IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 4 (Jul. - Aug. 2013), PP 63-68
www.iosrjournals.org

Protecting Attribute Disclosure for High Dimensionality and Preserving Publishing of Microdata

[1]Shaik. Mahammad Rafi, [2] P. Venkata Ramanaiah M.Tech,
[2]Assistant Professors in CSE Department , GCET ,Kadapa,YSR(D.t).
[1] PG (M.Tech) Student in CSE Department, Global College of Engineering and Technology ,Kadapa, YSR(D.t).

Abstract: *Generalization and Bucketization, have been designed for privacy preserving microdata publishing. Recent work has shown that generalization loses considerable amount of information, especially for high-dimensional data. Bucketization, on the other hand, does not prevent membership disclosure and does not apply for data that do not have a clear separation between quasi- identifying attributes and sensitive attributes.*

In this paper, we present a novel technique called slicing, which partitions the data both horizontally and vertically. We show that slicing preserves better data utility than gen- eralization and can be used for membership disclosure protection. Another important advantage of slicing is that it can handle high-dimensional data. We show how slicing can be used for attribute disclosure protection and develop an ef- ficient algorithm for computing the sliced data that obey the ℓ-diversity requirement.Our workload experiments confirm that slicing preserves better utility than generalization and is more effective than bucketization in workloads involving the sensitive attribute. Our experiments also demonstrate that slicing can be used to prevent membership disclosure.

I. Introduction

Privacy-preserving publishing of microdata has been stud- ied extensively in recent years. Microdata contains records each of which contains information about an individual en- tity, such as a person, a household, or an organization. Several microdata anonymization techniques have been pro- posed. The most popular ones are generalization [29, 31] for k-anonymity [31] and bucketization [35, 25, 16] for ℓ- diversity [23]. In both approaches, attributes are partitioned into three categories: (1) some attributes are identifiers that can uniquely identify an individual, such as Name or Social Security Number; (2) some attributes are Quasi-Identifiers (QI), which the adversary may already know (possibly from other publicly-available databases) and which, when taken together, can potentially identify an individual, e.g., Birth-date, Sex, and Zipcode ; (3) some attributes are Sensitive Attributes (SAs), which are unknown to the adversary and are considered sensitive, such as Disease and Salary.

In both generalization and bucketization, one first removes identifiers from the data and then partitions tuples into buckets. The two techniques differ in the next step. Gener- alization transforms the QI-values in each bucket into “less specific but semantically consistent” values so that tuples in the same bucket cannot be distinguished by their QI val- ues. In bucketization, one separates the SAs from the QIs by randomly permuting the SA values in each bucket. The anonymized data consists of a set of buckets with permuted sensitive attribute values.

II. Slicing

It has been shown [1, 15, 35] that generalization for k- anonymity losses considerable amount of information, espe- cially for high-dimensional data. This is due to the following three reasons. First, generalization for k-anonymity suffers from the curse of dimensionality. In order for generalization to be effective, records in the same bucket must be close to each other so that generalizing the records would not lose too much information. However, in high-dimensional data, most data points have similar distances with each other, forcing a great amount of generalization to satisfy k-anonymity even for relative small k’s. Second, in order to perform data analysis or data mining tasks on the generalized table, the data analyst has to make the uniform distribution assump- tion that every value in a generalized interval/set is equally possible, as no other distribution assumption can be justi- fied. This significantly reduces the data utility of the gen- eralized data. Third, because each attribute is generalized separately, correlations between different attributes are lost. In order to study attribute correlations on the generalized table, the data analyst has to assume that every possible combination of attribute values is equally possible. This is an inherent problem of generalization that prevents effective analysis of attribute correlations.

While bucketization [35, 25, 16] has better data utility than generalization, it has several limitations. First, buck- etization does not prevent membership disclosure [27]. Be- cause bucketization publishes the QI values in their original forms, an adversary can find out whether an individual has a record in the published data or not. As shown in [31], can be inferred from the bucketized table. Second, buck- etization requires a clear separation between QIs and SAs. However, in many datasets, it is unclear which attributes are QIs and which are SAs. Third, by separating the sensitive attribute from the QI attributes, bucketization breaks the attribute correlations between the QIs and the SAs. In this paper, we introduce a novel data anonymization technique called slicing to improve the current state of the art.

Slicing partitions the dataset both vertically and hori- zontally. Vertical partitioning is done by grouping attributes into columns based on the correlations among the attributes. Each column contains a subset of attributes that are highly correlated. Horizontal partitioning is done by grouping tu- ples into buckets. Finally, within each bucket, values in each column are randomly permutated (or sorted) to break the linking between different columns.

The basic idea of slicing is to break the association cross columns, but to preserve the association within each col- umn. This reduces the dimensionality of the data and pre- serves better utility than generalization and bucketization. Slicing preserves utility because it groups highly-correlated attributes together, and preserves the correlations between such attributes. Slicing protects privacy because it breaks the associations between uncorrelated attributes, which are infrequent and thus identifying. Note that when the dataset contains QIs and one SA, bucketization has to break their correlation; slicing, on the other hand, can group some QI at- tributes with the SA, preserving attribute correlations with the sensitive attribute.

III. Formalization

Age	Sex	Zipcode	Disease
22	M	47906	dyspepsia
22	F	47906	flu
33	F	47905	flu
52	F	47905	bronchitis
54	M	47302	flu
60	M	47302	dyspepsia
60	M	47304	dyspepsia
64	F	47304	gastritis

(a) The original table

Age	Sex	Zipcode	Disease
[20-52]	*	4790*	dyspepsia
[20-52]	*	4790*	flu
[20-52]	*	4790*	flu
[20-52]	*	4790*	bronchitis
[54-64]	*	4730*	flu
[54-64]	*	4730*	dyspepsia
[54-64]	*	4730*	dyspepsia
[54-64]	*	4730*	gastritis

(b) The generalized table

Age	Sex	Zipcode	Disease
22	M	47906	flu
22	F	47906	dyspepsia
33	F	47905	bronchitis
52	F	47905	flu
54	M	47302	gastritis
60	M	47302	flu
60	M	47304	dyspepsia
64	F	47304	dyspepsia

(c) The bucketized table

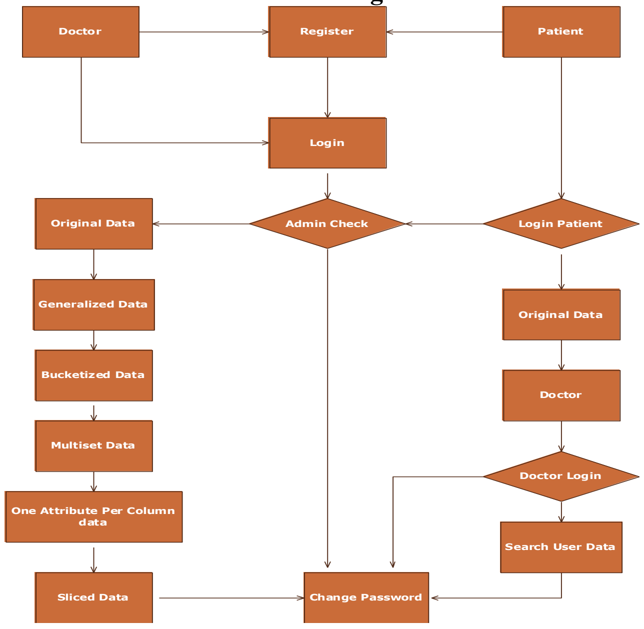
Age	Sex	Zipcode	Disease
22:2,33:1,52:1	M:1,F:3	47905:2,47906:2	dysp.
22:2,33:1,52:1	M:1,F:3	47905:2,47906:2	flu
22:2,33:1,52:1	M:1,F:3	47905:2,47906:2	flu
22:2,33:1,52:1	M:1,F:3	47905:2,47906:2	bron.
54:1,60:2,64:1	M:3,F:1	47302:2,47304:2	flu
54:1,60:2,64:1	M:3,F:1	47302:2,47304:2	dysp.
54:1,60:2,64:1	M:3,F:1	47302:2,47304:2	dysp.
54:1,60:2,64:1	M:3,F:1	47302:2,47304:2	gast.

(d) Multiset-based generalization

Age	Sex	Zipcode	Disease
22	F	47906	flu
22	M	47905	flu
33	F	47906	dysp.
52	F	47905	bron.
54	M	47302	dysp.
60	F	47304	gast.
60	M	47302	dysp.
64	M	47304	flu

(e) One-attribute-per-column slicing

Data Flow Diagram:



IV. Slicing Algorithms

4.1 Attribute Partitioning

Our algorithm partitions attributes so that highly- correlated attributes are in the same column. This is good for both utility and privacy. In terms of data utility, group- ing highly-correlated attributes preserves the correlations among those attributes. In terms of privacy, the association of uncorrelated attributes presents higher identification risks than the association of highly-correlated attributes because the association of uncorrelated attribute values is much less frequent and thus more identifiable. Therefore, it is better to break the associations between uncorrelated attributes, in order to protect privacy.

In this phase, we first compute the correlations between pairs of attributes and then cluster attributes based on their correlations.

4.2 Column Generalization

In the second phase, tuples are generalized to satisfy some minimal frequency requirement. We want to point out that column generalization is not an indispensable phase in Algorithm tuple-partition(T , ℓ) provides the same level of privacy protection as generaliza- tion, with respect to attribute disclosure.

```
1. Q = {T }; SB = ∅.
2. while Q is not empty
3. remove the first bucket B from Q; Q = Q − {B}.
4. split B into two buckets B1 and B2 , as in Mondrian.
5. if diversity-check(T , Q ∪ {B1 , B2 } ∪ SB , ℓ)
6. Q = Q ∪ {B1 , B2 }.
7. else SB = SB ∪ {B}.
8. return SB .
```

Figure 1: The tuple-partition algorithm

```
Algorithm diversity-check(T , T ∗ , ℓ)
1. for each tuple t ∈ T , L[t] = ∅.
2. for each bucket B in T ∗
3. record f (v) for each column value v in bucket B.
4. for each tuple t ∈ T
5. calculate p(t, B) and find D(t, B).
6. L[t] = L[t] ∪ {hp(t, B), D(t, B)i}.
7. for each tuple t ∈ T
8. calculate p(t, s) for each s based on L[t].
9. if p(t, s) ≥ 1/ℓ, return false.
10. return true.
```

Figure 2: The diversity-check algorithm

4.3 Tuple Partitioning

In the tuple partitioning phase, tuples are partitioned into buckets. We modify the Mondrian [17] algorithm for tuple partition. Unlike Mondrian k-anonymity, no generalization is applied to the tuples; we use Mondrian for the purpose of partitioning tuples into buckets.

Figure 1 gives the description of the tuple-partition algo- rithm. The algorithm maintains two data structures: (1) a queue of buckets Q and (2) a set of sliced buckets SB . Initially, Q contains only one bucket which includes all tu-

ples and SB is empty (line 1). In each iteration (line 2 to line 7), the algorithm removes a bucket from Q and splits the bucket into two buckets (the split criteria is described in Mondrian [17]). If the sliced table after the split satisfies ℓ-diversity (line 5), then the algorithm puts the two buckets at the end of the queue Q (for more splits, line 6). Other- wise, we cannot split the bucket anymore and the algorithm puts the bucket into SB (line 7). When Q becomes empty, we have computed the sliced table. The set of sliced buckets is SB (line 8).

The main part of the tuple-partition algorithm is to check whether a sliced table satisfies ℓ-diversity (line 5). Figure 2 gives a description of the diversity-check algorithm. For each tuple t, the algorithm maintains a list of statistics L[t] about t's matching buckets. Each element in the list L[t] contains statistics about one matching bucket B: the matching prob- ability p(t, B) and the distribution of candidate sensitive values D(t, B).

The algorithm first takes one scan of each bucket B (line 2 to line 3) to record the frequency f (v) of each column value v in bucket B. Then the algorithm takes one scan of each tuple t in the table T (line 4 to line 6) to find out all tuples that match B and record their matching probability p(t, B) and the distribution of candidate sensitive values D(t, B), which are added to the list L[t] (line 6). At the end of line
6, we have obtained, for each tuple t, the list of statistics L[t] about its matching buckets. A final scan of the tuples in T will compute the p(t, s) values based on the law of total probability described in Section 3.2. Specifically,
The sliced table is ℓ-diverse iff for all sensitive value s, $p(t, s) \leq 1/\ell$ (line 7 to line 10).

We now analyze the time complexity of the tuple-partition algorithm. The time complexity of Mondrian [17] or kd- tree [10] is O(n log n) because at each level of the kd-tree, the whole dataset need to be scanned which takes O(n) time and the height of the tree is O(log n). In our modification, each level takes O(n2) time because of the diversity-check algorithm (note that the number of buckets is at most n). The total time complexity is therefore O(n2 log n).

V. Membership Disclosure Pro- Tection

Let us first examine how an adversary can infer member- ship information from bucketization. Because bucketization releases the QI values in their original form and most indi- viduals can be uniquely identified using the QI values, the adversary can simply determine the membership of an in- dividual in the original data by examining the frequency of the QI values in the bucketized data. Specifically, if the fre- quency is 0, the adversary knows for sure that the individual is not in the data. If the frequency is greater than 0, the adversary knows with high confidence that the individual is in the data, because this matching tuple must belong to that individual as almost no other individual has the same QI values.
The above reasoning suggests that in order to pro- tect membership information, it is required that, in the anonymized data, a tuple in the original data should havea similar frequency as a tuple that is not in the original
Data.

VI. Experiments

We conduct two experiments. In the first experiment, we evaluate the effectiveness of slicing in preserving data utility and protecting against attribute disclosure, as compared to generalization and bucketization. To allow direct compari- son, we use the Mondrian algorithm [17] and ℓ-diversity for all three anonymization techniques: generalization, bucke- tization, and slicing. This experiment demonstrates that: (1) slicing preserves better data utility than generalization; (2) slicing is more effective than bucketization in workloads involving the sensitive attribute; and (3) the sliced table can be computed efficiently. Results for this experiment are presented in Section 6.2.

In the second experiment, we show the effectiveness of slicing in membership disclosure protection. For this pur- pose, we count the number of fake tuples in the sliced data. We also compare the number of matching buckets for origi-nal tuples and that for fake tuples. Our experiment results show that bucketization does not prevent membership dis- closure as almost every tuple is uniquely identifiable in the bucketized data. Slicing provides better protection against membership disclosure: (1) the number of fake tuples in the sliced data is very large, as compared to the number of orig- inal tuples and (2) the number of matching buckets for fake

	Attribute	Type	values
1	Age	Continuous	74
2	Workclass	Categorical	8
3	Final-Weight	Continuous	NA
4	Education	Categorical	16
5	Education-Num	Continuous	16
6	Marital-Status	Categorical	7
7	Occupation	Categorical	14
8	Relationship	Categorical	6
9	Race	Categorical	5
10	Sex	Categorical	2
11	Capital-Gain	Continuous	NA
12	Capital-Loss	Continuous	NA
13	Hours-Per-Week	Continuous	NA
14	Country	Categorical	41
15	Salary	Categorical	2

Table 2: Description of the Adult dataset

tuples and that for original tuples are close enough, which makes it difficult for the adversary to distinguish fake tu- ples from original tuples. Results for this experiment are presented in Section 6.3

Experimental Data. We use the Adult dataset from the UC Irvine machine learning repository [2], which is com- prised of data collected from the US census. The dataset is described in Table 2.

Tuples with missing values are elimi- nated and there are 45222 valid tuples in total. The adult dataset contains 15 attributes in total.

In our experiments, we obtain two datasets from the Adult dataset. The first dataset is the "OCC-7" dataset, which includes 7 attributes: QI = {Age, W orkclass, Education, M arital-Status, Race, Sex} and S = Occupation. The second dataset is the "OCC-15" dataset, which includes all 15 attributes and the sensitive attribute is S = Occupation. In the "OCC-7" dataset, the attribute that has the closest correlation with the sensitive attribute Occupation is Gen- der, with the next closest attribute being Education. In the "OCC-15" dataset, the closest attribute is also Gender but
the next closest attribute is Salary.

VII. Discussions And Future Work

This paper presents a new approach called slicing to privacy-preserving microdata publishing. Slicing overcomes the limitations of generalization and bucketization and pre- serves better utility while protecting against privacy threats. We illustrate how to use slicing to prevent attribute disclo- sure and membership disclosure. Our experiments show that slicing preserves better data utility than generalization and is more effective than bucketization in workloads involving the sensitive attribute.

The general methodology proposed by this work is that: before anonymizing the data, one can analyze the data char- acteristics and use these characteristics in data anonymiza- tion. The rationale is that one can design better data anonymization techniques when we know the data better. In [21], we show that attribute correlations can be used for privacy attacks.

This work motivates several directions for future research. First, in this paper, we consider slicing where each attribute is in exactly one column. An extension is the notion of over- lapping slicing, which duplicates an attribute in more than one columns. This releases more attribute correlations. For example, in Table 1(f), one could choose to include the Dis- ease attribute also in the first column. That is, the two columns are {Age, Sex, Disease} and {Z ipcode, Disease}. This could provide better data utility, but the privacy im- plications need to be carefully studied and understood. It is interesting to study the tradeoff between privacy and util-ity [22].

Second, we plan to study membership disclosure protec- tion in more details. Our experiments show that random grouping is not very effective. We plan to design more effec- tive tuple grouping algorithms.

Third, slicing is a promising technique for handling high- dimensional data. By partitioning attributes into columns, we protect privacy by breaking the association of uncorrelated attributes and preserve data utility by preserving the association between highly-correlated attributes. For example, slicing can be used for anonymizing transaction databases, which has been studied recently in [32, 37, 26].

Finally, while a number of anonymization techniques have been designed, it remains an open problem on how to use the anonymized data. In our experiments, we randomly gen- erate the associations between column values of a bucket. This may lose data utility. Another direction to design data mining tasks using the anonymized data [13] computed by various anonymization techniques.

References

[1] C. Aggarwal. On k-anonymity and the curse of dimensionality. In VLDB, pages 901–909, 2005.

[2] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[3] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the sulq framework. In PODS, pages 128–138, 2005.

[4] J. Brickell and V. Shmatikov. The cost of privacy: destruction of data-mining utility in anonymized data publishing. In KDD, pages 70–78, 2008.

[5] B.-C. Chen, R. Ramakrishnan, and K. LeFevre.Privacy skyline: Privacy with multidimensional adversarial knowledge. In VLDB, pages 770–781, 2007.

[6] H. Cramt'er. Mathematical Methods of Statistics.Princeton, 1948.

AUTHORS PROFILE

Mr. Shaik.Mahammad Rafi –He was born in Rajampet, Kadapa, A.P, India in 1990.He is studying M.Tech in the department of Computer Science and Engineering at **Global College of Engineering and technology**, Kadapa. He has done Bachelor's of Technology from JNTUA University in the year 2011 in Information Technology.

Mr. P.Venkata Ramanaiah –He was born in Khajipet,Kadapa,A.P,India. He is Master of Technology in Computer Science and Engineering at Madanapalle Institute of Technology and Science from JNTUA University in the year 2012. He has given guidance to many students in their thesis work of M.Tech. He has also contributed in the research work on Data Mining with his papers. He was presently working as Assistant. Professor in **Global College of Engineering and technology**,GCET, kadapa.