A Utility-Aware and Holistic Approach for Privacy Preserving Distributed Mining with Worst Case Privacy Guarantee

Madhushri Banerjee
University of Maryland Baltimore County
madhu2@umbc.edu

Zhiyuan Chen
University of Maryland Baltimore County
zhchen@umbc.edu

Aryya Gangopadhyay
University of Maryland Baltimore County
gangopad@umbc.edu

Abstract

Organizations often want to predict some attribute values collaboratively. However, they are often unwilling or not allowed to directly share their private data. Thus there is great need for distributed privacy preserving techniques. There exists a rich body of work based on Secure Multi-Party Computation techniques. However, most such techniques are tied to a specific mining algorithm and users have to run a different protocol for each mining algorithm. A holistic approach [11] was proposed in which all parties first use a SMC protocol to generate a synthetic data set and then share this data for different mining algorithms. However, this approach has two major drawbacks: 1) it provides no worst case privacy guarantee, 2) parties involved in the mining process often know what attribute to predict, but the holistic approach does not take this into account. In this paper, we propose a method that addresses these shortcomings. Experimental results demonstrate the benefits of the proposed solution.

Keywords: Privacy preserving data mining, privacy.

1 Introduction

Organizations often have their own private data and want to predict some attribute values collaboratively. For example, several credit card companies may wish to collaboratively build a mining model to predict credit card fraud. However, they are often unwilling or not allowed to directly share their private data. Thus there is great need for distributed privacy preserving data mining methods.

There exists a rich body of work based on Secure Multi-Party Computation techniques [12]. However, most such techniques are tied to a specific mining algorithm and can not be generalized to other mining algorithms. In practice, all parties usually know what attribute they want to predict (e.g., in the above case, whether the use of credit card is legitimate), but may try many different mining algorithms and then choose the one with the best results. This means that all parties need to run a different protocol for each mining algorithm and each such protocol could be quite expensive because of the communication overhead and also encryptions are often used.

A holistic approach [11] was proposed 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 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 [10]. However, this model does not prevent 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, which may lead to inferior mining quality.

In this paper, we address these shortcomings and propose an approach that not only provides worst case privacy guarantee, but also takes into account the specific mining objective (e.g., to predict the value of an attribute). The proposed method applies when miners know the objective of mining, but want to try different mining algorithms and then choose the algorithm with the best results. Using our method, all parties just need to sanitize the data once for that particular objective, and then they can directly apply different mining

algorithms on the sanitized data.

The rest of the paper is organized as follows. Section 2 describes the existing literature. Section 3 gives some preliminaries. Section 4 discusses our approach. Section 5 presents the experimental results. Section 6 concludes the paper.

2 Related Work

There is a rich body of work on privacy preserving data mining. Surveys can be found in [2] and [12]. Such work can be divided into those dealing with centralized case (i.e., when data is sent to a third party for mining) or distributed case. For centralized case, the commonly used techniques include random perturbation [3], generalization [10], and synthetic data generation [1].

Most work in distributed setting uses secure multi-party computation and survey articles can be found in [12]. However, as mentioned in Section 1, most such techniques are tied to a specific mining algorithm and can not be generalized. They also use encryption which could be quite expensive. As mentioned in Section 1, a holistic approach was proposed in [11] to address this problem. However, it does not provide worst-base privacy guarantee and also does not take into account the goal of mining. A perturbation method for the centralized case was proposed in [9], which provides worst case privacy guarantee by adding Laplace noise. This paper extends this method to distributed settings.

There has been recent work on utility aware privacy preserving data mining [6, 8]. However, all such techniques are for centralized settings while this paper focuses on distributed settings. Further, most such works use generalization rather than synthetic data generation or random perturbation.

3 Preliminaries

This section introduces some preliminaries. Section 3.1 describes the condensation approach and its problems. Section 3.2 explains the worst-case privacy framework proposed in [9]. Section 3.3 describes the adversarial model.

3.1 Condensation Approach

Algorithm 1 describes the original condensation approach [1]. The basic idea is to generate synthetic data while preserving the statistics of the original data. Step 1 to 3 generate clusters with size at least k. Each cluster is generated by picking a random seed and then assigning $k-1$ records closest to the seed to the cluster. Statistics (including mean and covariance) is then computed for each cluster. Synthetic data is then generated for each cluster based on the statistics.

One problem of this method is that the group formation depends not only on selection of seeds but also the order of which the seeds are considered. To address this problem, Vaidya proposed an improved condensation approach [11]. It first generates initial groups using K-means clustering. It then moves records in clusters with fewer than k (k is an input) records to nearest clusters.

Algorithm 1 Condensation Approach

Input: Dataset D, Group Size k,

1: **while** D contains at least k points **do**
2: Select a random data point X from D;
3: Find the closest $(k-1)$ records to X and add them to group G
4: Compute statistics for group G
5: Generate synthetic data using statistics of group G
6: Add the resulting synthetic data to output H
7: remove G from D
8: **end while**
9: Assign each remaining point in D to the closest group and generate synthetic data as in Step 4-5
10: **return** H

3.2 Worst-Case Privacy Model

A worst case privacy model was proposed in [4] for categorical data. Later it was extended to numerical data in [9]. 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 $y \in V_Y$*

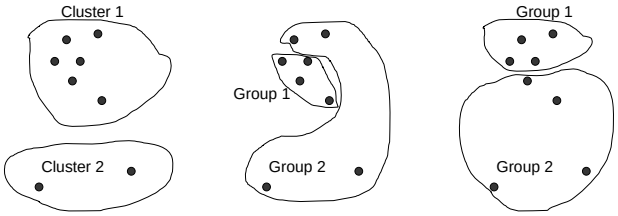
$$P[Q(X)] \leq \rho_1 \; and \; P[Q(X)|Y=y] \geq \rho_2$$

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

$$P[Q(X)] \geq \rho_2 \; and \; P[Q(X)|Y=y] \leq \rho_1$$

Definition 1 captures how much disclosing of a sanitized value increases predictability of the original value. E.g., suppose the original data contains salary 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 [9] to prevent such privacy breaches. The perturbation method first decorrelates data by applying Principal Component Analysis. It then adds noise following Laplace distribution to the principal components. Let D_P^i be the i-th column of the principal component matrix, 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



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

Figure 1. Example of moving records between clusters

distribution with mean zero and standard deviation b_i. [9] also proves that this perturbation method gives the following privacy guarantee.

Theorem 1 *The perturbation method proposed in [9] 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*

$$\frac{\rho_2}{\rho_1}\frac{(1-\rho_1)}{(1-\rho_2)} > e^{1/b}$$

For example, in the above example, suppose $b = 0.2$ and the probability of P having over 1 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 1 million dollar salary (ρ_2) will not exceed 0.13.

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 own data to other parties. We use the semi-honest adversarial model [12] 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 semi-honest model, parties will not collude.

A useful technique to prove the security of a SMC protocol is the composition theorem [12].

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 consider two cases: the data is either horizontally partitioned such that each party P_l contains a subset of records or vertically partitioned such that each party contains a subset of columns of the same set of records.

Algorithm 2 describes the method to generate a sanitized data set H. All parties can then directly share this data set for mining. The algorithm can be divided into 2 phases: 1) group generation (step 1 to 11), 2) perturbation (step 12 to 28). The next two subsections describe the details.

4.1 Group Generation

In the first phase, the algorithm generates g groups each containing at least $\lfloor |D|/g \rfloor$ records. Similar to the holistic approach, we use secure K-means clustering [7] (step 4) to generate g initial clusters. For horizontally partitioned data, we also use secure sum protocol to compute total data size $|D|$ to determine the group size constraint.

We propose two enhancements over the holistic approach in the first phase: 1) we use a weighted distance function to take into account the attribute that we want to predict, 2) we move records from a larger cluster to a smaller cluster to satisfy the group size constraint.

Weighted distance function: Suppose the goal of mining is to predict an attribute A_v. We call it *response variable*. Let m be the number of attributes, r_{iv} be the value of attribute A_v in record r_i, and $w, 0 \leq w \leq 1$ be a weight parameter decided by user. The weighted distance between records r_i and r_j equals

$$d_{ij}^2 = w(r_{iv} - r_{jv})^2 + \sum_{1 \leq l \leq m, l \neq v} \frac{(1-w)(r_{il} - r_{jl})^2}{m-1} \quad (1)$$

We can now assign a higher weight to the response variable so that records with similar response variable values are grouped together.

Moving records between clusters: Some clusters may have too few records (less than $\lfloor |D|/g \rfloor$). In [11], the author suggests to move records from these small clusters to other clusters. However, we consider a different approach to move records from larger clusters to smaller clusters.

The insight is shown in Figure 1. Figure 1(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. If we follow the approach in [11], records in cluster 2 will be moved to cluster 1 so we end up with only one group. This often leads to worse prediction accuracy because the group contains records with very different values. Of course, one can create more initial clusters (i.e., choose $K > g$), but it becomes quite difficult to predict the final number of groups.

Instead we move records from larger clusters to the smaller one. The question is which records to move. Figure 1(b) shows an example when some random records are moved to cluster 2. Cluster 2 becomes too scattered and too much distortion is introduced after sanitization. Instead, we move records in larger clusters that are closest to the center of the smaller cluster. Figure 1(c) displays the result. Now cluster 2 is much less scattered.

Let C_i be a cluster which does not contain enough records. We need to move records from other clusters to C_i. Clearly, it does not make sense to move records to a cluster C_j that is even smaller than C_i because then we need to move more records to C_j. Thus in line 5 we sort the clusters in ascending order of their sizes. In line 6-11, we securely select $\lfloor |D|/g \rfloor - |C_i|$ records in larger clusters to move. We

use the SMC protocol which finds the closest cluster center to a data point x in [7]. Since this protocol essentially finds the nearest neighbor of point x from a set of points (i.e., the cluster centers), here we use it to find records in larger clusters that are closest to cluster center c_i and move them to C_i.

4.2 Secure Perturbation

From line 13 to 28, we use the perturbation method described in Section 3.2 to provide worst-case privacy guarantee. The challenge is that the original method is for centralized case only. Thus we need to design a SMC protocol to securely compute it in a distributed setting.

The original perturbation method consists of 3 steps: 1) transforming data using PCA, 2) adding Laplace noise scaled to the range of each column in the principal component matrix, 3) applying a reverse PCA to map data back to original dimensions. The PCA transform matrix M can be inferred from the covariance matrix. Thus in line 13 the algorithm computes group statistics using the same SMC protocol as in the holistic approach [11] and then infer M. Next we describe the details of remaining steps for horizontally and vertically partitioned cases.

Horizontally partitioned case: When data is horizontally partitioned, the perturbation process can be largely done locally. Let D_l be party P_l's data. In line 16, each party P_l can locally transform their data by directly computing $D_{Pl} = D_l M$. The result will be still stored at party P_l. In line 18, all parties can use secure comparison [12] to compute the ranges of each column in the PCA component matrix without revealing their local data. In line 19, each party adds Laplace noise scaled to these ranges to their local principal components. Each party knows the noise added to its local data. However, they do not know the noise added to other party's data. Line 20 does a reverse PCA to map perturbed data back to original dimensions. This can be done by directly multiplying M^T (transpose of M) to perturbed local data (D'_{pl}).

Vertically partitioned case: When data is vertically partitioned, each party stores a subset of the columns of the same set of records. The main challenge is that now a data record is no longer locally owned by a single party.

PCA can be done using secure sum protocol as follows. The PCA transform matrix M can be divided into a number of smaller matrices $M_1, \ldots M_r$ where M_l corresponds to columns in party P_l. Each party P_l multiplies its local D_l to M_l (line 21). Note that $\sum_{l=1}^{r} D_l M_l = DM$. Thus all parties can use a secure sum protocol [12] to compute $\sum_{l=1}^{r} D_l M_l$ (line 22). The result will be stored as two random shares $D_{P1} = R$ at party P_1 where R is a random matrix and $D_{P2} = \sum_{l=1}^{r} D_l M_l - R$ at P_r. Clearly, $D_{P1} + D_{P2} = \sum_{l=1}^{r} D_l M_l$. Note that P_1 and P_r each has a random share of the principal components thus they can not infer the principal components without collusion. In line 23, P_1 and P_r compute ranges of each column in the PCA component matrix using secure comparison protocol.

Now the question is how to generate Laplace noise. We can not let any party to generate noise directly because then later that party can infer the original data from the sanitized data. Instead, we use the following property of Laplace noise. Suppose two independent and identically distributed random variables X_1, X_2 follows exponential distribution with parameter λ, then $X_1 - X_2$ follows Laplace distribution with $b = 1/\lambda$. Thus P_1 and P_r can independently generate two random noise matrices R_1 and R_2, each following the exponential distribution with $\lambda = 1/b_j$ for column j, and then they can use secure sum protocol to add up R_1, $-R_2$, D_{P1}, and D_{P2}. Note that P_1 (or P_r) only knows a random share of the noise and principal components, thus they can not infer the noise or data without collusion.

Security Analysis: Line 1 to 3 use secure sum so only the global data size is leaked. In line 4, the cluster membership and cluster sizes, as well as the cluster centers are revealed. In line 9, since secure minimal distance cluster protocol is used, the actual distance is not leaked. Parties only learn the IDs of records that are closest to c_i. In line 12 to 28, the group statistics as well as the ranges of principal components are revealed. Since secure comparison is used in line 17 and 23, and secure sum protocol is used in line 22 and 25, no additional information is leaked. In line 24, Party P_1 and P_r also only have random shares of the Laplace noise. Since each sub-protocol is secure, we can prove that the whole protocol is secure by the composition theorem. The formal proof is omitted due to lack of space.

Communication Cost: Let n be the number of records, m be the number of attributes, g be the number of groups, r be the number of parties, and r_k be the number of iterations in K-means clustering. When data is horizontally partitioned, most computation can be done locally and the communication cost is $O(mgrr_k + m^2 r)$ because it costs $O(mgrr_k)$ for secure clustering, $O(m^2 r)$ for computing group statistics, and $O(mgr)$ to compute b_j. When data is vertically partitioned, secure clustering takes $O(mngrr_k)$ and computing group statistics takes $O(m^2 nr)$. The perturbation method costs $O(mnr)$. Thus total communication cost is $O(mngrr_k + m^2 nr)$.

5 Experiments

5.1 Experiment 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 and Weka 3.6.

Datasets: We have used two data sets from UCI Machine Learning Repository [5]. One is the 'Concrete Compressive Strength' data set which has nine attributes and 1030 instances. The response variable is 'concrete compressive

strength'. The other data set is the 'Housing' data which contains 506 instances and 14 attributes with house price attribute as response variable. All attributes are numerical.

Algorithms: We implemented our approach (Algorithm 2). We set $b = 0.2$ in our approach. We also implemented an improved version of the condensation approach to use weighted distance. We call it "weighted condensation". We also implemented two improved versions of the holistic approach proposed in [11]. Both versions use weighted distance. One of them generates synthetic data and is called "weighted holistic" and the other uses the same perturbation method as our approach and is called "weighted holistic with guarantee".

Metrics: We measure both average privacy and quality of mining. The mining quality is measured as follows. We ran the M5P regression tree in WEKA 3.6 with 10 fold cross validation on the sanitized data and reported the correlation coefficient. The average privacy was measured using the 95% confidence interval metric proposed in [3]. The results reported were the average of 5 runs.

5.2 Results

We varied the weight on the response variable. Figure 2 shows the correlation coefficient of sanitized Concrete data when the weight w equals $1/m$ where m is number of attributes (i.e. equal weight to all attributes), 0.3, 0.5, 0.8, and 1. Figure 3 shows the results for Housing. In both data sets we have kept the group size greater than or equal to 100 and implemented all methods to have the same number of groups.

The results show that our approach leads to significantly better mining quality than other methods. The improvement over the original methods (without weighted distance) is even larger. For example, the highest correlation coefficient of our method is around 0.81 for both data sets. The correlation coefficient of the original condensation (with equal weight) is 0.41 for both data sets and the coefficient for original holistic approach (with equal weight) is 0.48 for Concrete and 0.42 for Housing.

All methods also generally perform better as weight increases because putting higher weight on response variable will put records with similar response variable values into the same group and thus improve the prediction accuracy. In practice, we can set default weight to 0.8 (about the optimal for all methods) and adjust it if necessary.

Figure 4 and Figure 5 show the tradeoff between average privacy (in terms of confidence interval) and correlation coefficient for our method and the original holistic approach. Our method uses the optimal weight while condensation uses equal weight. The X-axis is the privacy values for various number of groups (indicated as g) and Y-axis gives the corresponding correlation coefficients. Clearly, our method has better correlation coefficient than the holistic approach when they have similar average privacy. When

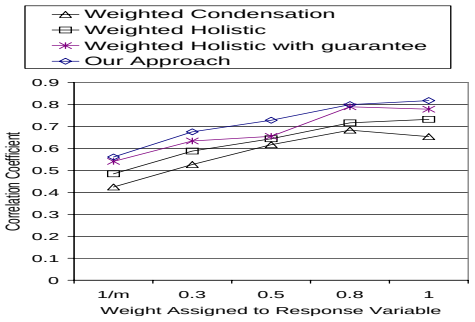


Figure 2. Correlation Coefficient for Concrete Dataset

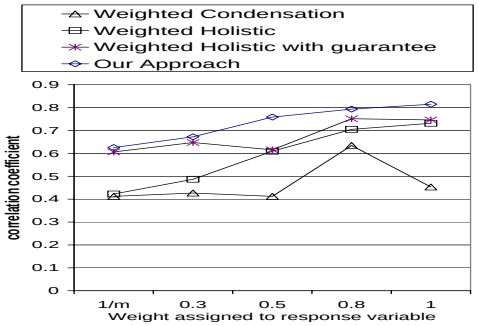


Figure 3. Correlation Coefficient for Housing Dataset

g is smaller, group size becomes larger, so privacy increases and correlation coefficient generally decreases for the holistic approach. However, for our method when g is very large (meaning group size is very small), the correlation coefficient of our method does not decrease as g decreases, probably because random noise introduces more error on smaller groups than on larger groups. This actually favors our approach because larger groups have better privacy.

6 Conclusion

In this paper we have proposed a distributed privacy-preserving method that allows data miners to sanitize the data only once for the same mining objective (e.g., to predict a certain attribute). Data miners can run different mining algorithms on the sanitized data. Unlike the holistic ap-

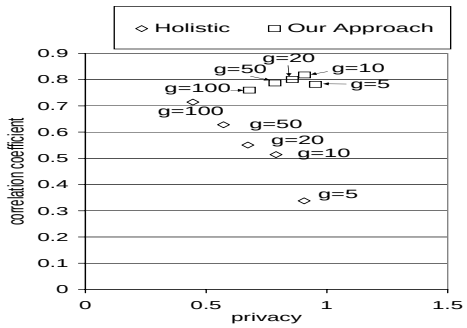


Figure 4. Privacy Vs Mining Quality for Concrete Dataset

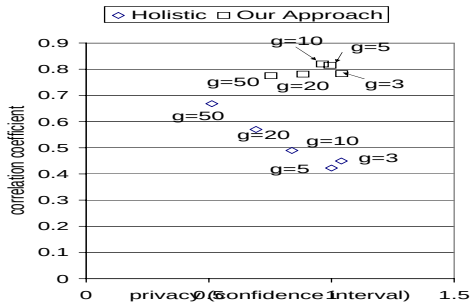


Figure 5. Privacy Vs Mining Quality for Housing Dataset

proach, our method also provides worst case privacy guarantee. Interestingly, by modifying the group generation process, our method also achieves better mining quality than the holistic approach. Our method is of course more expensive than the holistic approach due to more careful movement of records from larger clusters to smaller clusters (which requires nearest neighbor search). As future work we will investigate more efficient algorithms.

References

[1] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *EDBT*, 2004.

[2] C. C. Aggarwal and P. S. Yu. *Privacy-Preserving Data Mining: Models and Algorithms*. Springer, 2008.

[3] R. Agrawal and R. Srikant. Privacy preserving data mining. In *SIGMOD*, pages 439–450, Dallas, TX, May 2000.

[4] A. Evfimevski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, pages 211 – 222, San Diego, CA, June 2003.

[5] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[6] B. Fung and K. Wang. Anonymizing classification data for privacy preservation. *IEEE Trans. on Knowl. and Data Eng.*, 19:2007, 711-725.

[7] G. Jagannathan and R. N. Wright. Privacy-preserving distributed k-means clustering over arbitrarily paritioned data. In *SIGKDD'05*, pages 593–599, Chicago, IL, 2005.

[8] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *SIGKDD*, pages 277–286, New York, NY, USA, 2006. ACM.

[9] S. Mukherjee, M. Banerjee, Z. Chen, and A. Gangopadhyay. A privacy preserving technique for distance-based classification with worst case privacy guarantees. *Data Knowl. Eng.*, 66(2):264–288, 2008.

[10] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10:2002, 571-588.

[11] J. Vaidya. Towards a holistic approach to privacy-preserving data analysis. In *Workshop on Secure Knowledge Management*, 2008.

[12] J. Vaidya, C. Clifton, and M. Zhu. *Privacy Preserving Data Mining*. Springer-Verlag, 2005.

Algorithm 2 Utility Aware Sanitization Algorithm

Input: Data D (n rows m columns) distributed at parties $P_1, \ldots, P_r$, # of groups g, response variable A_v, weight w, noise level b.

if D is horizontally partitioned **then**
  all parties use secure sum to compute $|D|$
end if
All parties run secure K-Means clustering [7] on D with $K = g$ and with weighted distance function.
Sorts the clusters in ascending order of cluster size. If the data is horizontally partitioned then the parties need to first use a secure sum protocol to compute the number of points in each cluster.
for each cluster center c_i, $i = 1, \ldots, g$ **do**
  Add all points in cluster C_i to group G_i
  if $|C_i| < \lfloor |D|/g \rfloor$ **then**
    Use the protocol to find the closest cluster [7] to find $\lfloor |D|/g \rfloor - |C_i|$ points in $\bigcup C_j, j > i$ that are closest to c_i, and move them to group G_i.
  end if
end for
for each Group G_i **do**
  Use SMC protocol in [11] to compute group statistics
  Infer PCA transform matrix M from statistics
  if D is horizontally partitioned **then**
    Each party P_l locally computes $D_{Pl} = D_l M$ where D_l is its local data.
    All parties use secure comparison protocol to securely compute the range of each column as $b_j = b \cdot (\max\{D_P^j\} - \min\{D_P^j\})$ where D_P^j is the j-th column in the PCA component matrix
    Each party P_l locally adds Laplace noise with zero mean and standard deviation b_j to the j-th column in D_{Pl} to get D'_{Pl}
    Each party P_l computes $D'_{Pl} M^T$ where M^T is transpose of M and adds the results to output H
  else if D is vertically partitioned **then**
    Each party P_l computes $D_l M_i$ where M_i is the columns in M that correspond to columns in P_l.
    Use secure sum protocol [12] to compute $\sum_{l=1}^{r} D_l M_i$ and store the results as two random shares as D_{P1} at P_1 and D_{P2} at P_r
    P_1 and P_r use secure comparison protocol to compute $b_j = b \cdot (\max\{D_P^j\} - \min\{D_P^j\})$
    P_1 (P_r) generates $n \times m$ random noise matrix R_1 (R_2), which follows exponential distribution with $\lambda = 1/b_j$ for the j-th column.
    P_1 and P_r use secure sum to compute $D_{PN} = (D_{P1} + R_1 + D_{P2} - R_2)$
    adds $D_{PN} M^T$ to output H.
  end if
end for
Return H