Disclosure Risks of Distance Preserving Data Transformations

E. Onur Turgay, Thomas B. Pedersen, Yücel Saygın,
Erkay Savaş, and Albert Levi

Sabanci University
Istanbul, 34956, Turkey
onurturgay@su.sabanciuniv.edu,
{pedersen,ysaygin,erkays,levi}@sabanciuniv.edu

Abstract. One of the fundamental challenges that the data mining community faces today is privacy. The question "How are we going to do data mining without violating the privacy of individuals?" is still on the table, and research is being conducted to find efficient methods to do that. Data transformation was previously proposed as one efficient method for privacy preserving data mining when a party needs to outsource the data mining task, or when distributed data mining needs to be performed among multiple parties without each party disclosing its actual data. In this paper we study the safety of distance preserving data transformations proposed for privacy preserving data mining. We show that an adversary can recover the original data values with very high confidence via knowledge of mutual distances between data objects together with the probability distribution from which they are drawn. Experiments conducted on real and synthetic data sets demonstrate the effectiveness of the theoretical results.

1 Introduction

Data mining technology proved its success in many areas such as health, life-sciences, and security. On the other hand, the popularity of data mining ignited heated debates on the privacy aspects especially after the launch of large scale projects related to homeland security. In fact, some projects were stopped since they failed to meet the privacy concerns. According to a very recent article in Computer World by Jaikumar Vijayan "The chairman of the House Committee on Homeland Security, has asked Department of Homeland Security Secretary Michael Chertoff to provide a detailed listing of all IT programs that have been canceled, discontinued or modified because of privacy concerns" [15]. In addition to that, the Chairman also asked for information about the measures being taken to address privacy issues [15].

Measures to address privacy issues can be as simple as not collecting privacy sensitive information at all. Unfortunately, in many applications this is not possible. Therefore, advanced protocols based on statistics and cryptography are proposed to ensure privacy. Privacy preserving data management in general, is

B. Ludäscher and Nikos Mamoulis (Eds.): SSDBM 2008, LNCS 5069, pp. 79–94, 2008.
© Springer-Verlag Berlin Heidelberg 2008

still an ongoing research topic, and efficient as well as secure methods without strong assumptions are yet to be proposed. In fact, recent results showed that the data sets transformed with perturbation based techniques can be recovered by a principle component analysis based attack[7]. In this paper, we present a general attack which is applicable in cases where only the pairwise distances among objects are known.

The main contribution of this paper is to demonstrate that *any* data transformation, from which an attacker can learn the mutual distances between data objects may disclose private information: (1) If the attacker knows a few of the data objects in the database, he can recover *all data perfectly*, (2) If the attacker has *a-priory knowledge* of the probability distribution from which the data is drawn, he can recover *all data* with *high precision*. Our attack is based on an attack of Liu *et al.*[7], but is improved so that it is applicable even to the privacy preserving method which was proposed in [8] to prevent the attack from [7]. Our attack is also improved in the sense that it is applicable to a wider range of scenarios than the attack of Liu *et al.*. We demonstrate the attack with known probability distribution on the Adult Census dataset from the UCI Machine Learning Repository [13], which is the same dataset used in [8], and show that the original data can be recovered with an error as low as 2%.

2 Related Work

Data perturbation is a widely used technique for privacy preserving data mining. Additive perturbation techniques proposed by Agrawal and Srikant are based on adding random noise to the original data which can then be filtered to recover the distribution of the original data[3]. Another scenario is where a group of organisations would like to perform collective data mining but would not like to share their data. An encryption based protocol for privacy preserving association rule mining in distributed environments is proposed in[6]. Similarly, secure multiparty computation based methods are applied to privacy preserving clustering in distributed environments[14]. In [2], authors propose an approach for privacy preserving data mining which maps the original data set into a new anonymised data set preserving the correlations among the different dimensions.

Security of random perturbation methods against partial disclosure through successive querying of the database by snoopers is studied in [10]. The effect of high dimensionality in randomisation was studied by Aggarwal in [1].

Many techniques for classification such as clustering only relies on the mutual distances between the objects in the database. In consequence several privacy preservation techniques which preserves mutual distances have been proposed. The authors of [4,5,11] have proposed perturbation techniques based on geometric transformations such as translation, rotation, and re-scaling of the dataset. With the exception of rescaling, these operations preserve distances. Even rescaling, while it does not preserve the exact distances, preserves the relative distances. Oliveira and Zaïane, propose techniques for securely computing the distances between each pair of data objects, and only reveal the resulting

dissimilarity matrix to the third party, who can then perform clustering[12]. Oliveira and Zaïane prove that the dissimilarity matrix alone does not violate privacy *with the assumption that the attacker does not have domain knowledge*. However, in cryptography the well established Kerckhoffs' principle states that the security of a system must never rely on keeping the algorithm and/or the data secret — the only secret should be an easily exchangeable cryptographic key. The role of this principle in the case of privacy preserving data mining is well understood in the words of Bruce Schneier [9]:

> "Kerckhoffs' principle applies beyond codes and ciphers to security systems in general: every secret creates a potential failure point. Secrecy, in other words, is a prime cause of brittleness and therefore something likely to make a system prone to catastrophic collapse. Conversely, openness provides ductility."

While privacy is preserved in [12] when the adversary has no domain knowledge at all, it is unclear what happens if the adversary gains partial knowledge of the domain. A party involved in data mining, for instance, is likely to know the layout of the tables in the database, and anyone can easily gain access to national statistics about age, sex, income, e.t.c. Relying on this information to be kept secret from the adversary is unrealistic, and clearly violates Kerckhoffs' principle. Notice that knowing the distribution of the data is not the same as knowing the data. Even though anyone can see the distribution of patients with cancer according to e.g. age and income, we do not want anyone to learn the identity of a specific individual with cancer. In this paper we demonstrate that in a worst case scenario a secret database can be reconstructed very accurately if the adversary knows the table layout and knows the distribution of the data.

Our attack is based on the work by Liu *et al.*, where the authors point out that perturbation techniques which preserve distance between data objects can be attacked if the attacker knows a small set of data selected according to the same probability distribution as the original data set[7,8]. The attack applies principal component analysis to the perturbed data, and tries to fit it to the known data set. Liu *et al.* also propose an alternative transformation where the objects in the original data set are projected onto a subspace in a way that distance is preserved with high probability. They point out that the alternative approach is secure against the identified attack, but may not be secure against other attacks. Our attack is applicable to a wider range of scenarios than the attack of [7], since the attacker does not need the entire perturbed dataset: only the mutual distances and information about the probability distribution from which objects are chosen. Our attack is more general since: (1) In many cases the information about the probability distribution can be obtained from alternative sources (i.e. national statistical agencies), and (2) only the mutual distances from the original dataset are needed (not a perturbation of every object). Our attack also have some improvements in the computational cost: Our attack is polynomial in the number of attributes, whereas the attack in [7] is exponential in the number of attributes.

3 Problem Formulation

Throughout this paper we let n be the number of objects (i.e. rows) in the *target database* which is attacked. Each object in the database has d attributes. In other words, each object can be thought of as a vector in a d-dimensional vector space. For simplicity we assume that all attributes are from an alphabet Ω. We model the objects as random variables $X_1, \ldots, X_n$. We assume that these random variables are independent and identically distributed according to a *global probability distribution*, and let $P(x)$ denote the probability that $X_i = x$, for all $i \in \{1, \ldots, n\}$.

3.1 Distance Preserving Transformations and Dissimilarity Matrices

In this paper we show how to attack any distance preserving data transformation. The only thing we need for our attack are the pairwise distances between the objects in the database. We represent this information by a dissimilarity matrix as described below. We are not concerned with the actual transformation, or whether the data is centralised or distributed.

The dissimilarity matrix is an $n \times n$ matrix which contains the distances between each pair of data objects. We can describe the dissimilarity matrix as random variable D, which depends on the random variables $X_1, \ldots, X_n$ in that $D_{ij} = |X_i - X_j|$, for all $i, j \in \{1, \ldots, n\}$.

In our experiments we use databases containing numerical and boolean attributes, since they have well-defined distance measures. We use the Euclidean distance, and assign the values 1 and 0 to boolean values *true* and *false*, respectively. Textual and nominal data requires extra work, and are not addressed in this paper.

In data mining applications it is common to normalise the attributes before analysing the data. Normalisation prevents attributes of large magnitudes to dominate the small scaled attributes. In our work we assume that all attributes are normalised.

3.2 Motivating Scenario

Dissimilarity matrices of objects having only one attribute is a simple special case, where the distances between objects are equal to the differences in their attributes. In this section we will briefly study this special case to get some intuition.

Suppose we have a database containing the ages of randomly selected individuals within a country. For samples of sufficient sizes, it is acceptable to assume that the database has the same probability distribution of ages as nationwide.

Suppose we have a database of 5 individuals, x_1, x_2, x_3, x_4, x_5, with discrete ages 25, 95, 4, 60, 32, respectively. The corresponding dissimilarity matrix can be seen in Table 1.

If we know the age of two individuals, x_1 and x_2, say, we can easily find the age of all the other individuals: namely the unique age at the given distance

Table 1. Dissimilarity matrix of the age of 5 individuals

	x_1	x_2	x_3	x_4	x_5
x_1	0	70	21	35	7
x_2		0	91	35	63
x_3			0	56	28
x_4				0	28
x_5					0

from the two known ages. This corresponds to the "Hyper-lateration" attack described in Sec. 4 below.

To recover the ages from the dissimilarity matrix only, we start by finding the biggest distance in the matrix — in this case 91. The two individuals with biggest distance (x_2 and x_3) defines the boundaries of the database: one of them is the youngest, while the other is the oldest. We assign the age zero to any of the two points, x_2, say. Now the age of all the other individuals is their distance to x_2 (since all ages are known to be positive). The resulting ages can be seen in Table 2.

Table 2. Ages, if x_2 is assumed to be 0 years

x_2	x_4	x_5	x_1	x_3
0	35	63	70	91

Suppose that we know that more than half of the population is younger than 40 years. In that case the ages in Table 2 do not fit the probability distribution of the population — we have most likely chosen the wrong person as the youngest. When flipping the ages of x_2 and x_3 we get the ages seen in Table 3, which fits the global probability distribution better.

Table 3. Ages, if x_3 is assumed to be 0 years

x_3	x_1	x_5	x_4	x_2
0	21	28	56	91

Now that we have a good candidate dataset, the histogram of the candidate dataset is compared with the global probability distribution and the *statistical distance* between the two is computed (in our example, the dataset is too small to plot the histogram). By shifting the candidate dataset by small amounts (in this case by 1 year), and computing the statistical distance of the resulting probability distributions to the global probability distribution, we can find the best fitting candidate dataset. The shift trials are conducted until the oldest individual in the candidate dataset reaches the maximum possible age the global probability

distribution (at 120 years, say). The candidate dataset that has the minimum statistical distance to the global probability distribution is chosen as the winning set.

In Sec. 5 we give an attack which can rotate a multidimensional candidate dataset to the true dataset with low error.

3.3 Attack Scenarios

Our underlying model is that a secret data base of n objects is drawn according to a global probability distribution of d dimensional vectors. An attacker is given the dissimilarity matrix of the data base. Besides the dissimilarity matrix he has some extra information available. This information can be:

Known sample. An attacker might be able to learn a few of the objects in the data base. If he knows at least $d + 1$ objects (and knows the corresponding entries in the dissimilarity matrix) he will be able to reconstruct the data base with high probability, as described in Sec. 4.

Known probability distribution. The probability distribution from which the objects are drawn may be known to an attacker. In Sec. 5 we show an attack using this information.

There are several ways in which an attacker might recover the necessary information. To get a known sample of the database, an attacker might get insider information from a person within the organisation which owns the database, or he might be able to inject information. In some cases it may even be realistic to assume that the attacker already knows some entries in the database (he had an operation in the hospital which has the target database of medical data). Knowledge of the global probability distribution can, in some cases, be obtained from national statistical societies. In other cases the attacker could be in possession of his own database with objects drawn from the same global probability distribution (a competing hospital).

Finally we assume that an attacker knows the schema of the database. The schema of the database will often depend on the software which is used by the organisation, and may be readily available. It may also follow public standards.

3.4 Principal Component Analysis

In the scenario where the attacker does not have a known sample from the database, but has knowledge about the global probability distribution of the data, we apply principal component analysis (PCA). PCA is a statistical method which identifies correlations in a dataset. It takes a dataset of random variables drawn from the d-dimensional vector space, and creates a vector basis which is best suited to represent the dataset. The basis is such that when data is projected onto the subspace spanned by "the most significant" basis vectors, only little information is lost.

More precisely, PCA computes the covariance matrix of the dataset. On entry $(i, j) \in \{1, \ldots, d\}^2$ the covariance matrix has the covariance

$$\mathrm{Cov}(X_i, X_j) = \mathrm{E}((X_i - \mu_i)(X_j - \mu_j)), \tag{1}$$

where μ_i and μ_j are the expected values of X_i and X_j, respectively. The eigenvalues and the corresponding eigenvectors of the covariance matrix are then computed. The eigenvectors (i.e. principal components) form the new vector basis, which we refer to as the eigen-basis. The eigenvectors with the largest eigenvalues are the most significant components, and point in the direction of the highest correlation.

The central observation is that the eigenvalues do not change when the dataset is rotated and/or mirrored. In particular, a dataset which is obtained by hyperlateration will have the same eigenvalues as the original dataset, and we expect that rotating the corresponding eigen-basis to the original eigen-basis will recover the data.

4 Attacking with Known Sample

In this section we assume that the attacker has a known sample of at least $d+1$ objects from the data base, and knows the corresponding entries in the dissimilarity matrix.

Given three points in a two dimensional vector space, which do not lie on a line, a fourth point with known distances to these three points can be placed by triangulation or trilateration. Trilateration generalises to points in a d-dimensional vector-space, so that we can uniquely place a point with known distances to $d+1$ distinct points, which span the vector-space. We call this procedure "hyperlateration". Applying hyper-lateration to our case; if we have a database with d attributes and n objects, and we are only given the dissimilarity matrix, we can find the original data if we can correctly place $d+1$ distinct points (which span the full vector-space). In most databases there are considerably more data objects than attributes. We thus reduce the complexity of guessing all n objects in the database to guessing or obtaining $d+1 \ll n$ objects.

4.1 Hyper-lateration

We now give the algorithm for hyper-lateration. Given $d+1$ reference points, $p_0, \ldots, p_d$, in $\mathbb{R}^d$ the following theorem gives us a point x at distance δ_i to point p_i, for $i = 0, \ldots, d$.

Theorem 1. *Let $p_0, \ldots, p_d$ be $d+1$ distinct points which span $\mathbb{R}^d$. Any point x is uniquely determined by the set of distances $\{\delta_i\}_{i=0}^d$, where δ_i is the distance from x to point p_i, for $i \in \{0, \ldots, d\}$.*

Proof. Hyperlateration is the task of solving the equations

$$\delta_i^2 = \sum_{j=1}^{d} (\overline{x}_j - \overline{p}_{ij})^2 = \sum_{j=1}^{d} \overline{x}_j^2 - 2\overline{x}_j\overline{p}_{ij} + \overline{p}_{ij}^2, \tag{2}$$

for all $i \in \{0, \ldots, d\}$. These equations are simplified by subtracting the equaiton $\delta_0^2 = \sum_{j=1}^{d} \overline{x}_j^2 - 2\overline{x}_j\overline{p}_{0j} + \overline{p}_{0j}^2$ from all the other equaitons:

$$\delta_i^2 - \delta_0^2 = \sum_{j=1}^{d} 2\overline{x}_j(\overline{p}_{0j} - \overline{p}_{ij}) + \overline{p}_{ij}^2 - \overline{p}_{0j}^2, \tag{3}$$

for all $i \in \{1, \ldots, d\}$. The result is a system of d linear equations which we write as

$$M\overline{x} = \overline{d} + \overline{\delta}, \tag{4}$$

where $M_{ij} = 2(\overline{p}_{0j} - \overline{p}_{ij})$, and $\overline{d}_i = \sum_{j=1}^{d}(\overline{p}_{0j}^2 - \overline{p}_{ij}^2)$ only depend on the known points, and $\overline{\delta}_i = \delta_i^2 - \delta_0^2$ depends on the distances. This system of equations has a unique sollution exactly when M is non-singular, which is the case if and only if $\overline{p}_1, \ldots, \overline{p}_d$ are linearly independent.

4.2 The Hyper-lateration Attack

If an attacker knows a sample of $d+1$ objects from the database, he may be able to recover the entire database if he sees the dissimilarity matrix. The success of this attack depends on two things: (1) the known objects should be represented by distinct points which span the full vector-space, and (2) the attacker must know the corresponding entries in the dissimilarity matrix (i.e. he must know the distances between the known objects and any other object). If these two conditions are met, the attacker will be able to fully recover the database without any error.

The attack consists of the following steps:

1. Pre-compute the matrix M, it's inverse, and the vector $\overline{d}$ from Eq. 4.
2. For each row in the dissimilarity matrix, which corresponds to an unknown object, compute the vector $\overline{\delta}$ from Eq. 4, and solve the system of linear equations.

The first step can be done in time $O(d^3)$, while the second step (assuming that $n > d$) requires time $O((n-d)d^2)$. The overall time is $O(d^2 n)$.

5 Attacking without Known Sample

If the attacker does not have a known sample of the target database, he can still attack the dissimilarity matrix if he knows the "shape" of the data. In this section we show how the attacker can map a candidate dataset obtained from the dissimilarity matrix into the real data. The attacker obtains a candidate dataset by randomly fixing $d+1$ points so that they are consistent with the dissimilarity matrix. This gives a candidate dataset where the relative position of all points is true (up to mirroring in any axis) — in other words: a perturbation of the original data in the target database. By applying a principal component analysis attack, similar to the one presented by Liu *et al.*[7], to the candidate dataset, we show how the candidate dataset can be mapped to a dataset which fits well to the real data of the target database.

5.1 The PCA Attack

In the PCA attack the attacker is given only two pieces of information:

- The dissimilarity matrix of the database.
- A representation of the global probability distribution from which the data is drawn.

Given this data, the goal of the attacker is to reconstruct the secret database.

Generating a candidate dataset, which is consistent with the dissimilarity matrix, is straightforward by hyper-lateration as described in Sec 4.1. Unfortunately the candidate dataset found by hyper-lateration can be an arbitrarily rotated and mirrored version of the real data. Such a rotation and mirroring of a dataset can be seen as a perturbation of the dataset. Liu *et al.* proposed using PCA to rotate and mirror a perturbed dataset back to its original position by comparing the principle components of a known sample from the same global probability distribution with the perturbed data's principle components and then generate a rotation matrix which rotates the perturbed data to the actual position by a simple matrix multiplication[7]. The principle components generated by PCA are very good representatives of the general shape of the probability distribution, as they directly depend on the variances and covariance values of individual attributes. As the distance matrix we are attacking is assumed to be of a data mining application, the correlations between variables will most likely generate strong principle components. The results of [7] can directly be applied to our case, since the result of our hyper-lateration process is special case of data perturbation.

Although one of the main uses of PCA is to project data to lower dimensions without loosing statistical information, we only use it to find principle components of both the candidate dataset and the known probability distribution, and try to construct a rotation matrix to match the principle components of these two probability distributions.

One limitation of PCA is that it is invariant under mirroring of the data. In other words: when using PCA to rotate the candidate dataset, we may end up with a dataset which is mirrored along any of the principal components. There are 2^d possible mirror images of which we have to find the one that matches the global probability distribution best. To test the quality of a candidate we compute the statistical distance between the probability distribution which can be computed from the candidate dataset to the real probability distribution. For a candidate dataset C and global probability distribution P we compute

$$\delta(C, P) = \sum_{v \in \Omega^d} \left| \frac{\#v}{\|C\|} - P(v) \right|, \tag{5}$$

where $\#v$ is the number of occurrences of v in C (this can be computed efficiently by only summing over v which occur in C).

Our attack can be described in the following steps:

1. Perform hyper-lateration on the dissimilarity matrix, to get a candidate dataset.

2. Compute the covariance matrices of the global probability distribution, and the candidate dataset.
3. Find eigenvalues and eigenvectors of the two matrices.
4. Match the eigenvectors of the two covariance matrices pairwise, and find a rotation which will rotate the eigen-basis of the candidate dataset to the eigen-basis of the global probability distribution.
5. For each eigenvector, measure the statistical distance between the known probability distribution and the candidate dataset obtained from both directions of the eigenvector — choose the one that is closest to the known distribution, and continue to next eigenvector.

Steps 2 and 3 comprises the PCA. Notice that since the target database is drawn from the global probability distribution, the covariance matrix made from target database should match the covariance matrix of the global probability distribution fairly accurately.

Our attack differs from the attack of Liu *et al.* on two points:

- Liu *et al.* do an exhaustive search amongst all 2^n possible mirroring of the eigenvectors, whereas we find the direction of eigenvectors one at the time.
- Liu *et al.* use multivariate two-sample hypothesis test to find the best candidate dataset. We compute the statistical distance defined in Eq. 5. The approach by Liu *et al.* requires time $O((n+d)^2)$, whereas our approach can be done in time $O(n \log n)$ (sort the vectors in the candidate dataset and count their frequencies).

Step 1 has time complexity $O(d^2 n)$. Step 2 also takes time $O(d^2 n)$ (we assume that the covariance matrix of the global probability distribution has been precomputed) and Step 3 takes time $O(d^3)$. Finding the rotation in Step 4 takes $O(d^2 n)$. The final step has time complexity $O(dn \log n)$. In total our attack has time complexity $O(d^3 + d^2 n + dn \log n)$ (which is $O(n^3)$ when $d < n$).

5.2 Characteristics of Vulnerable Datasets

While our attack is based on a statistical method for mapping the hyper-latereted data to the global probability distribution, the characteristics of data can change the accuracy of the output considerably. For example, a dataset with a circular shape (no correlation between attributes) in 2-dimensional space cannot be mapped to its original position by using PCA.

The covariance matrix is the main identifier of the success of our attack, as it is the only input to PCA. Its eigenvalues and eigenvectors define the alignment of data and are the basis for finding the rotation which maps a candidate dataset to the real data of the target database. We therefore study the connection between the properties of the covariance matrix and the success of the attack. The covariance matrix contains the covariance values between each pair of attributes and the variances of single variables in its diagonal.

In order to see the effect of the covariance values, we implemented an algorithm that constructs a multivariate Gaussian distribution which has a given covariance matrix (see Sec. 5.2 below). By using this tool, we can observe error rates with different configurations of the covariance matrix.

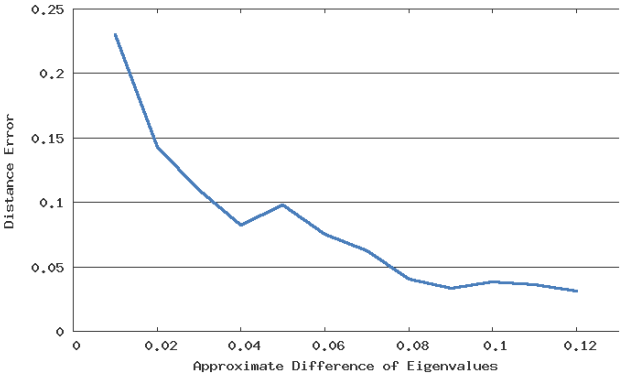

Fig. 1. Effect of eigenvalue differences (synthetic data)

Recall that the PCA-based attack works by finding the eigen-basis of the candidate dataset, and comparing it to the eigen-basis of the known properties distribution. In order to match the eigenvectors pairwise, all the corresponding eigenvalues must be different. We thus expect that the attack works best, when there is a large average difference between the eigenvalues of the covariance matrix (so that they are easily paired with the eigenvalues of the known probability distribution). To test this, we constructed data with difference covariance matrices, and plotted the average distance between eigenvalue versus the error percentage of the output of the attack. As can be seen from the result of the tests given in Fig. 1, there is a clear relationship between the eigenvalue difference and the error percentage. When the difference between the eigenvalues grows, the error drops. On the other hand, when the eigenvalues are very similar, the error percentage increases dramatically.

A dataset has a high average difference between eigenvalues of the covariance matrix when the correlations between different pairs of attributes differ. In other words: datasets where some attributes are highly correlated, while others are only weakly correlated are more vulnerable to the PCA attack.

In some scenarios it may not be realistic to assume that an attacker has access to statistical data which contains the correlations between attributes. If, for instance, the attacker can only obtain statistics for each attribute independently, he will not be able to apply the PCA attack, since his description of the global probability distribution does not contain the correlations necessary for the PCA attack to work. In this case, however, other methods may be applies. Recall that PCA is only used to recognise "geometrical characteristics" of the dataset which can be used to rotate the candidate dataset to the real data. Since we know that our hyper-latereted candidate dataset has *the same shape* as the original data — up to rotation, displacement, and mirroring, any technique which can give a simple representation of the shape of a dataset, can be used to find a rotation from the candidate dataset to the real data. Instead of using PCA an attacker may try to recognise each attribute independently in the candidate dataset. We leave it to future work to find alternatives to PCA.

Generating Synthetic Datasets. To characterise the vulnerable datasets, we generated synthetic datasets with a given covariance matrix, V. We do this as described in this section.

A d-dimensional data set, described by a d-dimensional random variable $X = (X_1, \ldots, X_d)$ has the covariance matrix:

$$\begin{bmatrix} E[(X_1 - \mu_1)(X_1 - \mu_1)] \cdots E[(X_1 - \mu_1)(X_d - \mu_d)] \\ E[(X_2 - \mu_2)(X_1 - \mu_1)] \cdots E[(X_2 - \mu_2)(X_d - \mu_d)] \\ \vdots \qquad\qquad\qquad\qquad \vdots \\ E[(X_d - \mu_d)(X_1 - \mu_1)] \cdots E[(X_d - \mu_d)(X_d - \mu_d)] \end{bmatrix},$$

where $\mu_i = E[X_i]$ is the expected value of the ith attribute. This matrix can be rewritten as

$$Cov_d(X) = E\left[(X - E[X])(X - E[X])^T\right]. \tag{6}$$

We now see, that for an d-dimensional random variable X, and orthogonal matrix U:

$$\begin{aligned} Cov_d(UX) &= E\left[(UX - E[UX])(UX - E[UX])^T\right] \\ &= E\left[U(X - E[X])(X - E[X])^T U^T\right] \\ &= UE\left[(X - E[X])(X - E[X])^T\right] U^T \\ &= UCov_d(X)U^T. \end{aligned}$$

We can now create a data set with the given covariance matrix V. Since V is self-adjoint it can be diagonalised. In other words, we can find orthogonal matrix U, and diagonal matrix D such that:

$$V = UDU^T. \tag{7}$$

The matrix U will have the ith eigenvector of V as the ith row, and the matrix D will have the ith eigenvalue in D_{ii}.

We now see that if $Cov_d(X) = D$ then

$$Cov(UX) = UCov(X)U^T = UDU^T = V. \tag{8}$$

In other words — by generating X with independently distributed variables, where X_I has variance D_{ii}, then the random variable UX has the desired covariance matrix.

6 Experimental Results

To demonstrate the potential power of our attack we apply it to both real and synthetic datasets. We use the "Adult Census" and "Auto-MPG" datasets from the UCI Machine Learning Repository [13]. For our experiments we use a 1.6 GHz Pentium Notebook with 2 MB cache and 512 MB RAM running the Windows XP

operating system. The implementation of our attack is programmed using ruby 1.8.6 with rb-gsl (gnu scientific library) bindings for mathematical operations.

The Adult Census dataset contains 48842 objects. The objects have 14 attributes, of which we are only using the attributes "age", "education-num", "sex", and "hours-per-week" (Liu *et al.* apply their attack to the attributes "age", "education-num", "hours-per-week").

The Auto-MPG dataset contains information on car brands, their engines, and their gas consumption. The dataset contains 398 objects with 8 attributes. We apply our attack to the attributes "mpg", "displacement" (engine volume), "horsepower", "weight", and "acceleration".

Before the attack is applied, we compute the global probability distributions, which is to be known to the attacker. To this end each test first selects a subset from the given dataset, and computes the statistics on that subset. The attack is subsequently applied to the dissimilarity matrix of another subset of objects (the target database). When the target database is overlapping with the data used for computing the probability distribution, the computed probability distribution fits closely to the target database. This may make the attack seem better than it is. We have selected non-overlapping sets, where possible. Unfortunately the Auto-MPG dataset only contains 398 objects, so some overlap is unavoidable.

As measure of success, we compute the average distance between the recovered data and their real values. The tests are repeated 30 times, and their average is taken. In the following graphs the distances are reported in percentage of the maximum distance[1], which is referred to as *distance error*, to make comparison easy.

In Fig. 2 the results of the attack on the Auto-MPG dataset is shown. Since the dataset is small, we use 200 objects for computing the known probability distribution. We attack target datasets of sizes from 50 to 400 in steps of 50. In the tests where the target database has less than 200 objects, we use non-overlapping sets. However, in the tests with more than 200 objects there is an overlap between the data used for computing the probability distribution and

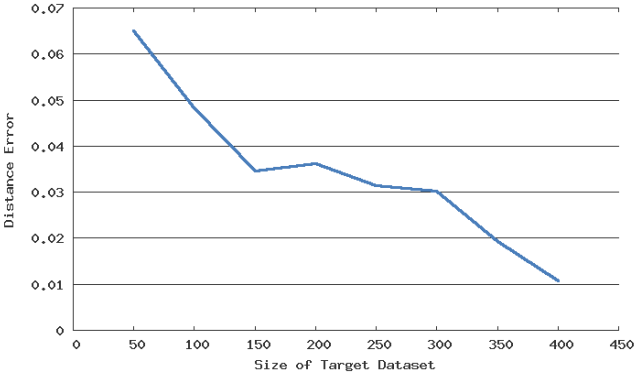

Fig. 2. Error percentage in Auto-MPG dataset with 5 attributes

[1] Since we normalise all data, the maximum distance of d-attribute objects is $\sqrt{d}$.

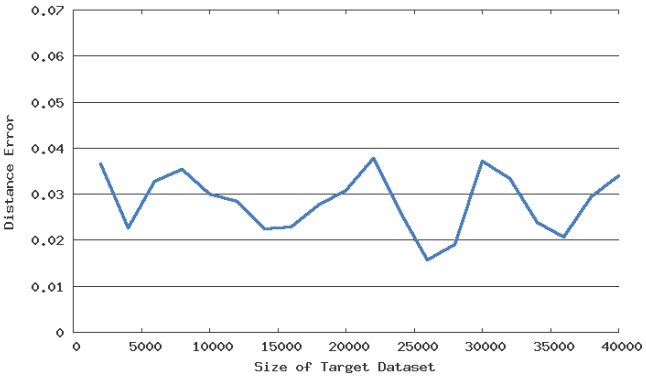

Fig. 3. Error percentage in adult census dataset with 4 attributes

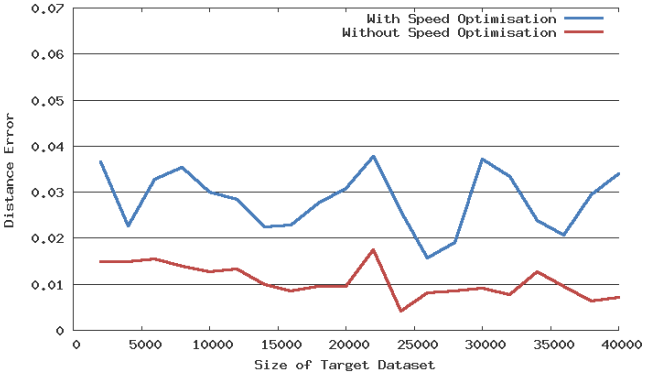

Fig. 4. Error percentage in adult census dataset with and without our speed optimisation

the target dataset. Even for a dataset as small as 50 objects, we can recover the data with an error of only 6.5%. When the size of the dataset grows, the error drops to approximately 2%, and therefore our attack becomes more effective.

In Fig. 3 the results of the attack applied to the Adult Census dataset is illustrated. The global probability distribution is computed from a set of 5000 objects, and the target datasets contain between 2000 and 40000 objects in steps of 2000. Since the Adult Census dataset contains 48842 objects the dataset used for computing the global probability distribution and the target dataset are non-overlapping. The tests show that in this very realistic scenario, we are able to recover the secret data with an error of only 3%, and on some cases as low as 2%. In terms of privacy this means that we can recover the age of individuals to a precision of 3 years; this is clearly a violation of privacy.

Since our attack does not do exhaustive search for the best mirroring of the dataset, but iteratively try to mirror in one direction at the time it will clearly not be as precise as when exhaustive search is used. To see how much the precision

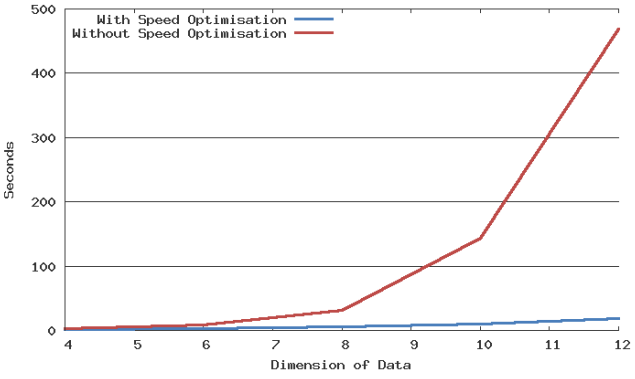

Fig. 5. Time of the attack on 1000 objects with and without our speed optimisation

suffers when using our optimisation we also make the above test on the Adult Census dataset. In Fig. 4 it is seen that the speed improvement reduces the precision of the attack, as expected.

To demonstrate the effect of our improvement on the speed of the attack, we perform a series of timings both with and without our improvement. The tests are performed on synthetic datasets of 1000 objects with $4 - 12$ attributes. As can be seen from Fig. 5 our approach greatly reduces the time of the attacks.

For databases with many attributes our improved attack offers a realistic attack, where the original attack becomes infeasible.

7 Conclusions

Privacy preserving data mining is still an ongoing research topic where off-the-shelf software solutions are yet to be developed. Two of the main reasons for the lack of software solutions are the strong assumptions made by the existing methods, and possible privacy breaches. In this work we showed that distance preserving data transformation techniques proposed for privacy preserving data mining (1) make too strong assumptions for real life scenarios, and (2) compromise the privacy of individuals. Current distance preserving transformations assume that the adversary does not have background information about the released data. To prove that this is not a realistic assumption, we showed how an adversary can utilise public data sets to obtain statistics about the transformed data. We further demonstrated how this background information can be used in conjunction with the distance values to obtain the original data set. We conducted experiments on US census and Auto MPG data obtained from UCI [13] to show that the actual data can be recovered with very high accuracy. Our attack is an improvement of the attack of Liu *et al.* since it is applicable to *any* distance preserving map (including the projection map proposed in [8] which is secure against the attack from [7]).

It is still an open problem to quantify how much information can be disclosed from the dissimilarity matrix of a given dataset. We argued that, when the PCA attack is applied, the amount of disclosed information is related to the characteristics of the eigenvalues of the correlation matrix, but for other techniques other properties may govern the amount of leakage. It is an interesting problem to find alternatives to PCA.

References

1. Aggarwal, C.C.: On randomization, public information and the curse of dimensionality. In: ICDE, pp. 136–145 (2007)
2. Aggarwal, C.C., Yu, P.S.: A condensation approach to privacy preserving data mining. In: EDBT, pp. 183–199 (2004)
3. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, USA, May 16-18, 2000, pp. 439–450. ACM, New York (2000)
4. Chen, K.: Geometric Methods for Mining Large and Possibly Private Datasets. PhD thesis, Georgia Institute of Technology (2006)
5. Chen, K., Sun, G., Liu, L.: Towards attack-resilient geometric data perturbation. In: Proceedings of the 2007 SIAM International Conference on Data Mining, pp. 78–89 (2007)
6. Kantarcioglu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. IEEE Trans. Knowl. Data Eng. 16(9), 1026–1037 (2004)
7. Liu, K., Giannella, C., Kargupta, H.: An attacker's view of distance preserving maps for privacy preserving data mining. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 297–308. Springer, Heidelberg (2006)
8. Liu, K., Kargupta, H., Ryan, J.: Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. IEEE Trans. Knowl. Data Eng. 18(1), 92–106 (2006)
9. Mann, C.C.: Homeland insecurity. The Atlantic Monthly 290(2) (2002)
10. Muralidhar, K., Sarathy, R.: Security of random data perturbation methods. ACM Trans. Database Syst. 24(4), 487–493 (1999)
11. Oliveira, S.R.M., Zaïane, O.R.: Privacy preserving clustering by data transformation. In: Proceedings of the 18th Brazilian Symposium on Databases, pp. 304–318 (2003)
12. Oliveira, S.R.M., Zaïane, O.R.: Privacy-preserving clustering by object similarity-based representation and dimensionality reduction transformation. In: ICDM 2004, pp. 21–30. IEEE Computer Society, Los Alamitos (2004)
13. UCI Machine Learning Repository, `http://mlearn.ics.uci.edu/MLsummary.html`
14. Vaidya, J., Clifton, C.: Privacy-preserving k-means clustering over vertically partitioned data. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, pp. 206–215. ACM Press, New York (2003)
15. Vijayan, J.: House committee chair wants info on cancelled dhs data-mining programs. Computer World (September 18, 2007)