Accuracy in Privacy-Preserving Data Mining Using the Paradigm of Cryptographic Elections

Emmanouil Magkos[1], Manolis Maragoudakis[2],
Vassilis Chrissikopoulos[1], and Stefanos Gridzalis[2]

[1] Department of Informatics, Ionian University
Plateia Tsirigoti 7, Kerkyra, Greece, 49100
{emagos,vchris}@ionio.gr
[2] Department of Information and Communications Systems Engineering
University of the Aegean, Karlovassi, Samos
{mmarag,sgritz}@aegean.gr

Abstract. Data mining technology raises concerns about the handling and use of sensitive information, especially in highly distributed environments where the participants in the system may by mutually mistrustful. In this paper we argue in favor of using some well-known cryptographic primitives, borrowed from the literature on large-scale Internet elections, in order to preserve accuracy in privacy-preserving data mining (PPDM) systems. Our approach is based on the classical homomorphic model for online elections, and more particularly on some extensions of the model for supporting multi-candidate elections. We also describe some weaknesses and present an attack on a recent scheme [1] which was the first to use a variation of the homomorphic model in the PPDM setting. In addition, we show how PPDM can be used as a building block to obtain a Random Forests classification algorithm over a set of homogeneous databases with horizontally partitioned data.

Keywords: Privacy and accuracy; Homomorphic encryption; Distributed systems; Frequency mining.

1 Introduction

Data mining aims at extracting valuable, non obvious information from large quantities of data [2]. This technology has broad applications in areas related to market research, as well as to financial and scientific research. Despite the potentials for offering valuable services, there have been concerns about the handling and use of sensitive information by data mining systems. The problem is even more intense nowadays with the proliferation of the Web and ICT technologies, and the progress in network, storage and processor capacity, where an enormous pool of private digital data can be easily gathered, or inferred from massive collections of public data such as Facebook.com, by using well-known data mining techniques. Even when access to sensitive data is controlled, public data can sometimes be used as a path towards private data [3].

J. Domingo-Ferrer and Y. Saygın (Eds.): PSD 2008, LNCS 5262, pp. 284–297, 2008.

Privacy concerns may also prevent building accurate data mining models. Traditionally, data mining algorithms have been applied in centralized collections of data. With distributed databases, data may be *horizontally* or *vertically* partitioned among a set of mutually mistrustful sites, where each site may hold similar data about different people or different data about the same set of people, respectively. This is also known as the *Server to Server* (S2S) model [4]. In a fully distributed setting, also known as the *Client to Server* (C2S) model [4], customers may be on hold of their own collections of sensitive data. Such data may need to be correlated with other clients' data, for example in order to provide some useful service. The traditional data warehousing approach, where dispersed data are gathered into a central site for building the data mining model, raises privacy concerns as organizations and people are reluctant to reveal their private data for legal, commercial or personal reasons. The simple approach of performing data mining at each site independently and then combine the results (*e.g.*, [5]) cannot always be possible (*e.g.*, in the $C2S$ setting) or accurate enough [6].

The need for privacy in statistical databases is driven by law, compliance, ethics, as well as for practical reasons: it would enable collaboration between data holders (*e.g.*, customers, organizations), if they were assured that their sensitive information would be protected. To this end, *Privacy Preserving Data Mining* (PPDM) has been evolved as a new branch of research in the data mining community [7]. Especially in distributed statistical databases, where there is a need to extract statistical information (*e.g.*, sum, average, entropy, Information Gain, etc) without compromising the privacy of the individuals [8].

A very common approach in the PPDM literature has been *data perturbation*, where original data are perturbed and the data mining model is built on the randomized data. For example, the data perturbation approach has been used for *classification* [9] and building *association rules* [10,11]. Typically, such approach involves a trade-off between two conflicting requirements: the privacy of the individual data and the accuracy of the extracted results [8,12]. In addition, there are cases where the disclosure of some randomized data about a client may reveal a certain property of the client's private information, an attack known as *privacy breach* [10,12]. Alternatively, and orthogonally to our research, the privacy preserving issue can also be regarded as an *access control* problem concerning aggregate data in more or less controlled environments. In this regard, multilevel and multilateral security in database information systems (*e.g.*, [13]), trusted platforms, query restriction policies and inference control [8,14,15] as well as anonymization techniques [16] have also been proposed in the literature.

Traditionally, the use of cryptographic primitives has also been well studied by the database security community [17]. In the academic literature for PPDM, following the line of work that begun with Yao [18], most theoretical results are based on the *Secure Multiparty Computation* (SMC) approach (*e.g.* [19,6,20]). SMC protocols are interactive protocols, run in a distributed network by a set of entities with private inputs, who wish to compute a function of their inputs in a privacy preserving manner. The goal is that no more information is revealed to an entity in the computation than can be inferred from that

participant's input and output [21]. SMC has been used for mining association rules on both horizontally [20] and vertically partitioned databases [6]. Classification models that use the SMC approach involve decision trees [19,22] and a naive Bayes Classifier for horizontally partitioned data [23], as well as decision trees for vertically partitioned data [24]. A disadvantage of this approach is that SMC protocols require multiple communication rounds among the participants, and privacy usually comes at a high performance and communication cost [22]. Most protocols in the SMC family are efficient as long as the number of participants is kept small (*e.g.*, two or three parties).

Our Contribution. In this paper we explore whether it is possible to use efficient cryptography in order to perform privacy preserving data mining, *e.g.*, in statistical databases, while maintaining the accuracy of the results. To this end we argue in favor of borrowing knowledge from a broad literature dealing with cryptographic elections via the Internet. We discuss some weaknesses and describe an attack on a recent PPDM scheme of Yang, Zhong and Wright [1] which, to our best knowledge, was the first work that used a variation of the classical homomorphic model [25] for online elections. Our PPDM approach will be based on the classical homomorphic model of Cramer, Gennaro and Schoenmakers [25] for online elections, and more particularly on some recent extensions proposed in [26,27] for multi-candidate elections. We show how this approach could be used to mine frequencies on a large set of customer databases. As an example, we also propose the use of PPDM as a building block to obtain a Random Forests classifier learning algorithm over a set of homogeneous databases with horizontally partitioned data.

2 PPDM Based on the Homomorphic (Election) Model

We argue that research for privacy preserving data mining could borrow knowledge from the vast body of literature on Internet voting systems [28]. These systems are not strictly related to data mining but they exemplify some of the difficulties of the multiparty case. Such systems also tend to balance well the efficiency and security criteria, in order to be implementable in medium to large scale environments. Furthermore, these systems fall within our distributed computing scenario and have similar architecture and security requirements. In an Internet election for example, an election authority receives several encrypted *yes/no* votes (*e.g.*, yes = 1 and no = 0) and declares the winning candidate. In this setting the goal is to protect the *privacy* of the voters (*i.e.*, unlinkability between the identity and the vote that has been cast), while also establishing eligibility of the voters, *accuracy* and verifiability for the election result.

The most efficient schemes in the literature for cryptographically secure online elections follow the *homomorphic model* [25]. This model is a general framework that allows usage of any randomized encryption scheme with several "nice" algebraic properties, in order to protect the privacy of the encrypted votes and establish accuracy of the decrypted results in a universally verifiable way. With

homomorphic encryption there is an operation $\oplus$ defined on the message space and an operation $\otimes$ defined on the cipher space, such that the "product" of the encryptions of any two private inputs is the encryption of the "sum" of the inputs: $E(M_1) \otimes E(M_2) = E(M_1 \oplus M_2)$. This property allows, for example, either to tally votes as aggregates or to combine shares of votes (*e.g.*, [29,30]), without decrypting single votes.

In [25] each client signs and then submits an encryption of her vote to a *bulletin board* [31], together with a *zero-knowledge* proof [32] that the vote is valid. The homomorphic model does not require interactions between clients, and only one flow of data is sent to the server. Privacy is established in a strong cryptographic sense: original inputs are encrypted using the randomized encryption scheme to preclude *chosen-plaintext* [33] attacks on the published encryptions; in addition, no encrypted input is ever decrypted, but instead it is combined with the other inputs to get the encrypted aggregate. The homomorphic property of the encryption scheme allows every participant to verify that the final results are accurate, by performing a multiplication of the encrypted inputs and comparing the encrypted aggregate to the value published on the bulletin board. Robustness in such protocols is established by using *threshold cryptography* [34], where the power of the election authority is divided among a set of n independent servers, in a way that a set of $t \leq n$ honest servers are able to cooperate and compute the decrypted outcome. As a result, the privacy of clients is assured against any coalition of less than t servers.

Compared with the election setting, the threat model in PPDM seems to be relaxed. Adversaries in distributed systems for data mining are considered as *semi-honest* (also referred to as *honest but curious*) [22]. This means that they are legal participants that follow the protocol specification but try to learn additional information given all the messages that were exchanged during the protocol. This fact favours the adoption of the homomorphic model in PPDM systems. First of all, there is no need for clients to construct complex zero-knowledge proofs on the correctness of their inputs. Furthermore, a strong notion of privacy for cryptographic elections, known as *receipt-freeness* or *uncoercibility* [35] is not an issue here, as the scenario of coercing the clients to reveal (or, sell) their private inputs does not seem realistic in the PPDM setting. In addition, the *universal verifiability* requirement in online elections, where any outsider is able to verify correctness of the final tally, can also be relaxed and replaced with a requirement for *atomic verifiability* (*i.e.*, where every participant in the protocol is able to verify the accuracy of the results). For all these reasons, we may be able to construct and choose among lightweight versions of some well-known cryptographic schemes for online elections that follow the homomorphic model, and adopt them to our PPDM setting.

2.1 Extending the Classical Homomorphic Model

In this section we look at some very efficient extensions of the homomorphic model, where *1-out-of-L* or *k-out-of-L* selections are allowed (*e.g.*, [26,27]). In

this way, the overall bits of information that a database sends to the miner could be increased, leading to new possibilities.

Multi-candidate protocols have been first investigated in [29] and further studied in [25], where the computation of the final tally grows exponentially with the number L of candidates: $\Omega(\sqrt{(}C)^{L-1})$, with C being the number of clients. Baudron et al [26] proposed the use of the *Paillier cryptosystem* [36] for conducting homomorphic elections with multiple candidates. The Paillier scheme provides a trapdoor to efficiently compute the discrete logarithm, thus making computation of the tally very efficient, even for large values of C and L. They also presented a threshold version of the Paillier cryptosystem, to be used in the election setting. We briefly recall the Paillier cryptosystem, leaving out some complex cryptographic details on the key generation and decryption functions [36]. Let $N = pq$ be an RSA modulus where p and q are two large primes, and g be an integer of suitable order modulo N^2. The public key is (N, g) while the secret key is the pair (p, q). To encrypt a message $M \in Z_n$, choose a random $x \in Z_n$ and compute $c = g^M x^N (mod N^2)$. The knowledge of the trapdoor (p, q) allows to efficiently decrypt c and determine M. The reader may refer to [36,26] for further details.

The protocol in [26] is a *1-out-of-L* protocol, where all the choices are in the set $(1, M, M^2, ...M^{L-1})$, with M being an integer larger than the number C of clients. A client, who wishes to select the m^{th} candidate, encrypts her input with the Paillier scheme, and then signs and publishes the result $c = g^{M^m} x^N (mod N^2)$ on a bulletin board. During the tallying stage, the authorities compute the "product" of the encrypted inputs and then cooperate to decrypt the tally using threshold Paillier decryption [26]. The decrypted tally can then be written in M-ary notation: $T = k_0 M^0 + k_1 M^1 + ... + k_{L-1} M^{L-1} (mod N)$, which will directly reveal all k_i's, where $0 \leq k_i \leq C$ is the number of selections for candidate i. The decryption process is publicly verifiable, due to the homomorphic property of the Pallier scheme [26].

In a more recent work, Damgard et al [27] also proposed a generalization of the Paillier cryptosystem and discussed its applicability to homomorphic elections. The size of each ciphertext in [27] is logarithmic in L, while the work for computing the final tally is also reduced, compared with [26]. They also proposed a threshold variant of the generalized system.

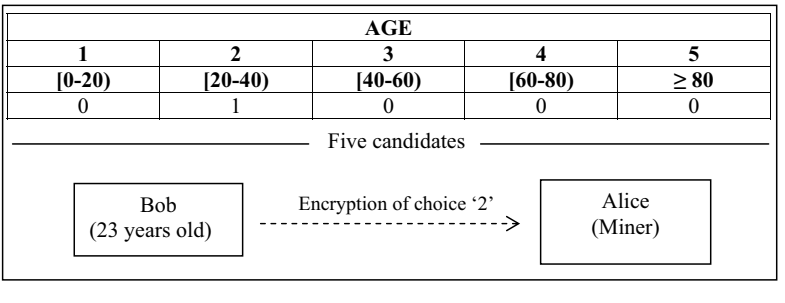

Fig. 1. A multi-candidate setting with *1-out-of-5* choices

PPDM-1 Approach: Mining with *1-out-of-L* Protocols. In the usual (*yes/no*) setting (*e.g.*, yes = 1 and no = 0), a client who does not want to participate may give false information. Or, in a fully distributed setting for example, where each client retains control of her transactions, the client may decide not to participate, although we consider this as a privacy violation. As a result, the null input should also be considered in homomorphic protocols. Furthermore, knowledge cannot always be represented with *yes/no* decisions. For example, a client may have to answer which group (*e.g.*, among L age groups) her age belongs to. These are some reasons why we are interested in multi-candidate schemes, where in the simplest *1-out-of-L* case each client makes one selection out of L candidates. For simplicity, we assume that all questions to a database can be reduced to a set of *yes/no* answers, as shown in Figure 1.

PPDM-2 Approach: Mining with *k-out-of-L* Protocols. *1-out-of-L* protocols can easily be adapted to support up to *k-out-of-L* selections. An easy generalization, with some loss of efficiency, would be to send up to k encrypted messages [27]. Our proposal is to encode all possible L-bit numbers as separate candidates, thus producing a set of 2^L candidates. Figure 2 depicts our trivial approach in the fully distributed setting, where the problem of allowing *k-out-of-L* selections from a database record with L features is reduced to a *1-out-of-*2^L multi-candidate protocol. Protocols with up to *k-out-of-L* selections could also be used in a partially distributed scenario, where the full database is horizontally partitioned into a small set of client partitions, with each client possessing R full records of customers' transactions. In this case, Bob would send R encrypted messages to the miner, where R is equal to the rows of the table in his database.

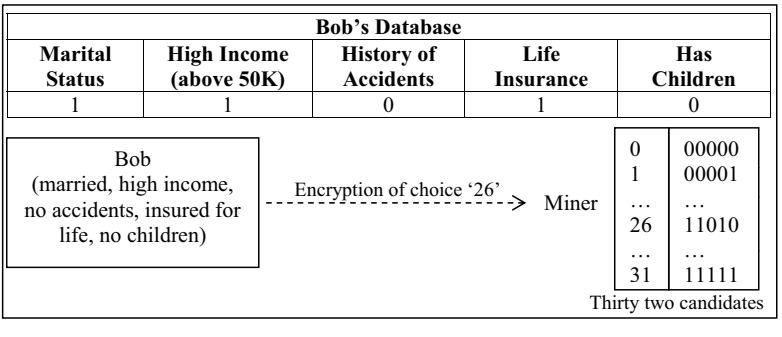

Fig. 2. A trivial way to turn a *1-out-of-L* scheme into a *k-out-of-L* scheme

3 Reviewing the (Yang et al) Scheme

In this section we briefly describe the work in [1], which is, to our best of knowledge, the first scheme that used a variant of the homomorphic election model in order to build a privacy preserving frequency mining algorithm. This algorithm

is then used in [1] as a building block to design a protocol for naive Bayes learning. The authors in [1] also discuss the application of this algorithm to other data mining techniques such as decision trees and association rule mining. A fully distributed setting is considered, where the clients' database is horizontally partitioned, and every client possesses her own data.

We briefly describe the PPDM protocol of [1], where a miner mines a large number of client data sets to compute frequencies of values. Let G be a group where the Discrete Logarithm problem is hard. All operations are done *modp*, where p is a publicly known and sufficiently large prime number. In a system with n clients, each client possesses two pairs of keys: $(x_i, X_i = g^{x_i})$, $(y_i, Y_i = g^{y_i})$, with g being a (publicly known) generator of the group G. Each client U_i knows her private keys (x_i, y_i), with values (X_i, Y_i) being the corresponding public keys. Furthermore, the protocol requires all clients to know the values X and Y, where $X = \prod_{i=1}^{N} X_i$, and $Y = \prod_{i=1}^{N} Y_i$. Each client is able to give a *yes/no* answer d_i to any question posed by the miner and the miner's goal is to learn $\sum_{i=1}^{N} d_i$. In the protocol of [1], depicted in Figure 3, all clients in the system use a variant of the ElGamal encryption scheme [37]. For correctness and privacy analysis, please refer to [1].

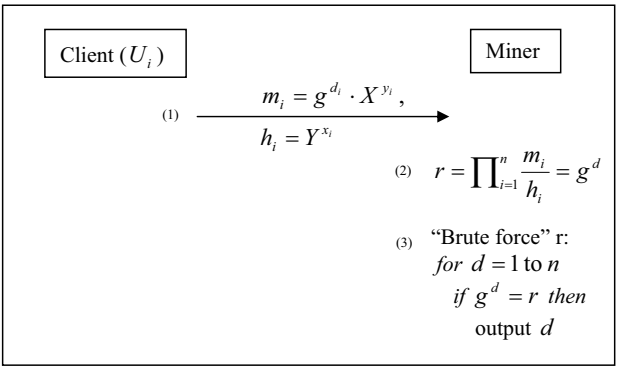

Fig. 3. A schematic representation of the protocol in [1]

Observe that the computation of the tally (*i.e.*, the result d that equals the sum of the plaintext inputs) in the scheme of [1], as well as in the classical homomorphic model of [25], involves a brute-force attack on the value g^d in order to find the discrete logarithm. This stands because there are no trapdoors to determine d from g^d in ElGamal variants [25]. In settings with only two candidates (*e.g.*, *yes/no*) this is a relatively easy computation, at least for a moderately sized number of clients. However the same is not true for multi-candidate selections in large-scale systems. To address this issue, in Section 2.1 we discussed some very efficient protocols for computing the tally in multi-candidate protocols with very large numbers of clients.

3.1 Weaknesses and Attacks

We briefly describe two weaknesses of the protocol in [1]. The first weakness is a minor one and refers to the need that each client must choose new x_i and y_i values after each execution of the protocol. This is actually a requirement in every randomized encryption scheme, where new randomness is used to increase the cryptographic security of the encryption process against chosen plaintext attacks [33]. For example, in Figure 3, if the client U_i uses the same x_i and y_i values during two successive runs, it will be trivial for an attacker (in the semi-honest model) to find out U_i's answers by trial and error.

The above weakness cannot be considered as an attack, since the authors in [1] write a remark about the need for updating the (x_i, y_i) values. However we rather consider this as a scalability issue: Prior to the execution of each run of the protocol (*e.g.*, possibly during a key setup phase) each client must obtain or compute the numbers X and Y which are functions of the public keys (X_i, Y_i) of all system clients. In a fully distributed and large-scale scenario, where a very large number of system clients hold their own records, it may be difficult to pre-compute and/or publicize these values, turning the key setup phase into a complex procedure, especially in cases where the number of participants is not constant through different runs of the system.

A DOS Attack. We also discuss a second weakness of the scheme in [1], which, under preconditions, could lead to a *Denial Of Service* (DOS) attack. We are unaware of any mention of this attack in the literature, and therefore briefly describe it here. We argue that a single client may be able to disrupt the system. Indeed, in a system with say three clients U_1, U_2, U_3, let us assume that U_2 does not send her input, because of a system crash. Then the protocol executes as in Figure 4 and a result cannot be found.

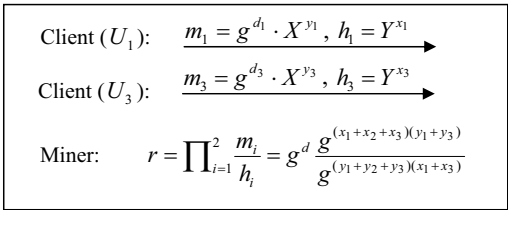

Fig. 4. A run with two active clients in a system with three registered clients

One could argue that in the semi-honest threat model, all clients will adhere to the protocol specification and will not abstain from the protocol, however this is an unrealistic assumption, especially in large-scale protocols (*e.g.*, 10000 was the number of clients used in the experimental results in [1]). Furthermore, the semi-honest model does not preclude non-malicious system crashes or network failures. Observe that a client does not know a priori who will participate in the

protocol, so the obvious fix of constructing the values X and Y as a function of the number of active participants will not work.

4 A Generic Random Forests (RF) Classifier

4.1 Introducing Standalone RF

Nowadays, numerous attempts in presenting ensemble of classifiers towards increasing the performance of the task at hand have been introduced. A plethora of them has portrayed state-of-the-art results in terms of precision and recall measures. Examples of such techniques are Adaboost, Bagging and Random Forests [38].

Random forests [39] are a combination of tree classifiers such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost, and are more robust with respect to noise. While traditional tree algorithms spend a lot of time choosing how to split at a node, Random Forests put very little effort into this. Compared with Adaboost, Random Forests portray the following characteristics:

1. The accuracy is as good as Adaboost and sometimes better.
2. They are relatively robust to outliers and noise.
3. They are faster than bagging or boosting.
4. They provide useful internal estimates of error, strength, correlation and variable importance.
5. They are simple and easily parallelized.

A random forest multi-way classifier $\Theta(\mathrm{x})$ consists of a number of trees, with each tree grown using some form of randomization. The leaf nodes of each tree are labeled by estimates of the posterior distribution over the data classes. Each internal node contains a test that best splits the space of data to be classified. A new, unseen instance is classified by sending it down every tree and aggregating the reached leaf distributions. The process is depicted in Figure 5:

Randomness can be injected at two points during training: in sub-sampling the training data so that each tree is grown using a different subset; and in selecting the node tests. In our approach, we shall discuss the former situation, and argue that using privacy preserving protocols in randomly selected instance vectors supports the creation of robust RF, thus allowing for effective Data Mining in horizontally-partitioned data sets. For vertically partitioned type of partitioned data, the latter approach needs to be taken into consideration. However, for the time-being this is out of the paper's scope.

4.2 Privacy-Preserving RF for Horizontally Partitioned (HP) Data

By the term horizontally partitioned data, we mean that parties (≥ 3) collect values from the same set of features but for different objects. Their goal is to

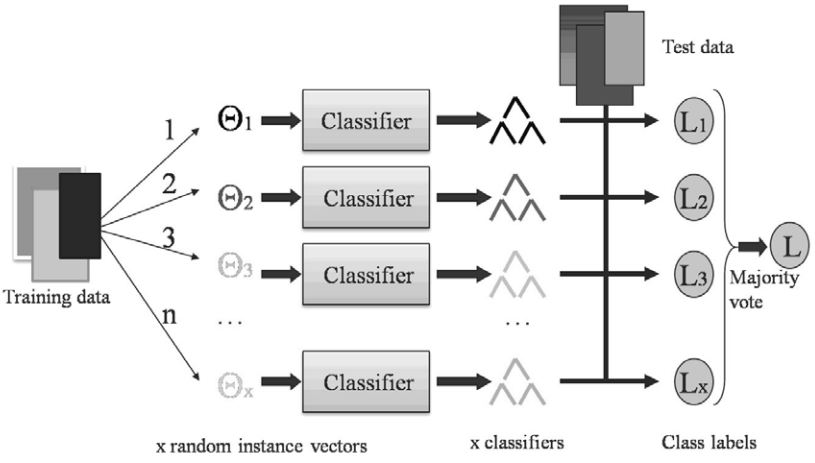

Fig. 5. Hierarchical decomposition of an RF classifier on a non-distributed data set

find an improved function for predicting class values of future instances, yet without sharing their data among each other. Thus, we enroll a collaborative approach where data need to be shared in a secure manner, and the final model will predict class labels without knowing the origin of the test instance. Similar to previous approaches such as [20], classification is performed individually, on each party's site, but the main contribution on the field is that during training, data from other parties are used in order to enhance randomness, thus increase the obtained classification accuracy. However, an assumption needs to be taken into account: data are sharing a common distribution. For example, suppose we have three different bank institutions, sharing financial information on their customers in a HP manner (*e.g.*, they all use features such as *age, occupation, income, marital status* and *sex*). In order to have a robust RF classifier, data has to follow a similar distribution among banks, meaning that if one bank owns data on a specific group of customers (*e.g.*, professors) and the others own data about a totally different group (*e.g.*, farmers), the obtained accuracy would be severely deteriorated. We exploit the two strengths of RF *i.e.*, randomness and voting. The former deals with the issue of premature termination of the tree learning process while the latter confronts data sparseness problems in an effective manner. In this work, we shall provide a protocol that allows for injecting randomness into trees during learning and allow voting over the majority class among all parties at classification time. More specifically, we shall discuss Random Input Forests (RI) learning from HP data sets abd using the forest structure to classify previous unseen test instances originating from one of the distributed database parties. Prior to this analysis, an introduction to *Out-Of-Bag* (OOB) selection of samples is included.

OOB Estimates to Monitor Error, Strength, and Correlation. Out-of-bag samples for tree T_i in a forest are those training examples that are not used to construct tree T_i. As [39] portrayed, they give unbiased estimates of

error on future data, since we do not need to use cross validation. Furthermore, oob samples enhance internal strength and correlation. By strength, we denote the notion of a tree being able to be a fairly good model on its own. Correlation among trees is related with the fact that errors are canceled out between different trees. Therefore, our framework uses the following procedure: Each new training set is drawn, with replacement, from the training set of the other parties. Then a tree is grown on the new training set using random feature selection. The trees grown are not pruned. Exact measures of strength and correlation are described in [39,40] and will not be explained so forth.

4.3 Random Input Forests

Our privacy-preserving protocol for training random forests at each party by inserting randomness from different ones is consisted of two distinct phases. At the former one, each party is collaborating using the procedure proposed by [20], in order to collect the whole set of available values per each attribute. This knowledge is particularly important for the next phase, where each party will require a certain number of instances from the others (again, we note that more than three parties are needed). The complete algorithm is as follows:

Each party selects K trees to grow:

- Build each tree by:
  - Selecting, at random, at each node a small set of features (F) to split on (given M features). From related research, common values of F are:
    1. $F = 1$
    2. $F = \log_2(M) + 1$

    F is held constant while growing the forest. Create a random instance based on the values of the complete feature set and ask the other parties to vote if they own it. (based on the afore-mentioned PPDM approach). Since F is significantly smaller that M, the number of candidate instances that each party will create is computationally efficient to be handled by the PPDM approach.
  - For each node split on the best of this subset (using oob instances)
  - Grow tree to full length. There is no pruning.

To classify an unseen, new instance X, collect votes (again using the PPDM approach) from every tree in the forest of each party and use general majority voting to decide on the final class label.

5 Conclusions

In this paper we discussed privacy issues in distributed data mining and argued in favor of borrowing knowledge from a broad literature dealing with cryptographic elections via the Internet. The goal is to use efficient cryptography in order to perform privacy preserving data mining in statistical databases, while maintaining the accuracy of the results. We proposed a PPDM approach based

on recent homomorphic schemes for multi-candidate elections [26,27] in order to cryptographically protect privacy in large-scale distributed data mining applications, without sacrificing scalability and efficiency. We reviewed a recent scheme [1] that used a variation of the classical homomorphic model [25] for online elections, discussed some weaknesses and described a security attack. Furthermore we proposed the use of the PPDM approach as a building block to obtain a Random Forests classification algorithm over a set of homogeneous databases with horizontally partitioned data. The introduction of a privacy preserving classifier from the domain of ensemble classifiers is a novelty of this work since such approaches have presented the most promising results as regards to precision and recall measures in real-world Data Mining applications.

We believe that research in cryptographic PPDM must be continued and practical solutions that balance the tradeoff between efficiency and security must be sought. More particularly, further research on cryptographic PPDM should take into account the various kinds of databases to work with, as well as the various data mining technologies that need to be supported.

References

1. Yang, Z., Zhong, S., Wright, R.N.: Privacy-preserving classification of customer data without loss of accuracy. In: SDM 2005 SIAM International Conference on Data Mining (2005)
2. Chen, M.S., Han, J., Yu, P.S.: Data mining: An overview from a database perspective. IEEE Transactions on Knowledge and Data Engineering 08, 866–883 (1996)
3. Clifton, C., Marks, D.: Security and privacy implications of data mining. In: 1996 ACM SIGMOD Workshop on Data Mining and Knowledge Discovery, Montreal, Canada, pp. 15–19 (1996)
4. Zhang, N., Wang, S., Zhao, W.: A new scheme on privacy-preserving data classification. In: KDD 2005: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pp. 374–383. ACM, New York (2005)
5. Prodromidis, A., Chan, P., Stolfo, S.J.: Meta-learning in distributed data mining systems: Issues and approaches. Advances in Distributed and Parallel Knowledge Discovery, 81–114 (2000)
6. Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: KDD 2002: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 639–644. ACM, New York (2002)
7. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y., Theodoridis, Y.: State-of-the-art in privacy preserving data mining. SIGMOD Rec. 33, 50–57 (2004)
8. Adam, N.R., Wortmann, J.C.: Security-control methods for statistical databases: A comparative study. ACM Comput. Surv. 21, 515–556 (1989)
9. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proc. of the ACM SIGMOD Conference on Management of Data, pp. 439–450. ACM Press, New York (2000)

10. Evfimievski, A., Srikant, R., Agrawal, R., Gehrke, J.: Privacy preserving mining of association rules. In: KDD 2002: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 217–228. ACM, New York (2002)
11. Rizvi, S.J., Haritsa, J.R.: Maintaining data privacy in association rule mining. In: VLDB 2002: Proceedings of the 28th international conference on Very Large Data Bases, VLDB Endowment, 682–693 (2002)
12. Liu, K., Kargupta, C.G.,, H.: A survey of attack techniques on privacy-preserving data perturbation methods. In: Aggarwal, C., Yu, P. (eds.) Privacy-Preserving Data Mining: Models and Algorithms. Springer, Heidelberg (2008)
13. Morgenstern, M.: Security and inference in multilevel database and knowledge-base systems. SIGMOD Rec. 16, 357–373 (1987)
14. Domingo-Ferrer, J. (ed.): Inference Control in Statistical Databases. LNCS, vol. 2316. Springer, Heidelberg (2002)
15. Woodruff, D., Staddon, J.: Private inference control. In: CCS 2004: Proceedings of the 11th ACM conference on Computer and communications security, pp. 188–197. ACM, New York (2004)
16. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information (abstract). In: PODS 1998: Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, p. 188. ACM, New York (1998)
17. Maurer, U.: The role of cryptography in database security. In: SIGMOD 2004: Proceedings of the 2004 ACM SIGMOD international conference on Management of data, pp. 5–10. ACM, New York (2004)
18. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: FOCS, pp. 162–167 (1986)
19. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000)
20. Kantarcioglu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. IEEE Trans. on Knowl. and Data Eng. 16, 1026–1037 (2004)
21. Goldwasser, S.: Multi party computations: past and present. In: PODC 1997: Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing, pp. 1–6. ACM, New York (1997)
22. Pinkas, B.: Cryptographic techniques for privacy-preserving data mining. SIGKDD Explor. Newsl. 4, 12–19 (2002)
23. Kantarcoglu, M., Vaidya, J.: Privacy preserving naive bayes classifier for horizontally partitioned data. In: IEEE ICDM Workshop on Privacy Preserving Data Mining, Melbourne, FL, pp. 3–9 (2003)
24. Du, W., Zhan, Z.: Building decision tree classifier on private data. In: CRPIT 1914: Proceedings of the IEEE international conference on Privacy, security and data mining, pp. 1–8. Australian Computer Society, Inc., Darlinghurst (2002)
25. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multiauthority election scheme. European Transactions on Telecommunications 8, 481–490 (1997)
26. Baudron, O., Fouque, P.A., Pointcheval, D., Stern, J., Poupard, G.: Practical multicandidate election system. In: PODC 2001: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing, pp. 274–283. ACM, New York (2001)
27. Damgard, I., Jurik, M., Nielsen, J.: A generalization of paillier's public-key system with applications to electronic voting (2003)

28. Gritzalis, D. (ed.): Secure electronic voting: trends and perspectives, capabilities and limitations. Kluwer Academic Publishers, Dordrecht (2003)
29. Cramer, R.J., Franklin, M., Schoenmakers, L.A., Yung, M.: Multi-authority secret-ballot elections with linear work. Technical report, Amsterdam, The Netherlands (1995)
30. Schoenmakers, B.: A simple publicly verifiable secret sharing scheme and its application to electronic voting. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 148–164. Springer, Heidelberg (1999)
31. Benaloh, J.D.C.: Verifiable secret-ballot elections. PhD thesis, New Haven, CT, USA (1987)
32. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. J. ACM 38, 690–728 (1991)
33. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory IT-22, 644–654 (1976)
34. Desmedt, Y.G., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 307–315. Springer, Heidelberg (1990)
35. Hirt, M., Sako, K.: Efficient receipt-free voting based on homomorphic encryption. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 539–556. Springer, Heidelberg (2000)
36. Paillier, P.: Public-key cryptosystems based on discrete logarithms residues. In: Eurocrypt 1999. LNCS, vol. 1592, pp. 221–236. Springer, Heidelberg (1999)
37. Gamal, T.E.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (1985)
38. Breiman, L.: Bagging predictors. Machine Learning Journal 26, 123–140 (1996)
39. Breiman, L.: Random forests. Machine Learning Journal 45, 32–73 (2001)
40. Breiman, L.: Looking inside the black box. In: Wald Lecture II, Department of Statistics, California University (2002)