International Journal of Research Studies in Science, Engineering and Technology
Volume 1, Issue 8, November 2014, PP 55-63
ISSN 2349-4751 (Print) & ISSN 2349-476X (Online)

A Novel Approach to Privacy Preserving Data Publishing Using Slicing Technique

CH. Srikanth Reddy [1], MD.John Saida [2]

[1] PG Scholar, Eswar College of Engineering, Narasaraopet, Guntur.
[2] Asst. Professor, Eswar College of Engineering, Narasaraopet, Guntur.

Abstract: *Several anonymization techniques, such as 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 generalization 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 efficient 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.*

Keywords: *anonymization techniques, Bucketization, zipcode, k-medoid method, slicing, attribute partitioning, column generalization, and tuple partitioning.*

1. Introduction

There are various situations in which a person might choose to withhold their identity. Acts of charity have been performed anonymously when benefactors do not wish to be acknowledged. A person who feels threatened might attempt to mitigate that threat through anonymity. In certain situations, it is illegal to remain anonymous. In the United States, 24 states have “Stop and identify” statutes that requires persons detained to self-identify when requested by a law enforcement officer. In recent years, due to increase in ability to store personal data about users and the increasing sophistication of data mining algorithms to leverage this information the problem of privacy preserving data mining has become more important. A number of anonymization techniques have been researched in order to perform privacy-preserving data mining. Data anonymization technique for privacy-preserving data publishing has received a lot of attention in recent years. Detailed data (also called as microdata) contains information about a person, a household or an organization. Most popular anonymization techniques are neralization and Bucketization. There are number of attributes in each record which can be categorized as 1) Identifiers such as Name or Social Security Number are the attributes that can be uniquely identify the individuals. 2) some attributes may be Sensitive Attributes(SAs) such as disease and salary and 3) some may be Quasi- Identifiers (QI) such as zipcode, age, and sex whose values, when taken together, can potentially identify an individual. The basic idea of slicing is to break the association cross columns, but to preserve the association within each column. 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.

2. Background

Anonymity is the condition of having one’s name or identity unknown or concealed. It serves valuable social purposes and empowers individuals as against institutions by limiting surveillance, but it is also used by wrong doers to hide their actions or avoid accountability the ability to allow anonymous access to services, which avoid tracking of user's personal information and user behavior such as user location, frequency of a service usage, and so on. If someone sends a file, there may be information on the file that leaves a trail to the sender. The sender's information may be traced from the data logged after the file is sent.

A. Anonymity vs. Security

Anonymity is a very powerful technique for protecting privacy. The decentralized and stateless design of the Internet is particularly suitable for anonymous behavior. Although anonymous actions can ensure privacy, they should not be used as the sole means for ensuring privacy as they also allow for harmful activities, such as spamming, slander, and harmful attacks without fear of reprisal. Security dictates that one should be able to detect and catch individuals conducting illegal behavior, such as hacking, conspiring for terrorist acts, and conducting fraud. Legitimate needs for privacy should be allowed, but the ability to conduct harmful anonymous behavior without responsibility and repercussions in the name of privacy should not.

B. Anonymity vs. Privacy

Privacy and anonymity are not the same. The distinction between privacy and anonymity is clearly seen in an information technology context. Privacy corresponds to being able to send an encrypted e-mail to another recipient. Anonymity corresponds to being able to send the contents of the e-mail in plain, easily readable form but without any information that enables a reader of the message to identify the person who wrote it. Privacy is important when the contents of a message are at issue, whereas anonymity is important when the identity of the author of a message is at issue.

3. Existing System

First, many existing clustering algorithms (e.g., k- means) requires the calculation of the “centroids”. But there is no notion of“centroids”in our setting where each attribute forms a data point in the clustering space. Second, k-medoid method is very robust to the existence of outliers (i.e., data points that are very far away from the rest of data points). Third, the order in which the data points are examined does not affect the clusters computed from the k-medoid method.

Disadvantages

1. Existing anonymization algorithms can be used for column generalization, e.g.,Mondrian . The algorithms can be applied on the subtable containing only attributes in one column to ensure the anonymity requirement.
2. Existing data analysis (e.g., query answering) methods can be easily used on the sliced data.
3. Existing privacy measures for membership disclosure protection include differential privacy and presence.

4. Proposed System

We present a novel technique called slicing, which partitions the data both horizontally and vertically. We show that slicing preserves better data utility than generalization 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 efficient 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.

Advantages

1. We introduce a novel data anonymization technique called slicing to improve the current state of the art.
2. We show that slicing can be effectively used for preventing attribute disclosure, based on the privacy requirement of ℓ-diversity.
3. We develop an efficient algorithm for computing the sliced table that satisfies ℓ-diversity. Our algorithm partitions attributes into columns, applies column generalization, and partitions tuples into buckets. Attributes that are highly-correlated are in the same column.

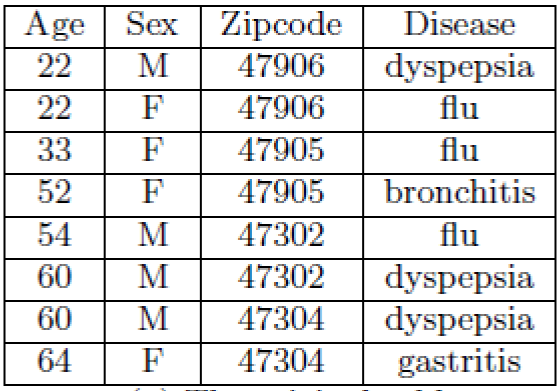
4. We conduct extensive workload experiments. Our results confirm that slicing preserves much better data utility than generalization. In workloads involving the sensitive attribute, slicing is also more effective than bucketization. In some classification experiments, slicing shows better performance than using the original data (which may overfit the model). Our experiments also show the limitations of bucketization in membership disclosure protection and slicing remedies these limitations.

5. MODULE DESCRIPTION

1. Original Data
2. Generalized Data
3. Bucketized Data
4. Multiset based Generalization Data
5. One-attribute-per-Column Slicing Data
6. Sliced Data

5.1 Original Data

We conduct extensive workload experiments. Our results confirm that slicing preserves much better data utility than generalization. In workloads involving the sensitive attribute, slicing is also more effective than bucketization. In some classification experiments, slicing shows better performance than using the original data.

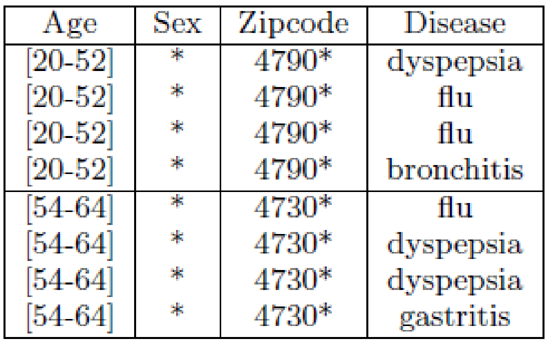


Age	Sex	Zipcode	Disease
22	M	47906	dyspepsia
22	F	47906	flu
33	F	47905	flu
52	F	47905	bronchitis
54	M	47302	flu
60	M	47302	dyspepsia
60	M	47304	dyspepsia
64	F	47304	gastritis

(a) The original table

5.2 Generalized Data

Generalized Data, in order to perform data analysis or data mining tasks on the generalized table, the data analyst has to make the uniform distribution assumption that every value in a generalized interval/set is equally possible, as no other distribution assumption can be justified. This significantly reduces the data utility of the generalized data.



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

5.3 Bucketized Data

We show the effectiveness of slicing in membership disclosure protection. For this purpose, we count the number of fake tuples in the sliced data. We also compare the number of matching buckets for original tuples and that for fake tuples. Our experiment results show that bucketization does not prevent membership disclosure as almost every tuple is uniquely identifiable in the bucketized data.

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

5.4 Multiset-Based Generalization Data

We observe that this multiset-based generalization is equivalent to a trivial slicing scheme where each column contains exactly one attribute, because both approaches preserve the exact values in each attribute but break the association between them within one bucket.

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

5.5 One-attribute-per-Column Slicing Data

We observe that while one-attribute-per-column slicing preserves attribute distributional information, it does not preserve attribute correlation, because each attribute is in its own column. In slicing, one groups correlated attributes together in one column and preserves their correlation. For example, in the sliced table shown in Table correlations between Age and Sex and correlations between Zipcode and Disease are preserved. In fact, the sliced table encodes the same amount of information as the original data with regard to correlations between attributes in the same column.

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

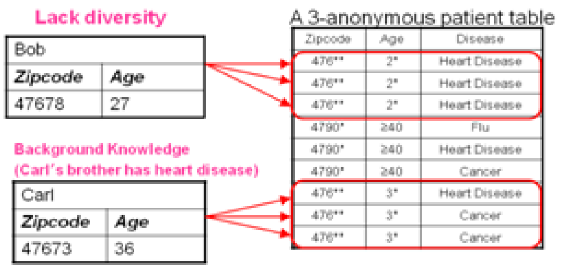
5.6 Sliced Data

Another important advantage of slicing is its ability to handle high-dimensional data. By partitioning attributes into columns, slicing reduces the dimensionality of the data. Each column of the table can be viewed as a sub-table with a lower dimensionality. Slicing is also different from the approach of publishing multiple independent sub-tables in that these sub-tables are linked by the buckets in slicing.

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

(f) The sliced table

System Architecture


Lack diversity
Background Knowledge
(Carl's brother has heart disease)
A 3-anonymous patient table

Bob	
Zipcode	*Age*
47678	27

Carl	
Zipcode	*Age*
47673	36

Zipcode	Age	Disease
476**	2*	Heart Disease
476**	2*	Heart Disease
476**	2*	Heart Disease
4790*	≥40	Flu
4790*	≥40	Heart Disease
4790*	≥40	Cancer
476**	3*	Heart Disease
476**	3*	Cancer
476**	3*	Cancer

6. Algorithms Used

We now present an efficient slicing. Our algorithm consists of three phases: attribute partitioning, column generalization, and tuple partitioning. We now describe the three phases.

6.1 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, grouping 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. An attribute partition consists of several subsets of A, such that each attribute belongs to exactly one subset. Each subset of attributes is called a column. Specifically, let there be columnsC1, C2, . . . , Cc, then and for any $1 \leq i1 \neq i2 \leq c$, $Ci1 \cap Ci2$ = For simplicity of discussion, consider only one sensitive attribute S. If the data contain multiple sensitive attributes, one can either consider them separately or consider their joint distribution. Exactly one of the c columns contains S. Without loss of generality, let the column that contains S be the last column Cc. This column is also called the *sensitive column*. All other columns {C1, C2,…..Cc- 1} contain only QI attributes. 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, grouping 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 attributes 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, first compute the correlations between pairs of attributes and then cluster attributes based on their correlations.

6.2 Measures of Correlation

Two widely used measures of association are Pearson correlation coefficient and mean square contingency coefficient. Pearson correlation coefficient is used for measuring correlations between two continuous attributes while mean-square contingency coefficient is a chi-square measure of correlation between two categorical attributes. Choose to use the mean-square contingency coefficient because most of our attributes are categorical. Given two attributes A1 and A2 with domains fv11;

v12; . . . ; v1d1g and fv21; v22; . . . ; v2d2g, respectively. Their domain sizes are thus d1 and d2, respectively. The mean-square contingency coefficient between A1 and A2 is defined as

$$\phi^2(A_1,A_2) = \frac{1}{\min\{d_1,d_2\}-1}\sum_{i=1}^{d_1}\sum_{j=1}^{d_2}\frac{(f_{ij}-f_{i\cdot}f_{\cdot j})^2}{f_{i\cdot}f_{\cdot j}}.$$

Here, *fi* and *fj* are the fraction of occurrences of v1i andv2j in the data, respectively. , *fi* and *fj* is the fraction of co-occurrences of v1i and v2j in the data.

6.3 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 our algorithm bucketization provides the same level of privacy protection as generalization, with respect to attribute disclosure. Although column generalization is not a required phase, it can be useful in several aspects. First, column generalization may be required for identity/membership disclosure protection. If a column value is unique in a column (i.e., the column value appears only once in the column), a tuple with this unique column value can only have one matching bucket. This is not good for privacy protection, as in the case of generalization/bucketization where each tuple can belong to only one equivalence-class/bucket. The main problem is that this unique column value can be identifying. In this case, it would be useful to apply column generalization to ensure that each column value appears with at least some frequency. Second, when column generalization is applied, to achieve the same level of privacy against attribute disclosure, bucket sizes can be smaller. While column generalization may result in information loss, smaller bucket-sizes allow better data utility. Therefore, there is a trade-off between column generalization and tuple partitioning. The trade-off between column generalization and tuple partitioning is the subject of future work. Existing anonymization algorithms can be used for column generalization, e.g., Mondrian. The algorithms can be applied on the subtable containing only attributes in one column to ensure the anonymity requirement.

6.4 Tuple Partitioning

In the tuple partitioning phase, tuples are partitioned into buckets, no generalization is applied to the tuples. The algorithm maintains two data structures 1) a queue of buckets Q 2) a set of sliced buckets SB. Initially, Q contains only one bucket which includes all tuples and SB is empty. For each iteration, the algorithm removes a bucket from Q and splits the bucket into two buckets. If the sliced table after the split satisfies l-diversity, then the algorithm puts the two buckets at the end of the queue Q. Otherwise, we cannot split the bucket anymore and the algorithm puts the bucket into SB. When Q becomes empty, we have computed the sliced table. The set of sliced buckets is SB. The main part of the tuple-partition algorithm is to check whether a sliced table satisfies '*l*-diversity.

Algorithm tuple-partition (T, ℓ)

```
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.
```

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.
```

7. 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 compare son, we use the Mondrian algorithm and ℓ-diversity for all three anonymization techniques: generalization, bucketization, 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.

(3) the sliced table can be computed efficiently. Results for this experiment are presented.

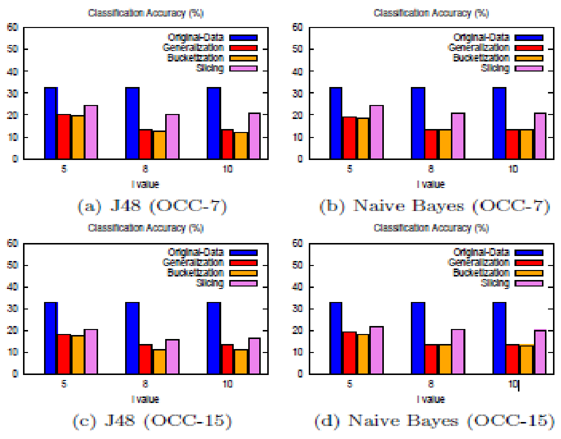


Figure 3. *Learning the sensitive attribute (Target: Occupation)*

In the second experiment, we show the effectiveness of slicing in membership disclosure protection. For this purpose, we count the number of fake tuples in the sliced data. We also compare the number of matching buckets for original tuples and that for fake tuples. Our experiment results show that bucketization does not prevent membership disclosure as almost every tuple is uniquely identifiable in the bucketized data.

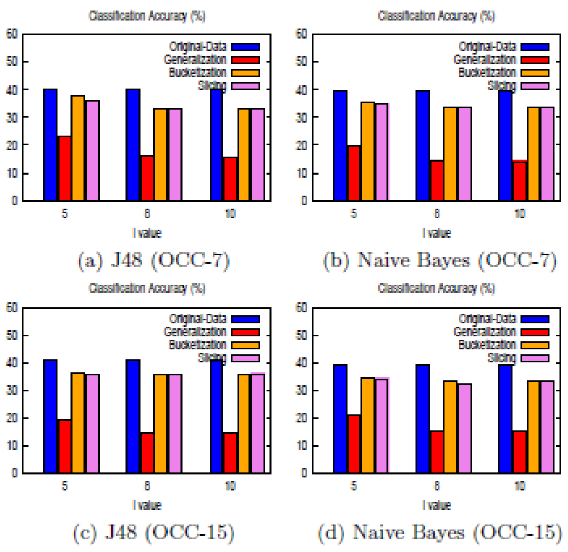
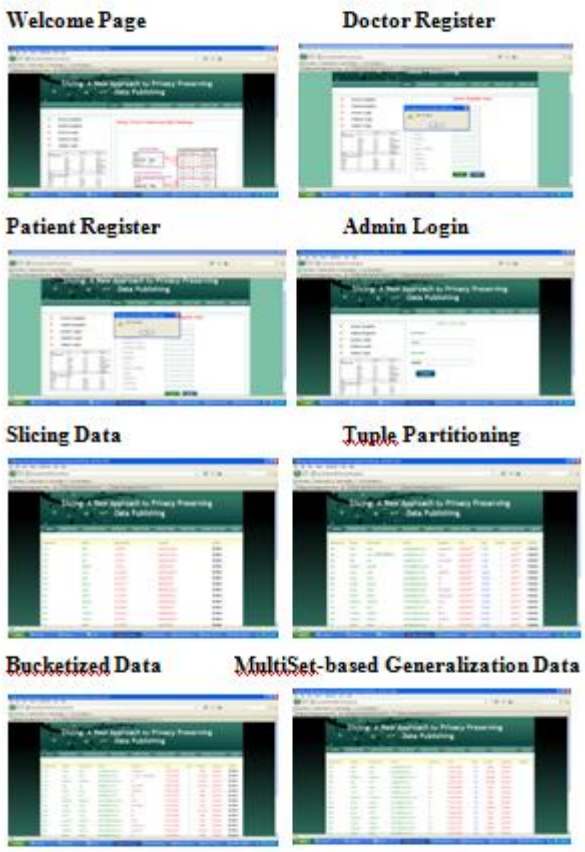


Figure 4. *Learning a QI attribute (Target: Education)*

8. Screen Shots



9. Conclusion

This paper presents a new approach called slicing to privacy-preserving microdata publishing. Slicing overcomes the limitations of generalization and bucketization and preserves better utility while protecting against privacy threats. We illustrate how to use slicing to prevent attribute disclosure 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 characteristics and use these characteristics in data anonymization. The rationale is that one can design better data anonymization techniques when we know the data better. 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 overlapping slicing, which duplicates an attribute in more than one column. These releases more attribute correlations. For example, in Table 1(f), one could choose to include the Disease attribute also in the first column. That is, the two columns are {Age, Sex and Disease} and {Zipcode, Disease}. This could provide better data utility, but the privacy implications need to be carefully studied and understood. It is interesting to study the tradeoff between privacy and utility Second; we plan to study membership disclosure protection in more details. Our experiments show that random grouping is not very effective. We plan to design more effective 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. 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 generate the associations between column values of a bucket. This may lose data utility. Another direction to design data mining tasks using the anonymized data 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.

[7] I. Dinur and K. Nissim. Revealing information while preserving privacy. In PODS, pages 202–210, 2003.

[8] C. Dwork. Differential privacy. In ICALP, pages 1–12, 2006.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In TCC, pages 265–284, 2006.

[10] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. TOMS, 3(3):209–226, 1977.

[11] R. J. Bayardo and R. Agrawal, "Data Privacy through Optimal k- Anonymization," in *Proc. of ICDE, 2005*, pp. 217–228.

[12] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-domain k-Anonymity," in *Proc. of ACM SIGMOD, 2005, pp. 49– 60*.

[13] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-Anonymity," in Proc. of ICDE, 2006.

[14] Gabriel Ghinita, Member IEEE, PanosKalnis, Yufei Tao," Anonymous Publication of Sensitive Transactional Data" in *Proc. of IEEE Transactions on Knowledge and Data Engineering* February 2011 (vol. 23 no. 2) pp. 161-174.

[15] D.J. Martin, D. Kifer, A. Machanavajjhala, J. Gehre, and J.Y. Halpern, "Worst-Case Background Knowledge for Privacy- Preserving Data Publishing," *Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE), pp. 126-135, 2007.*

AUTHORS' BIOGRAPHY

CH. Srikanth Reddy Pursuing MTech in Eswar college of Engineering, Narasaraopet

Md.John Saida M.Tech in Computer Science & Engineering. He is presently working as an Asst. Prof. in Eswar College of Engineering, Narasaraopet, Guntur, India. He is having about 10years of teaching experience in different Engineering Colleges and PG colleges. He attended national conference on "**Automating Oracle Business Intelligence**" and attended workshop on "**Empowering Students**" conducted by JKC (Jawaharlal Knowledge Center).