Privacy Preserving Unsupervised Clustering over Vertically Partitioned Data[⋆]

D.K. Tasoulis[1,2], E.C. Laskari[1,2], G.C. Meletiou[2,3], and M.N. Vrahatis[1,2]

[1] Computational Intelligence Laboratory, Department of Mathematics,
University of Patras, GR-26110 Patras, Greece
{dtas, elena, gmelet, vrahatis}@math.upatras.gr
[2] University of Patras Artificial Intelligence Research Center (UPAIRC),
University of Patras, GR-26110 Patras, Greece
[3] A.T.E.I. of Epirus, P.O. Box 110, GR-47100 Arta, Greece

Abstract. The exponential growth of databases containing personal information has rendered the task of extracting high quality information from collections of such databases very important. This task is hindered by the security concerns that arise, due to the confidentiality of the data records, and the reluctance of the organizations to disclose their data. This paper proposes a clustering algorithmic scheme that ensures privacy and confidentiality of the data without compromising the effectiveness of the clustering algorithm nor imposing high communication costs.

1 Introduction

Clustering, that is "grouping a collection of objects into subsets or clusters, such that those within one cluster are more closely related to one another than objects assigned to different clusters" [8], is a fundamental process in knowledge acquisition. With the availability of inexpensive storage and the progress of data capturing technology, many organizations have created heterogeneous databases of data, and this is expected to continue. Thus, any knowledge discovery methodology, such as clustering, must take into consideration the distributed and heterogeneous nature of the data. Evidently, clustering rules extracted from a collection of databases tend to reflect globally meaningful results, rather than cognition which is embedded in a particular database.

The scenario of having an individual's transactions divided among different organizations is common in real life [19]. This raises justifiable concerns among privacy advocates, that may prevent the necessary sharing of data, and hence discourage clustering projects involving more than one organization. Clustering and privacy are therefore, often perceived to be at odds. Clustering results rarely violate privacy as such, since they generally reveal high–level knowledge, rather than disclosing instances of sensitive data. However, the concern among privacy advocates is well founded, as bringing data together to support clustering and

[⋆] Partially supported by the "Archimedes" research programme awarded by the Greek Ministry of Education and Religious Affairs and the European Union.

data mining makes in general misuse easier [19]. The problem, therefore is not data clustering *per se*, but the manner in which it is performed. Thus, there is a growing need for the development of methods that have endogenous mechanisms to protect the confidentiality of sensitive data.

Clustering can conform with privacy preserving requirements by satisfying two conditions. Primarily, clustering algorithms need to be applicable without data sharing among the data proprietors; and secondly, no private information must be deducible from the extracted results. If these conditions are met clustering will not compromise privacy and it will contribute to obtaining globally meaningful results. One approach recently investigated is the addition of "noise" to the data before the data mining process [3, 5]. Another approach, restricted to classification, considers how much information can be inferred from the data made available through data mining algorithms, and how to minimize information leakage [3, 10]. Also, the extraction of association rules in horizontally partitioned data, was addressed in [9], while [18] addresses the same problem for vertically partitioned data. Concerning clustering, in [19] an adaptation of the k-means algorithm using several primitives from the secure multi-party computation literature, was proposed. Rather than sharing parts of the original or perturbed data, the authors of [11] suggest to transmit the parameters of suitable generative models, built at each local data site, to a central location that actually performs the clustering procedure. Finally, in [13], privacy preserving hierarchical data clustering methods are introduced using a family of geometric data transformation methods.

In this paper we assume a setting similar with that of [19], in which a number of different sites hold data for different attributes of a common set of entities (vertically partitioned data). The scope of each site is to obtain the clustering result over all its entities, but no site wants to reveal any information about its own attribute values. To this end, based on the recently proposed k-windows clustering algorithm [20], we develop a new algorithmic scheme that prevents the sharing of any meaningful information among the sites involved, results the same output as that obtained by the k–windows algorithm been applied to the unified database, and it does not raise any significant communication cost. Note that it is assumed that there does not exist a malicious site that provides wrong pattern lists in order to force the other sites to provide pattern lists, which can be used to gain information about the data. This assumption can be justified to the extent that the organizations that venture such projects have an established collaboration, rather than a one time partnership. Bad faith in this setting will be punished outside the algorithm.

2 Unsupervised k–Windows Clustering Algorithm

Intuitively, the k-windows algorithm tries to place a d–dimensional window that will contain all patterns belonging to a single cluster; for all clusters present in the dataset [20]. This goal is met by iteratively moving and enlarging the windows. During movement each window is centered at the mean of patterns that are included in it. This process is iteratively executed as long as the distance between the new and the previous center exceeds the user–defined variability threshold, θ_v. The enlargement process, takes place at each dimension separately. Each range of a window is enlarged by a proportion θ_e/l, where θ_e is user–defined and l stands for the number of previous successful enlargements. Next, the movement process is invoked. Once movement terminates, the proportional increase in the number of patterns included in the window is calculated. If this proportion does not exceed the user–defined coverage threshold, θ_c, the enlargement and movement steps are rejected and the position and size of the d–range are reverted to their prior to enlargement values. Otherwise, the new size and position are accepted. If enlargement is accepted for dimension $d' \geqslant 2$, then all dimensions d'', such that $d'' < d'$, undergo enlargement assuming as initial position the current position of the window. This process terminates if it does not result in a proportional increase in the number of patterns included in the window beyond the threshold θ_c.

The unsupervised k-windows algorithm is able to approximate the number of clusters, by applying the k-windows algorithm using a large number of initial windows. The windowing technique of the k-windows algorithm allows for a large number of initial windows to be examined, without any significant overhead in time complexity. Once movement and enlargement of all windows terminate, all overlapping windows are considered for merging. The merge operation is guided by a merge threshold, θ_m. Having identified two overlapping windows, the number of patterns that lie in their intersection is computed. Next, the proportion of this number to the total patterns included in each window is calculated. If the mean of these two proportions exceeds θ_m, then the windows are considered to belong to a single cluster and are merged, otherwise not. All these procedures are illustrated in Fig. 1, where (a) depicts the movement procedure, (b) the enlargement procedure and (c),(d),(e) are the three different instances of the merging procedure. For a comprehensive description of the algorithm and investigation of its capability to endogenously identify the number of clusters present in a dataset, refer to [15]. The unsupervised k-windows algorithm applied in both artificial and real life datasets, has proved to be efficient and effective in obtaining the actual number of clusters present in the dataset and achieving high classification results [17]. A high level description of the algorithm follows.

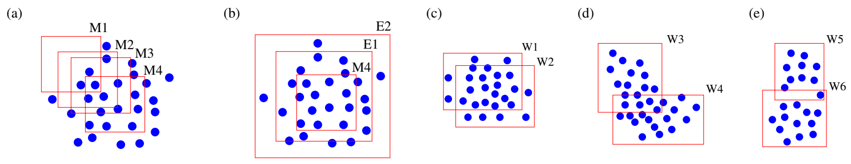

Fig. 1. (a) The movement procedure. (b) The enlargement procedure. (c) Windows that satisfy the similarity criterion of the merging procedure. (d) Windows that satisfy the merging criterion of the merging procedure. (e) Overlapping windows that do not satisfy any of the criteria of the merging procedure.

1. **Set** {the input parameters of k-windows algorithm}.
2. **Initialize** a set W of d–ranges.
3. **Perform** movements and enlargements of the d–ranges in W.
4. **Perform** the merging operation of the d–ranges in W.
5. **Report** the groups of d–ranges that comprise the final clusters.

3 Privacy Preserving Version of the *k*-Windows Algorithm

In this paper we consider the problem of privacy preserving unsupervised k-windows clustering, which can be formally defined as follows. Let r be the number of different sites, each holding a database with different attributes for the same set of n entities. All sites are interested in clustering through the unsupervised k-windows method the union of their databases, resulting in (a) the number of clusters over the union of data, (b) the final position of the centers of the clusters, and (c) cluster assignment for all points, under the following privacy conditions:

1. All databases are *private*, implying that there will be no disclosing of any database to any other site, or to a third party.
2. There is *minimal necessary information sharing* across the private databases, which means that the result of the clustering algorithm will be obtained without revealing any additional information.

To expose the workings of the proposed algorithmic scheme that enables the application of the k-windows algorithm in this setting, we separately describe each step of the methodology in the following subsections. Subsequently, in Section 4, the security of the scheme and privacy at each step are analyzed.

3.1 Determination of the Initial *d*–Ranges

The initialization phase requires the mutual agreement of all sites. Specifically, a set of k points that will comprise the centers of the initial d–ranges, should be mutually agreed upon. Each of these points represents a center around which a d-range will be initialized. Having decided on the identities of the patterns that will comprise the initial centers of the d–ranges, the size of the edges of these ranges must be set. The size of each edge can be decided locally; that is, by the site that holds the values for the corresponding coordinate (attribute). As it will be shown below, this information need not be communicated among sites.

3.2 Movements and Enlargements

After the initialization step has been completed for all k d–ranges, each site knows: (a) the coordinates of the centers of the ranges that correspond to the attribute values that it holds; and (b) the size of the edges of the d–ranges for the same coordinates. From this information alone, each site can conclude the set of points that are enclosed in a particular d–range with respect to the dataset it holds. The complete set of points that are included in the full–dimensional d–range is the intersection of the corresponding sets of all sites. The exact procedure for the computation of the set intersection and its privacy analysis are given in Section 4. This operation for the simple case of two sites, each holding one attribute, is illustrated in Fig. 2. The two dimensional range, Range 1, has as center the point $P1$. Site 1 has decided the size of the edge of Range 1 for attribute 1, while Site 2 has determined the size of the edge for attribute 2. As previously mentioned, this information is private and need not be communicated. Site 1, therefore, concludes that the patterns that are included in Range 1 are $V_1 = \{P1, P2, P3, P4, P5\}$; while Site 2 concludes that for the same Range the enclosed patterns are $V_2 = \{P1, P3, P4, P7\}$. As shown in Fig. 2 the patterns which are included in Range 1 with respect to all dimensions, lie in the intersection of the two sets, $V = V_1 \cap V_2 = \{P1, P3, P4\}$. To obtain the result of the set intersection the parties apply the set intersection protocol for private databases described in Section 4.

Subsequently, the mean of the patterns that lie within each d–range (i.e. the mean value of the d–dimensional points) needs to be calculated. This operation is straightforward as each site can compute the mean for its own coordinates and no information exchange is required. Each site can then update the position of the center of the d–range with respect to the specific coordinates, so as to coincide with the previously computed mean. The process of moving the window is iteratively applied as long as the number of patterns that lie in the d–range is significantly increased as a result of this operation. The stopping criterion for this operation is the user–defined variability threshold, θ_v, that corresponds to the least change in the center of a d–range that renders the re-centering of the d–range acceptable. In this setting, the stopping criterion must be satisfied for all the sites in order to stop the movement process. Once movement is terminated, the d–ranges are enlarged in order to enclose as many patterns as possible from the cluster.

The enlargement process is executed in a similar manner with movement. Since enlargement is considered at each dimension separately, each site can

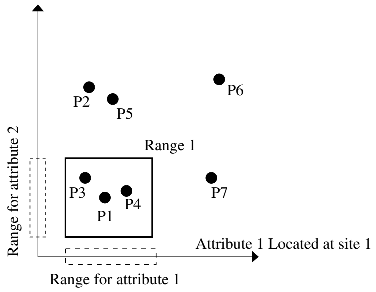

Fig. 2. Determination of points that are included in a d–range, over 2 sites

perform the operation for the coordinate(s) (i.e. attributes) it holds. Once a site has performed enlargement in a particular coordinate the new set intersection is computed for the enlarged d–range. After the enlargement in one dimension is performed, the window is re-centered, through the movement process described above. Once movement terminates, the proportional increase in the number of patterns included in the window is calculated. If this proportion does not exceed the user–defined coverage threshold, θ_c, the enlargement and movement steps are rejected and the position and size of the d–range are reverted to their prior to enlargement values. Otherwise, the new size and position are accepted. If enlargement is accepted for a dimension higher than one, the enlargement process for that d–range is reconsidered for all the lower dimensions (as described in Section 2). This process terminates if enlargement in any dimension does not result in a proportional increase in the number of patterns included in the window beyond the threshold θ_c.

3.3 Merging of the Resulting d–Ranges

To perform the merging operation no information need to be communicated among the sites. Since all the sites know the points that lie inside each window, they can determine the possible overlapping of any two windows. For each pair of overlapping d–ranges, each site can determine the proportion of common points with respect to the total number of patterns included in each window. Comparing this proportion with the threshold, θ_m, it is possible to determine whether the corresponding d–ranges belong to the same cluster.

4 Security and Privacy Analysis

To meet the privacy conditions for the complete algorithmic scheme, i.e., private databases and minimal necessary information sharing across them, it is sufficient to satisfy these conditions during the computation of the set of objects that lie in each d-range query. The computation of this set takes place in two stages. In the first stage each site individually computes the set of objects that lie in a d-range with respect to its attributes. The second stage involves the computation of the intersection of all individually computed sets. The privacy of the first stage is ensured as the computation is private to each site. Thus, only the privacy of the set intersection needs to be investigated.

To perform such a privacy analysis we first need to establish a security model. This is performed through the framework of *secure multi–party computation* [7, 21]. In this contribution we assume the security model to be the *semi-honest* [2, 7]. According to this model, the sites follow the protocol properly with the exception that they can retain a record of all their intermediate computations and received messages, in an attempt to obtain additional information if possible. Under this model, the proposed methodology considers several multi-party set intersections using a secure protocol that involves homomorphic encryptions and hashing [6].

The problem of set intersection of private databases in a multi-party environment is defined as follows. Assume that there are r parties, $S_1, \ldots, S_r$, with corresponding lists of inputs $V_1, \ldots, V_r$ from some domain. At the end all parties learn which specific inputs are shared among all databases, without obtaining any additional information. The correspondence to our case is direct.

Considerable effort has been devoted to the development of protocols that address the problem of finding the intersection of two lists while revealing only the intersection. In [12] two solutions to this problem are presented. The first solution requires the oblivious evaluation of n polynomials of degree n, while the second solution requires the evaluation of n^2 linear polynomials, where n denotes the cardinality of the databases. In [2], the problem of two set intersection, intersection size, equijoin and equijoin size are studied using commutative encryptions and hash functions, and secure protocols with low computation and communication costs are provided. In our contribution, we adapt the multi-party protocol introduced in [6]. This protocol involves homomorphic encryption schemes and oblivious polynomial evaluation, it considers a leader party and $r-1$ client parties and is briefly described in the following steps. Without loss of generality, it is assumed that each list contains l_c inputs. For more details refer to [6].

1. A client party S_i, for $1 \leqslant i \leqslant r-1$, generates a polynomial Q_i of degree l_c whose roots are its inputs, and uses its own public key to homomorphically encrypt the polynomial coefficients. S_i also chooses l_c sets of $r-1$ random numbers, $\{s^i_{j,1}, \ldots, s^i_{j,r-1}\}_{j=1}^{l_c}$, which can be viewed as a matrix with l_c rows and $r-1$ columns. This matrix is chosen such that the XOR of each row sums to zero. For each column l $(1 \leqslant l \leqslant l_c)$, the client party encrypts the corresponding shares using the public key of client S_l. Then, it sends all encrypted elements to a public bulletin board (or just to the leader party who acts in such a capacity).
2. For each data item y in his list, the leader S_r prepares $(r-1)$ random shares $\sigma_{y,l}$, one for each column of the matrix, where $\bigoplus_{l=1}^{r-1} \sigma_{y,l} = y$. Then, for each of the l_c elements of the matrix column representing client S_l, he computes the encryption of $(rn_{y,l} \cdot Q_l(y) + \sigma_{y,l})$ using S_l's public key and a new random number $rn_{y,l}$. Thus, the leader generates l_c tuples of $r-1$ elements each. Then, he permutes randomly the order of the tuples and sends the resulting data.
3. Each client S_l decrypts the r entries which are encrypted with its public key, i.e., the lth column generated by S_r, which has l_c items, and the $(r-1)$ lth columns generated by the clients (also of l_c items). Then, S_l computes the XOR of each row in the resulting matrix, $(\bigoplus_{i=1}^{r-1} s^i_{j,l}) \oplus \sigma_{j,l}$, and sends these l_c results.
4. Each site S_i checks if the XOR of the $(r-1)$ published results for each row is equal to the value y of its input. If this holds, then $\bigoplus_{l=1}^{r-1} \big((\bigoplus_{i=1}^{r-1} s^i_{j,l}) \oplus \sigma_{j,l}\big) = y$, and y is concluded to be in the list intersection.

The prescribed multi-party set intersection protocol is proved to be correct at evaluating the set intersection and secure with respect to all parties' privacy for the semi-honest model case [6].

Regarding the security control of the multiple queries, the *semi-honest* model on which the above security analysis is based allows the sites to keep a record of all their intermediate computations and received messages, to infer some previously unknown, confidential data about a given entity. Such threats may result in exact, or partial information disclosure [1]. A survey of methods that have been proposed to address the problem of security control for the multiple queries was published [1]. Evaluation criteria of such approaches include security, robustness, suitability and cost. For a more recent survey of such techniques see [4].

5 Complexity Issues

The computational complexity of the algorithm depends on the computational complexity of the range searches. To make this step efficient techniques from Computational Geometry can be employed [14]. All these techniques have in common the existence of a preprocessing stage at which they construct a data structure for the patterns. This data structure allows them to answer range queries in sub-linear time with respect to the size of the database. In this case, however, we must also consider the complexity of the multi-party set intersection protocol. The communication overhead of this protocol is $O(rl_c)$, where r represents the number of sites involved and l_c is the maximum size of each object set. The computation overhead comes up to $O(rl_c^2)$ which can be reduced to $O\big(r(l_c + l_c \ln\ln l_c)\big)$, through hash-to-bins method described in [6].

6 Discussion and Concluding Remarks

In this paper we present an algorithmic scheme that enables the application of the k-windows algorithm [20] on vertically partitioned data, with privacy. In this setting, the dataset is distributed over a number of sites and each site has information for all the entities, but only for a specific subset of the attributes of each entity. The goal is to cluster the known set of entities without revealing any of the values on which the clustering is based on. The work by Vaidya and Clifton [19] is directly comparable to the proposed setting. In [19] results from secure multi-party computation are employed in order to develop a privacy preserving k-means clustering algorithm. This approach ensures privacy, but imposes a high communication cost of $O(nrk)$, where r represents the number of sites involved, and n the total number of points. The advantages of the proposed approach reside in the clustering procedure per se, as well as, in the privacy preservation. Regarding clustering the k–windows algorithm has the ability to approximate the number of clusters present in a dataset [15, 16], provides high quality results, and has a low algorithmic complexity. With respect to the privacy issues, all privacy conditions are met through the adapted protocols for the semi-honest model. This work can be extended in order to be applicable to heterogeneous database models, as well as, to the case of horizontally partitioned datasets.

References

1. N. R. Adam and J. C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.
2. R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 86–97, 2003.
3. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, 2000.
4. J. Domingo-Ferrer and J. M. Mateo-Sanz. Current directions in statistical data protection. *Research in Official Statistics*, 1(2):105–112, 1998.
5. A. Evfimievski, R. Srikant, R. Agarwal, and J. Gehrke. Privacy preserving mining of association rules. *Inf. Syst.*, 29(4):343–364, 2004.
6. M. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology – EUROCRYPT 2004*, 2004.
7. O. Goldreich. *Secure multi-party computation.* Working Draft, Ver.1.4, 2002.
8. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer-Verlag, 2001.
9. M. Kantarcioglu and J. Vaidya. An architecture for privacy-preserving mining of client information. In C. Clifton and V. Estivill-Castro, editors, *IEEE International Conference on Data Mining Workshop on Privacy, Security, and Data Mining*, volume 14, pages 37–42. Australian Computer Society, 2002.
10. Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology, CRYPTO 2000*, pages 36–54. Springer-Verlag, Aug. 20-24, 2000.
11. S. Merugu and J. Ghosh. Privacy-preserving distributed clustering using generative models. In *Third IEEE International Conference on Data Mining*, 2003.
12. M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proc. of the 31st Annual ACM Symposium on Theory of Computing*, pages 245–254, 1999.
13. S. R. M. Oliveira and O. R. Zaane. Privacy preserving clustering by data transformation. In *18th Brazilian Symposium on Databases*, pages 304–318, 2003.
14. F. Preparata and M. Shamos. *Computational Geometry.* Springer Verlag, 1985.
15. D. K. Tasoulis and M. N. Vrahatis. Unsupervised distributed clustering. In *Proc. of the IASTED International Conference on Parallel and Distributed Computing and Networks*, pages 347–351. Innsbruck, Austria, 2004.
16. D. K. Tasoulis and M. N. Vrahatis. Unsupervised clustering on dynamic databases. *Pattern Recognition Letters*, 26(13):2116–2127, 2005.
17. D.K. Tasoulis and M.N. Vrahatis. Novel approaches to unsupervised clustering through the k-windows algorithm. In S. Sirmakessis, editor, *Knowledge Mining*, volume 185 of *Series Studies in Fuzziness and Soft Computing.* Springer Verlag, 2005.
18. J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 639–644. Edmonton, 2002.
19. J. Vaidya and C. Clifton. Privacy preserving k-means clustering over vertically partitioned data. In *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.
20. M. N. Vrahatis, B. Boutsinas, P. Alevizos, and G. Pavlides. The new k-windows algorithm for improving the k-means clustering algorithm. *Journal of Complexity*, 18:375–391, 2002.
21. A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 160–164, 1982.