Efficient Anonymizations with Enhanced Utility

Jacob Goldberger *, **Tamir Tassa** **

* School of Engineering, Bar-Ilan University, Ramat-Gan, Israel

** Division of Computer Science, The Open University, Ra'anana, Israel

E-mail: goldbej@eng.biu.ac.il, tamirta@openu.ac.il

Abstract. One of the most well studied models of privacy preservation is k-anonymity. Previous studies of k-anonymization used various utility measures that aim at enhancing the correlation between the original public data and the generalized public data. We, bearing in mind that a primary goal in releasing the anonymized database for data mining is to deduce methods of predicting the private data from the public data, propose a new information-theoretic measure that aims at enhancing the correlation between the generalized public data and the private data. Such a measure significantly enhances the utility of the released anonymized database for data mining. We then proceed to describe a new algorithm that is designed to achieve k-anonymity with high utility, independently of the underlying utility measure. That algorithm is based on a modified version of sequential clustering which is the method of choice in clustering. Experimental comparison with four well known algorithms of k-anonymity show that the sequential clustering algorithm is an efficient algorithm that achieves the best utility results. We also describe a modification of the algorithm that outputs k-anonymizations which respect the additional security measure of ℓ-diversity.

Keywords. k-anonymity, privacy-preserving data mining, clustering, mutual information, ℓ-diversity

1 Introduction

Our society experiences in recent years unprecedented growth in the amount of data that is collected on individuals, organizations, companies and other entities. Of particular interest are data containing structured information on individuals. Data holders are then faced with the intricate task of releasing data in order to detect interesting trends or correlations, while still protecting the privacy of individuals. Privacy-preserving data mining [3] has been proposed as a paradigm of exercising data mining while protecting the privacy of individuals. Many approaches were suggested, implemented and theoretically studied for playing this delicate game that requires finding the right path between data hiding and data disclosure. One of these approaches, proposed by Samarati and Sweeney [22],

*This article is an extended version of a paper presented at the 2009 IEEE International Workshop on Privacy Aspects of Data Mining (PADM 2009), Miami, Florida, USA, Dec. 6, 2009.

is k-anonymization. The method of k-anonymization suggests to modify the values of the public attributes of the data by means of generalization so that if the database is projected on the subset of the public attributes, each record of the table becomes indistinguishable from at least $k-1$ other records. Consequently, the private data may be linked to sets of individuals of size no less than k, whence the privacy of the individuals is protected to some extent. The model of k-anonymity has been shown to be insufficient to protect against all types of linking attack, whence it must be enhanced by additional security measures such as ℓ-diversity, e.g. [7, 17, 25].

The main challenge is to achieve k-anonymity with minimal loss of information or, alternatively speaking, with maximal utility. The definition of the target function, namely, the measure of utility, is critical in this discussion. Several measures of utility were suggested in the literature. The problem of finding the k-anonymization with maximal utility (or minimal information loss) was shown to be NP-hard [2, 10, 18]. Hence, the possible approaches are either heuristical algorithms [5, 8, 9, 20] or approximation algorithms with a guaranteed approximation factor [2, 10, 12, 18]. Usually, algorithms of the former type outperform algorithms of the latter type.

Our contribution in this study is twofold: First, we propose a new information-theoretic measure of utility that takes into account the private attributes and aims at enhancing the correlation between the generalized public data and the private data. This is in contrast to most of the measures of utility that were used in previous studies that rely only on the public data and quantify the correlation between the original public data and the generalized public data. A primary goal of data mining is to find frequent patterns or rules to predict the private data from the public data. Hence, we deem our measure as one that best serves the purpose of obtaining anonymized tables with maximal utility for such applications of data mining. Then, we proceed to describe a sequential clustering algorithm that obtains high-utility anonymizations. Our algorithm, which is independent of the underlying utility measure, is based on a modified version of sequential clustering that is the method of choice in clustering. Experimental comparisons with four well known algorithms of k-anonymity showed that the sequential algorithm, together with the agglomerative algorithm [9, 20], achieve the best results in terms of utility. As the sequential clustering is significantly faster than the agglomerative algorithm, it appears to be the algorithm of choice for achieving high-utility k anonymizations efficiently.

We would like to note that several studies in recent years focused on methods of increasing the utility of k-anonymized tables, e.g. [1, 13, 26]. Most of those studies achieve improved utility by breaking out of the basic k-anonymity framework. For example, Aggrawal et. al. [1] suggested clustering the database records into clusters of size no less than k, and then publishing the cluster centers, sizes and private information; Kifer and Gehrke [13] suggested publishing additional information in the form of anonymized marginals; Xiao and Tao [26] proposed a method called *Anatomy* which allows the release of exact public values while maintaining privacy by separating the private data from the records of public data. The first part of our paper proposes another way of increasing the utility of k-anonymized tables. In contrast to the above mentioned studies, it achieves that goal within the basic framework of k-anonymity (i.e., by generalizing the entries of the public database until it becomes k-anonymous); it does so by suggesting a more appropriate measure that is tailored to match the prediction task which is one of the main purposes of data mining the anonymized table.

The paper is organized as follows. In Section 2 we provide the basic notations and terminology. In Section 3 we review previously used utility measures. In Section 4 we discuss the mutual information utility measure [10]. That discussion sets the ground for the introduction of our *private* mutual information utility measure in Section 5. Then, we proceed to describe in Section 6 our proposed sequential clustering algorithm. That section includes also a detailed survey of other algorithms of k-anonymity, with which we compare our algorithm later on. In Section 7 we describe a modification of the sequential clustering algorithm that supports the additional security measure of ℓ-diversity. Section 8 is devoted to experimental results. The paper concludes in Section 9.

2 Notations and Terminology

Consider a database that holds information on individuals in some population. Each individual is described by a collection of r public attributes, $A_1, \ldots, A_r$ (e.g. gender, age, occupation), and a private attribute, A_{r+1} (that could represent, for example, a medical diagnosis for that individual, his credit limit etc.).[1] Each of the attributes consists of several possible values: $A_j = \{a_{j,\ell} : 1 \leq \ell \leq m_j\}$, $1 \leq j \leq r+1$. For example, if A_j is gender then $A_j = \{M, F\}$, while if it is the age of the individual, it is a bounded nonnegative natural number. The public database holds all publicly available information on the individuals; letting n denote the number of individuals, it takes the form

$$D = \{R_1, \ldots, R_n\}\,, \quad \text{where } R_i \in A_1 \times \cdots \times A_r\,. \tag{1}$$

The corresponding private database holds the private information,

$$D' = \{S_1, \ldots, S_n\}\,, \quad S_i \in A_{r+1}\,. \tag{2}$$

The complete database is the concatenation of those two databases, $D\|D' = \{R_1\|S_1, \ldots, R_n\|S_n\}$. We refer hereinafter to the tuples R_i and S_i, $1 \leq i \leq n$, as the public and private records, respectively. The jth component of the record R_i (namely, the (i,j)th entry in the database D) will be denoted hereinafter by $R_i(j)$.

The basic technique for obtaining k-anonymization is by means of *generalization*. By generalization we refer to the act of replacing the values that appear in the public database with subsets of values, so that each entry $R_i(j) \in A_j$ is replaced by a subset $\overline{R}_i(j) \subseteq A_j$ that includes that element.

Definition 1. Let A_j, $1 \leq j \leq r$, be finite sets and let $\overline{A}_j \subseteq \mathcal{P}(A_j)$ be a collection of subsets of A_j. Let $D = \{R_1, \ldots, R_n\}$ be a table where each record R_i, $1 \leq i \leq n$, is taken from $A_1 \times \cdots \times A_r$. A table $g(D) = \{\overline{R}_1, \ldots, \overline{R}_n\}$ is a generalization of D, if $\overline{R}_i \in \overline{A}_1 \times \cdots \times \overline{A}_r$, and $R_i(j) \in \overline{R}_i(j)$, for all $1 \leq i \leq n$ and $1 \leq j \leq r$.

A special kind of generalization is generalization by suppression, where $\overline{A}_j = A_j \cup \{A_j\}$ for all $1 \leq j \leq r$ (e.g. [18]). Namely, each entry is either left unchanged or is totally suppressed. A more refined scheme of generalization [2] is that in which there is a hierarchy of clusterings of A_j, the finest one consisting of all singleton subsets, and the coarsest one consisting of just the entire set.

It should be noted that Definition 1 poses no restrictions on the subsets in each collection. For example, if A_j is a numeric attribute (say, age), the subsets in $\overline{A}_j$ may consist of noncontinuous ranges. Our algorithm and analysis are indifferent to such anomalies, whence we do not pose here any restrictions on the collections of subsets. In practice, those subsets

[1]We assume one private attribute for the sake of simplicity; the extension to any number of private attributes is straightforward.

should be selected in accord with the underlying semantics of the attribute, so that each subset contains elements that have some semantic proximity or common denominator.

There are two main models of generalization. In *global recoding*, e.g. [4, 11, 14, 25], each collection of subsets $\overline{A}_j$ is a clustering of the set A_j (in the sense that $\overline{A}_j$ includes *disjoint* sets whose union equals A_j). In such cases, every entry in the jth column of the database is mapped to the unique subset in $\overline{A}_j$ that contains it. As a consequence, every single value $a \in A_j$ is always generalized in the same manner. On the other hand, in *local recoding*, e.g. [8, 9, 10, 12, 18, 21, 25], the collection of subsets $\overline{A}_j$ covers the set A_j but it is not a clustering (namely, the subsets in the collection may intersect). In such cases, each entry in the table's jth column is generalized independently to one of the subsets in $\overline{A}_j$ which includes it. Hence, if the age 34, for example, appears in the table in several records, it may be left unchanged in some, or generalized to 30 - 39, or totally suppressed in other records.

In this study we consider the case of local recoding that allows greater flexibility and, hence, enables achieving k-anonymity with (possibly) smaller information loss. As mentioned before, the problem of k-anonymization with minimal loss of information is NP-hard in the case of local recoding.

3 Previously Used Utility Measures

Let D be the original public database and $g(D)$ be a generalization of D. A critical question in the context of k-anonymization is how to define $\Pi(D, g(D))$ – the distance between D and $g(D)$ in the sense of the amount of information that was lost due to the application of the generalization operator g. Meyerson and Williams [18] considered the case of generalization by suppression, and their measure simply counted the number of suppressed entries in the generalized database. Aggarwal et al. [2] used the more general model of generalization by hierarchical clustering. Assume a monotone sequence of $h + 1$ clusterings, where the finest and coarsest clusterings in the sequence are the trivial ones. Then if we replace an exact database entry with a subset from the ith finest clustering that contains it, where $0 \le i \le h$, the corresponding loss of information is i/h. The overall loss of information is then defined as the average loss per entry.

The Loss Metric LM [11, 20] is a more precise and a more general version of the above defined measure. According to the LM measure, the cost per each table entry is a number between 0 (no generalization at all) and 1 (total suppression) that penalizes the generalization that was made in that entry according to the size of the generalized subset. The overall cost is the average cost per table entry:

$$\Pi_{\mathrm{LM}}(D, g(D)) = \frac{1}{nr} \cdot \sum_{i=1}^{n} \sum_{j=1}^{r} \frac{|\overline{R}_i(j)| - 1}{|A_j| - 1} . \tag{3}$$

In some works [5, 8], the two above measures were combined – the LM measure for numerical attributes and the measure of [2] for categorial attributes.

The Ambiguity Metric AM [20] is the average size of the Cartesian products of all generalized entries in each record in the table. This measure represents the number of (theoretically) possible combinations of original records that a generalized record can stand for:

$$\Pi_{\mathrm{AM}}(D, g(D)) = \frac{1}{n} \cdot \sum_{i=1}^{n} \prod_{j=1}^{r} |\overline{R}_i(j)| . \tag{4}$$

An immediate drawback of the AM cost measure is that it counts also combinations of attribute values that do not appear in the original database.

The Discernibility Metric DM [4] defines the cost of each generalized record $\overline{R}_i$ as the number of generalized records in the anonymized table that are indistinguishable from it. A suppressed record is penalized by the size of the entire database $|D|$. Therefore, the total cost of the DM measure is the sum of squares of the sizes of all non-suppressed clusters, plus the number of totally suppressed records multiplied by $|D|$. Since in k-anonymizations that are near-optimal, all clusters are of sizes close to k, all such anonymizations have approximately the same DM cost, what makes this measure less useful.

In [13] another measure was defined. They considered the probability distribution that is induced by the original table on the space of quasi-identifiers, $A_1 \times \cdots \times A_r$, and the probability distribution that is induced by the anonymized table on the same space, and then defined the information loss as the Kullback-Leibler divergence between the two distributions.

While all of the measures that were described above consider only the values of the public attributes of the database, the next measure takes into account also the private attribute. The Classification Metric CM [11] defines a possible penalty for each generalized record of the table, based on its private attribute. A record $\overline{R}_i$ is penalized either if its private value differs from the majority of the private values in its cluster, or if $\overline{R}_i$ is totally suppressed. The CM measure is then defined as the average of the penalties of all rows. The rationale behind the classification metric is that homogenous clusters have more utility than heterogenous clusters, because they indicate a stronger association between the public attribute values of the cluster and the trained classified attribute. The information-theoretic measure that we present in Section 5 is motivated by a similar rationale, but it is much more accurate than the above described crude measure.

4 The Mutual Information Utility Measure

None of the measures that were described in the previous section was information-theoretic, even though they aim to measure information. Two information-theoretic measures of information-loss were introduced in [10] - the entropy measure, and the non-uniform entropy measure. We concentrate here on the latter measure and describe it here in a manner that is based on the notion of mutual information. Our description is somewhat different from the one in [10]. In particular, we describe it as a *utility* measure (denoted $U(g(D))$ rather than a measure of *information-loss*; as such, the goal is to maximize it (while measures of information-loss are sought to be minimized). The discussion here of that utility measure, to which we refer as *the mutual information utility measure*, provides the technical background and motivation for the new utility measure that we introduce in the next section.

Let $D = \{R_1, \ldots, R_n\}$ be a database and let $A_1, \ldots, A_r$ be its public attributes. For each $1 \le j \le r$, denote by X_j the random variable that corresponds to the attribute A_j. By looking at the table's jth column – $\{R_1(j), \ldots, R_n(j)\}$ – as the sample space for the variable X_j, we get the probability distribution:

$$\Pr(X_j = a) = \frac{|\{1 \le i \le n : R_i(j) = a\}|}{n}, \quad a \in A_j \,. \tag{5}$$

The entropy of X_j is then defined as follows [6],

$$H(X_j) = -\sum_{a\in A_j} \Pr(X_j = a)\log \Pr(X_j = a)\,. \tag{6}$$

First, we derive the mutual information utility measure in the case of *global* recoding. In such settings, each column in $g(D)$ includes subsets that constitute a clustering of the corresponding attribute. Letting A_j, $1 \le j \le r$, be one of the public attributes, the corresponding column in the generalized table includes values from $\hat{A}_j = \{C_1, \ldots, C_{t_j}\}$ where $\hat{A}_j$ is just a clustering of A_j in the sense that $C_1, \ldots, C_{t_j}$ are disjoint subsets of A_j whose union equal A_j. (For example, if A_j is the age, $\hat{A}_j$ may consist of ranges of ages of the form $10-19$, $20-29$, $30-39$ etc.)

While the jth column in D defines a random variable X_j on A_j, the jth column in $g(D)$ defines a random variable $\hat{X}_j$ on $\hat{A}_j$, where for each $C_\ell \in \hat{A}_j$:

$$\Pr(\hat{X}_j = C_\ell) = \sum_{a\in C_\ell} \Pr(X_j = a)\,.$$

The conditional entropy of X_j given $\hat{X}_j$ is:

$$H(X_j|\hat{X}_j) = -\sum_{a\in A_j} \Pr(X_j = a)\log \Pr(X_j = a|\hat{X}_j = g(a))\,,$$

where $g(a)$ is the (unique) generalization of $a \in A_j$.

The mutual information between two random variables is a measure of the information that is disclosed on one of those variables by providing the value of the other one. The mutual information between X_j and $\hat{X}_j$ is:

$$I(X_j;\hat{X}_j) = H(X_j) - H(X_j|\hat{X}_j) = \sum_{a\in A_j} \Pr(X_j = a)\log \frac{\Pr(X_j = a|\hat{X}_j = g(a))}{\Pr(X_j = a)}\,.$$

Using Equation (5) we get that

$$I(X_j;\hat{X}_j) = \frac{1}{n}\sum_{i=1}^{n} \log \frac{\Pr(X_j = R_i(j)|X_j \in \overline{R}_i(j))}{\Pr(X_j = R_i(j))}\,. \tag{7}$$

The mutual information between the tuples $\langle X_1, ..., X_r\rangle$ and $\langle \hat{X}_1, ..., \hat{X}_r\rangle$ is a natural way to measure the information that the anonymized table reveals on the original table. However, the relative sparsity of the multidimensional data makes the empirical estimation unreliable. Hence, we use instead an approximation based on the assumption that the attribute random variables are independent (an assumption that implicitly underlies all previously used measures). This yields the mutual information utility measure $U(g(D)) := I(D; g(D))$ where $I(D; g(D))$ is the following mutual information,

$$I(D; g(D)) := \frac{1}{r}\sum_{j=1}^{r} I(X_j;\hat{X}_j)\,.$$

Hence, the goal is to find a clustering of each of the attributes that will render the database k-anonymized while keeping the mutual information, $I(D; g(D))$, maximal.

Having defined the mutual information utility measure in the case of global recoding, we proceed to define it in the case of local recoding. Assuming that $\overline{A}_j$ is the collection of subsets of A_j that may be used as generalized values, the generalized table $g(D)$ takes the form $g(D) = \{\overline{R}_1, \ldots, \overline{R}_n\}$ where $\overline{R}_i(j) \in \overline{A}_j$. Although we cannot formalize this local generalization as a joint distribution of the two random-variables (the original one and the generalized one) we can still apply the local interpretation of the mutual information between the jth column in the original table and the corresponding column in the anonymized table, (7). Therefore, the preserved information per attribute can still be written as

$$I(X_j; \overline{R}(j)) = \frac{1}{n} \sum_{i=1}^{n} \log \frac{\Pr(X_j = R_i(j) | X_j \in \overline{R}_i(j))}{\Pr(X_j = R_i(j))} \tag{8}$$

where $\overline{R}(j)$ stands for the jth column in $g(D)$. Finally, the mutual information (MI) utility measure is $U(g(D)) := I(D; g(D))$ where

$$I(D; g(D)) := \frac{1}{r} \sum_{j=1}^{r} I(X_j; \overline{R}(j)) \tag{9}$$

$$= \frac{1}{nr} \sum_{i=1}^{n} \sum_{j=1}^{r} \log \frac{\Pr(X_j = R_i(j) | X_j \in \overline{R}_i(j))}{\Pr(X_j = R_i(j))} .$$

The corresponding mutual information measure of information-loss is

$$\Pi_{MI}(D, g(D)) = -\frac{1}{nr} \sum_{i=1}^{n} \sum_{j=1}^{r} \log \Pr(X_j = R_i(j) | X_j \in \overline{R}_i(j)) . \tag{10}$$

Clearly, $\Pi_{MI}(D, g(D))$ is minimized when the utility measure $U(g(D)) = I(D; g(D))$ in (9) is maximized.

A natural property that one might expect from any utility measure is monotonicity. In other words, we expect that coarser generalizations will be characterized by smaller values of the utility measure.

Definition 2. Let D be a table and let $g_1(D)$ and $g_2(D)$ be any two generalizations of D. A utility metric $U(\cdot)$ is called monotone if $U(g_2(D)) \leq U(g_1(D))$ whenever $g_2(D)$ is a generalization of $g_1(D)$, in the sense that every entry in $g_2(D)$ is a superset of the corresponding entry in $g_1(D)$.

In other words, we expect that coarser generalizations will be characterized by smaller values of the utility measure.

Proposition 3. *The MI utility measure is monotone.*

Proof. Let $\overline{R}_i^1(j)$ and $\overline{R}_i^2(j)$ be generalized items associated with a generalization $g_1(D)$ and a coarser generalization $g_2(D)$ of the same database D. As $\overline{R}_i^1(j) \subseteq \overline{R}_i^2(j)$ and $R_i(j) \in \overline{R}_i^1(j)$, we infer that:

$$\Pr(X_j = R_i(j) | X_j \in \overline{R}_i^1(j)) \geq \Pr(X_j = R_i(j) | X_j \in \overline{R}_i^2(j)) .$$

Summing over all public attributes and all database records we obtain the monotonicity property $U(g_1(D)) \geq U(g_2(D))$ (or, alternatively, that $\Pi_{MI}(D, g_1(D)) \leq \Pi_{MI}(D, g_2(D))$). $\square$

5 The Private Mutual Information Utility Measure

All previously used measures of information-loss are based entirely on the public information and ignore the private information, the only exception being the Classification Metric CM. However, one should keep in mind that one of the main goals in publishing the database is to learn the relation between the public data and private data and to deduce methods of predicting the private data from the public data. Therefore, the information-loss caused by the generalization process should be measured in the context of this prediction task. Specifically, generalization of public attributes that are weakly correlated with the private data should be less penalized than generalization of other public attributes that are strongly correlated with the private data.

We proceed to present here a new utility measure that quantifies the mutual information between the generalized public data and the private data. So, instead of looking at $I(D;g(D))$ (namely, how much information do the generalized public data reveal on the original public data), we look at $U(g(D)) := I(D';g(D))$ – the amount of information that the generalized public data reveal on the private data. (Recall that $D' = \{S_1,\ldots,S_n\}$, where $S_i \in A_{r+1}$, is the ith private record, see Equation (2).) As before, let X_j denote the random variable that corresponds to the jth public attribute and let $\overline{R}(j)$ stand for the jth column in $g(D)$, $1 \le j \le r$. In addition, we introduce the random variable Y that corresponds to the private attribute. (The probability distribution of Y on the set of possible values for the attribute A_{r+1} is derived from the private database D' in similarity to the way that we defined the probability distribution of X_j according to the jth column in D.) In a way similar to definition (8) in the previous section, we define the mutual information between Y and the anonymized version of the jth public attribute X_j as follows:

$$I(Y;\overline{R}(j)) = \frac{1}{n}\sum_{i=1}^{n}\log\frac{\Pr(Y=S_i|X_j\in\overline{R}_i(j))}{\Pr(Y=S_i)}\,. \tag{11}$$

The mutual information $I(Y;\overline{R}(1),...,\overline{R}(r))$ can be utilized to measure the information that the anonymized table reveals on the private data. However, as discussed in the previous section, the relative sparsity of the multidimensional data makes the empirical estimation unreliable. Hence, we approximate that expression with the following one that can be easily computed:

$$I(D';g(D)) := \frac{1}{r}\sum_{j=1}^{r} I(Y;\overline{R}(j))\,. \tag{12}$$

The goal is then to maximize $U(g(D)) := I(D';g(D))$ that is defined through (11)+(12), i.e.,

$$I(D';g(D)) = \frac{1}{nr}\sum_{i=1}^{n}\sum_{j=1}^{r}\log\frac{\Pr(Y=S_i|X_j\in\overline{R}_i(j))}{\Pr(Y=S_i)}\,. \tag{13}$$

We refer to this utility measure as the *private mutual information utility measure* (PMI). The corresponding information-loss measure is

$$\Pi_{PMI}(D,g(D)) = -\frac{1}{nr}\sum_{i=1}^{n}\sum_{j=1}^{r}\log\Pr(Y=S_i|X_j\in\overline{R}_i(j))\,. \tag{14}$$

It expresses the amount of mutual information that is lost by replacing D with $g(D)$. Clearly, $\Pi_{PMI}(D,g(D))$ is minimized when $I(D';g(D))$ is maximized.

The PMI utility measure is defined in (12) as an average of the mutual information between the private attribute and each of the generalized public attributes. As such an averaging might hide a strong correlation of one of the generalized public attributes with the private attribute, another possible definition of that measure is

$$I(D'; g(D)) := \max_{1 \le j \le r} I(Y; \overline{R}(j)) . \tag{15}$$

A possible compromise between the ℓ_1-norm in (12) and the ℓ_∞-norm in (15) is the ℓ_2-norm version

$$I(D'; g(D)) := \frac{1}{r} \left(\sum_{j=1}^{r} I(Y; \overline{R}(j))^2 \right)^{1/2} . \tag{16}$$

In this study we focus on the first ℓ_1-version, (12). The comparison between the effectiveness of the different versions is left for future experiments and study.

5.1 Monotonicity

We now turn to discuss the monotonicity of the PMI utility measure, $U(g(D)) = I(D'; g(D))$. We begin by considering the case of global recoding. Assume that $g_2(D)$ is a generalization that is coarser than $g_1(D)$. Since both $g_1(D)$ and $g_2(D)$ are based on global recoding, they define for the jth attribute two random variables $\hat{X}_j^1$ and $\hat{X}_j^2$ respectively (see Section 4). The random variables $Y, X_j, \hat{X}_j^1$ and $\hat{X}_j^2$ form a Markov chain:

$$Y \longleftarrow X_j \longrightarrow \hat{X}_j^1 \longrightarrow \hat{X}_j^2 .$$

Hence, the Data Processing Lemma [6] implies that

$$I(Y; \hat{X}_j^1) \ge I(Y; \hat{X}_j^2) . \tag{17}$$

For completeness we provide here the proof. The chain rule for mutual information implies that:

$$I(Y; \hat{X}_j^1, \hat{X}_j^2) = I(Y; \hat{X}_j^1) + I(Y; \hat{X}_j^2 | \hat{X}_j^1) = I(Y; \hat{X}_j^2) + I(Y; \hat{X}_j^1 | \hat{X}_j^2) . \tag{18}$$

The Markovian chain structure (i.e. the fact that $g_2(D)$ is a coarsening of $g_1(D)$) implies that $I(Y; \hat{X}_j^2 | \hat{X}_j^1) = 0$. Hence,

$$I(Y; \hat{X}_j^1) = I(Y; \hat{X}_j^2) + I(Y; \hat{X}_j^1 | \hat{X}_j^2) .$$

Finally, as mutual information is always non-negative, inequality (17) immediately follows. Hence, by summing up inequalities (17) for all $1 \le j \le r$ and dividing by r we conclude that $I(D'; g_1(D)) \ge I(D'; g_2(D))$.

In the more general case of generalization based on local recoding, the PMI utility measure (13) is not always monotone. For example, let D be the following table with a single public attribute and a single private attribute:

D	a	a	a	a	b	b	b	b	c
D'	0	0	0	1	0	1	1	1	1

Consider the following 3-anonymization of D:

$g_1(D)$	a	a	a	*	*	b	b	b	*
D'	0	0	0	1	0	1	1	1	1

In this case (see (14)),

$$\Pi_{PMI}(D, g_1(D)) = -\frac{1}{9} \cdot \left[\left(6 \log \frac{3}{4} + 2 \log \frac{5}{9} + \log \frac{4}{9} \right) - \left(6 \log \frac{3}{4} + 2 \log \frac{1}{4} + \log 1 \right) \right] \approx -0.126 \,.$$

As $\Pi_{PMI}(D, g_1(D))$ is negative, we conclude that $I(D'; D) < I(D'; g_1(D))$, even though $g_1(D)$ is a generalization of D, whence monotonicity is violated. Nonetheless, the PMI utility measure is still meaningful for local recoding and serves well the purposes of data-mining, as exemplified in this example. All 3-anonymizations of D will have to generalize the c-record together with one of the a-records and one of the b-records. An easy calculation shows that any of the other selections of an 'a'-record or a 'b'-record results in a greater information loss as measured by the PMI measure. Consider, for example, the generalization

$g_2(D)$	a	a	*	a	*	b	b	b	*
D'	0	0	0	1	0	1	1	1	1

.

For that generalization we have

$$\Pi_{PMI}(D, g_2(D)) = \left[\left(5 \log \frac{3}{4} + \log \frac{1}{4} + 2 \log \frac{4}{9} + \log \frac{5}{9} \right) - \left(6 \log \frac{3}{4} + 2 \log \frac{1}{4} + \log 1 \right) \right] \cdot \left(-\frac{1}{9} \right) \approx 0.086 \,.$$

Indeed, the generalization $g_1(D)$ is the best one for data mining since it selects to obfuscate the outlier records. In fact, this is the reason why the PMI utility measure favors the generalization $g_1(D)$ over the original table D, since the latter has outlier records that blur the two prominent association rules "a implies 0" and "b implies 1", while the former eliminates those outliers and accentuates those two rules.

The above example exemplifies the advantage that our newly proposed utility measure has to offer with respect to the previous measures. All the measures that rely only on the public attributes, e.g. LM and MI measures, cannot distinguish between $g_1(D)$ and $g_2(D)$. Hence, using the PMI measure may yield anonymized tables with greater utility for data mining.

6 Algorithms for k-Anonymity

The problem of finding a k-anonymization of a given table with minimal information loss is NP-hard. Several polynomial-time approximation algorithms were devised for this problem. The first one [18] has an approximation guarantee of $O(k \log n)$ and runtime of $O(rn^2 + n^3)$ (for the case of suppressions only). The algorithm in [2] runs in time $O(kn^2)$ and approximates the optimal solution to within $O(k)$ (for the case of generalization by hierarchical clustering). As k may be relatively large in practice, those approximation factors might be unsatisfactory. A significant improvement was proposed in [10], with an $O(\log k)$-approximation algorithm that applies to any generalization and any measure. Alas, its run time, $O(n^{2k})$, renders it impractical. A more efficient $O(\log k)$-approximation algorithm was proposed in [21], but it is restricted only to generalizations by suppression.

Due to the poor performance and limitations of the provable approximation algorithms, heuristical algorithms are invoked. In Section 6.2 we describe three classes of k-anonymization algorithms, that include the most well known algorithms. Then, we proceed to describe in Section 6.3 an alternative approach, based on sequential clustering, that we propose in this context. As a preliminary discussion, we provide in Section 6.1 the basic definitions of cluster closure and generalization cost, terms which are utilized by any k-anonymization algorithm.

6.1 Generalization cost

Any k-anonymization induces a clustering of the records in D to clusters of size at least k. Conversely, every clustering of D into clusters of size at least k induces a k-anonymization, $g(D)$, in the following manner. Assume that $\{R_{i_1}, \dots, R_{i_m}\}$ is one of the clusters. Then the records $\overline{R}_{i_1}, \dots, \overline{R}_{i_m}$ in $g(D)$ will be all equal, and their jth entry will be the minimal set in $\overline{A}_j$ that includes the values $R_{i_1}(j), \dots, R_{i_m}(j)$. We aim at finding such a clustering that induces an optimal k-anonymization under a given measure of loss of information.

Let $\mathcal{C} = \{C_1, \dots, C_t\}$ be a clustering of the records in D, where all clusters are of size at least k. Such a clustering induces a k-anonymization $g(D)$ of D. Letting Π be some measure of information-loss, we proceed to define an anonymization cost, $\texttt{gc}$, for each of the clusters, C_i, $1 \le i \le t$. The generalization cost $\texttt{gc}$ will be defined so that the information-loss of the anonymization $g(D)$ will be given by

$$\Pi(D, g(D)) = \frac{1}{n} \sum_{j=1}^{t} \texttt{gc}(C_j) \cdot |C_j| \,. \tag{19}$$

In other words, we wish to define $\texttt{gc}$ so that the average over all n records in D of the $\texttt{gc}$ value of that record's cluster will be the information-loss of the anonymization $g(D)$ that is induced by that clustering.

Let C be one of the clusters in $\mathcal{C}$. Without loss of generality, we assume that $C = \{R_1, \dots, R_m\}$. The *closure* of C is the minimal generalized record $\overline{R}$ that generalizes every record in C. Namely, for all $1 \le j \le r$, $\overline{R}(j)$ is the minimal set in the collection $\overline{A}_j$ that includes all of the values $R_1(j), \dots, R_m(j)$. In the anonymized table $g(D)$ that corresponds to the clustering $\mathcal{C}$, all records in C will be replaced by the closure of C. Then the corresponding generalization cost of C, $\texttt{gc}(C)$, is the average information loss that is caused by replacing each of the records in C by the generalized record $\overline{R} = (\overline{R}(1), \dots, \overline{R}(r))$.

For example, for the LM measure, (3), we have

$$\texttt{gc}_{LM}(C) = \frac{1}{r} \cdot \sum_{j=1}^{r} \frac{|\overline{R}(j)| - 1}{|A_j| - 1} \,. \tag{20}$$

The generalization cost in that case is the same for all records that belong to the same cluster. For the MI measure, (10), and the PMI measure, (14), on the other hand, the generalization cost may differ from one record to another in the same cluster. In the former, it depends on the original public attributes in the record,

$$\texttt{gc}_{MI}(C) = -\frac{1}{mr} \sum_{i=1}^{m} \sum_{j=1}^{r} \log \Pr(X_j = R_i(j) | X_j \in \overline{R}_i(j)) \,,$$

while in the latter it depends on the private attribute in the record,

$$\mathtt{gc}_{PMI}(C) = -\frac{1}{mr}\sum_{i=1}^{m}\sum_{j=1}^{r}\log\frac{\Pr(Y = S_i|X_j \in \overline{R}_i(j))}{\Pr(Y = S_i|X_j = R_i(j))}\,.$$

6.2 Previous k-anonymity algorithms

Here we overview some of the prominent algorithms of k-anonymity. The survey is separated to three sections, each one is devoted to a different type of algorithms.

6.2.1 Agglomerative algorithms

Agglomerative algorithms were proposed in [9, 20]. The basic idea in such algorithms is to define a distance function dist$(\cdot,\cdot)$ between clusters, where the distance between two given clusters is correlated with the change in the overall utility in case we unify them. With such a definition of distance, agglomerative algorithms begin to build a clustering from the bottom upwards. Namely, they start with the trivial clustering that consists of singleton clusters and then keep unifying the two closest clusters until all clusters become larger than k. A key ingredient in such algorithms is the definition of the distance function. It is natural to define the distance so that it best fits the cost function of the k-anonymization. We used in our experiments one of the distance functions that were proposed in [9],

$$\text{dist}(A,B) = |A \cup B| \cdot \mathtt{gc}(A \cup B) - |A| \cdot \mathtt{gc}(A) - |B| \cdot \mathtt{gc}(B)\,,$$

which, in view of Equation (19), expresses the difference in the overall generalization cost if we unify the clusters A and B.

A different agglomerative algorithm was considered in [5]. That algorithm, which was called k-member clustering, selects one of the records as a center for a new cluster, and then looks for the $k-1$ closest records, where, as before, a distance between records is defined in terms of the generalization cost, i.e.,

$$\text{dist}(R_i, R_j) = \mathtt{gc}(\{R_i, R_j\})\,. \tag{21}$$

The center record together with its $k-1$ closest records are grouped together as a cluster, and then the process continues similarly with the remaining records.

6.2.2 Top-down algorithms

The basic agglomerative algorithm is a bottom-up clustering algorithm. Another approach in clustering is the top-down approach. The most well-known representative of that approach in k-anonymity is the Mondrian algorithm [15]. In that algorithm one starts with the other trivial clustering, in which all records are placed in a single cluster, and then keeps greedily splitting the clusters into smaller clusters until no further splits are possible without violating the k-anonymity constraint.

The Mondrian algorithm assumes that each attribute is totally ordered, so that the table records may be viewed as points in an r-dimensional space. The top-down approach is then implemented by separating those points by means of *cuts*; a cut of an r-dimensional domain A is a separation of that domain to two sub-domains along one of its dimensions, where the two sub-domains are of the form $\{\mathbf{x} \in A : x_j \le a_0\}$ and $\{\mathbf{x} \in A : x_j > a_0\}$, for some $1 \le j \le r$. Hence, when the algorithm stops, the set of r-dimensional points is separated to rectangular regions with boundaries of the form $a_j \leq x_j \leq b_j$. That separation defines the clustering. All points (records) within the same region (cluster) will be generalized to the closure of that cluster. Here lies the main difference between this approach and other clustering approaches that are not guided by geometry. In this approach, two identical records will always be generalized in the same way (with the exception of records that fall on the boundary between two regions and then we may choose to associate such records with either of the two regions). Such a model of generalization is called *global multi-dimensional recoding*. It falls between local and global recoding, in the sense that any global recoding is a special case of global multidimensional recoding, which is a special case of local recoding. Hence, local recoding is a more flexible generalization model than global multidimensional recoding. As a result, it usually yields anonymizations with better utility, as exemplified later in Section 8.1.

6.2.3 An algorithm based on space-filling curves

Even though the problem of optimal k-anonymity is NP-hard in general, it is possible to solve it optimally and efficiently in polynomial time in the case of one-dimensional data, using dynamic programming [8]. Hence, a possible approach is to map the multidimensional data to one-dimensional data, in a manner that attempts to preserve proximity, and then to apply on the one-dimensional data the optimal k-anonymization algorithm. Space-filling curves are good candidates for such mappings. Two types of such curves were examined in [8]; one of them was the Hilbert space-filling curve [19], and it was found to yield the best results. The Hilbert curve of order t maps all points of integral coordinates in the box $[0, 2^t - 1]^r$ to integers in the interval $[0, 2^{tr} - 1]$.

It turns out that such algorithms are extremely sensitive to the different ranges of the attributes. Consequently, they tend to operate well when the ranges of all attributes are similar, but may produce bad anonymizations otherwise. As a toy example, consider a table that has the 4D-records that are listed in the left column of Table 1, where each one of them appears exactly $k/2$ times (namely, there are $n = 2k$ records in total). The second column in the table gives the 1D-Hilbert curve mapping of the 4D-records. An optimal

original record	1D value	re-scaled record	modified 1D value
$R_1 = (0, 0, 0, 0)$	0	$(0, 0, 0, 0)$	0
$R_2 = (1, 1, 1, 0)$	9	$(2, 2, 2, 0)$	220
$R_3 = (0, 0, 0, 2)$	118	$(0, 0, 0, 2)$	118
$R_4 = (1, 1, 1, 2)$	123	$(2, 2, 2, 2)$	162

Table 1: 4D records and their Hilbert curve mappings

clustering of those $2k$ records would place the $k/2$ records that equal R_1 together with the $k/2$ records that equal R_3 in one cluster, the closure of which would be $(0, 0, 0, *)$, and the remaining k records in a cluster whose closure is $(1, 1, 1, *)$. The overall suppression cost of such a clustering is $2k$. However, as the Hilbert-curve induced order of those records is R_1, R_2, R_3, R_4, the resulting clustering would put together the k records that equal R_1 or R_2 in a cluster whose closure is $(*, *, *, 0)$, and the other k records in a cluster whose closure is $(*, *, *, 2)$. The overall suppression cost in this case is $6k$.

The reason for this malfunction lies in the mapping strategy of the Hilbert curve. It first maps all points that are confined to the box $[0, 2^1 - 1]^4$. Only then it takes the remaining points in the larger box $[0, 2^2 - 1]^4$, and so forth. Note that in the above example, R_1 and

R_2 are from the smaller box, while R_3 and R_4 are outside that box. Therefore, even though in the 4D space, the point R_1 is closer to R_3 than it is to R_2 (in the sense of the distance definition (21)), the Hilbert curve fails to preserve proximity in this case.

A simple way to improve the algorithm is to first rescale all attributes so that they range along the same interval. If we stretch the first three attributes to range in the interval $[0, 2]$, just like the fourth attribute, we get the set of 4D-points and their 1D image as given in the two right columns of Table 1. Applying the optimal k-anonymity algorithm on this 1D data gives the sought-after optimal clustering for the original set of records.

We verified that such a preprocessing of the input records improves significantly the performance of the Hilbert curve based anonymization algorithm also on real data. In the experimental section we report the performance of the above described improved version of the algorithm.

6.3 The sequential clustering algorithm

The most fundamental non-agglomerative clustering technique is K-means [16]. As the number of clusters is unknown, but is bounded from above by $\lfloor n/k \rfloor$, we may set K to a number in the vicinity of that upper bound, select K random centers, and then use any of the utility measures that were defined in the previous sections as the underlying metric. Alternatively, we can apply a greedy sequential algorithm that can be viewed as a sequential version of the K-means algorithm. The sequential greedy algorithm is known to perform well in terms of both clustering quality and computational complexity [23].

The basic sequential algorithm starts with a random partition of the data into clusters and then it keeps scanning the data points and attempts to improve their allocation to clusters in order to increase the value of the underlying utility. Usually, when one looks for optimal clustering of data, the number of clusters is given as an input. In such settings, the clustering algorithm will look for an optimal clustering that has the required number of clusters. For example, agglomerative (bottom-up) algorithms will execute the merging process until the desired number of clusters is obtained. As another example, the original sequential clustering algorithm [23] will be initialized with a random clustering having the specified number of clusters. However, in our k-anonymization clustering problem, the constraint is on the size of the clusters rather than on their number. To cope with this constraint, the scheduling of the sequential algorithm should be modified. We proceed to describe an adaptation of the sequential algorithm for the problem of finding k-anonymizations with high utility.

The sequential clustering algorithm starts with a random partition of the records into clusters. Then, it goes over the n records in a cyclic manner and for each record checks whether it may be moved from its current cluster to another one while increasing the utility of the induced anonymization. Specifically, if R_i belongs currently to cluster C_j, then the change in the information loss if we move it to another cluster C_h is (in view of (19)),

$$\Delta_{i:j\to h} = \frac{1}{n}\cdot\{\ [\mathtt{gc}(C_j\setminus\{R_i\})\cdot(|C_j|-1)+\mathtt{gc}(C_h\cup\{R_i\})\cdot(|C_h|+1)]- \\ -[\mathtt{gc}(C_j)\cdot|C_j|+\mathtt{gc}(C_h)\cdot|C_h|]\ \}\ . \qquad (22)$$

If that difference is negative, then we gain from such a transition. Therefore, in each step of the algorithm we reexamine the current location of each of the records R_i in D and then look for an alternative location (cluster) that provides the best improvement in terms of information loss. This loop may be iterated until either we reach a local optimum (i.e., a stage in which no single-record transition offers an improvement) or the local improvements of the utility become sufficiently small.

At this point, some of the clusters are large, in the sense that their size is at least k, while others are small. If there exist small clusters, we apply the agglomerative algorithm on those clusters in order to merge them into larger clusters of size k or more. Finally, if we are left at the end with a single cluster that is small, we merge it with the closest large cluster.

The initial number of clusters in the random clustering is set to $\lfloor n/k_0 \rfloor$ and the initial clusters are chosen so that all of them are of size k_0 or k_0+1, where $k_0 = \alpha k$ is an integer and α is some parameter that needs to be determined. Then, during the sequential algorithm, we allow the size of the clusters to vary in the range $[2, \omega k]$, for some predetermined fixed parameter ω, where $1 < \omega \leq 2$. When a cluster becomes a singleton, we remove it and place that record in one of the other clusters where it fits best. If a cluster becomes too large (i.e., its size becomes larger than the upper bound ωk), we split it into two (almost) equal-sized clusters in a random manner.

It is preferable to have in the final clustering clusters of size close to k, since larger clusters imply lesser utility. One way of controlling the cluster sizes is by selecting properly the size of the initial clusters, $k_0 = \alpha k$, and by selecting the upper limit of cluster size, ωk. Our tests indicated that it is preferable to set α to a value smaller than 1 (namely, initially all clusters are smaller than k), and to set ω to a value smaller than 2. In all of our tests we used $\alpha = 0.5$ and $\omega = 1.5$.

This approach is summarized in Algorithm 1.

Algorithm 1 Sequential clustering algorithm for k-anonymization

```
input  Table $D = \{R_1, \ldots, R_n\}$, an integer $k$.
output  A clustering of $D$ into clusters of size at least $k$.
1:  Choose a random partition of the data records into $t := \lfloor n/k_0 \rfloor$ clusters of sizes either
    $k_0$ or $k_0 + 1$. Denote the clusters by $C_1, \ldots, C_t$.
2:  for $i = 1, \ldots, n$ do
3:      Let $C_j$ be the cluster to which record $R_i$ currently belongs.
4:      For each of the other clusters, $C_h$, $h \neq j$, compute the difference in the information
        loss if we move $R_i$ from $C_j$ to $C_h$ — $\Delta_{i:j \to h}$.
5:      Let $C_{h_0}$ be the cluster for which $\Delta_{i:j \to h}$ is minimal.
6:      If $C_j$ is a singleton, move $R_i$ from $C_j$ to $C_{h_0}$ and remove cluster $C_j$.
7:      Else, if $\Delta_{i:j \to h_0} < 0$, move $R_i$ from $C_j$ to $C_{h_0}$.
8:  end for
9:  If there exist clusters of size greater than $\omega k$, split each of those clusters randomly into
    two (almost) equal-sized clusters.
10: If at least one record was moved during the last loop, go to Step 2.
11: while the number of clusters of size smaller than $k$ is greater than 1 do
12:     Unify the two closest small clusters.
13: end while
14: If there exists a small cluster, unify it with the cluster to which it is closest.
15: Output the resulting clustering.
```

As there is no guarantee that such a procedure finds the global optimum, it may be repeated several times with different random partitions as the starting point, in order to find the best local optimum among those repeated searches.

Note that the agglomerative algorithm is in fact a special case of this sequential algorithm that corresponds to the selection $k_0 = 1$.

7 Supporting ℓ-Diversity

The notion of ℓ-diversity was introduced in [17] as an enhancement to k-anonymity. In that model, each cluster of size at least k of indistinguishable records must be sufficiently diverse, in the sense that it has at least ℓ "well-represented" distinct values in the private attribute. The basic diversity measure that was suggested in [17] is the entropy: Given a cluster of records, its diversity is the entropy of the distribution that it induces on the private attribute; the diversity of the entire clustering is the minimal diversity over all clusters. The entropy measure is hard to enforce in practice. Hence, a simpler measure of diversity was proposed in [25, 26]. A cluster of records respects ℓ-diversity according to that measure, if the frequency of each of the private values in the cluster does not exceed $1/\ell$. All definitions of ℓ-diversity are monotone in the following sense. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two clusterings of the same set of records and assume that $\mathcal{C}_1$ is coarser than $\mathcal{C}_2$ (i.e., every cluster in $\mathcal{C}_1$ is a union of clusters in $\mathcal{C}_2$). Then if $\mathcal{C}_2$ is ℓ-diverse, so is $\mathcal{C}_1$.

In this section we modify the basic sequential algorithm to respect also ℓ-diversity, according to the latter definition of ℓ-diversity. Namely, the input to the modified algorithm will consist of a public table D and its corresponding private table D', together with two parameters – k and ℓ – and it will issue k-anonymizations of $D||D'$ that respect also ℓ-diversity.

The private value in each record is taken from the attribute domain A_{r+1}, see Equation (2). Let us denote the values in A_{r+1} by $a_1, \ldots, a_m$. Then the frequency of $a_i \in A_{r+1}$ is

$$f_i := \frac{|\{1 \leq j \leq n : S_j = a_i\}|}{n} .$$

Let ℓ_0 be the inverse of the maximal frequency, i.e., $\ell_0 = (\max_i f_i)^{-1}$. Owing to the above discussed monotonicity of the diversity measure, all anonymizations of that table respect ℓ-diversity only if $\ell \leq \ell_0$.

In practice, in order to avoid anonymizations which are too coarse (namely, ones in which the clusters are much too large, whence they carry very small utility), it might be necessary to take ℓ which is strictly smaller than ℓ_0. To exemplify that, let us consider the case of a binary private attribute, $A_{r+1} = \{0, 1\}$, and assume that among the n records in the table, h records have the private value 0 and $n - h$ records have the private value 1. If $h \geq n - h$ then $\ell_0 = n/h$. Let us attempt to split the table into two clusters, each of which still respects ℓ_0-diversity. Let p and q denote the number of 0 records and 1 records, respectively, in the first of the two clusters. It is easy to see that such a split respects ℓ_0-diversity if and only if $p(n - h) = qh$. Since p and q must be integers in the range $0 \leq p \leq h$ and $0 \leq q \leq n - h$, it is possible that the only integral solutions of that equation are $(p, q) = (0, 0)$ or $(p, q) = (h, n - h)$. In that case, such a split is just the trivial split, where one of the clusters is empty and the other one contains the entire table. Hence, the only anonymization that respects ℓ_0-diversity in this case is the one in which all records are clustered together; in such an anonymized table, all records are totally suppressed.

In what follows, we describe a modification of Algorithm 1 that respects also ℓ-diversity. The modified algorithm is given in Algorithm 2.

In Step 1 of Algorithm 1, we created a random partition of the data records to t clusters of (almost) equal size. In Algorithm 2, we create a random partition so that the distribution of the private attribute in each of the t clusters is as close as possible to its distribution in the entire table. A strategy for performing such splits is described in Algorithm 3 in Section 7.1.

Let $C_1, \ldots, C_t$ be the resulting clusters and let p_i^j denote the number of records in C_j that have the sensitive value a_i, $1 \leq j \leq t$, $1 \leq i \leq m$. The diversity in C_j is $\text{div}(C_j) := \left(\max_i p_i^j / |C_j|\right)^{-1}$. Then the initial clustering is ℓ_1-diverse with $\ell_1 := \min_j \text{div}(C_j)$. As discussed above, ℓ_1 is no larger than ℓ_0, which is the diversity of the whole table. However, thanks to the splitting strategy that is described in Section 7.1, ℓ_1 is close to ℓ_0, since the splitting strategy attempts to preserve in each cluster the global private distribution, as much as possible.

We assume hereinafter that ℓ – the input parameter that indicates the required level of diversity for the output anonymization, is no larger than ℓ_1. As ℓ must be no larger than ℓ_0, and ℓ_1 is close to ℓ_0 (as implied by the splitting strategy and as indicated also by our experiments), this assumption is reasonable. In case the input parameter ℓ is larger than ℓ_1, we output the trivial clustering that consists of one cluster that includes all records (Step 3).

After computing the initial clustering, which by our assumption is ℓ-diverse, we proceed with the normal operation of the algorithm, while making sure that we do not create a cluster with diversity smaller than ℓ. By doing so, we guarantee that all intermediate clusterings are also ℓ-diverse, and, consequently, so is the final output.

There are three types of operations that are made on the clustering during the sequential clustering:

1. Transitions of records from one cluster to another.
2. Splitting of large clusters.
3. Unification of small clusters.

As for record transitions, Algorithm 2 performs such an action only if it does not violate ℓ-diversity in neither the originating cluster, nor in the destiny cluster. Specifically, we consider the option of moving a record from a cluster C_j only if such a removal would not decrease the diversity of C_j to below ℓ; if it would, we move on to the next record (Step 6). Otherwise, we look for a better cluster for that record only among those clusters that can receive that record without violating ℓ-diversity (Steps 7-12).

The operation of unifying small clusters (Steps 17 and 19) needs no further testing, since the unification of two ℓ-diverse clusters is also ℓ-diverse. Hence, we are left with the problem of splitting large clusters to smaller clusters. Here, whenever we receive a large cluster that needs to be split, we apply on it the splitting strategy that is described in Section 7.1; namely, the same strategy that we applied on the entire table in order to split it to t clusters, may be applied also on a given large cluster in order to split it to two clusters. If the resulting two clusters have diversity which is at least ℓ, then we were successful. However, if one of the resulting clusters has a diversity smaller than ℓ, we do not split the large cluster (Step 14).

This approach is described in Algorithm 2. We proceed to describe the splitting procedure, Algorithm 3.

7.1 A splitting strategy that retains diversity

Let $D||D' = \{R_1||S_1, \ldots, R_n||S_n\}$ be a set of records that needs to be split into t (almost) equal-sized clusters such that the distribution of the private values S_i in each of those clus-

Algorithm 2 Sequential clustering algorithm for k-anonymization and ℓ-diversity

```
input  Table $D||D' = \{R_1||S_1, \dots, R_n||S_n\}$, an integer $k$, a real parameter $\ell \geq 0$.
output  A clustering of $D||D'$ into clusters of size at least $k$ that respect $\ell$-diversity.
1: Compute $\ell_0 = \mathrm{div}(D||D')$. If $\ell > \ell_0$ stop and output "Input diversity parameter is illegal".
2: Call Algorithm 3 with inputs $D||D'$, and $t := \lfloor n/k_0 \rfloor$. The algorithm returns a partition of the data records into $t$ clusters, $C_1, \dots, C_t$, of sizes either $k_0$ or $k_0 + 1$.
3: Compute $\ell_1 = \min_{1 \leq i \leq t} \mathrm{div}(C_i)$. If $\ell > \ell_1$ stop and output the trivial clustering $\{D||D'\}$.
4: for $i = 1, \dots, n$ do
5:     Let $C_j$ be the cluster to which record $R_i||S_i$ currently belongs.
6:     If $\mathrm{div}(C_j \setminus \{R_i||S_i\}) < \ell$ continue to the next $i$.
7:     Find the set of clusters $B := \{C_h : h \neq j, \text{and } \mathrm{div}(C_h \cup \{R_i||S_i\}) \geq \ell\}$.
8:     If $B = \emptyset$, continue to the next $i$.
9:     For all $C_h \in B$, compute the difference in the information loss if we move $R_i||S_i$ from $C_j$ to $C_h$ — $\Delta_{i:j \to h}$.
10:    Let $C_{h_0}$ be the cluster for which $\Delta_{i:j \to h}$ is minimal.
11:    If $C_j$ is a singleton, move $R_i||S_i$ from $C_j$ to $C_{h_0}$ and remove cluster $C_j$.
12:    Else, if $\Delta_{i:j \to h_0} < 0$, move $R_i||S_i$ from $C_j$ to $C_{h_0}$.
13: end for
14: For each cluster $C_j$ where $|C_j| > \omega k$, call Algorithm 3 with inputs $C_j$ and 2. If the two resulting clusters respect $\ell$-diversity, replace $C_j$ with them; otherwise, retain $C_j$.
15: If at least one record was moved during the last loop, go to Step 4.
16: while the number of clusters of size smaller than $k$ is greater than 1 do
17:    Unify the two closest small clusters.
18: end while
19: If there exists a small cluster, unify it with the cluster to which it is closest.
20: Output the resulting clustering.
```

ters will be as close as possible to the distribution in the entire set. Recall that the sensitive values are taken from the domain $A_{r+1} = \{a_1, \dots, a_m\}$.

Let ℓ denote the diversity of D'. Namely, if p_j is the number of occurrences of a_j in D', then $\max_{1 \leq j \leq m} p_j = n/\ell$. Given an integer $t \geq 2$, the goal is to split D' into t disjoint subsets, $D' = \bigcup_{1 \leq i \leq t} D'_i$, such that the frequency of each a_j in each subset will be no larger than $1/\ell$, and the subset sizes will be as close as possible, in the sense that the difference between the sizes of the largest and smallest subsets is minimal. We may formalize this problem as a problem of integer linear programming. The goal is to find mt integers, $\{x_{i,j} : 1 \leq i \leq t, 1 \leq j \leq m\}$, where $x_{i,j}$ is the number of occurrences of a_j in D'_i, such that

1. $\sum_{i=1}^{t} x_{i,j} = p_j$, $1 \leq j \leq m$ (consistency);
2. $x_{i,j} \leq \frac{1}{\ell} \cdot \sum_{j=1}^{m} x_{i,j}$, $1 \leq i \leq t$, $1 \leq j \leq m$ (ℓ-diversity condition);
3. $\sum_{j=1}^{m} x_{1,j} \geq \sum_{j=1}^{m} x_{2,j} \geq \cdots \geq \sum_{j=1}^{m} x_{t,j}$ (monotonicity);
4. $\sum_{j=1}^{m} x_{1,j} - \sum_{j=1}^{m} x_{t,j}$ is minimized (target function).

The last minimality condition is requested because in the original sequential algorithm the set of records was split to equal-sized subsets; also, if we remove that minimality condition, then the trivial split, where $D'_1 = D'$ and $D'_i = \emptyset$ for all $i > 1$, will be a feasible solution.

This linear program has the following non-integral solution,

$$x_{i,j} = \frac{p_j}{t}\,, \quad 1 \le i \le t\,, 1 \le j \le m\,. \tag{23}$$

But, as we seek an integral solution, the number of possible roundings is typically exponential and consequently the integer linear program is NP-hard. Therefore, we select at random one of the possible rounded solutions. To that end, we define:

$$y_{i,j} = \begin{cases} \left\lceil \frac{p_j}{t} \right\rceil & 1 \le i \le p_j \bmod t\,, \\ \left\lfloor \frac{p_j}{t} \right\rfloor & p_j \bmod t + 1 \le i \le t\,. \end{cases} \tag{24}$$

Then, in order to select at random one of the possible rounded solutions in the neighborhood of the optimal non-integral solution (23), we select for each $1 \le j \le m$ a random permutation π_j on $\{1, \ldots, t\}$, and then set

$$x_{i,j} = y_{\pi_j(i),j}\,, \quad 1 \le i \le t\,, 1 \le j \le m\,. \tag{25}$$

This procedure is summarized in Algorithm 3.

Algorithm 3 ℓ-diversity respecting splitting procedure

```
input  A set of records $D||D' = \{R_1||S_1, \ldots, R_n||S_n\}$, an integer $t$.
output  A clustering of $D||D'$ into $t$ (almost) equal-sized clusters while attempting to pre-
    serve in each cluster the overall distribution of the private values in $D||D'$.
1:  Compute $m$, the number of distinct private values in $D'$, and $\{p_1, \ldots, p_m\}$, the number
    of occurrences of those values in $D'$.
2:  For each $1 \le j \le m$ generate an independent and random permutation $\pi_j$ on $\{1, \ldots, t\}$.
3:  For all $1 \le i \le t$ and $1 \le j \le m$ compute $x_{i,j}$ using Eqs. (24)+(25) and $\pi_j$.
4:  for $i = 1, \ldots, t$ do
5:    Set $C_i = \emptyset$.
6:    for $j = 1, \ldots, m$ do
7:      Select at random $x_{i,j}$ records from $D||D'$ with private value that equals the $j$th
        private value. Add them to $C_i$ and remove them from $D||D'$.
8:    end for
9:  end for
10: Output $C_1, \ldots, C_t$.
```

As π_j is selected at random, we get that the expected value of $x_{i,j}$ is

$$E(x_{i,j}) = \left\lceil \frac{p_j}{t} \right\rceil \cdot \theta_j + \left\lfloor \frac{p_j}{t} \right\rfloor \cdot (1-\theta_j)\,, \quad \theta_j := \frac{p_j \bmod t}{t}\,.$$

Therefore, the expected value of the size of the ith cluster is

$$E\left(\sum_{j=1}^{m} x_{i,j}\right) = \sum_{j=1}^{m} \left(\left\lceil \frac{p_j}{t} \right\rceil \cdot \theta_j + \left\lfloor \frac{p_j}{t} \right\rfloor \cdot (1-\theta_j) \right)\,. \tag{26}$$

As the right hand side in Equation (26) is independent of i, we see that the expected size of all clusters is the same. Hence, such a heuristic solution complies with our attempt to split the original set of records into clusters of similar sizes.

We would like to note that there exist more advanced algorithms for solving integer linear programs, e.g. the *branch and bound* or *branch and cut* methods. Such methods may be applied in the invocation of the splitting procedure in Step 1 of Algorithm 1, where the entire table is split to the initial clusters, since that stage occurs only once. However, it is too costly to apply such methods whenever we need to split a large cluster to two, since such splits occur many times throughout the execution of Algorithm 1.

8 Experiments

In Section 8.1 we describe experiments that demonstrate the advantages of the sequential clustering algorithm over other well known k-anonymization algorithms. In Section 8.2 we report additional experiments that examined properties of the sequential clustering. Section 8.3 is devoted to experimentation of the modified sequential clustering that respects ℓ-diversity. In Section 8.4 we describe the experiments that compare our proposed PMI measure to the MI measure.

8.1 Comparing the sequential to other k-anonymity algorithms

We tested the sequential clustering algorithm versus the following algorithms that we described in Section 6.2: The Mondrian algorithm, the agglomerative algorithm, k-member algorithm due to Byun et al., and the improved Hilbert-curve algorithm. The experiments were conducted on the dataset `Adult` from the UCI Machine Learning Repository.[2]. That dataset was extracted from the US Census Bureau Data Extraction System. It contains demographic information of a small sample of US population with 14 public attributes such as age, education-level, marital-status, occupation, and native-country. The private information is an indication whether that individual earns more or less than 50 thousand dollars annually. The `Adult` data contains 45,222 records after tuples with missing values are removed. The algorithms were implemented in C and ran on a Pentium (R) 4 CPU 3.40 GHz, 1.49 GB of RAM.

In the experiments that we report herein we restricted our attention to generalization by suppression. In addition, we implemented a non-repetitive version of the sequential algorithm, since, as we report later on, the repetitive version that performs the sequential clustering several times, each time starting with a different random initial clustering, does not offer a significant improvement in the results. We ran each of the five algorithms with seven values of the anonymity parameter, $k = 10, 20, 30, 40, 50, 75, 100$. The LM-utility results are given in Figure 1. (The plot does not include a curve for the agglomerative algorithm since it issued results almost identical to those of the sequential algorithm.) The corresponding runtimes are given in Figure 2. (Here we do not include the runtime of the agglomerative algorithm since it was much higher than that of the other algorithms.)

We tested also the scalability of the algorithms. We ran them on extracts of different sizes from the `Adult` dataset. The runtimes (for the case $k = 50$) are shown in Figure 3.

We see that the sequential and agglomerative algorithms offer the best results in terms of utility, while the algorithm due to Byun et al. offers similar results, with only slightly smaller utility values. Those three algorithms significantly outperform the Mondrian and the improved Hilbert algorithms. In terms of runtime, the latter two are much faster than the first three. From among the first three algorithms, the sequential algorithm offers the

[2]http://mlearn.ics.uci.edu/MLSummary.html

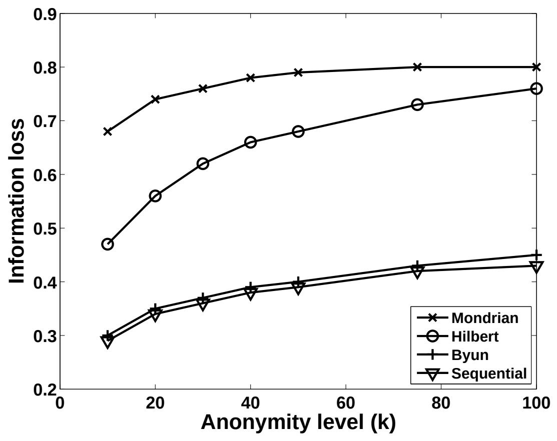

Figure 1: Algorithm comparison – utility

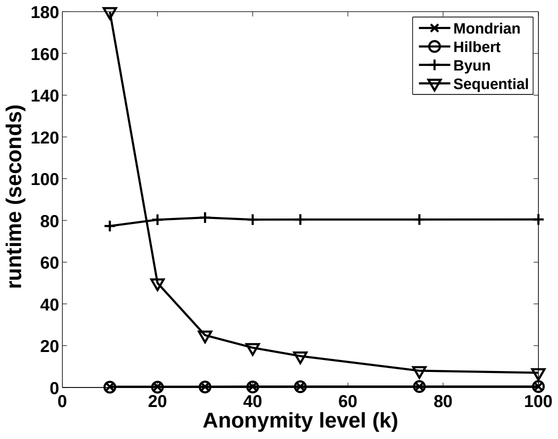

Figure 2: Algorithm comparison – runtime

best runtime for most values of k. The runtime of the agglomerative and k-member algorithms remains $O(n^2)$, almost independently of k. The runtime of the sequential algorithm, on the other hand, decreases with k. Since the number of clusters is $O(n/k)$, then each pass over all records in the table involves $O(n^2/k)$ computations of utility gain by moving a record from one cluster to another. The faster than $O(1/k)$ decrease in the runtime stems from the fact that the number of iterations also reduces with k. Since, in practice, higher values of k are required for greater privacy, the advantage offered by the sequential algorithm over the agglomerative and k-member algorithms in terms of runtime is prominent.

8.2 Additional testing of the sequential clustering algorithm

In the previous section we used a non-repetitive version of the sequential algorithm, since the algorithm exhibits very weak dependence on the random choices that it makes. In the next set of experiments, we ran the sequential algorithm ten times on the same input with

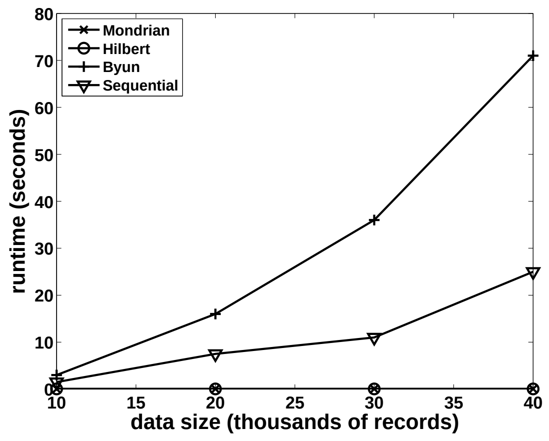

Figure 3: Algorithm comparison – scalability

the same value of k. Table 2 shows the minimal value of the LM cost measure, the average value and the standard variation along those ten repetitions. As can be seen, while repeated runs do converge to different local minima, the change in the utility of the output is small. Hence, repeated runs may be attempted, but they probably might not lead to a significant improvement over the results of the non-repetitive version.

k	minimum	average	standard deviation
10	0.298	0.302	0.03
20	0.338	0.340	0.03
30	0.361	0.364	0.03
40	0.378	0.380	0.03
50	0.390	0.394	0.03
60	0.416	0.419	0.04
100	0.433	0.439	0.04

Table 2: LM results over repeated runs of sequential clustering

Next, we tested the dependence of the runtime of the algorithm on the dimension r, which is the number of quasi-identifiers. We ran the sequential clustering algorithm on extracts of 5,8,11 and 14 quasi-identifiers from the `Adult` dataset (the original number of attributes is 14). Figure 4 shows the runtime as a function of r for $k = 20$ and $k = 50$. As expected, the dependence is roughly linear.

8.3 Testing the diversity-respecting version of the algorithm

We implemented the modified version of the sequential clustering algorithm that was described in Section 7 in order to examine the effects of adding the diversity constraint on the utility of the output anonymization.

The `Adult` database has a binary private attribute. Out of 45222 records, 11208 records have a zero private value and 34014 have one as their private value. Hence, the table respects ℓ_0-diversity with $\ell_0 = 1.3295$. We ran the modified version of the sequential clustering algorithm on the `Adult` database with several values of the diversity parameter ℓ.

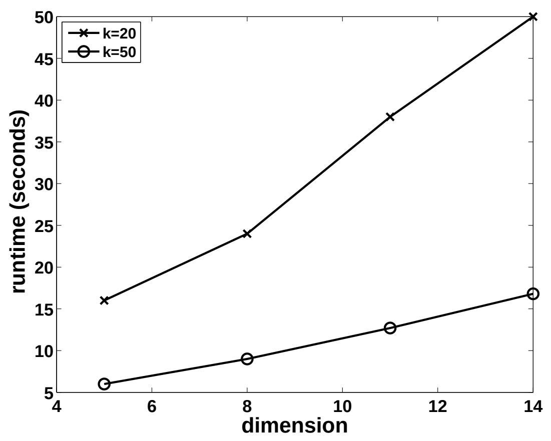

Figure 4: Runtime versus dimension

Figure 5 shows the resulting LM information loss values when the value of k was set to $k = 50$. Each point on the curve corresponds to one experiment, where the horizontal coordinate indicates the resulting measure of diversity of the output (namely, the diversity of the least diverse cluster in the output), while the vertical coordinate indicates the LM information loss. As can be seen, when $\ell = 1$ (namely, when no diversity restriction is imposed), we recover the result of the basic algorithm, as reported earlier in Figure 1. (In both experiments the information loss was roughly 0.4; the slight difference stems from the randomness of the algorithm.) The quality of the output, in terms of the LM information loss, decreases when the diversity demand increases. The highest level of diversity is very close to the theoretical bound of $\ell_0 = 1.3295$. The information loss in the most diverse anonymization is still smaller than the information loss of the output of the modified Hilbert and the Mondrian algorithms (as reported in Figure 1) in which no diversity demands were imposed.

As another illustration of the dependence of the information loss on diversity we repeated the above experiment when the education attribute was considered sensitive and the remaining 13 attributes served as the public ones. The education attribute has 16 possible values, among which 'Bachelors', 'Some-college' and 'HS-grad' are relatively frequent. The entire table respects ℓ_0-diversity with $\ell_0 = 3.06$. Figure 6 shows the resulting LM information loss values when the value of k was set to $k = 50$. The information loss in the most diverse anonymization is similar to the information loss of the output of the modified Hilbert algorithm (0.645) and less than the Mondrian (0.735) in which no diversity demands were imposed.

8.4 Comparing the PMI and MI measures

After establishing the superiority of sequential clustering over agglomerative clustering, we proceeded to test the PMI utility measure and compare it to the MI measure. We exemplify the effects of using the PMI utility measure, as compared to the MI measure, through the entropy diversities of the anonymized clusters. Given a cluster of anonymized records, its entropy diversity is the entropy of the distribution that it induces on the private attribute [17]. For example, if all records in that cluster have the same private value then that clus-

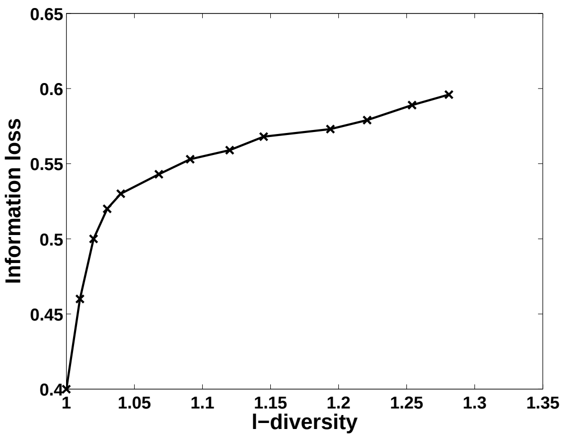

Figure 5: Information loss versus the diversity of the anonymization

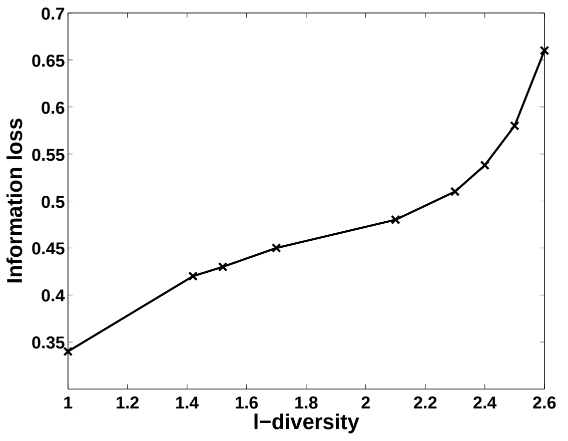

Figure 6: Information loss versus the education-diversity of the anonymization

ter's diversity is zero; but if, on the other hand, all records have a unique private value then the diversity is $\log m$ where m is the cluster size. On one hand, we wish to arrive at a clustering in which every cluster has a low diversity since that would indicate a strong correlation between the generalized public data and the private data. On the other hand, a too low diversity (that helps learning) might jeopardize the privacy of the individuals in that cluster. Therefore, Machanavajjhala et. al. [17] suggested to impose a minimal diversity as a privacy measure.

We ran the sequential clustering on the `Adult` database with a weighted MI measure,

$$U_{wMI} = w \cdot U_{MI} + (1-w) \cdot U_{PMI} \, ,$$

with $w = 0, .25, .5, .75, 1$, for $k = 50, 75, 100$. The average diversities of the resulting clusterings in each of those cases are shown in Table 3.

We see that when $w = 0$ (which corresponds to the PMI measure) the correlation between the generalized public data and the private one is much stronger than in the case $w = 1$

$k\backslash w$	0	0.25	0.5	0.75	1
50	0.07	0.14	0.31	0.51	0.54
75	0.08	0.14	0.32	0.51	0.56
100	0.08	0.15	0.34	0.54	0.58

Table 3: Average diversities for different values of k and w.

(which corresponds to the MI measure). Hence, it is apparent that anonymizations that were obtained by using the PMI measure clearly are more valuable for mining association rules.

It should be pointed out that in all of our experiments (either with the above U_{wMI} measures, for all values of w, or with other measures such as the LM) there were clusters with zero diversity. Hence, the problem that was identified in [17] does occur, regardless of the utility measure (or the clustering algorithm). Hence, it is necessary to impose also ℓ-diversity, as described in Section 7. Therefore, using the PMI measure, as opposed to the MI or other utility measures, could help minimizing the diversities, within the ℓ-diversity condition, and hence increase the utility of the resulting anonymized tables.

9 Conclusions

In this study we proposed the private mutual information (PMI) utility measure that aims at maximizing the correlation between the generalized public data and the private data. We showed that this measure is much more adequate for the purposes of data mining that aims at finding association rules to predict the private data from the public data. We then described the sequential clustering algorithm. That algorithm, which is independent of the underlying utility measure, appears to be the algorithm of choice for efficiently finding k-anonymizations with high utility, as indicated by experimental comparison with other popular algorithms.

In a recent work [24], other significant advantages of the sequential clustering algorithm were demonstrated. That study considered the problem of k-anonymizing distributed databases. Given a database that is partitioned between several sites, either horizontally or vertically, it is required to devise secure distributed anonymization algorithms that allow the different sites to obtain a k-anonymized view of the union of their databases, without disclosing sensitive information. The algorithms in [24] are based on the sequential clustering algorithm, which offers anonymizations with utility that is significantly better than that offered by previous algorithms that were implemented in the distributed setting (the Mondrian algorithm [15] and the approximation algorithm of [18]). The distributed versions of the sequential algorithm apply to any number of sites, any generalization technique and any utility measure, and can support ℓ-diversity. The most important advantage of basing the distributed algorithm on sequential clustering is that such a solution relies on minimal cryptographic assumptions, as opposed to previous distributed algorithms that depend on costly cryptographic primitives.

Our initial experiments regarding the diversity show that the PMI measure is much more suitable when the goal is to achieve anonymizations from which association rules or methods of predicting the private data from the public data can be mined. A more thorough experimental validation of this claim will proceed as follows: We intend to obtain several k-anonymizations of the same table using different measures of information-loss. Then each of those tables will be used either for mining association rules or for the computation of a classifier. Our conjecture, which is supported by our initial experiments that we reported here, is that the PMI-related table will produce a set of association rules which is closer to the set of association rules that can be mined from the original table; also, the PMI-derived classifier is believed to be more accurate than a classifier that is based on anonymizations that are based on other measures of information-loss.

References

[1] G. Aggarwal, T. Feder, K. Kenthapadi, S. Khuller, R. Panigrahy, D. Thomas, and A. Zhu. Achieving anonymity via clustering. In *PODS*, 2006.

[2] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Approximation algorithms for k-anonymity. *J. of Privacy Tech.*, 2005.

[3] R. Agrawal and R. Srikant. Privacy-preserving data mining. *ACM SIGMOD Record*, 29(2), 2000.

[4] R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, 2005.

[5] J.W. Byun, A. Kamra, E. Bertino, and N.Li. Efficient k-anonymization using clustering techniques. In *DASFAA*, 2007.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley, New York, 1991.

[7] J. Domingo-Ferrer and V. Torra. A critique of k-anonymity and some of its enhancements. In *ARES*, 2008.

[8] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis. A framework for efficient data anonymization under privacy and accuracy constraints. *ACM Trans. Database Syst.*, 34, 2009.

[9] A. Gionis, A. Mazza, and T. Tassa. k-Anonymization revisited. In *ICDE*, 2008.

[10] A. Gionis and T. Tassa. k-Anonymization with minimal loss of information. *IEEE Trans. Knowl. Data Eng.*, 21, 2009.

[11] V. Iyengar. Transforming data to satisfy privacy constraints. In *SIGKDD*, 2002.

[12] B. Kenig and T. Tassa. A practical approximation algorithm for optimal k-anonymity. *Submitted*.

[13] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *ICDM*, 2006.

[14] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *ACM-SIGMOD International Conference on Management of Data (SIGMOD)*, pages 49–60, 2005.

[15] K. LeFevre, David J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *International Conference on Data Engineering (ICDE)*, 2006.

[16] S.P. Lloyd. Least square quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[17] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-Diversity: privacy beyond k-anonymity. In *ICDE*, 2006.

[18] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *PODS*, 2004.

[19] B. Moon, H. Jagadish, and C. Faloutsos. Analysis of the clustering properties of the Hilbert space-filling curve. *The New England Journal of Medicine*, 13(1):124–141, 2001.

[20] M. E. Nergiz and C. Clifton. Thoughts on k-anonymization. In *ICDE Workshops*, 2006.

[21] H. Park and K. Shim. Approximate algorithms for k-anonymity. In *SIGMOD*, 2007.

[22] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *PODS*, 1998.

[23] N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *ACM SIGIR*, 2002.

[24] T. Tassa, I. Tamir, and E. Gudes. Secure distributed computation of anonymized views of shared databases. *Submitted*.

[25] R.C.W. Wong, J. Li, A.W.C. Fu, and K. Wang. (α, k)-anonymity: An enhanced k-anonymity model for privacy preserving data publishing. In *KDD*, pages 754–759, 2006.

[26] X. Xiao and Y. Tao. Anatomy: Simple and Effective Privacy Preservation. In *International Conference on Very Large Data Bases (VLDB)*, 2006.