Distrib Parallel Databases (2014) 32:5–35
DOI 10.1007/s10619-013-7126-6

A generic and distributed privacy preserving classification method with a worst-case privacy guarantee

Madhushri Banerjee · Zhiyuan Chen · Aryya Gangopadhyay

Published online: 1 May 2013

Abstract It is often necessary for organizations to perform data mining tasks collaboratively without giving up their own data. This necessity has led to the development of privacy preserving distributed data mining. Several protocols exist which deal with data mining methods in a distributed scenario but most of these methods handle a single data mining task. Therefore, if the participating parties are interested in more than one classification methods they will have to go through a series of distributed protocols every time, thus increasing the overhead substantially. A second significant drawback with existing methods is that they are often quite expensive due to the use of encryption operations. In this paper a method has been proposed that addresses both these issues and provides a generic approach to efficient privacy preserving classification analysis in a distributed setting with a worst-case privacy guarantee. The experimental results demonstrate the effectiveness of this method.

Keywords Data mining · Privacy preserving data mining · Classification

1 Introduction

Data mining is the process of discovering new knowledge from large volumes of data. The importance of data mining has increased manifold with the ability to store huge

Communicated by Elena Ferrari.

M. Banerjee (✉) · Z. Chen · A. Gangopadhyay
Department of Information Systems, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250, USA
e-mail: madhu2@umbc.edu

Z. Chen
e-mail: zhchen@umbc.edu

A. Gangopadhyay
e-mail: gangopad@umbc.edu

amounts of data. While dealing with sensitive data like personal data, business data, medical data, crime data, and others, knowledge discovery techniques might threaten the goal of preserving privacy of the data. Therefore, the data needs to be protected from attacks that might reveal sensitive information. But, as the primary goal is to retrieve useful information from the data, the privacy protected data should still yield useful data mining results.

The issue of preserving privacy in data mining is further complicated by the constraints that automatically arise when the data analysis are performed by third parties. The data often contains critical business information which if divulged might undermine the utility of data mining. Nowadays, many companies are opting for business process outsourcing (BPO) as it is in most cases low cost and profitable. But, since preserving the privacy of data and knowledge is one of the primary objectives for corporate strategies, they are often reluctant to share data with a third party.

Another critical issue with privacy preserving data mining occurs when the data is present in more than one location and owned by more than one party. This situation has given rise to the relatively new area of research named privacy preserving distributed data mining [51]. As a result, collaborative data analysis from every data location may yield useful results but the parties may be reluctant or not allowed to share their own data.

Data can be distributed horizontally or vertically. Horizontal data partitioning means each party contains a subset of records, vertical data partitioning means each party contains a subset of columns of the same set of records. Below are two examples.

1. Horizontally partitioned: More than one bank have data from their corresponding sources about fraud detection and will benefit from collaborative analysis by building a classifier but they are not allowed to do so as it will reveal their business data. The data at different banks have the same attributes but contain different rows.
2. Vertically partitioned: A hospital and the environmental department wish to collaborate in order to analyze outbreak of certain diseases without actually revealing their data. Both data sets are about the same locations or areas, but have different attributes.

There exists a rich body of work in privacy preserving distributed data mining based on Secure Multi-Party Computation techniques [35, 51]. These techniques develop data mining algorithms that will work in a distributed scenario and will not reveal critical information of the parties involved. However, these algorithms generally suffer from two major setbacks: (1) most of these algorithms handle only a single data mining method; therefore, if the participating parties are interested in exploratory data analysis (e.g., trying a few mining algorithms or different parameter settings for the same algorithm), they will have to go through a different secure distribution protocol for every mining task they intend to perform. (2) These protocols are often expensive as they involve many expensive encryption operations.

In this paper, both the above shortcomings have been addressed. The contribution of this research work comprises the following:

1. The proposed algorithm deals with centralized data, horizontally partitioned data, and vertically partitioned data to provide a worst-case privacy guarantee for classification methods.
2. This algorithm will enable the end user to sanitize the data once and then perform exploratory data analysis by applying more than one classification technique.
3. The effectiveness of the algorithm has been validated by extensive experiments.

The rest of the paper is organized as follows. Section 2 describes related work. Section 3 gives some background information needed for the foundation of this research. Section 4 provides details about the methodology in this research. Section 5 presents the extensive experimental results. Section 6 concludes the paper.

2 Related work

There are extensive body of work in privacy preserving data mining. Surveys on various privacy preserving techniques can be found in books [2] and [51]. The discussion of existing work in this area has been divided into two broad categories for better understanding. The first category deals with situations when data is in a centralized location, and the second category handles distributed data mining scenarios.

Privacy preserving data mining on centralized data Privacy preserving data mining in a non-distributed setting assumes that the mining will be outsourced to a third party and the data holder is only responsible for the perturbation of the data. The most simple technique in this regard is additive perturbation. A decision tree classifier was build by [5] where data was sanitized using additive perturbation. However, as mentioned in [25, 30] additive perturbation is vulnerable to attacks based on data correlations.

As additive perturbation was not that efficient, Kim et al. [31] proposed multiplicative perturbation methods. The first technique multiplies individual data values by random numbers generated with mean 1 and small variance. The second approach generates the noise by transforming the data to their logarithms and calculates the covariance matrix and then generates random numbers which have mean zero and variance as some multiple of the covariance matrix. Finally it adds the generated noise to the data. The limitations of this method are that it fails to be accurate in distance based mining methods like K-nearest neighbor classification and is also vulnerable to correlation attacks as is additive perturbation.

A few transform-based methods have been proposed for distance-based mining [9, 37, 40, 42, 43]. However, such methods can not be applied in distributed mining because each party has to use the same transform thus they can reverse the transform to infer others' data.

[11, 20] proposed another approach of perturbation called data swapping where some values of attributes are swapped over certain mutually selected rows. Thus the data is perturbed by keeping the t-order statistics unchanged but the method is unsafe as there is no guarantee that the swapped values will be very different from the original values. [1] suggests a condensation approach of using properties of clusters to

regenerate data simulating the actual one but obfuscating the original values. However the privacy levels for this method are not very high.

Another technique is the randomized response method proposed in [13, 53]. Here, a method is proposed for preserving the privacy of respondents while asking sensitive questions in an interview. The concept here is to protect the identity of the interviewee by randomizing the response. [13] proposes a decision-tree classification method to support the above mentioned scheme. Another technique is k-anonymity proposed in [46] where the data of an individual will be indistinguishable from other $k - 1$ individuals. The concept of k-anonymity is more related to the area of securing an individual's identity than the area of privacy preserving data mining but since data is often about individuals this concept is worth mentioning.

All the methods discussed above deal with average case privacy and fail to handle the worst-case privacy guarantee. To address this problem [41] proposes a privacy preservation method to preserve Euclidean distances within data with a worst-case privacy guarantee. Evfimievski et al. first proposed a worst-case privacy model and used it for association rule mining in [18]. [4] further developed an optimal matrix theoretic scheme for random perturbations of categorical data and provided guarantees through amplification. However both [18] and [4] suggest some specific randomization schemes suited for discrete data and only consider mining of association rules from the perturbed data. [41] extends the model to continuous data where the authors use Principal Component Analysis to eliminate attacks based on correlation followed by adding Laplace Noise to distort the data. The authors also propose modifications to the K-nearest neighbor classification to improve its accuracy over the perturbed data. However, this method has three problems: (1) it only deals with centralized data; (2) it is unclear how to generalize the modified KNN to other mining algorithms; (3) it uses PCA which is often inferior to linear discriminant analysis (LDA) for classification. Here, we propose a method that uses LDA and provides a worst-case privacy guarantee for multiple classification methods and also works for distributed data.

Another concept that needs to be mentioned here is differential privacy. Dwork has proposed a differential privacy model that provides a worst-case privacy guarantee for statistical databases [14]. The basic idea is to perturb the results of statistical queries such that whether an individual is included in the database will only have limited impact on the results. As a result, attackers can infer very limited information from the results.

There are two possible use scenarios: the interactive scenario when users do not have direct access to data and can only ask statistical queries against the database, and the non-interactive scenario when data is sanitized and given to users. Most research on differential privacy focuses on the interactive scenario [6, 15, 16, 33, 39, 45, 55]. There has been some work on applying differential privacy to non interactive scenario [7, 16, 17, 19, 56]. However, it remains an open problem to find efficient solutions in many domains for the non interactive scenario [17]. Further, existing work also assumes that the set of queries asked over the database is given.

In this paper we will focus on the *non interactive* scenario and use a worst-case privacy model that is quite similar to differential privacy but allows users to have direct access to sanitized data. We also assume that users have not decided on the

data mining algorithms, which means the set of queries is not known in advance. Our approach does not require a known set of queries. It is also very efficient.

Privacy preserving data mining on distributed data With the developments in network technologies data can be located in more than one locations and be held by more than one party who are not interested in divulging their own data and as a result there is a need for algorithms that can perform data mining involving all the parties but can still keep their data private. There has been extensive research in this area. Most work in a distributed setting uses secure multi-party computation. Survey articles can be found in [51]. [27, 48] deal with association rules for vertically partitioned data and horizontally partitioned data respectively. [34] and [49] handle clustering over vertically partitioned data. [8, 12, 23, 29, 50] explore the problem in context of decision trees. [28, 38, 54] consider the process of Bayesian learning while [52] concentrates on Naive Bayes classification for both horizontally and vertically partitioned data.

All these methods mentioned above are very specific to a certain class of mining algorithms and because the information available to create global models is very much situation dependent. This has resulted in devising a huge number of distributed algorithms each suitable for a particular mining algorithm. No single general algorithm exists. In other words, multiple SMC protocols must be executed to perform multiple mining algorithms.

The only exception is a holistic approach proposed in [47] in which all parties first use a SMC protocol to generate a synthetic data set and then directly share this data for different mining algorithms. The synthetic data generation is based on the condensation approach [1]. This approach first generates size-k clusters and then generates synthetic data based on the statistical properties of these clusters. This approach has one major benefit: data miners only need to sanitize the data once and then can try many different mining algorithms on the same sanitized data.

However, the condensation approach has two major drawbacks. First, it provides no worst-case privacy guarantee. Since each cluster contains at least k records, it satisfies the k-anonymity model [46]. However, this model does not prevent the disclosure of sensitive values. For example, some records in the synthetic data may have very similar values to records in the original data. Second, in distributed mining, all parties often know what attribute to predict, but the condensation approach does not take this into account. This may lead to inferior mining quality.

The algorithm proposed in this paper addresses these issues and provides a perturbation technique which will enable the end user to perform multiple classification methods. The key idea of this approach includes a utility-aware projection step that removes unnecessary information for the mining method followed by a group-based perturbation step that provides a worst-case privacy guarantee.

3 Background

This section gives the background necessary to understand the proposed method. Section 3.1 explains the Linear Discriminant Analysis method used in this article as the first stage of the perturbation technique proposed here. Next, in Sect. 3.2 the

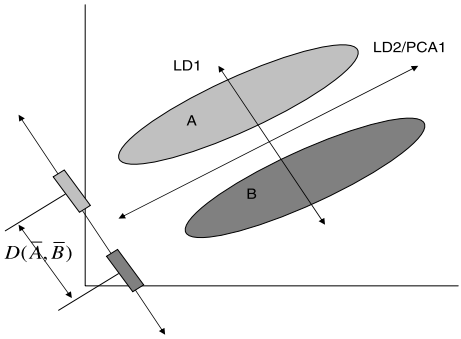


Fig. 1 Linear discriminant analysis

worst-case privacy model is explained with details also used in this article in order to provide a worst-case privacy guarantee in this proposed framework. Finally, Sect. 3.3 explains the adversarial model commonly used in a private and distributed setting.

3.1 Linear discriminant analysis

Linear Discriminant Analysis (LDA) finds a linear transformation of predictor variables which provides a more accurate discrimination between class labels. LDA is often used for dimensionality reduction before classification is performed on the data. Principal Component Analysis (PCA) is very similar to LDA except that PCA is unsupervised and does not take the class labels into consideration. This is evident from Fig. 1. The figure shows two classes: A and B. PCA returns the direction with largest data variance (PCA1 in Fig. 1), but not the direction that best separates the two classes (LD1 in Fig. 1). LDA finds the direction that best separates the two classes.

Let us consider a dataset x with n rows or tuples and m columns or attributes where the last attribute is the class label. Equation (1) computes S_b, between class scatter matrix where μ_c is the vector of dimension $m \times 1$ representing the mean of each class c and $\bar{x}$ is the vector representing the global mean of the dataset x. The product of $(\mu_c - \bar{x})(\mu_c - \bar{x})^T$ and their summation over all the classes gives the covariance matrix of the centers of classes. Similarly, in (2), S_W can be explained as the within class scatter matrix, related to the covariance matrix between classes.

$$S_b = \sum_c (\mu_c - \bar{x})(\mu_c - \bar{x})^T \tag{1}$$

$$S_W = \sum_c \sum_{i \in c} (x_i - \mu_c)(x_i - \mu_c)^T \tag{2}$$

$$J(w) = w^T S_b w / w^T S_W w \tag{3}$$

The idea is to find a linear combination that offers the maximal separation for the classes. Therefore, once S_b and S_W are computed, LDA will find a set of weight vectors (the number is no more than the number of classes minus one) that maximize $J(w)$. The data can then be mapped to the subspace formed by these vectors.

3.2 Worst-case privacy model

A worst-case privacy model was proposed in [18] for categorical data. Later it was extended to numerical data in [41]. This model prevents privacy breaches called ρ_1-to-ρ_2 privacy breach. Let X be a random variable whose values represent the original data points. The sanitized data is represented by another random variable Y. Further let V_X and V_Y represent the set of possible values of X and Y respectively.

Definition 1 A ρ_1-to-ρ_2 privacy breach with respect to some property $Q(x)$ of a random variable X is said to occur if for some $y \in V_Y$

$$P\big[Q(X)\big] \leq \rho_1 \quad \text{and} \quad P\big[Q(X)\big|Y = y\big] \geq \rho_2$$

Where $0 < \rho_1 < \rho_2 < 1$ and $P[Y = y] > 0$. In a similar way a ρ_2-to-ρ_1 downward privacy breach occurs if

$$P\big[Q(X)\big] \geq \rho_2 \quad \text{and} \quad P\big[Q(X)\big|Y = y\big] \leq \rho_1$$

Definition 1 captures how much disclosing a sanitized value increases the predictability of the original value. E.g., suppose the original data contains the salaries of employees. The probability of having a salary over one million dollars equals 0.001. Now, suppose we observe a perturbed salary of a person P, which is ten million dollars. One may infer that P's salary is over one million with 0.99 probability. This is a 0.001 to 0.99 privacy breach.

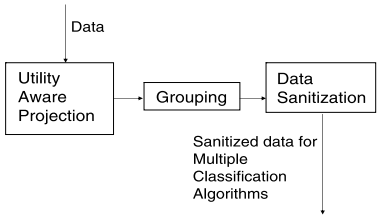
A perturbation method was proposed in [41] to prevent such privacy breaches. The perturbation method first decorrelates data by applying Principal Component Analysis. It then adds noise following Laplace noise to the principal components. The authors in [41] have proved that only noise satisfying Laplace distribution provides a worst-case privacy guarantee, while the widely used uniform or Gaussian noise does not provide such guarantee. Let D_P^i be the i-th column of the principal component matrix, and b_i be the range of that column, i.e., $b_i = b \cdot (\max\{D_P^i\} - \min\{D_P^i\})$, where b is a parameter. The noise added to the i-th column follows Laplace distribution with a mean zero and standard deviation b_i. [41] also proves that this perturbation method gives the following privacy guarantee.

Theorem 1 *The perturbation method proposed in* [41] *will neither cause an upward ρ_1-to-ρ_2 nor a downward ρ_2-to-ρ_1 privacy breach with respect to any property of X if the following is satisfied*

$$\frac{\rho_2}{\rho_1}\frac{(1-\rho_1)}{(1-\rho_2)} > \gamma$$

As mentioned in [41] γ is the amplification factor and $\gamma = e^{1/b}$. For example, in the above example, suppose $b = 0.2$ and the probability of P having over a million dollar salary is 0.001 (ρ_1). After seeing the sanitized data, using Theorem 1 we can infer that the probability of P having over a million dollar salary (ρ_2) will not exceed 0.13.

Fig. 2 Overview of the method



3.3 Adversarial model

In distributed settings, each party has only a portion of data and all parties want to collaboratively compute some output without revealing their data to other parties. We use the semi-honest adversarial model [51] where parties follow the protocol faithfully, but may try to infer private information of the other parties from the messages they see during the execution of the protocol. Under the semi-honest model, parties will not collude.

The security of SMC protocols is defined such that every party learns no more than in an ideal model where there is a trusted third party who does all the computation and distributes the result to all participants. So each party can only infer information from the final results and its own share of data. In reality, however, some SMC protocols do reveal some extra information (typically as intermediate results) for performance reasons. Following traditional SMC methods, we require that our sanitization process not reveal extra information besides the output (i.e., the sanitized data) and some non critical information (e.g., the intermediate cluster positions in K-Means clustering). In addition, we require a worst-case privacy guarantee (as mentioned in Sect. 3.2) that prevents attackers from inferring original data from the sanitized data.

A useful technique to prove the security of a SMC protocol is the composition theorem [51].

Theorem 2 *If a protocol is shown to be secure except for several invocations of sub-protocols, and if the sub-protocols are proved to be secure, then the entire protocol is secure.*

4 Our approach

We have proposed a privacy-preserving approach that works for centralized, horizontally partitioned, and vertically partitioned classification. Our approach assumes that data miners know which attribute they want to classify, but they may try multiple mining algorithms or setting different parameters for the same mining algorithm. This assumption is often satisfied in practice because for multiple parties to collaborate they need to specify a common goal (e.g. the classification task) first. In the distributed mining case we also assume each party follows the semi-honest model.

Figure 2 presents the overview of our framework. Our approach consists of three major steps:

1. Utility aware projection which removes information unnecessary for the classification method. This step improves privacy protection without sacrificing mining quality.
2. Grouping of each class such that similar data is put into the same group. This step improves mining quality.
3. Data sanitization which adds Laplace noise to data set to provide a worst-case privacy guarantee.

Algorithm 1 presented below is the algorithm for centralized data, that is when the data is present in a single location. Each step is described below.

Utility aware projection The first step in our approach is to use Fisher's Linear Discriminant Analysis to transform the original data. As already explained in Sect. 3.1 the use of LDA has several advantages. It helps in drastic reduction of data dimension and also eliminates the correlation between columns in the transformed data thus preventing correlation attack. The authors in [41] have used PCA but Fig. 1 explained the advantages of LDA over PCA for classification data. Step 1 in Algorithm 1 computes this step.

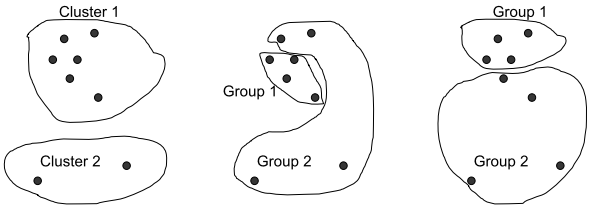
Grouping of each class Here we further group each class with a predetermined group size after transforming the data using LDA. Grouping each class ensures that each group only consists of records from the same class. This will improve mining quality after adding noise (the data sanitization phase is explained below). Lines 3 to 10 in Algorithm 1 presents the steps for this phase.

Step 3 runs a K-Means clustering on each class where k is predetermined depending on the intended group size. The clusters are then sorted in ascending order of their

Algorithm 1 Algorithm for centralized case

Input: Dataset D, Number of classes L, predetermined group size S, Noise parameter b, Number of selected coefficients s.
Output: Sanitized Dataset D'

1: Infer LDA transform of the dataset D by $D_P \leftarrow \text{LDA}(D)$;
2: **for** each class L_i **do**
3: Run K-Means clustering on class L_i with $K = g$ where g is the number of groups to be formed in class L_i and $g = \lfloor |L_i|/S \rfloor$
4: Sort the clusters in ascending order of cluster size.
5: **for** Each cluster center c_j, $j = 1, \ldots, g$ **do**
6: Add all points in cluster C_j to group G_j
7: **if** $|C_j| < S$ **then**
8: find the closest cluster and move $S - |C_j|$ points in $\bigcup C_l, l > j$ that are closest to c_j, and move them to group G_j.
9: **end if**
10: **end for**
11: For each group x compute $\max_j^x$ and $\min_j^x$ as the maximum and minimum values of transformed column j in group x
12: For each group x compute $b_j^x = b \cdot (\max_j^x - \min_j^x)$ where $1 \le j \le s$ and $D_{NP_x}(l, j) = D_{P_x}(l, j) + n(b_j^x)$ where $n(b_j^x) \sim \text{Laplace}(0, b_j^x)$ and $D_{P_x}(l, j)$ is the transformed value in j-th column of every row l in group x.
13: **return** $D' = \bigcup_{1 \le x \le g} D_{NP_x}$
14: **end for**



(a) K-means (b) Random movement (c) Moving closest points

Fig. 3 Example of moving records between clusters

size (line 4). In our approach we move elements from larger clusters to the smaller clusters depending on their affinity to each other (lines 5 to 10). The insight for this step is shown in Fig. 3. Figure 3(a) shows the initial groups generated after K-Means clustering. Cluster 1 contains 6 points and cluster 2 contains only 2 points. Each final group should have 4 points as we need to form approximately equal sized groups. The question is which records to move. Figure 3(b) shows an example when some random records are moved to cluster 2. Cluster 2 becomes too scattered and thus too much distortion will be introduced after data sanitization. Instead, we move records in larger clusters that are closest to the center of the smaller cluster. Figure 3(c) displays the result. Now cluster 2 is much less scattered.

In our proposed algorithm we take the group size into account while grouping the data, but do not take the data range into consideration. A very small range in a group of data can lead to lower privacy. One way to handle such a situation (if it occurs) is to use a binary search method as proposed in [32] to handle the group size and minimum range parameters in the proposed algorithm here. In [32] the authors use a binary search method to determine the optimal group size. In our algorithm it may not be feasible to use minimum data range as a parameter but we can use the group size as a parameter and tweak the group size following the algorithm (rule-based approach) proposed in [32] and check if the data range parameter is satisfied by adjusting the group size.

The algorithm will first select the minimal and maximal number of groups using the method in [32]. It will then run our grouping algorithm using the maximal number of groups to check whether the range for each generated group exceeds a threshold. If so, the range condition will be satisfied. Otherwise, we will check whether the minimal number of groups satisfies the range condition. If it does, we will use a binary search algorithm to find the maximal group size that still satisfy the range condition. Otherwise, we will not group the data because grouping will lead to violation of the minimal range condition. We will further analyze and optimize this algorithm in our future research.

Once the groups are formed, each group is then sanitized separately as explained in the data sanitization phase below.

Data sanitization The data sanitation phase is very similar to [41]. Each group from the previous phase is sanitized by adding noise generated using Laplace distribution. The reason for using Laplace noise is because it provides a worst-case privacy guarantee as explained in details in Sect. 3.2 and in [41].

In [41] Laplace noise is added over the entire dataset. The noise added to the dataset is scaled by the range of the whole dataset. As a result it distorts the data extensively and therefore decreases classification accuracy. The modified KNN algorithm as presented in [41] tries to address this data distortion problem. However, in this research since we are trying to develop a generic method for both distance based and non-distance based classification methods, modified-KNN is not appropriate.

To reduce data distortion, we group each class and generate noise for each group separately and add this noise to the corresponding group. We could have added the noise generated separately for each class but if the class is not very well formed then this method of noise addition can produce significant data distortion as well. Therefore, we have further grouped the classes and produced groups of approximately equal size and then sanitized each group separately. Lines 11 and 12 in Algorithm 1 present the steps for noise addition for the case of centralized data. The range of each column of each group is computed in line 11. In line 12, noise with Laplace distribution is generated scaled to the ranges computed in line 11. This noise is then added to the group. These steps are repeated for every group to obtain the final sanitized data in line 13.

Complexity analysis The computational complexity of LDA is $O(m^2n)$ and k-means clustering is $O(snk)$ where m is the number of attributes in the original data, n is the number of data elements, k is the number of clusters to be generated, and s is the dimension of the data after LDA transformation. The noise addition process takes $O(ns)$ time.

We have extended our algorithm for a distributed scenario, that is where data is not present in one single location but is distributed in multiple locations and often with multiple parties. The data can be distributed horizontally or vertically. In order to collaboratively sanitize the whole data as if the data was present in a single centralized location, we need to securely perform all the above three steps such that none of the parties needs to reveal their own private information to other parties. Sections 4.1 and 4.2 present the methods and protocols in details for horizontally partitioned data and vertically partitioned data, respectively.

4.1 Horizontally partitioned data

This section provides the details for utility-aware projection, a grouping of each class and data sanitization for horizontal partitioning followed by a security analysis of the methods used and the computational complexity.

Utility aware projection for horizontally partitioned data Steps 1 to 5 in Algorithm 2 explain this phase. When the data is horizontally partitioned the calculation of S_b and S_W can be mostly done locally. In order to calculate S_b all the parties will need to go through a series of secure sum protocols to calculate N_c, μ_c and $\bar{x}$. Computing of S_W can also be done locally as $(x_i - \mu_c) * (x_i - \mu_c)^T$ can be calculated locally by each party and then secure sum protocol can be used to calculate the final S_W. $J(w)$ can be computed by each party locally once S_b and S_W are calculated. Here, the only information revealed to each party is the class mean, the overall mean of the dataset, S_b and S_W which we assume are not sensitive information.

Algorithm 2 Algorithm for horizontally partitioned data

Input: Dataset D distributed as $D_1, D_2, \ldots, D_p$ in p participating parties, number of classes L, minimum number of elements in each group S, noise parameter b, number of retained coefficients after LDA s.
Output: Sanitized Dataset D'

1: All parties run secure sum protocol to compute number of elements in each class N_c, class mean μ_c, and global mean $\bar{x}$
2: Each party computes S_b by $S_b = \sum_c (\mu_c - \bar{x})(\mu_c - \bar{x})^T$;
3: Each party t where $1 <= t <= p$ computes S_{W_t} by $S_{W_t} = (x_i^t - \mu_c)(x_i^t - \mu_c)^T$ where x_i^t are the elements of party t.
4: All parties run secure sum protocol to calculate S_W
5: Each party t where $1 \leq t \leq p$ computes the LDA transformed data by $\hat{D}_t = \text{LDA}(S_b, S_W, D_t, s)$
6: **for** each class L_i **do**
7: All parties run secure K-Means Clustering [26], on each class with $k = g$ and $g = \lfloor |L_i|/S \rfloor$
8: The clusters are sorted in an ascending order based on their size
9: All Parties need to first use a secure sum protocol to compute the number of points in each cluster
10: **for** each cluster center c_j, $j = 1, \ldots, g$ **do**
11: Add all points in cluster C_j to group G_j
12: **if** $|C_j| < S$ **then**
13: Use the protocol to find the closest cluster [26] to find $S - |C_j|$ points in $\bigcup C_l, l > j$ that are closest to c_j, and move them to group G_j.
14: **end if**
15: **end for**
16: All the parties run secure max and secure min protocol (secure comparison protocol) [26, 49] to compute $\max_j^x$ and $\min_j^x$ for each group x where $1 \leq x \leq g$ and each column j where $1 \leq j \leq s$ in the transformed data $\hat{D}$
17: For each group x each party computes $b_j^x = b \cdot (\max_j^x - \min_j^x)$ where $1 \leq j \leq s$ and $\hat{D}_{t_x}(l, j) = \hat{D}_t(l, j) + n(b_j^x)$; where $\hat{D}_t(l, j)$ is the transformed value in the j-th column of every row l in group x at party t and $n(b_j^x) \sim \text{Laplace}(0, b_j^x)$
18: Each party t outputs $\bigcup_{1 \leq x \leq g} \hat{D}_{t_x}$ to form the whole sanitized dataset D'
19: **end for**

Grouping of each class for horizontally partitioned data Steps 6 to 15 in Algorithm 2 explain the phase of grouping each class. All the parties run secure k-means clustering [26] on each class with predetermined $k = g$. Then the parties engage in a secure sum protocol to calculate the number of elements in each cluster and then order the clusters in ascending order based on the size of each cluster. Then the records from larger clusters are moved to the smaller clusters using secure protocols to satisfy a predetermined minimum number of elements in each group.

Data perturbation for horizontally partitioned data Steps 16 to 18 in Algorithm 2 explain the data perturbation/sanitization phase.For data perturbation purposes, each party needs to add Laplace noise to the dataset. All parties use secure max and secure min protocols to calculate the range for each column in the transformed data, without revealing their local data. Each party then adds Laplace noise scaled to these ranges to their local data points. The algorithm for horizontally partitioned data is detailed in Algorithm 2.

Security analysis We want to show that our sanitization process does not reveal extra information besides the output (i.e., the sanitized data) and some non-critical information. In line 1 of Algorithm 2 all parties engage in secure sum protocol and

the information revealed is the number of elements in each class N_c, class mean μ_c and global mean $\bar{x}$ which we assume are not private. Line 2 and 3 are computed locally by each party. In line 4 all the parties engage in another secure sum protocol to calculate S_W. We assume that these statistics S_b and S_W are not private. Line 5 is again computed locally by each party. Line 7 to Line 15 performs the grouping of each class. Only secure protocols from existing literature [26, 49] are used in these steps and therefore critical information is not divulged.

The next step is data sanitization. In line 16 all parties run secure max and secure min protocol and the information revealed are the highest and the lowest values for each column in each group in the transformed dataset which we again assume is not a security breach. Lastly, line 17 is computed locally and in line 18 each party outputs their sanitized dataset.

Communication cost Let n be the number of records, m be the number of attributes, s be the number of attributes in the transformed data, c be the number of classes, g be the number of groups, r be the number of parties and r_k be the number of iterations for k-means clustering. When data is horizontally partitioned, most computation can be done locally for statistics needed for the LDA transform which is $O(mcr)$, and $O(scr)$ to compute b_j needed to add the noise to the dataset. The computational complexity for secure k-means clustering is $O(sgrr_k)$. Therefore, the overall complexity can be considered as $O(mcr + sgrr_k)$ for the case of horizontally partitioned data.

4.2 Vertically partitioned data

This section provides the details for utility aware projection, grouping of each class, and data sanitization for vertical partitioning followed by a security analysis of the methods used and computational complexity. Algorithm 3 describes our approach for vertically partitioned data.

Utility aware projection for vertically partitioned data Steps 1 through 5 explain the phase of utility aware projection in Algorithm 3. In the case of vertically partitioned data, N_c is known to each party and each party can locally compute μ_c and $\bar{x}$ for their local subset of columns. In order to compute S_b and S_W the parties will involve a series of secure dot product protocol. For example, let us consider the computation of S_W as in (2). Since the data is vertically partitioned each party can subtract the class means corresponding to their local attributes from their local attributes with the corresponding class labels (i.e. if there are p parties each party computes $x_{i_p} - \mu_{i_p}$). As it is evident $x_i - \mu_c$ is distributed in more than one party, in order to compute $(x_i - \mu_c)(x_i - \mu_c)^T$ all the parties need to engage in secure dot product protocols to contribute to the total computation of the matrix $(x_i - \mu_c)(x_i - \mu_c)^T$. $(x_i - \mu_c)(x_i - \mu_c)^T$ is needed to be computed for all the data points for a single class and then all the parties engage in a secure sum protocol to compute $\sum_{i \in c}(x_i - \mu_c)(x_i - \mu_c)^T$. These steps are repeated for each class in the dataset and finally another secure sum protocol results in the computation of $S_W = \sum_c \sum_{i \in c}(x_i - \mu_c)(x_i - \mu_c)^T$. The steps are similar for the calculation of S_b as in (1).

Algorithm 3 Algorithm for vertically partitioned data

Input: Dataset D (n rows and m columns) distributed as $D_1, D_2, \ldots, D_p$ in p participating parties, number of classes C, minimum number of elements in each group S, noise parameter b, number of retained coefficients after LDA s.
Output: Sanitized Dataset D'

1: Each party t calculates the class mean μ_{c_t} and global mean $\bar{x}_t$ for the subset of columns stored by t where $1 \leq t \leq p$;
2: All parties engage in secure dot/scalar product protocol [26, 49] to compute S_b and S_W;
3: All parties infer LDA Transform matrix M from the statistics computed from the previous step;
4: Each party t computes $D_t * M_t$ where D_t is the local data and M_t is the columns in M that correspond to the columns in t;
5: All parties engage in a secure sum protocol to compute $\hat{D}_t = \sum_{t=1}^{p} D_t * M_t$. The results are stored in two random shares as $\hat{D}_t^1$ at party P_1 and $\hat{D}_t^2$ at P_r
6: **for** each class L_i **do**
7: Parties P_1 and P_r run secure K-Means Clustering [26] on their random shares, on each class with $k = g$ and $g = \lfloor |L_i|/S \rfloor$
8: The clusters are sorted in an ascending order locally based on their size
9: **for** each cluster center c_j, $j = 1, \ldots, g$ **do**
10: Add all points in cluster C_j to G_j
11: **if** $|C_j| < S$ **then**
12: Use the protocol to find the closest cluster as presented in [26], to find $S - |C_j|$ points in $\bigcup C_l, l > j$ that are closest to c_j, and move them to group G_j
13: **end if**
14: **end for**
15: Parties P_1 and P_r use secure comparison protocol to compute $b_j^x = b \cdot (\max_j^x - \min_j^x)$ for each column j in $\hat{D}_t^x$ (the transformed rows in group x) where $1 \leq j \leq s$ and $1 \leq x \leq g$
16: P_1 (P_r) generates random noise matrix R_1 (R_2), which follows exponential distribution with $\lambda = 1/b_{jx}$ for each column and each group x in the transformed data
17: P_1 and P_r use secure sum protocol to compute $\hat{D}_{t_x} = (\hat{D}_{t_x}^1 + R_1 + \hat{D}_{t_x}^2 - R_2)$
18: **return** $\bigcup_{1 \leq x \leq g} \hat{D}_{t_x}$ for party t and class L_i to form the whole sanitized dataset D'
19: **end for**

The final S_b and S_W are revealed to each party which we assume is not sensitive information. From S_b and S_W each party can compute the LDA transform matrix M. At step 4 of Algorithm 3 each party t computes its fraction of LDA transformed data by multiplying D_t by M_t, which contains columns in M that corresponds to data columns at party t. E.g., if party 1 contains data columns 1, 2, and 4 then M_1 contains columns 1, 2, and 4 in M. The final LDA transformed data ($\hat{D}_t$) at party t is computed as the sum of $D_t \times M_t$ ($1 \leq t \leq r$) using a secure sum protocol. Note that for vertically partitioned data, the transformed data should not be revealed directly to any party because otherwise that party can apply a reverse LDA to find out data at other parties. Instead, $\hat{D}_t$ is stored as two random shares in two parties (say P_1 and P_r) such that $\hat{D}_t^1 = R$ is stored by P_1 and $\hat{D}_t^2 = \hat{D}_t - R$ is stored by P_r.

Grouping of each class for vertically partitioned data Steps 6 through 14 show the grouping phase for each class in Algorithm 3. All parties run secure k-means clustering on each class based on the SMC protocol mentioned in [26]. Next each party orders the clustering in the ascending order of their sizes, the size of each cluster is known to each party. Then the parties engage in a series of secure protocols to move elements from larger clusters to smaller clusters after comparing distance between the smaller cluster centers and the points in the larger clusters.

Data perturbation for vertically partitioned data The phase of data perturbation is explained in steps 15 through 18 in Algorithm 3. In order to generate the noise with Laplace distribution, both the parties engage in a secure comparison protocol to compute $\max_j^x$ and $\min_j^x$ for each column j in the transformed dataset in each group x. In case of vertically partitioned data, we can not let a single party generate the Laplace noise because that party can infer the original data values from the last shared sanitized data and the generated noise. In horizontally partitioned cases this is not an issue because each party contains different records. Instead, we use the property that a Laplace$(0, b)$ noise can be generated as the difference of two i.i.d exponential noise. Parties P_1 and P_r generate noise R_1 and R_2 respectively with exponential distribution where $\lambda = 1/{b_j}^x$ and ${b_j}^x$ is the noise level b scaled to the range of the column j and group x in the LDA transformed data. The final sanitized data is then calculated using secure sum protocol by P_1 and P_r using the equation $\hat{D}_{t_x} = (\hat{D}^1_{t_x} + R_1 + \hat{D}^1_{t_x} - R_2)$.

Security analysis Line 1 in Algorithm 3 is computed locally by each party. Line 2 uses secure dot product protocol and the only information revealed is S_b and S_W which we assume are not private as each single party cannot identify other parties data from the privately computed statistics. Lines 3 and 4 are also computed locally by each party. Line 5 uses secure sum protocol and the transformed data is stored as two random shares with two parties (P_1 and P_r), therefore, nothing is revealed to all the parties except to parties P_1 and P_r holding the random shares. The information revealed to P_1 and P_r is also secured as P_1 holds a random data R and P_r holds the difference of the transformed data and R, that is, no one party gets hold of the transformed data; also by the assumption of semi-honest model the parties will not collude. Lines 7 to 14 perform the grouping of each class. Only secure protocols from existing literature [26] and [49] are used in these steps and therefore critical information is not divulged.

The next step is data sanitization. In line 15 parties P_1 and P_r use secure comparison protocol to compute the maximum and minimum values for each group and each column in the transformed data. The only information revealed from line 15 is the corresponding maximum and minimum values which we assume are not private. In line 16 P_1 and P_r locally generate the noise data. In line 17 secure sum protocol is used which reveals the sanitized data and is provided to all other parties in line 18.

Communication cost Let n be the number of records, m be the number of attributes, s be the number of attributes in the transformed data, c be the number of classes, g be the number of groups, r be the number of parties and r_k be the number of iterations needed for k-means clustering. When data is vertically partitioned, computing statistics for LDA requires a communication cost of $O(mncr)$. The perturbation method costs $O(snr)$. The computational complexity of the secure k-means clustering is $O(sngrr_k)$. Thus overall communication cost is $O(mncr + sngrr_k)$. As mentioned before, if a small sample of the dataset is selected by each party to compute the utility aware projection (LDA) the communication cost can be reduced drastically.

Random sampling We also propose the method of stratified random sampling to reduce the overhead of computing $J(w)$ while computing LDA. In case of vertical

Table 1 Comparison of order of communication cost for various methods

Methods	Vertical	Horizontal
Our method	$O(mncr + sngrr_k)$	$O(mcr + sgrr_k)$
Naive Bayes	$O(mncr)$	$O(mcr)$
SVM	$O(n^2r)$	$O(mcpr)$

partitioning each party selects randomly a fixed percentage of their data if the data size is huge. The percentage varies between 10 %–30 % based on the size of the dataset. We use stratified sampling such that all the classes will be equally represented in the sampled dataset based on the proportion of the classes in the actual datasets. For example, if there are two classes in the original data and 30 % of the data is to be selected for the sampled dataset, then 30 % of data points with class label 1 are selected randomly and similarly 30 % of data points with class label 2 are selected randomly. Therefore, it will result in a sampled data where the proportion of the class labels are same as the proportion of the class labels in the actual dataset.

This technique of stratified random sampling will reduce the communication cost for the initial statistical analysis substantially. Statistics needed for the LDA, that is S_b and S_W will be computed based on this random sampling of that data. Experiments in Sect. 5 will show the effectiveness of random sampling.

4.3 Comparison of communication cost

The communication cost for the vertically partitioned case involves a number of data points and can be significant if the dataset is very big. This problem is also present in the existing algorithms that handle vertically partitioned datasets like [48, 49, 52]. But, the advantage of the algorithm presented here is that the secure distributed protocols are run only once and then the data can be provided to the user for exploratory data analysis, unlike other algorithms that handle only a single mining task and for each task the users have to go through all the protocols. This increases the overhead substantially.

In Table 1 a comparison of the order of communication cost is provided. We have compared the communication cost of the existing methods in [28, 52, 58, 59] with our proposed method and shown that our communication cost is either low or similar to the existing methods. In Table 1 n is number of data points; m is the number of dimensions in original data; c is the number of classes; r is the number of parties involved; s is the number of dimensions preserved after LDA; p is the average number of values in each attribute; g is the number of groups; and g_r is the number of iterations in k-means clustering.

4.4 Privacy of the proposed method

The proposed method combines LDA with additive perturbation. LDA completely decorrelates the data thereby guards additive perturbation against the typical noise filtering attacks based on data correlations [25]. The additive perturbation step guards against attacks that can recover the LDA transform (e.g., the attacking methods described in [36]) because such attacks can not remove the added noise.

Next we compute the amplification for the proposed method. The result will show that the proposed method also provides a worst-case privacy guarantee for ρ_1 to ρ_2 privacy breach.

One could certainly compute amplification using the original data values. However this paper computes the amplification using the linear discriminant values, not the original data values. The reason is that in the worst-case, attackers can find out the LDA transform (e.g., if the attacker learns the covariance matrix of original data) and will be able to infer the original data values from the principal component values.

To compute amplification for the combined framework, let $\overrightarrow{w_i} = (w_{i1}, \ldots, w_{in})$ denote the i-th weight vector and $\overrightarrow{x_1} = (x_{11}, x_{12}, \ldots, x_{1n})$ and $\overrightarrow{x_2} = (x_{21}, x_{22}, \ldots, x_{2n})$ be two arbitrary row vectors in the original data. Also let X_1 and X_2 denote the i-th linear discriminant scores to which the two data vectors are mapped respectively(i.e., X_1 and X_2 equal the dot product of $\overrightarrow{w_i}$ with $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$, respectively). Let D_P^i and D_{NP}^i denote the i-th column of D_P (the result of LDA transform) and D_{NP} (the perturbed LDA scores), respectively. According to [41] the amplification can be defined for continuous data as follows:

$$\begin{aligned} A_{max} &= \max_{x_1,x_2 \in V_x, y \in V_y} \lim_{dx \to 0} \frac{P([x_1, x_1 + dx] \to y)}{P([x_2, x_2 + dx] \to y)} \\ &= \max_{x_1,x_2 \in V_x, y \in V_y} \lim_{dx \to 0} \frac{P([x_1, x_1 + dx] \to y)/dx}{P([x_2, x_2 + dx] \to y)/dx} \\ &= \max_{x_1,x_2,y} \frac{f_\Delta(y - x_1)}{f_\Delta(y - x_2)} \end{aligned} \tag{4}$$

Here f_Δ is the density function of noise. Using the results from (4), amplification for the i-th linear discriminant can be computed directly over these linear discriminant values as follows:

$$\begin{aligned} A_{\max} &= \max_{X_1,X_2,y} \lim_{dx \to 0} \frac{P([X_1, X_1 + dx] \to y)/dx}{P([X_2, X_2 + dx] \to y)/dx} \quad \forall X_1, X_2 \in D_P^i, y \in D_{NP}^i \\ &= \max_{X_1,X_2,y} \frac{f_\Delta(y - X_1)}{f_\Delta(y - X_2)} \quad \text{by (4)} \end{aligned}$$

If the noise follows Laplace distribution with zero mean, as proved in [41] amplification is bounded by the support bounds of data distribution and has the value of

$$\text{Amplification} = e^{\frac{\{\max(x) - \min(x)\}}{b}} \tag{5}$$

Using (5) the amplification equals

$$\begin{aligned} A_{\max} &= e^{\frac{\{\max(D_P^i) - \min(D_P^i)\}}{b_i}} \\ &= e^{\frac{\{\max(D_P^i) - \min(D_P^i)\}}{b \cdot \{\max(D_P^i) - \min(D_P^i)\}}} \\ &= e^{1/b} \end{aligned} \tag{6}$$

where b_i is the scale parameter of the Laplace noise added to the i-th linear discriminant. The bounds of D_P^i are computed from the linear discriminant scores of the

data. Note that b_i is proportional to the range of the i-th linear discriminant, thus the amplification for each column is the same and dependent only on b. Further the linear discriminants being independent, amplification is calculated on each column individually since one cannot infer values of a column from another column.

Although the data is grouped, grouping is performed over the linear discriminants. Therefore, the data is already decorrelated. The Laplace noise is added over each group and the noise is scaled by the range of each column in each group. If there is reasonable amount of data points in each group and the range of each column is significant there will be limited adverse effect on privacy.

As discussed before, in distributed cases we use the semi-honest model. To prove that the distributed protocols are secure we have used the composition theorem. The security analysis for both horizontally and vertically partitioned data in the previous sections show that the distributed protocols are secure as the sub-protocols are secure. Some intermediate statistics such as range of each group and scatter matrices are revealed as described in Sects. 4.1 and 4.2. However if we set appropriate minimal group size (set by the user) with a significant range of values in each column, then knowing these statistics will not allow attackers to get good estimate of the values of an individual data point. A large range may indicate an individual with an extreme value is present in the data. However, the attacker will not know which person in the group has that value because the privacy model described above explicitly prevents him from doing that (all people in the same group are indistinguishable to the attacker).

5 Experiments

This section presents experimental evaluation of the proposed method. Section 5.1 describes setup of the experiments. Section 5.2 describes results for the centralized case. Section 5.3 describes results for the distributed case.

5.1 Setup

The experiments were conducted on a machine with Pentium Dual Core, 2.0 GHz CPU, 3.0 GB of RAM, and running Windows Vista. All algorithms were implemented using Matlab 7.0.

Datasets The experiments were run over six real datasets and two synthetic datasets. Among the real datasets, five are from UCI Machine Learning Repository namely Iris, Wine, Pendigits, Breast Cancer and Adult [21]. A real life dataset for Lung Cancer [22] has been used as well. The synthetic datasets were generated in Matlab such that each dataset follows a similar structure as in Fig. 1. The Synthetic1 dataset has 2000 rows and 4 columns (including class label) and 4 classes (each class with 500 points) and Synthetic2 data has 2500 rows with 5 classes (each class having 500 data points) and 17 columns (including class label).

Privacy measures Three approaches have been proposed in the literature to measure privacy: the first using confidence interval [5], the second using information theory [3], and the third using amplification to measure worst-case privacy breach as mentioned. However, the information theory approach is inappropriate for distance-based mining algorithms such as KNN because Euclidean distance is based on individual data values, while information theory only considers the distribution of values [60].

This paper uses amplification to measure the *worst-case* privacy breach and uses the confidence interval to measure the *average* privacy. In our approach, the amplification γ equals to $e^{1/b}$ according to Theorem 1 where b is the level of Laplace noise. The confidence interval measure is as follows: If a transformed attribute x can be estimated with c % confidence in the interval $[x_1, x_2]$, then the privacy equals $\frac{x_2 - x_1}{x_U - x_L}$ where x_U is the maximal value of x and x_L is the minimal value. 95 % confidence interval was used in experiments. All the experiments were run five times and the average measure has been reported.

Comparison with other algorithms The following algorithms are compared in the experimental evaluation.

1. LDA + groupwise-noise: this is our approach. Here, noise generated using Laplace distribution is added separately on each group created after grouping each class (range of each group is used to generate the noise and added to each corresponding group) in the dataset.
2. PCA + noise: this is the method proposed in [41]. This method is the only existing perturbation method that provides a worst-case privacy guarantee. However, it uses PCA rather than LDA and does not have the grouping step.
3. LDA + noise: this algorithm is the same as our approach except that the grouping step is skipped and the added Laplace noise is scaled to the ranges of the whole data set. So the difference between this method and PCA + noise will show the difference between LDA and PCA.
4. LDA + classwise-noise: this is the same as our approach except that Laplace noise is added separately on each class and the noise is scaled to the ranges of each class. This method can be seen as somewhat in between our approach and LDA + noise.

The mining quality of all algorithms is the same in the distributed case as in the centralized case, so we conducted extensive experiments to compare mining quality in the centralized case and focus on communication cost and cost of encryption for the distributed case. In distributed case, we also compared our generic approach with using specific SMC algorithms for each mining algorithm.

5.2 Results for centralized case

Figure 4 reports accuracy vs. noise Level for each dataset. The noise level is varied at 0.1, 0.2, and 0.3. The classification algorithm is KNN with best k (found through cross-validation). The number of retained coefficients s after LDA is 1 for all the datasets except Pendigits where $s = 4$ (number of classes in Pendigits is 10). The

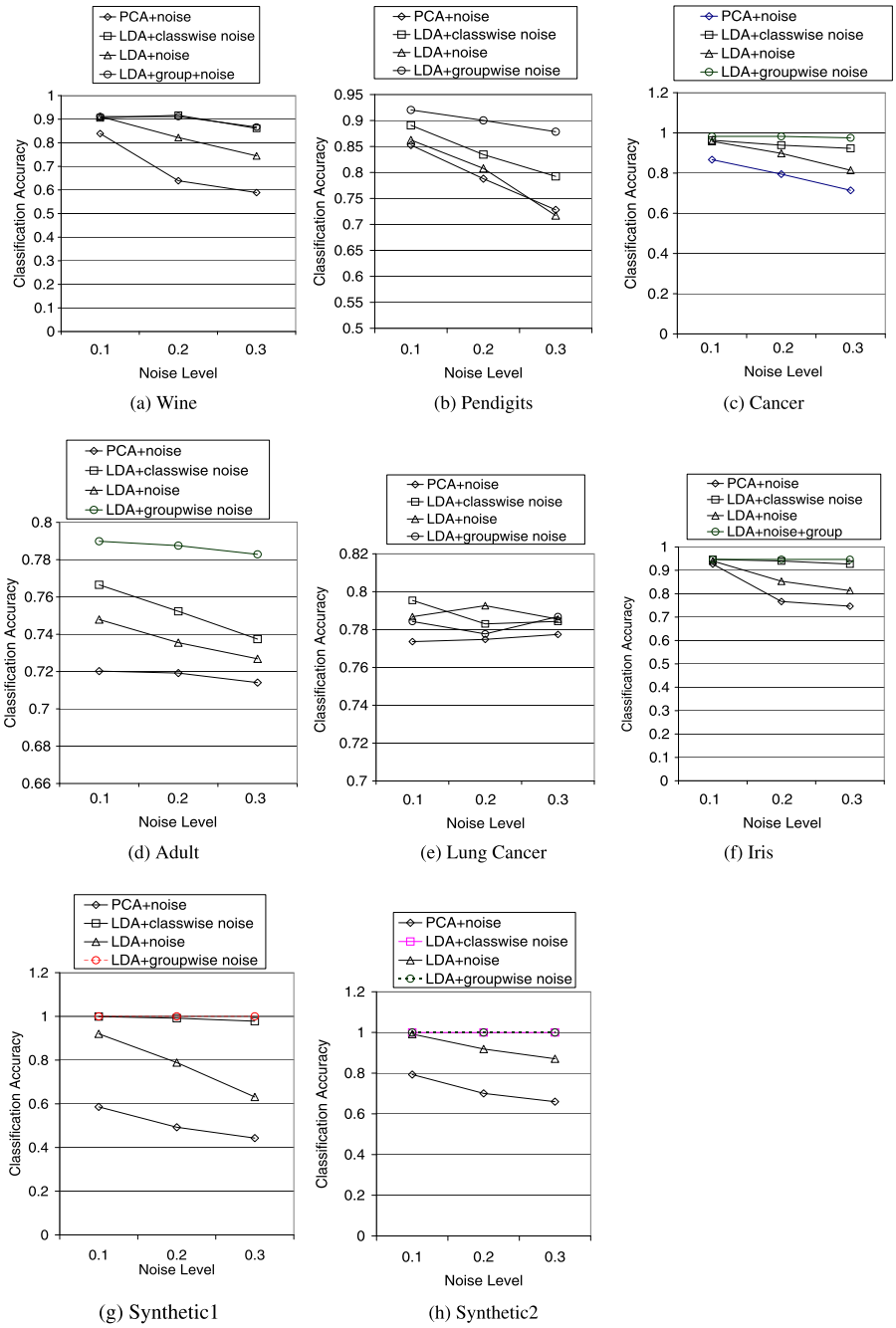


Fig. 4 Accuracy vs. noise level for various datasets

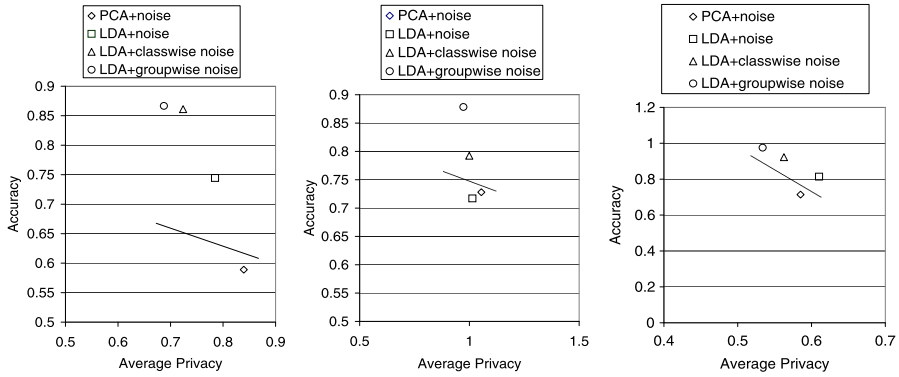
minimum predetermined group size varies with dataset (e.g. 20 for Iris to 1000 for Adult). The results show that the accuracy of all privacy preserving methods decreases as noise level increases. However, the decrease of LDA + groupwise noise is the least significant among all methods and it leads to the highest accuracy in almost all cases except for Lung Cancer data where all methods more or less behave similarly (the difference of accuracy among all methods is about 2 % for that data set). Our approach leads to higher accuracy than PCA + noise for two reasons: (1) our approach uses LDA which takes class label into account while PCA does not; (2) our approach adds group-wise noise while PCA + noise adds global noise which distorts data a lot more than our approach.

Among the three LDA based methods, LDA + classwise noise is usually the second best. The gap between it and our approach (LDA + groupwise noise) is significant only for Pendigits and Adult and the gap is insignificant for all other data sets. This proves that groupwise noise is only useful when the classes are not very well formed and thus further grouping of each class after LDA transformation increases the mining accuracy of the perturbed data. On the other hand, LDA + noise is often better than PCA + noise, showing the benefit of LDA over PCA. But it is significantly worse than the other two LDA based methods for most data sets, due to the same problem of adding global noise.

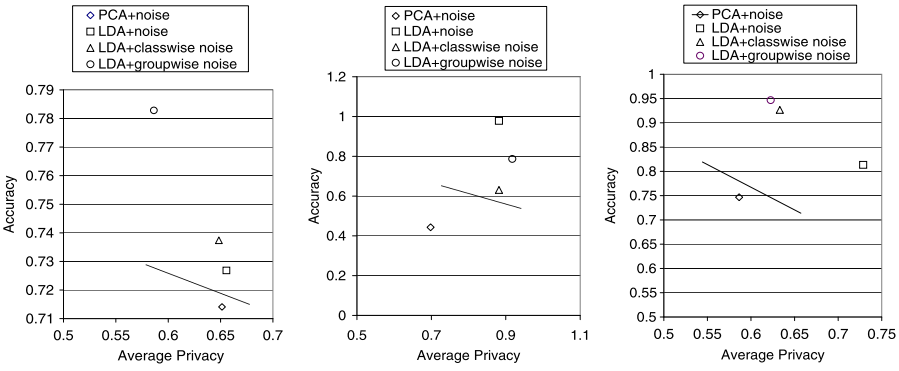
Figure 5 shows the average privacy vs. accuracy for each dataset. The noise level is fixed at 0.3. The x-axis represents the average privacy measure while the y-axis depicts the accuracy. Thus methods lying above the line (manually drawn) have better accuracy-privacy tradeoff. The results show that the three LDA based methods provide better accuracy and in most cases similar or even higher privacy measures than the existing method [41]. Among the three LDA based methods, our approach usually gives the highest accuracy but slightly lower privacy. So in practice if users are more concerned with accuracy then our approach is the choice. Otherwise users may choose the other two LDA based methods.

Figure 6 reports the worst-case privacy measure for all datasets. Figure 6(a) reports the amplification for all data sets using our proposed method in Algorithm 1. Note that the amplification for all data sets are exactly the same because as shown in Theorem 1, they depend only on noise scaling parameter b. Thus one figure is used for all data sets.

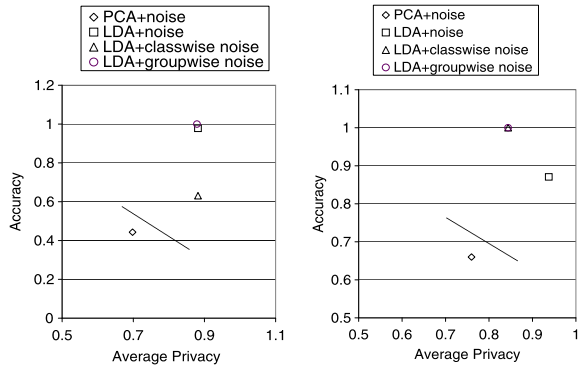
Equation in Theorem 1 describes the relationship of amplification (γ) and the maximal value of ρ_2 in ρ_1 to ρ_2 privacy breach. Suppose ρ_1, the probability of a privacy sensitive property in the original data, is 0.1 %. Figure 6(b) reports the maximal possible value of ρ_2 (the probability of the privacy sensitive property given sanitized data) for various values of b. For example, for $b = 0.3$, the maximal possible value of ρ_2 is 2.8 %. This means for $b = 0.3$, the probability of this privacy sensitive property will not exceed 2.8 % given the sanitized data. Typically most privacy-sensitive data properties (can be the data values themselves) in real life appear in original data with low probabilities. The results show that the proposed method does not increase the conditional probabilities of such properties too much thus effectively restricting worst-case privacy breaches.In both the figures the noise level b is varied from 0.1 to 0.4.



(a) Wine (b) Pendigits (c) Cancer

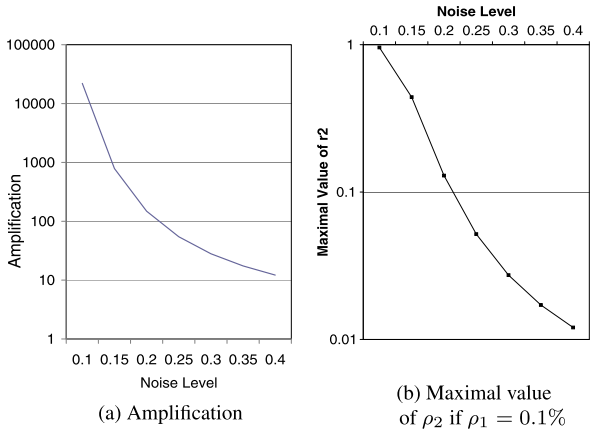


(d) Adult (e) Lung Cancer (f) Iris



(g) Synthetic1 (h) Synthetic2

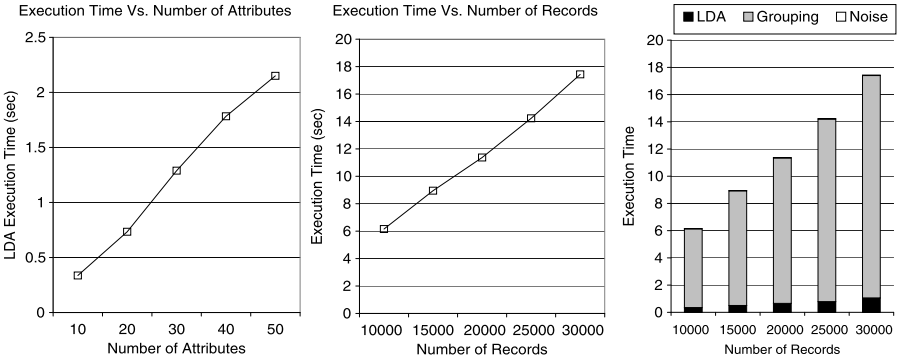
Fig. 5 Accuracy vs. privacy for various datasets



(a) Amplification

(b) Maximal value of ρ_2 if $\rho_1 = 0.1\%$

Fig. 6 Worst-case privacy for all data sets



(a) (b) (c)

Fig. 7 Performance analysis over synthetic datasets

Execution time We have also analyzed the performance measures of the algorithm presented here. We have used laboratory generated synthetic data sets for this purpose. Figure 7(a) provides the execution time for the step of Linear Discriminant analysis while we change the number of attributes in the dataset as provided in the x-axis. The y-axis shows the execution time. The number of records and number of groups after LDA is maintained constant at 20,000 and 1,000 respectively. The execution time for the step of grouping and noise addition is not presented in this set of experiments as the number of attributes after LDA remains same after LDA. In Fig. 7(b) the execution time is presented as we vary the number of records while the number of attributes is constant at 20 and group size is maintained as 1,000. In Fig. 7(c) we present the execution time for each of the step in our algorithm: LDA, grouping and noise addition. As apparent from Fig. 7 the execution time of the presented method is linearly proportional to the size of the dataset.

5.3 Results for distributed case

Since mining quality of all algorithms in the distributed case is the same as the centralized case except when our method uses stratified sampling for vertical partitioned data set. So we only reported mining quality for vertical partitioned case.

Since our approach is also generic compared to existing SMC algorithms, we also report our approach's performance for various mining algorithms.

We are also interested in the overhead for the distributed mining case. Since network bandwidth and encryption operations are two typical bottlenecks in distributed mining, we compare the communication cost (in terms of bytes) and the number of encryption operations of our approach to that of traditional SMC algorithms (i.e., one algorithm for each mining algorithm).

We used Synthetic2 in our experiments. In the horizontally partitioned case, we randomly partitioned the data set into 5 partitions, each with about 500 rows. In the vertically partitioned case, we randomly partitioned the data set into 3 partitions, each with 6 columns plus a row ID. The class label is stored in the first partition. We also used 20 groups and selected 4 columns after LDA. We used 30 % of the random sample in the vertical partitioned case.

Stratified sampling for vertically partitioned datasets The results using stratified sampling have been presented in Fig. 8. We have also conducted experiments to measure the impact of stratified sampling for vertically partitioned data sets. The goal of stratified sampling is to reduce communication cost and the benefit is significant only for large data sets. Thus the experiments have been performed in datasets which have more than 1000 data points. The X-axis represents the noise level which is varied from 0.1 to 0.3. The Y-axis represents the KNN accuracy. We compared our proposed method of 'LDA + groupwise noise' with and without sampling (the sampling percentage is 30 %) and reported accuracy of KNN. We noticed less than 1 % difference in terms of accuracy for the method with and without sampling. The detail results are omitted due to space constraints. As it is apparent from the figures, the accuracy of 'LDA + groupwise Noise' algorithms for vertical partitioning of data is not affected due to the sampling in all the datasets.

Classification accuracy for other classification methods We have tried exploratory data analysis by running multiple classification method over the perturbed data. The classification methods include Support Vector Machine, Naive Bayes, Bayesian Network and Decision Tree (J48). We have used the Data Mining Tool Weka Version 3.6 for this purpose. The results are displayed in Fig. 9. The noise level has been fixed at 0.2 for this set of experiments. The results show that there is a loss in accuracy from the original data after perturbing the data but the drop is not significant for our approach in most of the datasets. Our approach also has better accuracy than PCA + noise for most cases except for Iris and Pendigits. Iris contains only 4 data attributes so PCA and LDA makes little difference. For Pendigits our method does not work well only for SVM, probably because SVM has difficulty in separating many classes (Pendigits has 10 classes) after LDA.

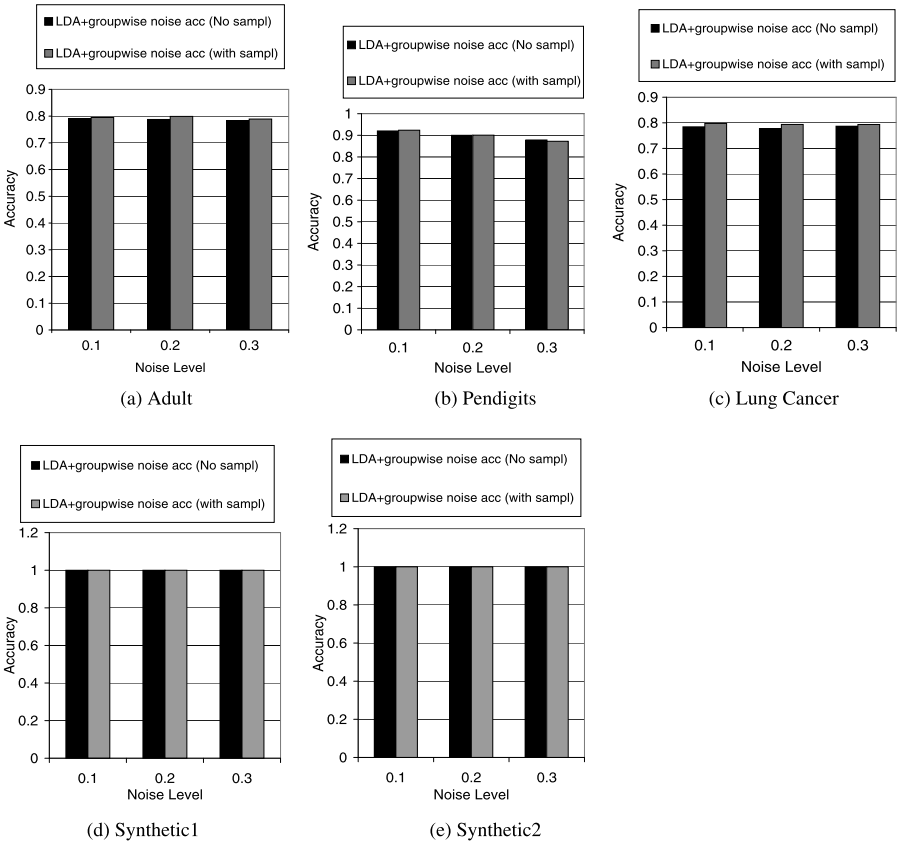
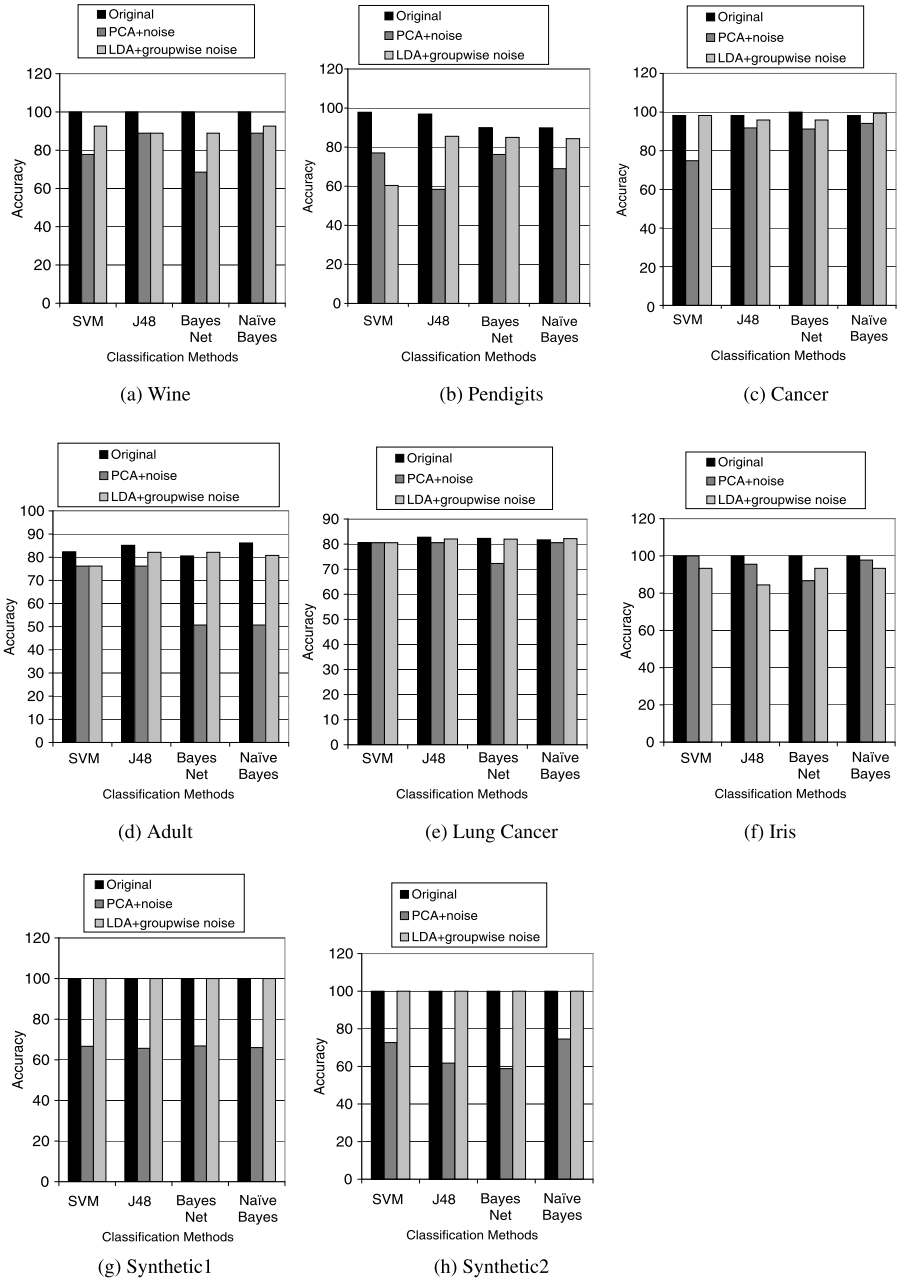


Fig. 8 Accuracy vs. noise level on 30 % sampling of datasets

Communication cost and number of encryption operations We considered a case where all parties try both Naive Bayes and SVM algorithms. Using our approach, all parties only need to sanitize data once. Using traditional SMC approaches, all parties would first run the SMC protocol for Naive Bayes and then the protocol for SVM.

Table 2 reports the communication cost in terms of MB. The results show that our method leads to significant savings (in about 2 orders for horizontally partitioned case and about a factor of 6 for vertically partitioned case).

The overhead of SMC protocols are dominated by the cost of encryption. Table 3 reports the number of encryption operations (in thousands) as well as the execution time (in parenthesis). We used Pallier encryption [44] in SMC protocols. The results show that our method leads to significant savings in terms of encryption cost. Further, the cost of encryption is quite small for the horizontally partitioned case because most computation can be done locally without the need of encryption. The cost is higher for the vertically partitioned case, but is still tolerable because our algorithm only needs to be run once. The unit execution time of a single encryption operation is 0.0017 second and we expect it to be reduced further with the rapid improvement in hardware performance.



(a) Wine (b) Pendigits (c) Cancer

(d) Adult (e) Lung Cancer (f) Iris

(g) Synthetic1 (h) Synthetic2

Fig. 9 Classification accuracy for various classification methods

6 Conclusion and future work

This paper proposes a method to provide a worst-case privacy guarantee for classification algorithms. It provides an algorithm for the centralized case and then extends the

Table 2 Comparison of communication cost (in MB) for various methods

Methods	Horizontal	Vertical
Our method	0.15	203.5
Traditional SMC	64.02	1238.4

Table 3 Number of encryption operations (in thousands) and execution time in seconds (shown in parenthesis) for various methods

Methods	Horizontal	Vertical
Our method	2.4 (4.13 sec)	3180 (5469.6 sec)
Traditional SMC	1000.4 (1720.7 sec)	19350 (33282 sec)

work to distributed scenarios providing secure protocols for both horizontal as well as vertical partitioning of the datasets. In order to provide a worst-case privacy guarantee this paper uses Laplace noise framework as presented in [41] but uses Linear Discriminant Analysis to eliminate correlation between attributes instead of Principal Component Analysis as LDA is proved to be more effective in case of classification. Experiments also prove that this method works well for multiple classification algorithms.

For better accuracy we have proposed a grouping technique. Privacy is related to the range of values in a group. Privacy is also problematic if a group contains too few points. So we assume user provides a threshold for minimal group size and minimal range. Right now our proposed method takes the minimal group size into consideration but finding the optimal setting of noise level, minimal group size, and minimal range will be our future research work.

Another future research direction will be to minimize communication cost for the distributed protocols. The distributed protocols have been presented for both horizontally and vertically partitioned datasets but further optimization of the protocols will reduce the communication cost. Therefore, our future research plan is also to explore algorithms to optimize communication cost.

Appendix

In the appendix we give brief summary of the SMC protocols we use in this paper.

Secure sum protocol The secure sum protocol [10] uses homomorphic encryption. More specifically, suppose we want to compute the sum of x_i $(1 \leq i \leq r)$ where x_i belongs to party i. Two parties are first randomly selected. Without loss of generality, suppose party 1 and 2 are chosen. Party 1 then generates a pair of public and private keys for a homomorphic encryption scheme. It then sends the public key to the other parties. Each party uses the public key to encrypt x_i and then sends the encrypted x_i to party 2. Party 2 then computes the encrypted sum without decrypting each individual x_i using the property of homomorphic encryption. Party 2 then sends the result back to party 1, which will decrypt the result and distribute the sum to each party.

The secure sum protocol is used in our algorithm for horizontally partitioned case to compute number of elements in class, class mean, global mean, global S_b, and

global S_w, all of which are essentially sum of local shares. The secure sum protocol is also used by our algorithm for vertically partitioned case to compute the result of LDA transform ($\hat{D}_t$).

The secure sum protocol just needs r encryption operations (r as the number of parties) and 1 decryption operation. So both the communication overhead and computational overhead is $O(r)$.

Secure scalar product We use the secure scalar product protocol that was first proposed in [24] and later used in [26]. Suppose two parties A and B each has a vector x^A and x^B. We want to compute the scalar product of $x^A \cdot x^B$. The protocol also uses homomorphic encryption. Party A generates a pair of public and private keys and sends the public key to B. Party A then computes encrypted value for x^A and sends it to Party B. Party B selects a random value s_B and uses the property of homomorphic encryption to compute the encryption of $x^A \cdot x^B - s_B$ and sends it back to Party A. Party A decrypts the result and gets $s_A = x^A \cdot x^B - s_B$. Now each party has a random share (s_A and s_B) for the scalar product.

The secure scalar product is used in the secure K-Means clustering [26] algorithm to compute distances. We also use it in our algorithm for the vertically partitioned case to compute S_b and S_w. Let us first consider S_b. Let vector $x_i (1 \leq i \leq m)$be a 1 by c vector where the j-th value x_{ij} is the value of $\mu_j - \bar{x}$ on column i. So the value at i-th row and j-th column of S_b is essentially scalar product of x_i and x_j. Similarly, for S_w, suppose y_{lj} is a 1 by $|c_l|$ vector where the i-th element is the value of the i-th row in class c_l on column j minus the value of μ_l on column j. The value at i-th row and j-th column of S_w is the sum of a series of c (c as the number of classes) scalar products where each scalar product equals $y_{li} \cdot y_{lj}$ (i.e., the scalar product computed in class c_l but over the column i and j).

The protocol requires n encryption operations and one decryption operation for a length n vector. Since the encrypted value of a data element may be used in several scalar product computations (e.g., column 1 will be used to compute the scalar product with column 2, 3, . . . ,), we can reuse the encrypted value. Thus each data element needs to be encrypted at most once. Our algorithm requires $O(mn)$ encryption operations to compute S_b and S_w in the vertically partitioned case.

Secure comparison protocol We use the secure comparison protocol to find the closest cluster and the range of each column in our algorithms for horizontally and vertically partitioned cases. The protocol was proposed by Yao at [57].

Secure K-Means clustering We use the secure K-Means clustering protocol proposed in [26]. This protocol also calls secure sum, secure scalar protocol, and Yao's secure comparison protocol. The secure K-Means clustering protocol requires $O(sngrr_k)$ encryption operations where n is number of rows in data, s is number of columns after LDA, r is number of parties, g is the number of clusters, and r_k is the number of iterations in K-Means clustering.

The overhead of these SMC protocols are dominated by the cost of encryption. In the horizontally partitioned case, our algorithm requires $O(mcr + sgrr_k)$ encryption operations. Note that this is irrelevant to the number of rows in the data set because

each party can do most computations locally. So the algorithm is efficient. In the vertically partitioned case, our algorithm requires $O(mn + sngrr_k)$ encryption operations. Encryption can be quite expensive in practice, especially for large data sets under the vertically partitioned case. However, as stated in [24], the current hardware and some optimization tricks (such as reusing encrypted values) can make the computational overhead tolerable.

References

1. Aggarwal, C.C., Yu, P.S.: A condensation approach to privacy preserving data mining. In: 9th International Conference on Extending Database Technology, Heraklion, Crete, Greece (2004)
2. Aggarwal, C.C., Yu, P.S.: Privacy-Preserving Data Mining: Models and Algorithms. Springer, Berlin (2008)
3. Agrawal, D., Aggarwal, C.C.: On the design and quantification of privacy preserving data mining algorithms. In: 20th ACM PODS, Santa Barbara, CA, pp. 247–255 (2001)
4. Agrawal, S., Haritsa, J.R.: A framework for high-accuracy privacy-preserving mining. In: ICDE (2005)
5. Agrawal, R., Srikant, R.: Privacy preserving data mining. In: 2000 ACM SIGMOD, Dallas, TX, May 2000, pp. 439–450 (2000)
6. Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: the sulq framework. In: PODS (2005)
7. Blum, A., Ligett, K., Roth, A.: A learning theory approach to non-interactive database privacy. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC'08, pp. 609–618. ACM, New York (2008). http://doi.acm.org/10.1145/1374376.1374464
8. Caragea, D., Silvescu, A., Honavar, V.: Decision tree induction from distributed, heterogeneous, autonomous data sources. In: Conference on Intelligent Systems Design and Applications (2003)
9. Chen, K., Liu, L.: A random rotation perturbation approach to privacy-preserving data classification. In: ICDM 2005, Houston, TX, November 2005
10. Clifton, C., Kantarcioglu, M., Vaidya, J., Lin, X., Zhu, M.: Tools for privacy preserving distributed data mining. ACM SIGKDD Explor. **4**, 28–34 (2002)
11. Dalenius, T., Reiss, S.P.: Data-swapping: a technique for disclosure control. J. Stat. Plan. Inference **6**, 73–85 (1982)
12. Du, W., Zhan, Z.: Building decision tree classifier on private data. In: IEEE International Conference on Privacy, Security and Data Mining, Maebashi City, Japan, December 2002, pp. 1–8 (2002)
13. Du, W., Zhan, Z.: Using randomized response techniques for privacy preserving data mining. In: 9th ACM SIGKDD, Washington, DC, August 2003, pp. 505–510 (2003)
14. Dwork, C.: Differential privacy. In: ICALP, pp. 1–12 (2006)
15. Dwork, C.: Differential privacy: a survey of results. In: Proceedings of the 5th International Conference on Theory and Applications of Models of Computation, TAMC'08, pp. 1–19. Springer, Berlin (2008). http://dl.acm.org/citation.cfm?id=1791834.1791836
16. Dwork, C., Mcsherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd Theory of Cryptography Conference, pp. 265–284. Springer, Berlin (2006)
17. Dwork, C., Naor, M., Reingold, O., Rothblum, G.N., Vadhan, S.: On the complexity of differentially private data release: efficient algorithms and hardness results. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC'09, pp. 381–390. ACM, New York (2009). http://doi.acm.org/10.1145/1536414.1536467
18. Evfimevski, A., Gehrke, J., Srikant, R.: Limiting privacy breaches in privacy preserving data mining. In: 22nd ACM PODS, San Diego, CA, June 2003, pp. 211–222 (2003)
19. Feldman, D., Fiat, A., Kaplan, H., Nissim, K.: Private coresets. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC'09, pp. 361–370. ACM, New York (2009). http://doi.acm.org/10.1145/1536414.1536465
20. Fienberg, S.E., McIntyre, J.: Data-swapping: variations on a theme by Dalenius and Reiss. Tech. rep., National Institute of Statistical Sciences (2003)
21. Frank, A., Asuncion, A.: UCI machine learning repository (2010). http://archive.ics.uci.edu/ml

22. Gal, T., Chen, Z., Gangopadhyay, A.: A privacy protection model for patient data with multiple sensitive attributes. Int. J. Inf. Secur. Priv. **2**(3), 28–44 (2008)
23. Giannella, C., Liu, K., Olsen, T., Kargupta, H.: Communication efficient construction of decision trees over heterogeneously distributed data. In: Fourth IEEE International Conference on Data Mining (2004)
24. Goethals, B., Laur, S., Lipmaa, H., Mielikainen, T.: On secure scalar product computation for privacy-preserving data mining. In: The 7th Annual International Conf. in Information Security and Cryptology (2004)
25. Huang, Z., Du, W., Chen, B.: Deriving private information from randomized data. In: SIGMOD 2005, Baltimore, MD, June 2005, pp. 37–48 (2005)
26. Jagannathan, G., Wright, R.N.: Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In: SIGKDD'05, Chicago, IL, pp. 593–599 (2005)
27. 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)
28. Kantarcioglu, M., Vaidya, J.: Privacy preserving naïve Bayes classifier for horizontally partitioned data. In: IEEE ICDM Workshop on Privacy Preserving Data Mining, Melbourne, FL, November 2003, pp. 3–9 (2003)
29. Kargupta, H., Park, B.H.: A Fourier spectrum-based approach to represent decision trees for mining data streams in mobile environments. IEEE Trans. Knowl. Data Eng. **16**(2), 216–229 (2004)
30. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the privacy preserving properties of random data perturbation techniques. In: ICDM, pp. 99–106 (2003)
31. Kim, J.J., Winkler, W.E.: Multiplicative noise for masking continuous data. Tech. rep. 2003-01, Statistical Research Division, U.S. Bureau of the Census, April 2003
32. Kim, D., Chen, Z., Gangopadhyay, A.: Optimizing privacy-accuracy tradeoff for privacy preserving distance-based classification. Int. J. Inf. Secur. Priv. **6**(2), 16–33 (2012)
33. Li, C., Hay, M., Rastogi, V., Miklau, G., McGregor, A.: Optimizing linear counting queries under differential privacy. In: Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'10, pp. 123–134. ACM, New York (2010). http://doi.acm.org/10.1145/1807085.1807104
34. Lin, X., Clifton, C., Zhu, Y.: Privacy preserving clustering with distributed em mixture modeling. Int. J. Knowl. Inf. Syst. **8**(1), 68–81 (2005)
35. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Advances in Cryptology (CRYPTO'00). Lecture Notes in Computer Science, vol. 180, pp. 36–53 (2000)
36. Liu, K., Giannella, C., Kargupta, H.: An attacker's view of distance preserving maps for privacy preserving data mining. In: The 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'06) (2006)
37. 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)
38. Ma, D., Sivakumar, K., Kargupta, H.: Privacy sensitive Bayesian network parameter learning. In: 4th IEEE International Conference on Data Mining (ICDM'04), Brighton, UK, November 2004, pp. 487–490 (2004)
39. Mcsherry, F.: Mechanism design via differential privacy. In: Proceedings of the 48th Annual Symposium on Foundations of Computer Science (2007)
40. Mukherjee, S., Chen, Z., Gangopadhyay, A.: A privacy preserving technique for Euclidean distance-based mining algorithms using Fourier-related transforms. VLDB J. **15**(4), 292–315 (2006)
41. Mukherjee, S., Banerjee, M., Chen, Z., Gangopadhyay, A.: A privacy preserving technique for distance-based classification with worst case privacy guarantees. Data Knowl. Eng. **66**(2), 264–288 (2008)
42. Mukherjee, S., Chen, Z., Gangopadhyay, A.: A fuzzy programming approach for data reduction and privacy in distance based mining. Int. J. Inf. Comput. Secur. (in press)
43. Oliveira, S., Zaïane, O.R.: Privacy preserving clustering by data transformation. In: 18th Brazilian Symposium on Databases, pp. 304–318 (2003)
44. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT, pp. 223–238. Springer, Berlin (1999)
45. Roth, A., Roughgarden, T.: Interactive privacy via the median mechanism. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC'10, pp. 765–774. ACM, New York (2010). http://doi.acm.org/10.1145/1806689.1806794
46. Sweeney, L.: K-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. **10**(5), 557–570 (2002)

47. Vaidya, J.: Towards a holistic approach to privacy-preserving data analysis. In: Workshop on Secure Knowledge Management (2008)
48. Vaidya, J.S., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: 8th ACM SIGKDD, Edmonton, Canada, July 2002, pp. 639–644 (2002)
49. Vaidya, J.S., Clifton, C.: Privacy-preserving k-means clustering over vertically partitioned data. In: 9th ACM SIGKDD, Washington, DC, August 2003, pp. 206–215 (2003)
50. Vaidya, J., Clifton, C.: Privacy-preserving decision trees over vertically partitioned data. In: Proceedings of the IFIP WG 11.3 International Conference on Data and Applications Security, pp. 139–152. Springer, Berlin (2005)
51. Vaidya, J., Clifton, C., Zhu, M.: Privacy Preserving Data Mining. Springer, Berlin (2005)
52. Vaidya, J., Kantarcioglu, M., Clifton, C.: Privacy-preserving naïve Bayes classification. VLDB J. **17**(4), 879–898 (2008)
53. Warner, S.: Randomized response: a survey technique for eliminating evasive answer bias. J. Am. Stat. Assoc. **60**(309), 63–69 (1965)
54. Wright, R., Yang, Z.: Privacy-preserving Bayesian network structure computation on distributed heterogeneous data. In: 10th ACM SIGKDD Conference (SIGKDD'04), Seattle, WA, August 2004, pp. 713–718 (2004)
55. Xiao, X., Bender, G., Hay, M., Gehrke, J.: Ireduct: differential privacy with reduced relative errors. In: Proceedings of the 2011 International Conference on Management of Data, SIGMOD'11, pp. 229–240. ACM, New York (2011). http://doi.acm.org/10.1145/1989323.1989348
56. Xiao, X., Wang, G., Gehrke, J.: Differential privacy via wavelet transforms. IEEE Trans. Knowl. Data Eng. **23**, 1200–1214 (2011). doi:10.1109/TKDE.2010.247
57. Yao, A.C.: How to generate and exchange secrets. In: 27th IEEE Symposium on Foundations of Computer Science, pp. 162–167 (1986)
58. Yu, H., Jiang, X., Vaidya, J.: Privacy-preserving svm using nonlinear kernels on horizontally partitioned data. In: Proceedings of the 2006 ACM Symposium on Applied Computing, SAC'06, pp. 603–610 (2006)
59. Yu, H., Vaidya, J., Jiang, X.: Privacy-preserving svm classification on vertically partitioned data. In: PAKDD, pp. 647–656 (2006)
60. Zhu, Y., Liu, L.: Optimal randomization for privacy preserving data mining. In: KDD, pp. 761–766 (2004)