaluation Method 36
4.3.3 Results 37
5 Customized K-Anonymity 38
5.1 Concept and motivation 38
5.2 User policy 39
5.3 Bisecting One-K-Gather 39
6 Experimental Results for Customized K-Anonymity 45
6.1 Distribution Analysis 45
6.2 Utility Loss Evaluation 46
7 Conclusion 47
8 Future Work 48
8.1 Larger Database 48
8.2 Approximation Algorithm 48
8.3 Non-numerical Data 48
8.4 General Customized K-Anonymity 49
Bibliography 50
Table of Figures 54
Abbreviations 55

1 Background

This chapter first introduces the background of the thesis, including concept of privacy preserving data publishing and recommender system, after that, a brief definition of thesis motivation and purpose will be provided. At last, a short overview of the rest chapters is given which provides an outline to readers.

1.1 Privacy Preserving Data Publishing

The collection of information by governments and corporations has created massive opportunities for knowledge-based decision making. Driven by mutual benefits, or by regulations that require certain data to be published, there is a demand for the exchange and publication of data among various parties. For example, Netflix, the world's largest online DVD rental service, recently published a data set contains 100M ratings of 17K movies by 500K users, announced the $1-million Netflix Prize in a drive to improve the quality of movie recommendations based on user preferences (Hafner, 2006).

Releasing data to the public or other parties for research is an inevitable trend and has substantial benefits to the company and the society. However, such activities have been strongly opposed by their users since the released data often contain their sensitive information and by publishing data directly, will violate users' privacy. Hence users argue that their safety of integrity would be intruded and the privacy issue has been raised with increasing importance today. This undertaking is in the scope of privacy preserving data publishing (PPDP) (Fung, et al., June, 2010).

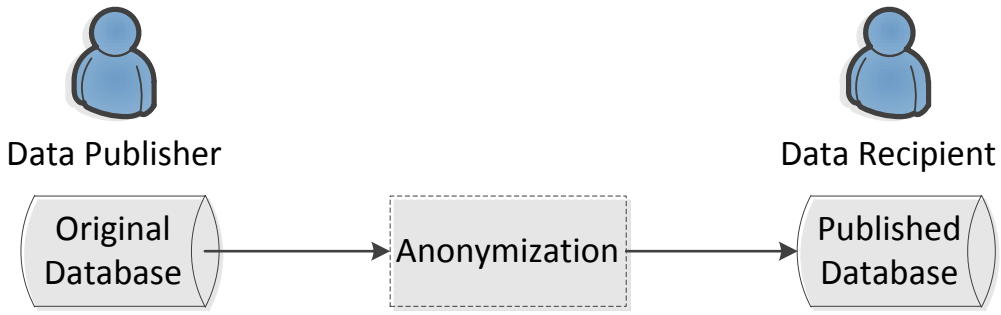

Figure 1.1 A Simple Model of PPDP

A typical PPDP scenario is described in the above figure. Assume there is a centralized trusted server, called data publisher, who has a collection of data from users and wants to release the collected data to a data miner or to the public for research or other purposes. A task of the utmost importance here for the data publisher is to anonymize data before it being published such that the data recipient cannot learn the privacy information about users while still get meaningful data and perform data mining activities in a decent accuracy.

One trivial anonymization method is that before dataset to be released, user names and IDs are replaced with random numbers or simply removed. However, this kind of trivial anonymization is not good enough to protect users' privacy. Private or sensitive user information can still be mined from the remaining user data, so called re-identification (Samarati, 2001).

For example, Netflix disclosed what it considered was anonymized user data to those trying to come up with solutions. This, however, led to a lawsuit by a mother who argued that Netflix had not sufficiently anonymized the information and that she (among others) could be easily outed according to her own rental history. Indeed, within weeks of the data being released, researchers like Narayanan and Shmatikov had found a way to use an external data source (e.g. IMDb) to decode an individual's viewing history with surprising accuracy (Narayanan, et al., 2007). The result has shown that with 8 movies ratings (of which 2 may be completely wrong) and dates that may have a 14-day error, 99% of subscribers can be uniquely identified in the dataset.

1.2 Recommender Systems

This thesis mainly studies the privacy preserving data publishing for recommender systems (e.g. Netflix Prize dataset). To help consumers make intelligent buying decisions, lots of websites provide so called recommender systems. Recommender systems form or work from a specific type of information filtering system technique that attempts to recommend information items (films, television, video on demand, music etc.) that are likely to be of interest to the user.

Typically, recommendations are usually based on user ratings and logs. These data may contain sensitive user information such as buying history and movie-rating records. One of the most commonly used recommendation technique is called "collaborative filtering", by which predicts a subscriber's future choices from his past behavior using the knowledge of what similar consumers did. I.e. try to make prediction of values of some items using a combination of other attributes.

A typical recommender system consists of the following components:

- A set of users $U = \{u_1, \dots, u_m\}$,
- A set of items $O = \{o_1, \dots, o_n\}$,
- A set of possible rating values V (e.g. $V = \{1,2,3,4,5\}$),
- A set of user ratings $R = \{(u, o, r)\}$ where $u \in U$, $o \in O$ and $r \in V$, which is the rating value assigned by the user u to an item o (only if u has rated o).

Given a recommender system, the data can be represented as an $m \times n$ matrix A as the following figure shows. Each entry r_{ij} is either a real number $r \in V$, or 0 if user u_i has not rated item o_j.

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	Movie 6	Movie 7	Movie 8	Movie 9	...
User 1	1	0	2	0	0	5	5	0	0	...
User 2	2	0	0	0	0	4	5	0	0	...
User 3	5	4	0	4	0	0	0	0	0	...
User 4	5	4	0	0	0	3	0	0	0	...
User 5	4	5	0	4	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...

Figure 1.2 Matrix Representation of Recommender System

There are three main features for a recommender system.

- **Ordinal data**. The data is always within some numerical scales, for example, from 1 (bad) to 5 (excellent).
- **High dimensionality**. It is always in a very high dimensional space. Because the number of dimensions equals to the number of items. So for example, if we have 10000 items, we will get 10000 dimensions.
- **Sparsity**. Each individual record contains values only for a small fraction of attributes; users may only rate a limited number of movies from the entire set of movies. So real-world recommender system is always very sparse.

1.3 Motivation and Purpose

In this thesis project, we will try to find solutions to the privacy and user profile anonymization problem that specific to recommender system, which do not lead to disclosure of individual information, but preserves the informational content as much as possible.

This thesis consists of two parts, a theoretical part where we will investigate different approaches of user profile anonymization and identify their advantages and disadvantages. In this part of work, a study of the state of art should be performed in which candidate approaches will be analyzed. Meanwhile, after gaining knowledge about existing methods, we are supposed to come up with a better solution or make improvement of available methods.

In the second part, “proof of concept” part, is a practical part where we are required to implement our new algorithm and verify the findings of the theoretical studies with our available data sets.

1.4 Organization of the Thesis

Chapter 2 investigates different privacy models, especially perturbation based approaches and k-anonymity. The pros and cons of each approach will be pointed out. In this thesis, we focused on achieving k-anonymity for recommender system, the challenges will also be listed in the end of the chapter. Chapter 3 presents our general anonymization process which eliminates the problems mentioned in chapter 2 and invented a new algorithm to achieve k-anonymity. Chapter 4 is a series of experiments did on the purpose of analyzing the performance of our general process. In chapter 5, we raise a new concept, that is "customized k-anonymity" along with a candidate algorithm to achieve it. Evaluation towards customized k-anonymity will be provided in chapter 6. Chapter 7 and 8 are conclusion and future work respectively.

As a convention, I will use upper case to represent matrices or databases, bold lower case to represent vectors/records, and lower case to represent values.

2 Privacy Models Investigation

In this chapter, we investigate different privacy models for privacy preserving data publishing as well as a brief discussion on pros and cons of each method. Furthermore, we state theoretical challenges in high dimensionality of data along with related work for overcoming it.

2.1 Taxonomy

Publishing personal micro-data for analysis or seeking better data mining performances, while maintaining individual privacy, is a problem of increasing importance today. Privacy preserving data publishing (PPDP) techniques have been deserved serious thinking and widely studied in recent years. A number of different methods and models have been developed toward PPDP, such as randomization and k-anonymity. Furthermore, the problem has been discussed in several communities such as the database community, the statistical disclosure control (SDC) community and the cryptography community.

A survey on some of the techniques used for privacy preserving data mining (publishing) may be found in (Verykios, et al., 2004) and (Aggarwal, et al., 2008). In (Verykios, et al., 2004), authors classified privacy preserving techniques based on five dimensions: data distribution, data modification, data mining algorithm, data or rule hiding and privacy preservation. Here we introduce the first two dimensions, based on it, a rough categorization of some well-known PPDP techniques will be shown subsequently.

The first dimension refers to the distribution of the data records. Many models have been designed for centralized data, while others aim to solve distributed data sets scenario.

The second dimension refers to the data modification schema. Approaches can in turn be differentiated depending on their modification on the original data. In general, methods of data modification include:

- **Perturbation,** or random perturbation, which is always accomplished by adding noise or altering of an attribute value with a new one.
- **Generalization**, which consists in substituting the values of a given attribute with more general values,
- **Suppression**, in which the value of the attribute is removed completely,
- **Swapping**, which refers to interchanging the values of individual entries in the dataset while still maintaining the overall distribution, and
- **Sampling**, that only releases a sample of the whole data population.

According to aforementioned dimensions, some existing techniques can be categorized as follows:

- **Data perturbation method**, which tends to preserve privacy by perturbing or swapping the input data and reconstructing distributions at an aggregate level before performing the mining.
- ***K*-anonymity model and its variations**, k-anonymity demands that every tuple in the micro-data table released are indistinguishably related to no fewer than k respondents. In this model, we reduce the granularity of data representation with the use of techniques such as generalization and suppression.
- **Distributed techniques**, in the scope of distributed PPDM, overlaps closely in cryptography for secure multiparty computation, which tends to compute functions over multiple recipients without actually sharing the input data with one another.

We delimitate our goal as the privacy preserving of the released datasets owned by one trusted party, thus data is centralized and available to the public. Distributed techniques are of the scope of this thesis work. In section 2, we will introduce the perturbation based method. In section 3, we will discuss the k-anonymization model along with its variations.

2.2 Perturbation Based Approach

Perturbation based method is a technique address the issue of privacy preservation by perturb data in a smart way such that original data would be masked but some important properties or distribution are preserved. After data perturbation process, it is not possible to accurately estimate original values individually, instead, the data mining process is rely on a reconstructed distribution at an aggregated level.

The idea was first introduced by Warner (Warner, 1965) in the context of survey techniques in order to eliminate evasive answer bias by interviewees due to privacy concerns. He developed the “randomized response method” for survey results in which involve interviewee only furnishes a truthful answer on a given probability p which is less than 1. It was suggested that this kind of method may encourage greater cooperation for certain survey problems.

2.2.1 Randomization

Randomization based approaches try to hide data by noise addition. It is based on the observation that sometimes data mining problems do not necessarily require individual records, but only distributions. Noise addition can be described as follows: The sensitive attributes $\mathrm{x_i} \in X$ will be modified to $(\mathrm{x_i} + \mathrm{y_i})$ according to a randomizing distribution function $\mathrm{f_Y}\ (\mathrm{y})$ so that they cannot be recovered with sufficient precision. For example, assume $f_Y\ (y) = Y(0, \sigma^2)$, Original database A would be perturbed by $G(A) = A + Y$.

Here, the main problem of randomization method is reconstruction, i.e. after perturbation, for each dimension, we are given: visible noisy data: $w_1 = x_1 + y_1, w_2 = x_2 + y_2 \ldots w_n = x_n + y_n$ and the known probability distribution of Y. The aim is to estimate the probability distribution of X as accurate as possible. Notice that data reconstruction should be performed at data recipient's side after data publishing.

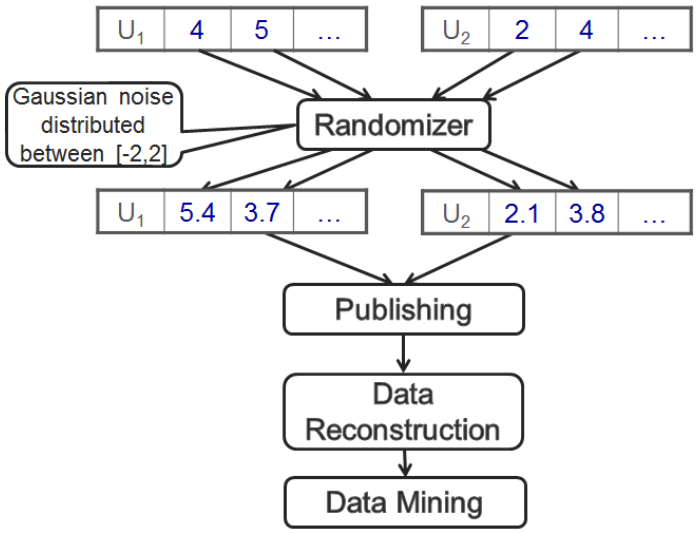

Figure 2.1 General process of randomization based PPDP

Agrawal and Srikant invented a reconstruction procedure based on Bayesian to reconstruct the original data distribution given a perturbed distribution (Agrawal, et al., 2000). The reconstructed distribution is then used to build accurate decision trees.

A subsequent work (Agrawal, et al., 2002) illustrated that the method suggested in (Agrawal, et al., 2000) did not take into account the distribution of reconstructed data, by which could cause privacy leakage. They raised a privacy measure based on the differential entropy of a random variable. They also provided an improved distribution reconstruction method using Expectation Maximization (EM) algorithm and proved that EM algorithm converges to the maximum likelihood estimate of the original distribution based on the perturbed data.

Polat and Du extended it to SVD based collaborative filtering in recommender system in (Polat, et al., 2005). In which a data disguising scheme was introduced to perturb the data sent by users before the server derives it, such that the server cannot get the truthful information about the user's private information while still allows the server to conduct SVD-based CF with decent accuracy. This work diverges to this thesis since it focuses on the privacy protection against the data owner, while we focus on the privacy preserving of the released data from a trusted data owner. Also, we try to provide an anonymization technique that suitable on any collaborative filtering algorithm, not specific to a particular collaborative filtering method.

Pros and Cons: One of the key advantages of randomization based approaches is that they are relatively simple, since the noise added to a given record is independent of the behavior of other records. Thus, randomization method can be achieved at data collection time. However this is also a weakness of randomization method since outlier records can often be difficult to mask thus are vulnerable to be attacked. Therefore, outlier records need to be covered by adding much more noise to all the records in the data, which definitely reduces the utility of data. Another key weakness of the randomization framework is that it does not consider the possibility that publicly available records can be used to identify the identity of the owner of that record. It has been proved in (Aggarwal, 2007) that the use of public databases can heavily influence the performance of randomization based approaches especially in high dimension cases.

2.2.2 Rotation Perturbation

Rotation perturbation is one kind of multiplicative data perturbation technique, which aims at perturb the data while preserving the multi-dimensional information w.r.t inter-column dependency and distributions.

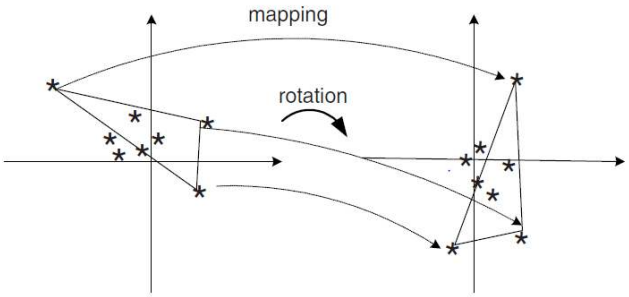

Figure 2.2 2-dimensional rotation perturbation

Intuitively, we can view this process as the figure above, but it is not only traditional "rotation", it includes all orthonormal perturbation. Chen and Liu first introduced this technique in (Chen, et al., 2005). The idea is easy, for a d-dimensional database, we generate a $d \times d$ orthonormal matrix $\mathrm{R_{d\times d}}$. Notice the feature of an orthonormal matrix:

- Each column is perpendicular to other $d-1$ columns so the dot product of each pair of columns is 0.
- Each column is normalized to unit length, so the magnitude of each column is equal to 1.

They multiply the original database A with some randomly generated orthonormal matrix R:

$$\mathrm{G(A)} = \mathrm{RA}$$

The data is rotated to some other coordinates, but preserve the geometric shapes in hyper-plane:

- Preserving length of a vector x $\|\mathrm{Rx}\| = \|\mathrm{x}\|$, and
- Inner product of x and y $\langle \mathrm{Rx}, \mathrm{Ry} \rangle = \mathrm{x}^{\mathrm{T}}\mathrm{R}^{\mathrm{T}}\mathrm{Ry} = \langle \mathrm{x}, \mathrm{y} \rangle$.

Thus all the data mining algorithms which only utilize the above properties, so called rotation invariant, can be performed with exactly same accuracy before and after rotation.

Rotation perturbation is vulnerable to so called (Independent Component Analysis) ICA technique, it is derived from the research of signal processing, can be applied to estimate the independent components of original dataset X from the perturbed data (in this case, the orthonormal matrix), under some conditions.

Pros and Cons: The good thing is high accuracy preserved as long as the data mining algorithm is rotation invariant. And it is not necessarily require redesigning algorithm. But also it is a shortcoming that it is limited to those data mining models which is invariant to certain data properties, another problem is that the loss of truthfulness at the record level. And also, ICA-based technique may violate the privacy.

2.2.3 Geometric Perturbation

Chen and Liu further proposed the idea of geometric perturbation by combination of randomization and rotation perturbation in (Chen, et al., 2007).

Given a d-dimensional orthonormal matrix R, and a Gaussian noise $\mathrm{N}(0, \sigma^2)$, they perturbed the original database A by the following fomula:

$$\mathrm{G}(\mathrm{A}) = \mathrm{RA} + \mathrm{N}|\mathrm{R}(\mathrm{A} + \mathrm{N})$$

They used noise addition to further protect the data privacy with a compensation of certain about of utility loss. It is all about a trade-off of privacy preserving and information loss.

2.3 *K*-Anonymity

Released data often contains information, which is not explicit identifier, but in combination can be linked to publicly available information to re-identify the data respondents (privacy users). The k-anonymity model was developed in the context of this kind of indirect identification attack from public databases. It tries to protect data by constructing groups of anonymous records. Simply speaking, it demands that every tuple in the micro-data table released be indistinguishably related to no fewer than k respondents.

The concept of k-anonymity was first introduced in (Samarati, et al., 1998), a subsequent work (Samarati, 2001) provided a formal foundation for k-anonymity model, it not only introduced the formal definition of quasi-identifiers, k-anonymity requirement, k-anonymity and k-minimal generalization, but also proposed the first algorithm for achieving k-anonymity.

2.3.1 Generalization/Suppression Based *K*-Anonymity

The original modification methods for achieving k-anonymity are generalization and suppression. The figure below shows an example of 2-anonymity table. Generalization consists in substituting the values of a given attribute with more general values. For example, zip code can be generalized by masking the least significant digits with stars. Suppression is just deleting the outlier records where generalization would cause huge information loss.

Birthdate	Sex	Zipcode	Disease
21/1/79	male	53715	Cancer
10/1/79	female	55410	Hypertension
1/10/44	female	90210	Cancer
21/2/83	male	02274	Obesity
19/4/82	male	02237	Cancer

→

	Birthdate	Sex	Zipcode
group 1	*/1/79	person	5****
	*/1/79	person	5****
suppressed	1/10/44	female	90210
group 2	*/*/8*	male	022**
	//8*	male	022**

Figure 2.3 An example of 2-anonymity achieved by generalization/suppression

The novel idea of k-anonymity raised an active study for algorithms that finding minimal (optimal) k-anonymous table. A quite detailed survey and overview of k-anonymity technique may be found in (Ciriani, et al., 2007). It provided taxonomy of different k-anonymity techniques according to the different levels on which generalization and suppression are performed.

The first algorithm for guaranteeing k-anonymity was proposed in conjunction with the definition of k-anonymity in (Samarati, 2001). The approach exploits a binary search on the DGH (domain generalization hierarchies) to avoid an exhaustive visit of the whole generalization space. Though this approach is simple, the time complexity is exponential to the number of quasi-identifiers.

Several exact algorithms have been proposed after that. In (Meyerson, et al., 2004), it proved that the complexity of finding optimal k-anonymization is NP-hard. Nevertheless, many approximation algorithms and heuristic algorithms were emerged.

One representative heuristic method proposed by Bayardo and Agrawal (Bayardo, et al., 2005), called k-Optimize algorithm, can often achieve a highly sub-optimal solution effectively. The key idea is assign an order to quasi-identifiers and associates an index, with each interval in every domain of the quasi-identifiers. A generalization is then represented through the union of the individual index values for each attribute. A set enumeration tree over the set of index values is built and by searching the tree using a standard traversal strategy, it is equivalent to the evaluation of each possible solution to the k-anonymity problem. Moreover, k-Optimize prunes a node when it can determine that none of its descendants could be optimal. They claimed that it is possible to fix a maximum computational time and obtain a good, but not optimal solution.

Pros and Cons: The k-anonymity model is elegant privacy preserving models for released data. The merit of k-anonymity can be concluded as its simple idea, truthfulness of the data and measurability. However, Generalization/Suppression based approaches are not suitable in our recommender system case.

On one hand, suppression will destroy the format of the original database, which is undesirable. On the other hand, as for the problem of generalization, from looking the example below:

	M_1	M_2	M_3
U_1	5	5	5
U_2	0	5	5
U_3	5	0	5
U_4	5	5	0

→

	M_1	M_2	M_3
U_1	[0,5]	[0,5]	[0,5]
U_2	[0,5]	[0,5]	[0,5]
U_3	[0,5]	[0,5]	[0,5]
U_4	[0,5]	[0,5]	[0,5]

Figure 2.4 4-anonymity table achieved by generalization

If we would like to anonymize these four records using generalization, the first problem is it will change the original data representation from a certain value to an interval which will cause problems on data miners side, cause a modification of data mining algorithm, and the second problem is that if not every users within a group have rated some certain movies (in the above example) we will have to generalize that attribute to be the whole set to be universal set ([0,5]), whose information is totally lost.

Fortunately, Suppression/Generalization are not the only data modification technique for implementing k-anonymity, next sub-section will introduce clustering based approach, which is more suitable for recommender system.

2.3.2 Clustering Based K-Anonymity (K-Gather)

While k-anonymity forces one to suppress or generalize an attribute value even if all but one of the records in a cluster have the same value, clustering based anonymization allows us to pick a cluster center whose value along this attribute dimension is the same as the common value, thus enabling us to release more information without losing privacy.

This thesis focus on achieving clustering based k-anonymity for recommender system. Clustering-based approaches first partition database into groups of similar records, and then anonymize records within each group, which is more meaningful and suitable for continues and ordinal attributes.

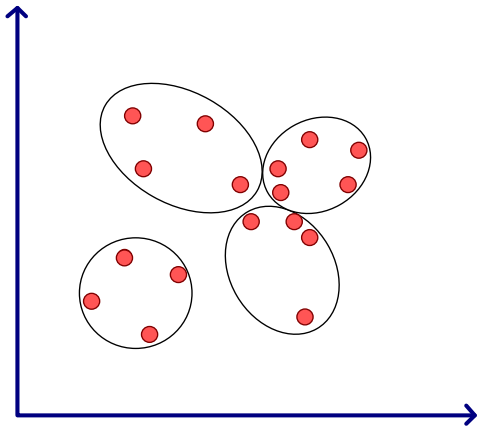

Figure 2.5 A possible solution for 4-gather clustering

Then we can call this partition problem as "***k*-gather problem**", borrowed from (Aggarwal, et al., 2006), which formalize the problem as classifying a set of points, say N points, into a number of clusters such that each cluster has at least k points, and the objective is to minimize the average radius among the clusters. It has been proved that this problem is in the class of NP-completeness and hardness. The above figure shows a trivial example of 4-gather clustering in a two-dimensional space.

The figure below illustrates how a toy 5×7 recommender system anonymized by this technique. We first apply some k-gather clustering algorithm to find groups of similar users. Intuitively in this example, we need to cluster the first two users into a group and last three users to another group. Then within each group, we anonymized the rating of each movie as the average rating over only users who rated that movie, i.e. $\bar{r}_j = \frac{\left(\sum_{u_i \in G, r_{ij} \neq 0} r_{ij}\right)}{m}$, where m is the number of users in G who rated movie o_j. The data publisher just needs to publish the central point and the number of records of each anonymous group.

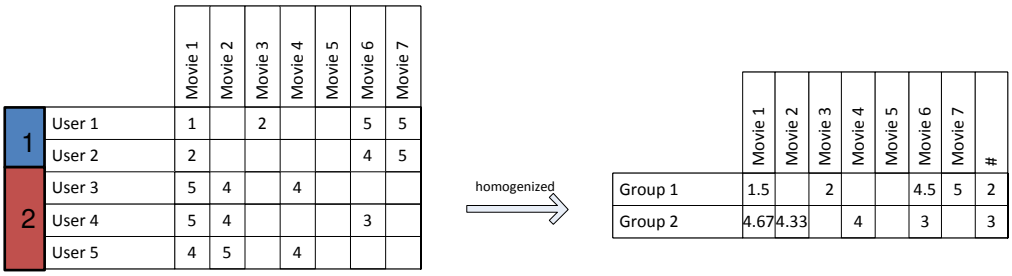

Figure 2.6 Clustering based 2-anonymity for a toy RS

Clustering based k-anonymity has been studied widely in different communities. In the statistical disclosure control community, Domingo-Ferrer et al. discussed how to use microaggregation, in which clusters of records are constructed, to achieve k-anonymity (Domingo-Ferrer, et al., 2002), (Domingo-Ferrer, et al., 2005), (Domingo-Ferrer, et al., 2006). K-gather and k-cellular clustering problems was defined in (Aggarwal, et al., 2006), and they has shown the proof for NP-completeness and hardness of both problems and further provided a constant-factor approximation algorithm. Byun et al. (Byun, et al., 2007) formulated this kind of clustering problem for k-anonymity as k-member clustering problem and proposed a greedy k-member algorithm in which finds a solution in a greedy heuristic manner. Thompson and Yao (Thompson, et al., 2009) proposed the bounded t-means and the union-split clustering algorithm.

Borrowed the concept from (Domingo-Ferrer, et al., 2005), we classify the existing k-gather heuristics into two classes:

Fixed-size *k*-gather: These heuristics consists in constructing $\left\lfloor \frac{n}{k} \right\rfloor$ anonymous groups which all have size *k*, and assign rest records to one or some constructed groups. Bounded t-means (Thompson, et al., 2009), Byun's greedy algorithm (Byun, et al., 2007), MDAV-generic (Domingo-Ferrer, et al., 2002) are in this class.

Variable-size *k*-gather: These heuristics yield anonymous groups which all have sizes varying between k and $2k-1$. Example is MHM algorithm (Domingo-Ferrer, et al., 2005).

Fixed-size k-gather heuristics are often very efficient and simple. However, Variable-size methods often achieve lower information loss since they are able to adapt the choice of group sizes to the structure of the dataset.

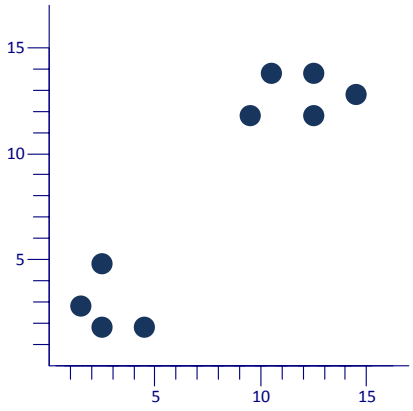

Figure 2.7 Variable vs. fixed-sized groups

The idea behind variable group size is to avoid unnatural k-partition. For instance, consider the dataset in Figure 2.7, applying fixed sized 3-gather is quite unnatural. The optimal solution for this dataset obviously consists of a group with four records and another group with five records.

2.3.3 Challenges

Most k-gather algorithms proposed are only designed for relational datasets. In recommender systems, sensitive values and quasi-identifier values are joint sets. i.e., all values in the recommender systems could be sensitive, but also could be used as quasi-identifiers. Furthermore, the inherent high dimensionality and sparsity of recommender system both raise the difficulty in applying k-gather clustering algorithm.

2.3.3.1 Curse of dimensionality problem

Recommender system is always in a very high dimensional space. In most cases, we depend critically on distance or similarity measurement in k-gather algorithm, and require tuples within an anonymous group are, in general, closer to each other than to tuples in other groups. (Steinbach, et al., 2003) showed that in high dimensional spaces, distances between points become relatively uniform, which means distance functions begin to show loss of intra-record distinctiveness thus seem to be meaningless.

$$\lim_{d\to\infty}\frac{MaxDist-MinDist}{MinDist}=0$$

This is so called curse of dimensionality problem, and therefore, distance function like Euclidean distance tends to be meaningless with the raise of dimensions. Even we can utilize other similarity function like correlation coefficient based distance function; we still suffer from high time-consuming which makes a lot of candidate algorithms inapplicable.

One candidate proposal is subspace clustering approach which consists in automatically detecting clusters in subspaces of the original feature space that in high dimensional space, different subspaces may contain different, meaningful clusters. There are many branches of subspace clustering approach, e.g. grid based approach:

- Divide the space over which the data ranges into rectangular cells
- Searching in some way (e.g. Apriori)…
- Discard low-density grid cells.
- Combine adjacent high-density cells to form clusters.

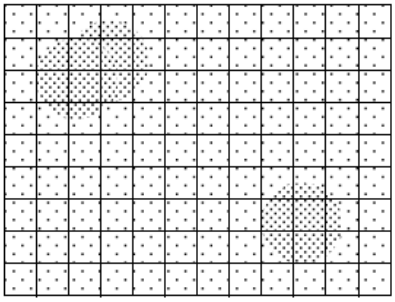

Figure 2.8 Grid based subspace clustering

One of the representative gird based approaches is CLIQUE in (Agrawal, et al., 1998). It finds subspaces automatically by using an apriori-like method, which navigates possible subspaces recursively in a bottom-up way. One observation they found is a “dense” region in a specific subspace must be dense when projected onto lower dimensional subspaces. Utilizing this observation, after finding dense one-dimensional intervals, it is possible to investigate all the potential dense two-dimensional intervals, thus we can find actual dense two-dimensional intervals. The procedure can be extended to find dense units in any subspace in this “bottom-up” way.

Some other subspace clustering approaches are projected clustering approaches like PROCLUS (Aggarwal, et al., 2000), ORCLUS and density based approach SUBCLU (Kailing, et al., 2004).

One fatal deficiency of subspace clustering approach is that the number of grids normally increases exponentially with the raise of dimensionality. If each dimension is only split in two (e.g. {[1, 3], (3, 5]}), we will have 2^d cells. For Netflix example, d = 17700, we will have 2^{17700} cells to explore, even take a square root of it, we have no expect to see the result before we die. We say it suffers from another aspect of the curse of dimensionality.

2.3.3.2 Inherent sparsity problem

Real-life recommender system is always very sparse, for example, only 1.16% entries are non-zero in Netflix database.

Since it is essential to group users with similar movie preferences to minimize information loss, while due to the sparsity of recommender system, even users with very similar preference may have only a small overlap in the items they have rated, thus applying k-gather clustering algorithm directly on the original data may not be sufficient to explore similar users.

	Terminator II	Terminator III
User1	5	
User2		5

Figure 2.9 Sparsity problem

For example, from looking the table above, we know Terminator II and III are movies with similar genres and similar casts. Two users give high rate to one of the two movie shows their similar movie preferences, but they have no overlap in the movies they rated.

2.3.3.3 Easier variable-size k-gather heuristic

Another challenge is about the k-gather clustering algorithm, notice that the problem of optimal k-gather is NP-hard, so a lot of heuristics have been proposed, but many of existing heuristics are fixed methods, this can cause unnatural k-partition, like this example of Figure 2.7 **Variable vs. fixed-sized groups**, the optimal solution for this dataset is obviously consists of a group of 4, and another group of 5. The existing variable-size algorithms sometimes are too complicated and hard to be implemented, so we need to work out a simple, efficient variable-size approach.

2.3.4 L-Diversity and T-Closeness

Machanavajjhala et al. (Machanavajjhala, et al., 2006) presented two attacks, i.e. homogeneity attack and the background knowledge attack, to compromise a k-anonymous table. They pointed out that k-anonymity is susceptible to these two attacks due to the lack of preventing inference of the sensitive values of the attributes. Thus they define a stronger notion of privacy model that is l-diversity, in which not only maintains the minimum group size of k, but also contains at least l different values for the sensitive attributes.

Li N. et al further enhanced the concept of k-anonymity and l-diversity to the t-closeness framework (Li, et al., 2007). They showed skewness attack and similarity attack towards l-diversity, argued that global distribution may be used by adversary, they proposed the principle of t-closeness, in which uses the Earth Mover distance metric to measure whether the distance between the distribution of a sensitive attribute within an anonymized group is different from the global distribution by more than a threshold t.

There are many more variances of k-anonymity proposed afterwards, L-diversity and t-closeness are out of the scope of this thesis.

3 General Solution

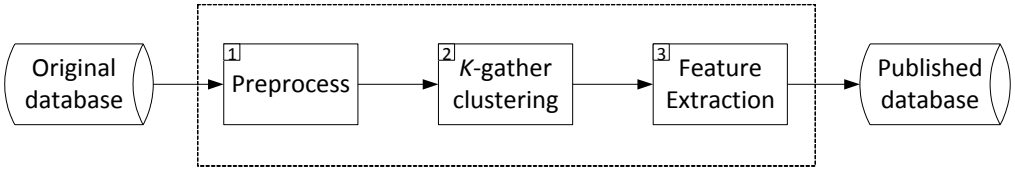

Figure 3.1 General PPDP system components

Here is the general process. It consists of three major phases.

- A preprocess step strategically eliminates the high dimensionality and sparsity problem.
- Then we invent some variable k-gather clustering heuristic (Bisecting *K*-Gather) which applies on the processed data to find group of similar users.
- Finally we anonymize data within each cluster.

3.1 Preprocess (User Preference Matrix Extraction)

We take a neat approach to preprocess the data before applying k-gather clustering algorithm. An important observation here is that similar users are not necessarily those who have rated the same movies but users who have similar movie preferences.

So, how to extract user preferences from the original database? In today's mathematical world, this technique is called SVD (singular value decomposition). SVD is a matrix approximation method that factors any $\mathrm{m} \times \mathrm{n}$ matrix A into three matrices as the following:

$$\mathrm{A}_{\mathrm{m}\times\mathrm{n}} = \mathrm{U}_{\mathrm{m}\times\mathrm{m}}\Sigma_{\mathrm{m}\times\mathrm{n}}\mathrm{V}^{\mathrm{T}}_{\mathrm{n}\times\mathrm{n}} \qquad (1)$$

Here Σ is a diagonal matrix which contains singular values of A. The columns of U and V are, respectively, left- and right-singular vectors for the corresponding singular values. If we order the singular values in descending order and choose r most significant values, SVD will output a matrix A' of rank r with the minimum approximation error in terms of least squared distance from A. I.e.:

$$\mathrm{A}'_{\mathrm{m}\times\mathrm{n}} = \mathrm{U}_{\mathrm{m}\times\mathrm{r}}\Sigma_{\mathrm{r}\times\mathrm{r}}\mathrm{V}^{\mathrm{T}}_{\mathrm{r}\times\mathrm{n}} \qquad (2)$$

There is some latent meaning behind singular values, i.e. we can treat singular values as implicit aspects or features of movies. For example, if we set r to be 10, that means we limit movies to have 10 features such that each movie is described only by 10 values say how much that movie exemplifies each feature, and correspondingly each user can be described by another 10 values say how much they prefer each aspects. Here I explicit describe this insight as following tables. And the rate is nothing but the product of these two vectors (r_{11}=3*1+ (-0.3)*0.1+……). Notice here a negative value is meaningful, it shows the extent a user dislikes this feature or the movie is of opposite feature.

	Action	Animation	Comedy	Crime	Romance	Sci-Fi	Thriller	…
User 1	3	-0.3	0.1	2.1	-3	1.5	1	

	Action	Animation	Comedy	Crime	Romance	Sci-Fi	Thriller	…
Movie 1	1	0.1	1.4	-2	2	-0.2	-0.2	

Figure 3.2 User preference vector and movie feature vector

$$A'_{m\times n} = U'_{m\times r} V'^{T}_{r\times n} \text{ , where } U'_{m\times r} = U_{m\times r}\sqrt{\Sigma_{r\times r}}^{T} \text{ , } V'^{T}_{r\times n} = \sqrt{\Sigma_{r\times r}} V^{T}_{r\times n}. \quad (3)$$

We connect aforementioned neat idea with our matrix terms. We transform the equation (2) a bit to equation (3). In equation (3), the original matrix has been decomposed into two very oblong matrices: a $n \times r$ movie aspect matrix $V'^{T}_{r\times n}$, and a $m \times r$ user preference matrix $U'_{m\times r}$. Matrix $U'_{m\times r}$ is exactly what we are looking for.

There are hundreds complicated ways of computing singular value decompositions in the literature doesn't mean there isn't a really simple way too, we use the method invited by Simon Funk (Funk, 2006). It computes the approximate singular value decomposition using a form of regularized stochastic gradient descent. Each left and right singular vector make up a singular factor and is computed one at a time. Each factor is estimated by iteratively looping over the data points and training the singular vectors making up the current factor.

At the end of the preprocessing phase, we get our low-dimension linear model $U'_{m\times r}$ which describes user preferences. Empirically, it is highly expected that we can reduce dimensionality of original database to the user preferences matrix by a magnitude of 10^2 without loss of accuracy, which effectively eliminates curse of dimensionality problem. Furthermore, since we actually extract user preferences from the original recommender system, we have already solved sparse rating problem.

3.2 K-Gather Clustering (Bisecting K-Gather)

After the preprocess phase, we have extracted user preference matrix out of the original database which is in a much lower dimensional space and indicate users' movie preference, then we need to explore similar users from user preference matrix.

As I mentioned before, fixed k-gather clustering algorithms like bounded T-means (Thompson, et al., 2009), has drawbacks, and may suffer higher information loss, so here we invented a new k-gather heuristic for our own, called bisecting k-gather (BKG for short). This section first proposed the Bisecting k gather algorithm. And then evaluate its time complexity in both best and worst case scenarios.

3.2.1 Algorithm Description

BKG derives from the conventional bisecting k-means algorithm, which is a variance of k-means tends to produce clusters of similar sizes with smaller entropy. The algorithm exploits a top-down partition strategy that iteratively bisects the original dataset with possible least entropy. One can view the process akin to constructing a BSP (Binary Space Partitioning) tree, root is the entire dataset, BKG recursively divides the entire dataset into two until every leaf contains at least k records and the union of all leaves is the entire dataset. Our bisecting k gather clustering method is described as follows.

ALGORITHM 1 [BISECTING_K-GATHER **(**R: dataset, k: integer)]**.**

1. Compute the centroid $\bar{\mathbf{x}}$ of $\boldsymbol{R}$**.**
2. Randomly pick one record, $\mathbf{c_L} \in \mathbf{R}$**,** and compute $\mathbf{c_R} = \bar{\mathbf{x}} - (\mathbf{c_L} - \bar{\mathbf{x}})$**.**
3. Bisect $\mathbf{R}$ into two clusters $\mathbf{G_L}$ and $\mathbf{G_R}$ according to the following rule:

   $\begin{cases} \boldsymbol{x_i} \in \boldsymbol{G_L}\ \boldsymbol{if}\ \|\boldsymbol{x_i} - \boldsymbol{c_L}\| \le \|\boldsymbol{x_i} - \boldsymbol{c_R}\| \\ \boldsymbol{x_i} \in \boldsymbol{G_R}\ \boldsymbol{if}\ \|\boldsymbol{x_i} - \boldsymbol{c_L}\| > \|\boldsymbol{x_i} - \boldsymbol{c_R}\| \end{cases}$, where $\|\cdot\|$ stands for specific similarity function.

4. Repeat step 2 and 3 *number_of_iterations* times, select clusters from bisection with lowest SSE.
5. Count number of records in each cluster, without losing generality, assume $\boldsymbol{n_L} \le \boldsymbol{n_R}$**.**
6. If $\boldsymbol{n_L} \in (\boldsymbol{0},\boldsymbol{k})$**,**
   A. Compute the centroid $\overline{\boldsymbol{x_L}}$ of $\boldsymbol{G_L}$.
   B. Form a cluster containing $\boldsymbol{G_L}$ and $\boldsymbol{k} - \boldsymbol{n_L}$ records closest to $\overline{\boldsymbol{x_L}}$**.**
7. For either cluster $\boldsymbol{G_i}$ of $\{\boldsymbol{G_L}, \boldsymbol{G_R}\}$**:**
   A. If $\boldsymbol{n_i} \in [\boldsymbol{3k},+\infty)$**,** *call* BISECTING_K-GATHER$(\boldsymbol{G_i}, \boldsymbol{k})$**.**
   B. If $\boldsymbol{n_i} \in [\boldsymbol{2k},\boldsymbol{3k})$**,**
      I. Compute the centroid $\overline{\boldsymbol{x_i}}$**.**
      II. Find the most distant record $\boldsymbol{x_r}$ from $\overline{\boldsymbol{x_i}}$**.**
      III. Form a cluster containing $\boldsymbol{x_r}$ and$\boldsymbol{d}$ $\boldsymbol{k} - \boldsymbol{1}$ records closest to $\boldsymbol{x_r}$**.**
      iv. Form another cluster containing the rest of records.

For a large-scale dataset that $k \ll n$, it is expected that the dataset will be partitioned into a number of "bins" (step 7.a) before dealing with clusters of size between 0 and $3\mathrm{k}$. For cluster of size $\mathrm{n_L} \in (0,k)$, we combine $\mathrm{k} - \mathrm{n_L}$ closest records in its sibling to form a group; We satisfy with clusters of size in $[k,2k)$; as for cluster of size $[2k,3k)$, we heuristically divide the cluster into two cluster, one of size *k*, the other of size in $[k,2k)$. The corresponding flow chart of the algorithm is shown below.

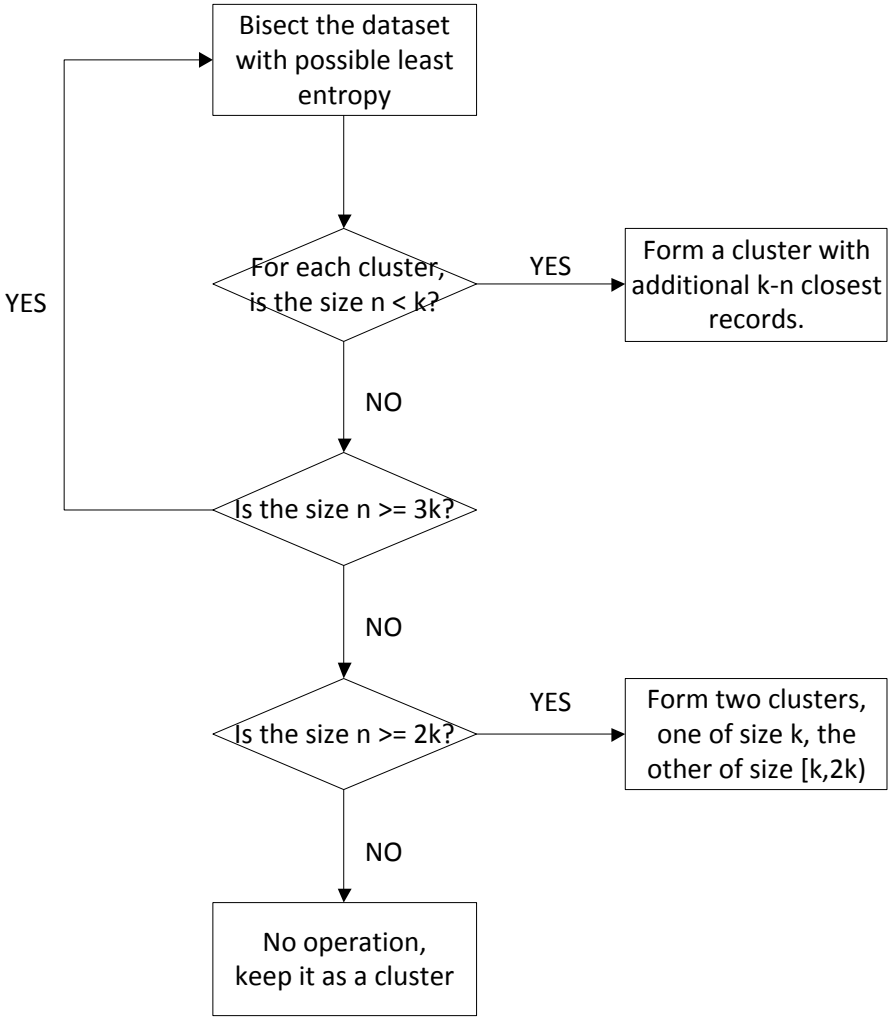

Figure 3.3 Flow chart for BKG algorithm

It is much clearer to explain the algorithm with a concrete example. Figure 3.4 below shows an example about the process of how BKG works for a 3-gather problem given a set of 12 randomly generated points.

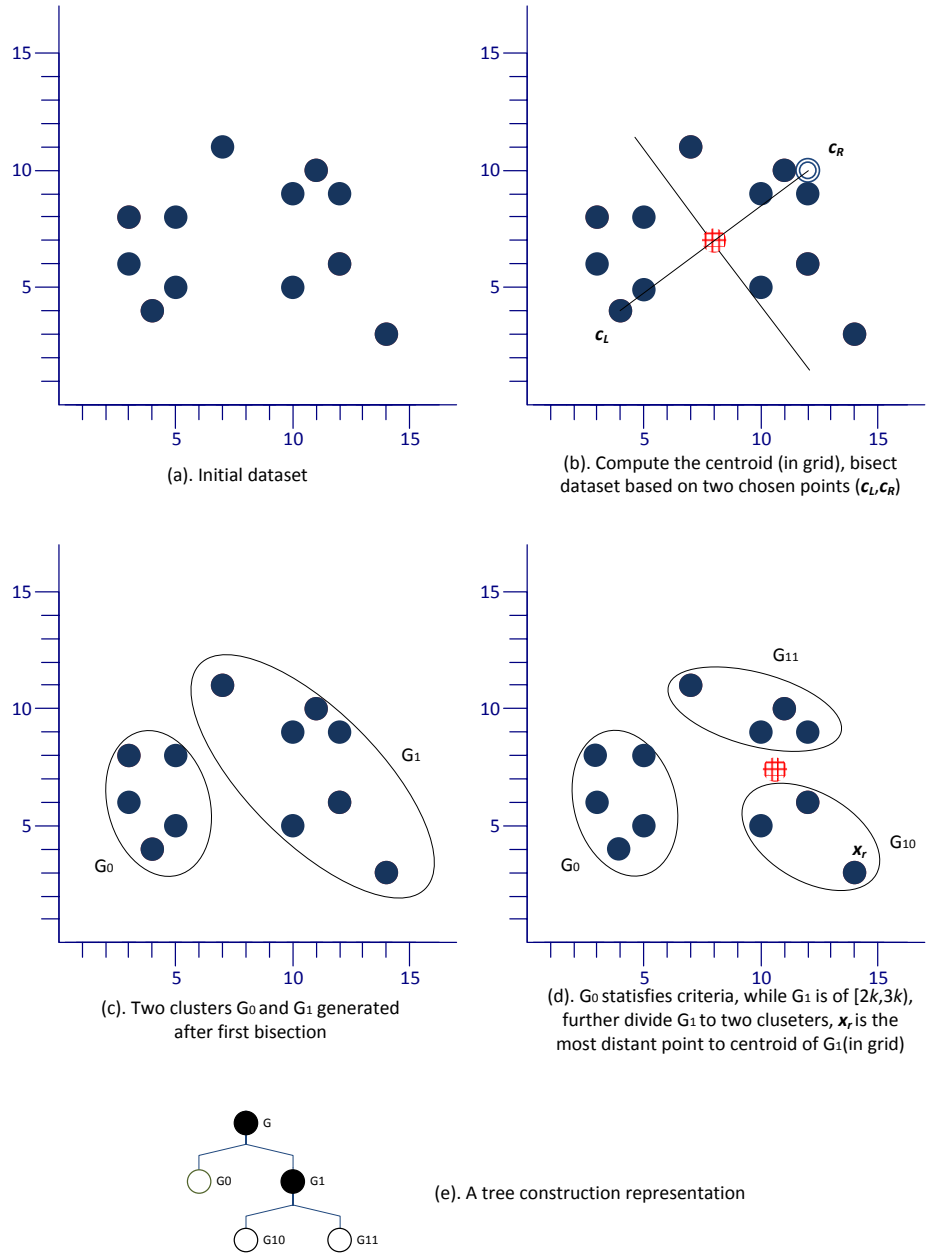

Figure 3.4 3-anonymity achieved by BKG for a set of 12 random points

It computes the centroid of the dataset first which is in grid (b), and we bisect the dataset based on one randomly picked point and its symmetrical point., this bisection operation can be repeated several times and we will choose one with least SSE. Two clusters generated after first iteration (c). G_0 is of size $[k,2k)$, we satisfy with this, and G_1 is of size $[2k,3k)$, we further divide G_1 into two clusters (d), we do this, we compute the centroid, get the most distant point to the centroid $\mathbf{x_r}$ and form a cluster of $\mathbf{x_r}$ and its two closest neighbors. The rest forms the other cluster.

3.2.2 Time Complexity Analysis

Since BKG is a non-deterministic algorithm, it is not easy to measure the average time complexity of the algorithm. Here we measure the best case and worst case scenarios.

Lemma 1 the best case time complexity of BKG is $O(n\ log n)$.

Proof: the complexity of BGK algorithm can be measured as the number of required distance computations. In best scenario, the dataset is not skewed much, means the data is relatively uniformly distributed, such that we are constructing a quite balanced BSP tree, we need about $log\frac{n}{k}$ iterations to produce nearly $\frac{n}{k}$ leaves (groups), and the best case happens when all the clusters' sizes are in $[k, 2k)$, therefore, it is more like a pure binary search and no extra work to assign the points. So therefore, in each iteration, to determine which cluster a point belongs to, we need n distance computations. The total number of distance computations is about $n\ log\frac{n}{k}$. Thus the time complexity is $O(n\ log n)$. ■

Even if the clusters are all with size $[2k,3k)$, let's measure the number of distance computations required in step 7.b. The centroid computation takes n additions; find the most distant record from centroid also takes n additions and finding $k-1$ closest points costs about $(k-1)n$ additions. Thus the time complexity is $O\left(kn\ log\frac{n}{k}\right)$. Thus the time complexity is also $O(n\ log n)$.

Lemma 2 the worst case time complexity of BKG is $O(n^3)$.

Proof: The worst case happens when following requests are met:

The dataset is extremely skewed such that we lose our nice tree structure degraded to a linked list.

After each iteration, a cluster of size $(0,k)$ is separated from the entire set.

For step 1, we now need $\frac{n}{k}$ iterations instead of nice $\log\frac{n}{k}$. Then we calculate the number of distance computations for step 6, just to make it simpler, we assume that the number n of records in the dataset is a multiple of k and assume we get no records in c_L's cluster, then we need n distance computations to find the closest record from c_L, $n-1$ distance computations for the second closest record, up to $n-k+1$ distance computations for the k closest record. Thus, the total number of distance computations is

$$\sum_{i=0}^{n-2k} n-i \approx \frac{n(n-1)}{2}$$

Multiply the times of iterations with number of distance computations in each iteration, we get $\frac{n^2(n-1)}{2k}$, which means the complexity of the worst case is $O(n^3)$.■

In real life, the worst case scenario rarely happens. We expect complexity comparable to $O(n\ log\ n)$.

3.3 Feature Extraction

After user preference extraction and k-gather clustering phases, a set of clusters with each cluster contains user(s) with similar movie preferences are produced. Notice what data publisher want to publish is the anonymized recommender system, not the anonymized user preferences table, so for each cluster of records, we need refer back to our original recommender system, and anonymize the rating of each movie as the average rating over only users who rated that movie, i.e. $\bar{r}_j = \frac{\left(\sum_{u_i \in G, r_{ij} \neq 0} r_{ij}\right)}{m}$, where m is the number of users in group G who rated movie o_j.

So by anonymizing the rating of each movie within the group, some missing entries will be replaced by group average values , such that the total number of entries increases with the raise of k, the figure below shows the total number of entries we would get with the increase of k (MovieLens database).

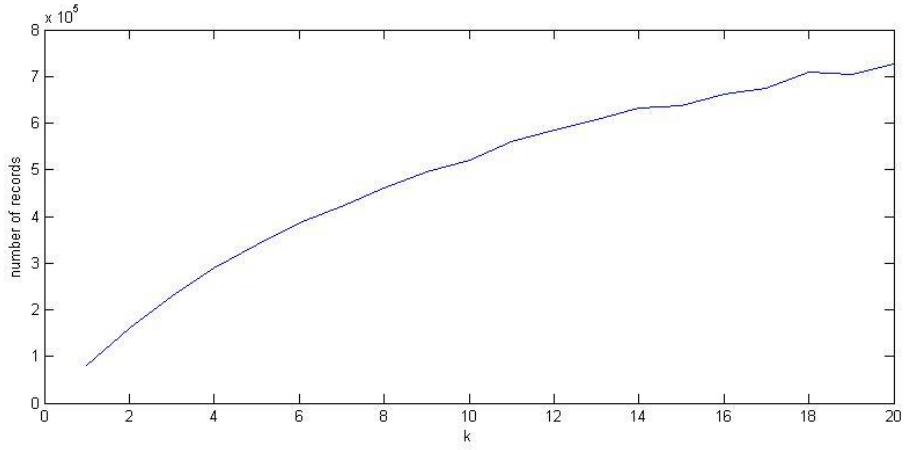

Figure 3.5 The increment of entries with the raise of k

Now this anonymized database is ready to be published.

4 Evaluation of the General Process

We implemented our general solution in a JAVA environment. Chapter three goes through our general process of anonymization, this chapter, we evaluate the performance of our newly invented k-gather clustering algorithm (bisecting k-gather) and also more importantly, the utility loss evaluation.

4.1 Data Set

We use MovieLens data in our experiments. It was collected by MovieLens website for seven months by the GroupLens Research Project. It contains 100K ratings of 1682 movies by 943 users. Ratings follow the 1 (bad) to 5 (excellent) numerical scales. The sparsity of the data set is high, at a value of 93.7%.

4.2 Evaluation of BKG

We evaluate the performance of BKG as well as comparison between bisecting k-gather and bounded t-means (Thompson, et al., 2009) (a representative of fixed-size k-gather) with respect to the clustering quality.

4.2.1 Measure of Clustering Quality

The root mean square error (RMSE) is used to evaluate the clustering quality of a single anonymous group. RMSE measures the standard deviation between predicted and centroid point as follows

$$RMSE_j = \sqrt{\frac{1}{T}\sum_{u_i \in G_j, i=1}^{T} (\mathbf{x_i} - \bar{\mathbf{x}})^2}$$

Where T is the size of the group, $\mathbf{x_i}$ is the individual point and $\bar{\mathbf{x}}$ is the centroid of G_j. After calculating RMSE of each anonymous group, we calculate the average RMSE (ARMSE) of all the groups by the following equation.

$$ARMSE = \frac{1}{N}\sum_{j=1}^{N} RMSE_j$$

Where N is the number of groups, and ARMSE is the measure of clustering quality.

4.2.2 Evaluation Method

The experiments are of two parts, one for the evaluation of bisecting k/gather itself and the other for the comparison of performance between BKG and BTM.

The first part assesses BKG with respect to time consumption, scalability and stability. We run BKG for variable k from $\mathrm{k}=3, \mathrm{k}=5$ to $\mathrm{k}=10$, with each case run the algorithm twenty times and calculate the ARMSE of each time using the equations shown in 4.2.1.

The other part is about the comparison between BKG and BTM. Using the same dataset, we fixed k to be 3 and ran BKG and BTM for twenty times respectively and compare the average radius of each cluster.

4.2.3 Results

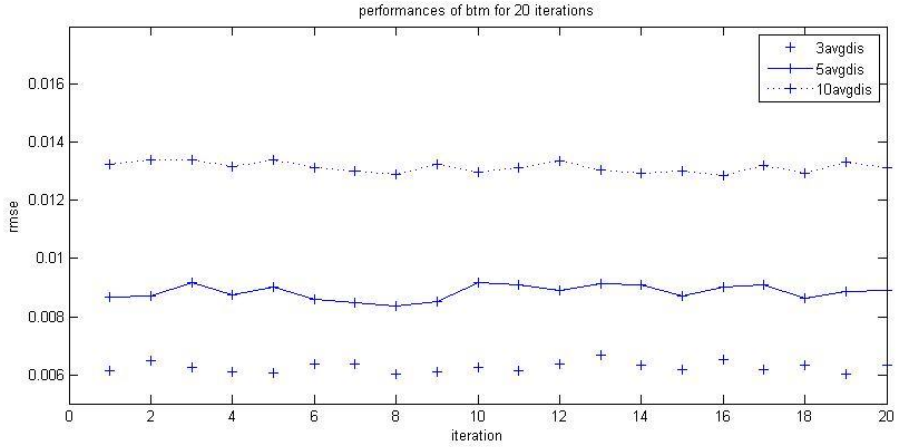

Figure 4.1 The performance of BKG of k=3, 5, 10 respectively

The speed of BKG is very fast, it only takes several seconds to cluster user preference matrix. That's partly due to the impact of dimension reduction performed in the early stage, but also proved the efficiency of the algorithm.

And from the performance result, by looking at the figure above, the average radius of cluster increases with the raise of k, but the raise is in sort of linear manner, and with the same k, the discrepancy of result for twenty rounds is quite stable, at a maximum difference of around 0.0005.

The result shows the efficiency, stability and scalability of BKG.

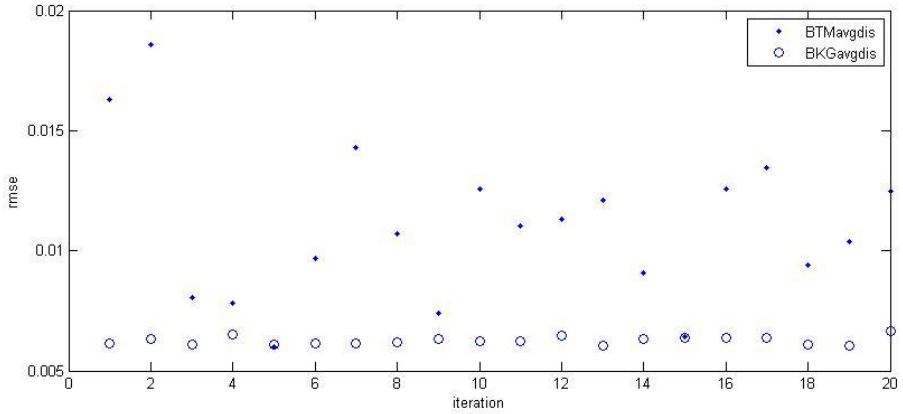

Figure 4.2 The performance comparison between BKG and BTM of k=3

This plot shows the average radius of clusters produced by BKG and BTM for twenty tries respectively. Obviously, BKG outperforms BTM a lot. Though the best result from BTM is competitive to BKG, the performance of BTM fluctuates and not deterministic, in worst case, the ARMSE is about 0.018. BKG, on the other hand, produced more stable results with low average radius.

4.3 Evaluation of Utility Loss

Next, more importantly, now that we successfully manage to anonymize the database, we need to evaluate how much utility we will loss as a compensation.

4.3.1 Measure of Prediction Accuracy

The mean absolute error (MAE) is used to evaluate the prediction ability. MAE measures the deviation between predicted and actual ratings of users as follows.

$$MAE = \frac{1}{\mathrm{T}}\sum_{\mathrm{t}=1}^{\mathrm{T}}\left|\mathrm{x_p} - \hat{\mathrm{x}}\right|$$

Where T is the number of ratings of test set, and $\mathrm{x_p}$ and $\hat{\mathrm{x}}$ are the predicted and actual ratings, respectively.

4.3.2 Evaluation Method

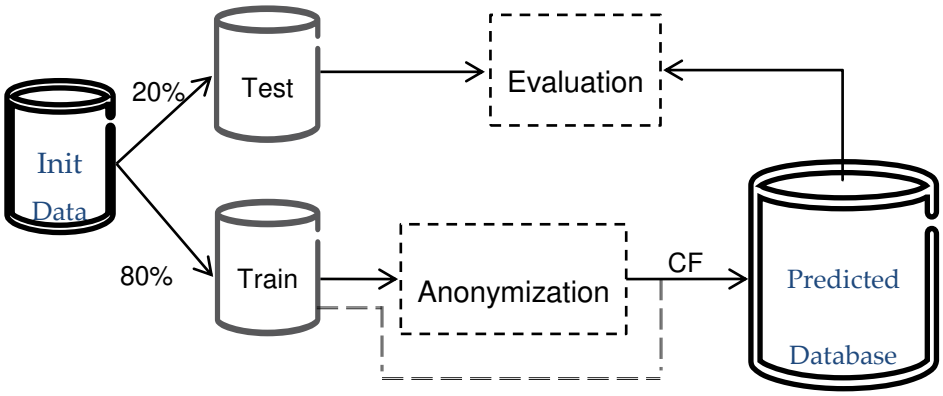

Figure 4.3 Utility evaluation model

4.3.2.1 Benchmark

Start from the initial data set, we randomly split the initial data set into two subsets, one is the training set which contains 80% of the ratings, and the other is test (20%) set that assumed to be missing. Each entry in the test set is regarded as unknown rating in the training process, and after applying some collaborative filtering algorithm (incremental SVD) directly (follow the dashed line), we got the predicted database, we calculate the MAE between the test set (which contains actual ratings) and corresponding predicted values. The result MAE is our base line.

4.3.2.2 Evaluation for anonymized dataset

In order to guarantee the comparability, we split the original data into training set and test set the same way as mentioned above, the training set is regarded as real dataset need to be anonymized.

After anonymization process, anonymized dataset is generated by computing the average record among each anonymous group. Notice that the number of entries increases with the raise of k as shown in the figure 3.5.

The same CF algorithm is performed for anonymized training dataset, adjust the number of iterations for the algorithm converges and avoids over training. Since the number of anonymized users (i.e. group) decreases with the raise of k, so less iterations is needed for SVD to converge. Then we calculate the corresponding MAE for each k as an average MAE of 10 tries.

4.3.3 Results

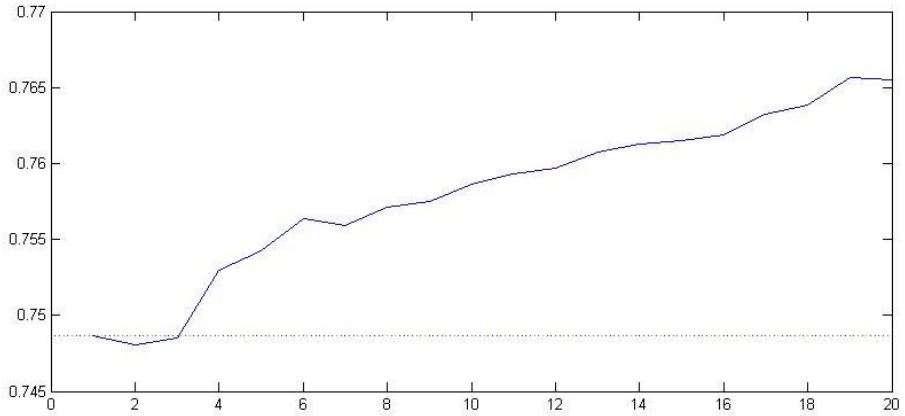

Figure 4.4 The increment of entries with the raise of k

The horizontal dotted line is the base line. I.e. the MAE when applies some CF algorithm directly on the original database. The MAE is about 0.748. And if we take the average ratings of all the records, all the rows are anonymized to be one unique record, the MAE is about 0.815.

The solid line illustrates the MAE of *k*-anonymity achieved by BKG with the raise of k. It shows, when *k* equals to 2 or 3, the MAE are even slightly better than the benchmark,

That is because, by anonymize similar users, we actually fill in many missing values with high accuracy. So we actually get more latent information by anonymization. It shows. If we combine 2 or 3 similar users together, we can get benefits from the additional information we padded, in other word, the quality of information we gained is good enough to improve the recommendation accuracy.

But with the raise of k, though the amount of additional information also increases, the accuracy of information will become worse and worse. It's a trade-off between the amount of information gained and the quality of information. Thus MAE increases with the raise of k, but the increment extent is under control, I will say we can anonymize the data at a very low cost of information loss.

5 Customized K-Anonymity

In this chapter, we introduce a new concept named “customized k-anonymity”, which is a more flexible privacy model. This concept is mainly for the purpose of flexible anonymization for different users, we aim to separate users with different privacy levels. I.e. make the privacy level customized, and try to come up with a method to solve this variable privacy level circumstance.

The concept and motivation will be proposed in the first section and then the introduction of user policy module and the modified anonymization schema will be provided. In 6.3, we modified our BKG to be a candidate algorithm for customized k-anonymity (basic case).

5.1 Concept and motivation

In some PPDP scenarios, not all the records are required the same privacy concern. For instance, in movie recommender system case, some subscribers may not treat their movie ratings as their privacy. They don’t care if someone knows his/her rating information. Then with the potential to improve the overall anonymization quality with respect to the information loss, data publisher can provide subscribers a set of privacy concern levels (PL in short), and with the potential to improve the individual recommendation accuracy, subscribers who doesn’t care about his/her information leakage has reason to choose lower privacy levels. Here we raised a hypothesis.

Hypothesis: Users of lower privacy level can get better recommendation accuracy.

If the hypothesis is true, then it will be major reason for those users who don’t care about their information leakage to choose lower privacy levels.

In general, it is common that the collection of data records owned by data publisher may have different privacy concern requests, thus k is not a global fixed number, but an individual flexible privacy parameter. And therefore, each record is tagged with an attribute of privacy level, generally, from $[k_{min}, k_{max}]$.

We can define clustering based customized k-anonymity problem as follows.

Definition 1 (customized k-gather problem) The customized k-gather problem is to cluster n records (with individual privacy level k) in a metric space into a set of clusters, such that for each cluster, the number of records satisfies each individual’s privacy level. The objective is to minimize the maximum radius among the clusters.

In this thesis, we have studied the basic case of customized k-anonymity, also the most meaningful and popular case, i.e. there are only two alternative privacy levels $k \in \{1, c\}$, $k = 1$ means no privacy concern, and constant c is the maximal privacy concern level. We name this problem as 1-k-gather problem.

5.2 User policy

After pre-processing phase, we now get dataset in a reduced dimensional space. In next phase, in order to support customized k-anonymity, we introduce a user policy module, which renders privacy requirements from the user side. Here, privacy requirement varies, the basic requirement could be assignment of privacy level k. However, user policy can be more specific and complex, for example, assign different privacy levels to each record, which is what we name "customized k-anonymity".

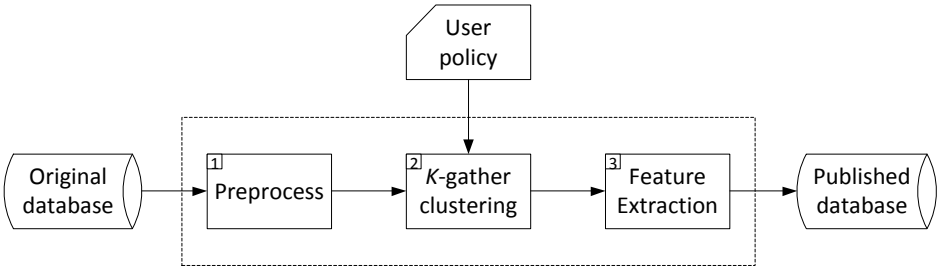

Figure 5.1 The anonymization schema with user policy

When received privacy requirements from the users, we can apply appropriate algorithm specific to the requirements. While there exist many heuristics for traditional k-gather problem, but since customized k-anonymity is a new concept thus no existing algorithm can be leveraged. We work out a new heuristic for our own.

5.3 Bisecting One-*K*-Gather

We have modified our BKG with additional tool functions to be a heuristic for solving 1-k-gather problem. The new heuristic is called bisecting one-k-gather (BOKG for short). The figure below describes the algorithm with a flowchart.

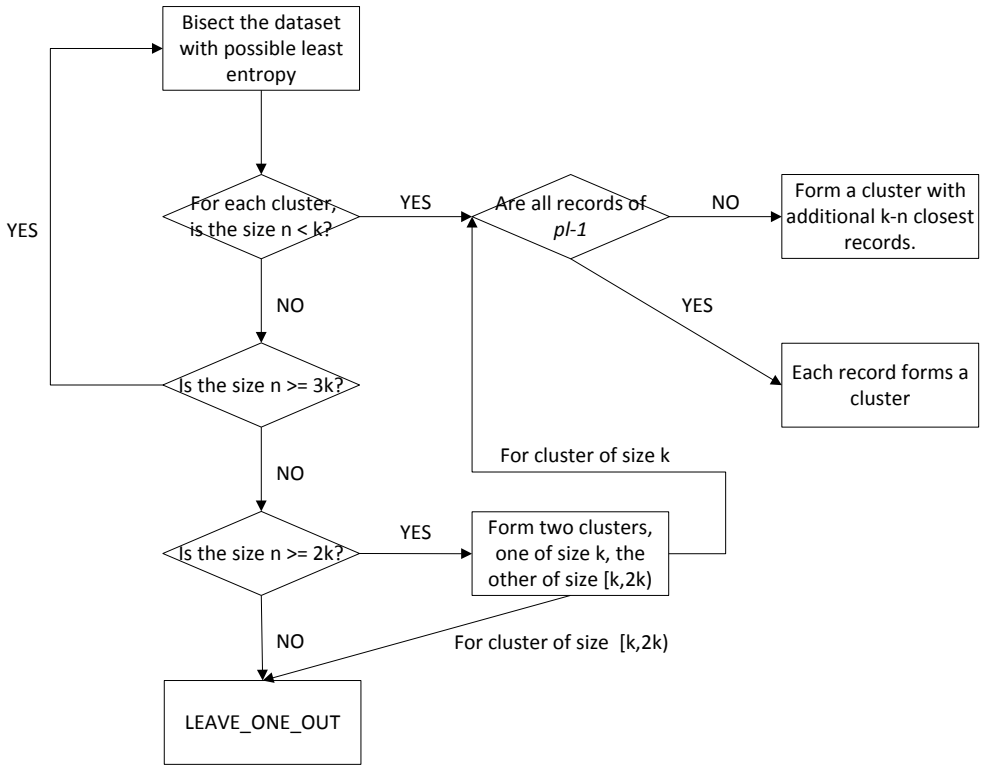

Figure 5.2 Flowchart for BOKG algorithm

We introduce two tool functions based on original BKG algorithm.

Function 1 [CHECK_ALL_ONE (input r: dataset, output is_all_one: boolean)].

1. **if** privacy levels of all the records in r are 1
2. each record forms a cluster;
3. return true;
4. **else** return false;
5. **end if**.

CHECK_ALL_ONE simply checks a set of records to see whether all records belong to privacy-level-one (PL-1) class, if so, no doubt it is better to form groups with each individual.

Function 2 [LEAVE_ONE_OUT (input r: dataset, input k: integer)].

1. **if** CHECK_ALL_ONE(r) = true
2. return;
3. **end if;**
4. **while** $n > k$
5. compute centroid $\bar{x}$ of r and standard deviation σ;

6. find the most distant privacy-levlel-1-record $\mathbf{x_{1r}}$ from $\bar{\mathbf{x}}$;
7. **if** rse $(\mathbf{x_{1r}}, \bar{\mathbf{x}}) \geq \sigma$ →rse is root squared error
8. $\mathbf{x_{1r}}$ forms an individual cluster itself;
9. n − −; **continue**;
10. **else** return;
11. **end if**;
12. **end while**;

For clusters of size $n \in [k,2k)$, LEAVE_ONE_OUT tries to separate PL-1 record out of cluster by evaluating contribution it made to improve the cluster quality with respect to root squared error. PL-1 records which maximize the RSE of original cluster are chosen to be separated.

ALGORITHM 2 [BISECTING_1-K-GATHER (r: dataset, k: integer)].

1. Compute the centroid $\bar{\mathbf{x}}$ of r.
2. Randomly pick one record, $\mathbf{c_L} \in$ r, and compute $\mathbf{c_R} = \bar{\mathbf{x}} - (\mathbf{c_L} - \bar{\mathbf{x}})$.
3. Bisect r into two clusters G_l and G_r according to the following rule:

$$\begin{cases} \mathbf{x_i} \in G_l \text{ if } \|\mathbf{x_i} - \mathbf{c_L}\| \leq \|\mathbf{x_i} - \mathbf{c_R}\| \\ \mathbf{x_i} \in G_r \text{ if } \|\mathbf{x_i} - \mathbf{c_L}\| > \|\mathbf{x_i} - \mathbf{c_R}\|' \end{cases}$$ Where $\|\cdot\|$ stands for specific similarity function.

4. Repeat step 2 and 3 *number_of_iterations* times, select clusters from bisection with lowest sse.
5. Count number of records in each cluster, without losing generality, assume $n_l \leq n_r$.
6. if $n_l \in (0,k)$,
   A. If CHECK_ALL_ONE(g_l) = false, do b. and c.
   B. Compute the centroid $\overline{x_L}$ of g_l.
   C. Form a cluster containing g_l and $k - n_l$ records closest to $\overline{x_L}$.
7. for either cluster G_i of $\{G_l, G_r\}$:
   A. If $n_i \in [3k,+\infty)$, call BISECTING_1-K-GATHER (G_i, k).
   B. If $n_i \in [2k,3k)$,
      I. Compute the centroid $\bar{\mathbf{x}}_\mathbf{i}$.
      II. Find the most distant record $\mathbf{x_r}$ from $\bar{\mathbf{x}}_\mathbf{i}$.
      III. Form a cluster G_{i0} containing $\mathbf{x_r}$ and $k - 1$ records closest to $\mathbf{x_r}$.
      IV. Form another cluster G_{i1} containing the rest of records.
      V. CHECK_ALL_ONE(G_{i0})
      VI. LEAVE_ONE_OUT (G_{i1}).
   C. If $n_i \in [k,2k)$, LEAVE_ONE_OUT (G_i).

It is much clearer to explain the algorithm with a concrete example. Figure shows an example of how BOKG works for the same dataset in

Figure **3.4 3-anonymity achieved by BKG for a set of 12 random points**, but with six points of PL-1, denoted by ○ , and the rest of points of PL-3, denoted by ●.

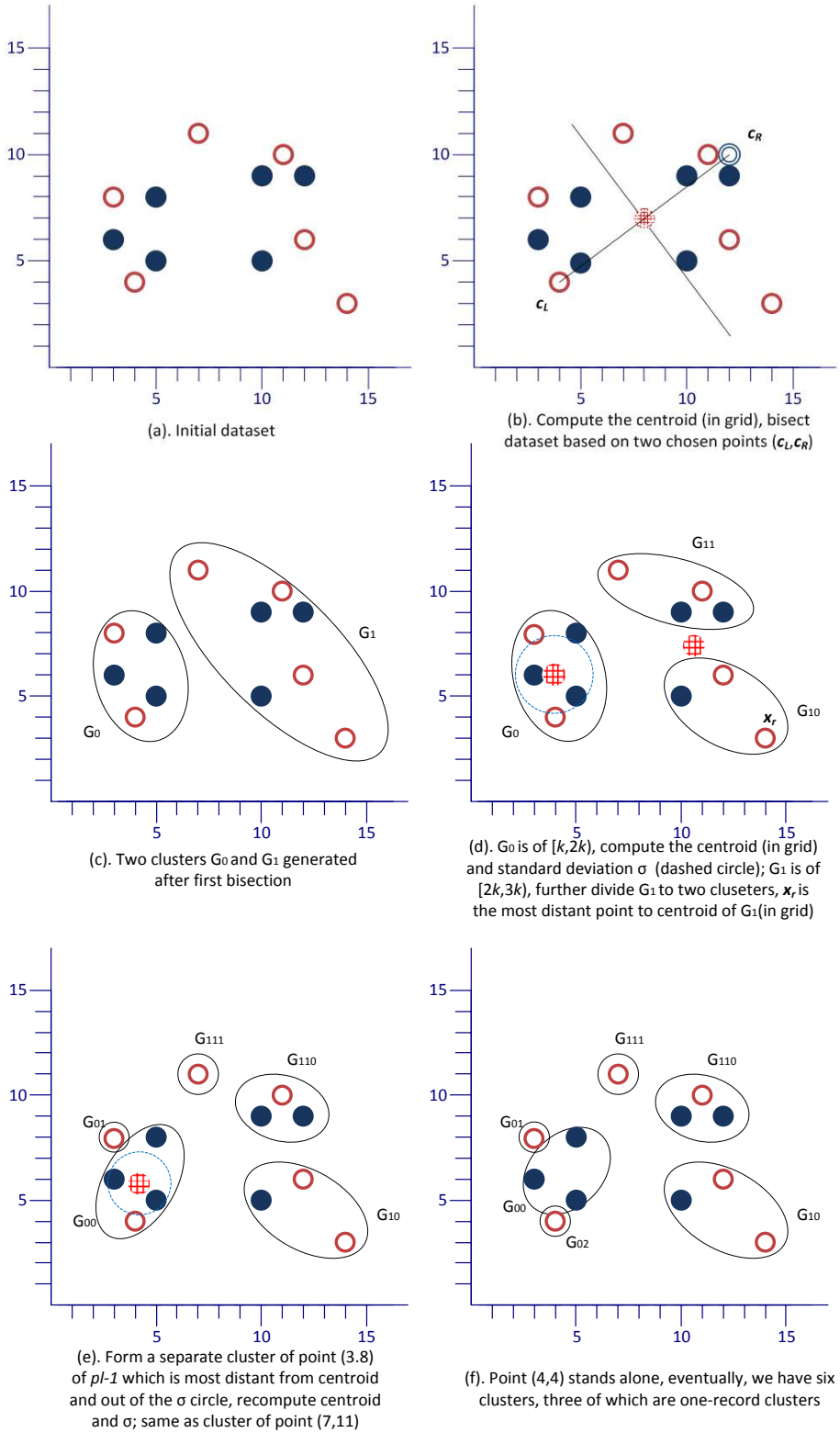

Figure 5.3 1-3-anonymity achieved by BOKG for a set of 12 random points

Look at this example, the first two steps are the same as BKG. In (d), since size of G_0 is bigger than k, we compute its centroid which is in grid and the standard deviation, in dashed circle, found the most distant point of PL-1, since this point is out of the circle, so, it means, by separating this point will improve the cluster quality, and then we recursively do this, until the size of cluster is equal to k, or no PL-1 point is greater than “standard deviation circle”, the same thing happened to G_1.Finally we got 6 clusters with three cluster of size one and three of size three.

6 Experimental Results for Customized K-Anonymity

Experiments on customized k-anonymity also includes two parts, one is the analysis of the distribution of users in groups of different sizes, on the purpose to verify the BOKG algorithm that how many PL-1 users actually can be separated to be clusters itself.

Second part is the “Proof of hypothesis”, to users of no privacy concern, and users with privacy concern, we evaluate how much utility we will loss respectively, in order to prove that users with lower privacy level do will get higher recommendation quality.

6.1 Distribution Analysis

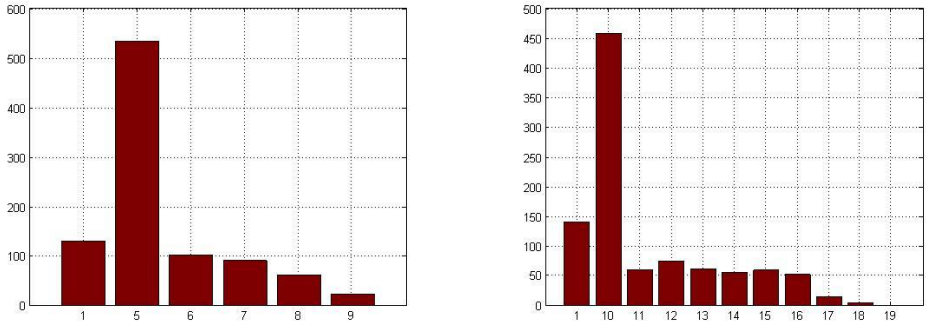

Figure 6.1 User distributions $k = 5$ (left plot), $k = 10$ (right plot)

We still used MovieLens database as our test database. We separated users into two categories, with half of the users are assigned privacy level 1 (471 users). Then BOKG was applied with $k = 5$ and $k = 10$, we calculated the number of users in groups of different sizes. For each case, we plot the result of the average numbers with ten tries.

The above figure shows the distribution. According to the experimental results, in both cases, about one-third of PL-1 users were separated to be clusters itself, which is pretty neat result. Actual number are 130 when $k = 5$ and about 140 when $k = 10$, notice that with the raise of k, the number of users who can be separated slightly increases.

6.2 Utility Loss Evaluation

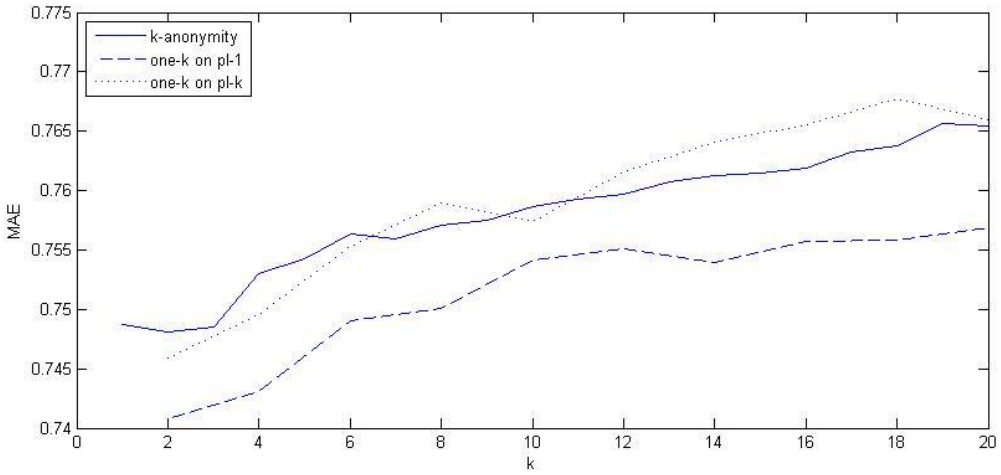

Figure 6.2 Utility loss between conventional and customized k-anonymity

And here is the final results, for k from 2-20, if the test records chosen from those PL-1 users, the line in dashed line, if the test records are chosen from those of privacy level k users, the plot is in dotted line. The solid line is the utility loss result for conventional k-anonymity, plotted here as a reference.

First we can see that on average users who require no privacy concern shows a great improvement of the prediction accuracy. They can get lower information loss and get better recommendations. This result proves our hypothesis.

The reason for that is hard to interpret fully, but we think the main reason is that since users who form cluster itself will not loss any information, and by separating those distant users out of the cluster, the original cluster quality improves, that is the users in the same cluster are more similar, and by anonymization, we can get much more accurate information, so PL-1 user can get more information now from anonymized users.

Second, other users with normal privacy concerns may get similar prediction accuracy as before, which is fine. Thus people who care about their privacy information can still get competitive accuracy as before.

And also, compared to conventional k-anonymity, the overall utility loss of customized k-anonymity outperforms.

7 Conclusion

The subject of this thesis is about privacy preserving data publishing for recommender system. We mainly focused on adapting k-anonymity to recommender system and also innovatively raised some new concept and algorithms.

Our main contributions are summarized as follows:

1. We propose a general scheme for PPDP of recommender system. It eliminates the high dimensionality and sparsity problem by a preprocess phase, and present a new k-gather heuristic, bisecting k-gather algorithm (BKG), which is a top-down, data-oriented approach. It forms groups of similar records by iteratively bisecting dataset with least entropy.

2. We advance the concept of "customized k-anonymity", which consists in anonymize database with records of different privacy concerns (required by user policy).

3. We study the basic case of customized k-anonymity, i.e. $\mathrm{k} \in \{1, \mathrm{c}\}$ and we further modify BKG to satisfy the basic case of customized k-anonymity, called bisecting 1-k-gather (BOKG).

We implemented our anonymization system under a JAVA environment. The presented solutions are supported by a series of experiments. According to the experimental results, it proves our k-gather heuristic (BKG) is efficient and stable.

Since we anonymize user records according to other users of similar preferences, thus the padded values are of high accuracy, when k a low number or the scale of database is huge, we even can expect some benefits from anonymization. Utility test also showed that we can anonymize data of recommender system, which is extremely sparse and large, with a really tiny compensation or even some benefits.

The new concept "customized k-anonymity", a more flexible privacy model, disclosed meaningful use cases in reality. Though we have only studied the basic case of customized k-anonymity, the result of the algorithm turns out to be neat and sufficient to prove our hypothesis. I.e. users with lower privacy level are likely to get better recommendation quality.

8 Future Work

There are several challenges awaiting future development of this anonymization system and also some experiments that need to be done but not due to the limit of time. Some challenges are common for both conventional k-anonymity while some are specific to customized k-anonymity.

This chapter will list several future works that are valuable to study. The scalability issue, accuracy of k-clustering algorithm, "how to produce a more general solution to all types of data", and "how to deal with more general customized k-anonymity problem" will be pointed out and discussed in the following sections.

8.1 Larger Database

Due to the time limit, we have only verified our anonymization system using MovieLens database. We need to scale it up to even larger database like Netflix, to test the scalability, to measure the time cost in a very large database, and the trend of information loss rate with the raise of the scale. Maybe it is more suitable when data getting larger, since there are only 943 users in MovieLens, so if k = 10, that means after anonymization, there are only about 9 "unique" users left. But with the raise of the size, we can expect to achieve stronger anonymization with respect to the raise of k, while still have many "unique" users.

8.2 Approximation Algorithm

Since we only need to apply anonymization once, that is before data being published, so here time requirement is not so strict for privacy preserving data publishing, there are some approximation k-gather algorithms (Aggarwal, et al., 2006), approximation algorithms are algorithms used to find approximate solutions to optimization problems, which require much higher time complexity, like $O(n^{30})$, but can get better clustering quality that improve the utility.

8.3 Non-numerical Data

One extension may be adapting the solution to solve non-numerical data. Now we are dealing with numerical database, think about if the database is not of numerical attribute, can our mechanism still applicable to be a more general solution?

Since SVD needs numerical data and k gather clustering also need numerical data. One possible solution in my mind is that for numerical database, we may add another preprocessing phase, in which transforms non-numerical data to numerical representation, and then normalize it. Then put the processed data to our system.

Non-numerical data can be transformed to numerical data in some way. For example, gender “male” and “female” can be transformed to 0 and 1 respectively. Worth to think about transformation rules for different types of data, but one may still need to think about how to weight different attribute, straight normalization may not be proper in some cases.

8.4 General Customized K-Anonymity

For customized k-anonymity, in the Chapter 5 and 6 we have just studied the basic case, that is separating users into two groups, one with no privacy concern, one with privacy concern.

This is actually the most popular case for customized k-anonymity, while in order to make customized k-anonymity more general, one future work will be supporting multiple privacy levels ,like [k_{min} , k_{max}]. E.g. $k \in \{1,5,10\}$.

It will be a big challenge to come up with modification to our existing BKG to support multiple privacy levels, or find another possible algorithm to achieve it.

Bibliography

[Online] // IMDb. The Internet Movie Database. - 2007. - http://www.imdb.com/.

Aggarwal AvCharu C. and Yu Philip S. Privacy-Preserving Data Mining: Models and Algorithms [Book]. - [s.l.] : Springer, 2008. - 1st edition. - ISBN: 978-0-387-70991-8.

Aggarwal C. C. and Procopiuc C. Fast Algorithm for Projected Clustering [Conference] // ACM SIGMOD Conference. - 1999.

Aggarwal C. C. and Yu P. S. A Condensation Approach to Privacy Preserving Data Mining [Conference] // EDBT Conference. - 2004.

Aggarwal C. C. and Yu P. S. Finding Generalized Projected Clusters in High Dimensional Spaces [Conference] // ACM SIGMOD Conference. - 2000.

Aggarwal C. C. On k-anonymity and the Curse of Dimensionality [Conference] // VLDB Conference. - 2005.

Aggarwal C. C. On Randomization, Public Information and the Curse of Dimensionality [Conference] // ICDE Conference. - 2007.

Aggarwal G. [et al.] Achieving Anonymity via Clustering [Conference] // ACM PODS Conference. - 2006.

Agrawal D. and Aggarwal C. C. On the Design and Quantification of Privacy-Preserving Data Mining Algorithms [Conference] // ACM PODS Conference. - 2002.

Agrawal R. [et al.] Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications [Conference] // ACM SIGMOD Conference. - 1998.

Agrawal R. and Srikant R. Privacy-Preserving Data Mining [Conference] // Proceedings of the ACM SIGMOD Conference. - 2000.

Agrawal R. and Srikant R. Privacy-Preserving Data Mining [Conference] // Proceedings of the ACM SIGMOD Conference. - 2000.

Bayardo R. J. and Agrawal R. Data Privacy through Optimal k-Anonymization [Conference] // Proceedings of the ICDE Conference. - 2005. - pp. 217-228.

Byun J. W. [et al.] Efficient k-Anonymization Using Clustering Techniques [Journal] // Lecture Notes in Computer Science / ed. Ramamohanarao K.. - [s.l.] : Springer, 2007. - Vol. 4443. - pp. 188-200.

Chang C. C. and Thompson B. Towards Publishing Recommendation Data with Predictive Anonymization [Conference] // ASIACCS. - 2010.

Chen K. and Liu L A Random Rotation Perturbation Approach to Privacy Preserving Data Classification [Conference] // ICDM Conference. - 2005.

Chen K. and Liu L. Towards Attack-Resilient geometric data perturbation [Conference] // SIAM Data Mining Conference. - 2007.

Ciriani V. [et al.] k-Anonymity [Book Section] // Advances in Information Security / book auth. Jajodia S. and Yu T.. - [s.l.] : Springer, 2007.

Domingo-Ferrer J. [et al.] Efficient Multivariate Data-Oriented Microaggregation [Journal] // VLDB Journal. - 2006. - Vol. 15. - pp. 355-369.

Domingo-Ferrer J. and Mateo-Sanz J. Practical Data-Oriented Microaggregation for Statistical Disclosure Control [Journal] // IEEE Transactions on Knowledge and Data Engineering. - 2002. - 1 : Vol. 14.

Domingo-Ferrer J. and Torra V. Ordinal, Continuous and Heterogeneous k-Anonymity through Microaggregation [Journal] // Data Mining and Knowledge Discovery. - 2005. - 2 : Vol. 11. - pp. 195-212.

Dwork C. Differential Privacy: A Survey of Results [Journal] // TAMC, Lecture Notes of Computer Science / ed. Agrawal M.. - Heidelberg : Springer, 2008. - Vol. 4978. - pp. 1-19.

Evfimievski A. [et al.] Privacy-Preserving Mining of Association Rules [Conference] // ACM KDD Conference. - 2002.

Fung Benjamin C. M. [et al.] Privacy-preserving data publishing: A survey of recent developments [Journal] // ACM Computing Surveys (CSUR). - June, 2010. - 4 : Vol. Volume 42.

Funk S. Netflix update: Try this at home [Online]. - 12 11, 2006. - http://sifter.org/~simon/journal/20061211.html.

Goil S., Nagesh H. and Choudhary A. MAFIA: Efficient and Scalable Subspace Clustering for Very Large Data Sets [Report] / Electrical and Computer Engineering ; Center for Parallel and Distributed Computing. - [s.l.] : Northwestern University, 1999. - CPDCTR-9906-010.

Hafner K. And if you liked the movie, a Netflix contest may reward you handsomely [Report] / New York Times. - [s.l.] : New York Times, 2006.

Kailing K., Kriegal H. P. and Kröger P. Density-Connected Subspace Clustering for High-Dimensional Data [Conference] // SIAM Conference. - 2004.

Kantarcioglu M. and Clifton C. Privacy-Preserving Distributed Mining of Association Rules on Horiaontally Partitioned Data [Journal] // IEEE TKDE Journal. - 2004. - 9 : Vol. 16.

Li N., Li T. and Venkatasubramanian S. t-Closeness: Privacy beyond k-Anonymity and l-Diversity [Conference] // ICDE Conference. - 2007.

Li N., Li T. and Venkatasubramanian S. t-Closeness: Privacy beyond k-Anonymity and l-Diversity [Conference] // ICDE Conference. - 2007.

Machanavajjhala A [et al.] l-Diversity: Privacy Beyond k-Anonymity [Conference] // ICDE Conference. - 2006.

McSherry F. and Mironov I. Differentially Private Recommender Systems: Building Privacy into the netflix Prize Contenders [Conference] // ACM KDD Conference. - 2009.

Meyerson A. and Williams R. On the Complexity of Optimal k-Anonymity [Conference] // ACM PODS Conference. - 2004.

Narayanan A. and Shmatikov V. Robust De-anonymization of Large Sparse Datasets (How to Break Anonymity of the Netflix Prize Dataset) [Report] / The University of Texas. - Austin : [s.n.], 2007.

Polat H. and Du W. SVD-based Collaborative Filtering with Privacy [Conference] // ACM SAC Symposium. - 2005.

Rizvi S. and Haritsa J. Maintaining Data Privacy in Association Rule Mining [Conference] // VLDB Conference. - 2002.

Samarati P. and Sweeney L. Generalizing Data to Provide Anonymity When Disclosing Information (Abstract) [Conference]. - Seattle, WA : [s.n.], 1998. - p. 188.

Samarati P. Protecting Respondents' Identities in Microdata Release [Journal] // IEEE Trans. Knowl. Data Eng. - 2001. - 6 : Vol. 13. - pp. 1010-1027.

Sen W. [et al.] Clustering Algorithm Based on Sparse Feature Vector for Interval-Scaled Variables [Conference] // ViCom Conference. - 2007.

Sen W. and Wei G. high Dimensional Data Clustering Algorithm Based on Sparse Feature Vector for Categorical Attributes [Conference] // LSIM Conference. - 2010.

Sen W., Gu S. and Gao X. High Dimensional Sparse Data Clustering Algorithm Based on Concept Feature Vector (CABOCFV) [Conference] // IEEE/SOLI Conference. - 2008.

Steinbach M., Ertöz L. and Kumar V. The Challenges of Clustering High Dimensional Data [Book Section] // New Vistas in Statistical Physicas - Applications in Econophysics, Bioinformatics, and Pattern Recognition / ed. Wille L. T.. - [s.l.] : Springer-Verlag, 2003.

Thompson Brian and Yao D. F. The Union-Split Algorithm and Cluster-Based Anonymization of Social Networks [Conference] // ASIACCS Conference. - Sydney : [s.n.], 2009.

Vaidya J. and Clifton C. Privacy-Preserving Association Rule Mining in Vertically Partitioned Databases [Conference] // ACM KDD Conference. - 2002.

Velcin J. and Ganascia J. G. Default Clustering with Conceptual Structures [Conference] // Journal on data semantics VIII. - 2007.

Velcin J. and Ganascia J. G. Default Clustering with from Sparse Data Sets [Conference] // ECSQARU Conference. - 2005.

Verykios V. S. [et al.] State-of-the-art in Privacy Preserving Data Mining [Conference] // ACM SIGMOD Record. - 2004. - Vol. 33.

Warner S. L. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias [Journal] // Journal of American Statistical Association. - March 1965. - 309 : Vol. 60. - pp. 63-69.

Wille R. Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts [Conference] // Ordered Sets. - 1982.

Wolff K. E. A First Course in Formal Concept Analysis [Conference] // StatSoft / ed. Faulbaum F.. - 1993.

Table of Figures

Figure 1.1 A Simple Model of PPDP 9

Figure 1.2 Matrix Representation of Recommender System 11

Figure 2.1 General process of randomization based PPDP 15

Figure 2.2 2-dimensional rotation perturbation 16

Figure 2.3 An example of 2-anonymity achieved by generalization/suppression 18

Figure 2.4 4-anonymity table achieved by generalization 19

Figure 2.5 A possible solution for 4-gather clustering 20

Figure 2.6 Clustering based 2-anonymity for a toy RS 21

Figure 2.7 Variable vs. fixed-sized groups 22

Figure 2.8 Grid based subspace clustering 23

Figure 2.9 Sparsity problem 24

Figure 3.1 General PPDP system components 26

Figure 3.2 User preference vector and movie feature vector 27

Figure 3.3 Flow chart for BKG algorithm 29

Figure 3.4 3-anonymity achieved by BKG for a set of 12 random points 30

Figure 3.5 The increment of entries with the raise of k 32

Figure 4.1 The performance of BKG of k=3, 5, 10 respectively 34

Figure 4.2 The performance comparison between BKG and BTM of k=3 35

Figure 4.3 Utility evaluation model 36

Figure 4.4 The increment of entries with the raise of k 37

Figure 5.1 The anonymization schema with user policy 39

Figure 5.2 Flowchart for BOKG algorithm 40

Figure 6.1 User distributions $k = 5$ (left plot), $k = 10$ (right plot) 45

Figure 6.2 Utility loss between conventional and customized k-anonymity 46

Abbreviations

BKG	Bisecting K-Gather
BOKG	Bisecting One-K-Gather
BSP	Binary Space Partitioning
CF	Collaborative Filtering
ICA	Independent Component Analysis
MAE	Mean Absolute Error
PPDM	Privacy Preserving Data Mining
PPDP	Privacy Preserving Data Publishing
PL	Privacy Level
RSE	Relative Standard Error
SSE	Sum of Squared Error
SVD	Singular Value Decomposition