Hindawi Publishing Corporation
Journal of Computational Engineering
Volume 2013, Article ID 397096, 6 pages
http://dx.doi.org/10.1155/2013/397096

Research Article

Random Response Forest for Privacy-Preserving Classification

Gábor Szűcs

Department of Telecommunications and Media Informatics, BME, Hungary and Inter-University Centre for Telecommunications and Informatics, Kassai str, Debrecen 4028, Hungary

Correspondence should be addressed to Gábor Szűcs; szucs@tmit.bme.hu

Received 23 April 2013; Revised 25 September 2013; Accepted 3 October 2013

Academic Editor: André Nicolet

The paper deals with classification in privacy-preserving data mining. An algorithm, the Random Response Forest, is introduced constructing many binary decision trees, as an extension of Random Forest for privacy-preserving problems. Random Response Forest uses the Random Response idea among the anonymization methods, which instead of generalization keeps the original data, but mixes them. An anonymity metric is defined for undistinguishability of two mixed sets of data. This metric, the binary anonymity, is investigated and taken into consideration for optimal coding of the binary variables. The accuracy of Random Response Forest is presented at the end of the paper.

1. Introduction

In the data mining area, the privacy is emphatic issue to preserve anonymity of persons in the models. The goal of privacy-preserving data mining (PPDM) [1] is to develop data mining models without increasing the risk of misuse of the data used to generate those models. There are two broad approaches in the literature based on the different points of view of the privacy [2]. The randomization approach focuses on individual privacy, and fortunately data mining models do not necessarily require individual records, but only distributions. So this approach preserves the privacy by perturbing the data, and since the perturbing distribution is known, it can be used to reconstruct aggregate distributions, that is, the probability distribution of the data set. In another—so-called Secure Multi-party Computation (SMC)—approach, the aim is to build a data mining model across multiple databases without revealing the individual records in each database to the other databases [3], but this paper does not deal with this approach.

PPDM methods for data modification can include perturbation, blocking, merging, swapping, or sampling. Perturbation is accomplished by the alteration of an attribute value by a new value (i.e., changing a 1-value to a 0-value, or adding noise). Blocking means the replacement of an existing attribute value with a fix sign or character representing the missing value. Merging is the combination of several values into a coarser category [4]. Data swapping refers to interchanging values of individual records [5]. Sampling refers to releasing data for only a sample of a population.

A wide approach in PPDM literature is data perturbation—this paper focuses on only this method—where original data are perturbed and the data mining model is built on the randomized data. The data perturbation should take two opposite requirements into consideration: the privacy of the individual data and the accuracy of the extracted results. The performance of PPDM techniques can be measured in terms of the accuracy of data mining results and the privacy protection of sensitive data [6]. In randomization methods, the goal is to randomize the values in individual records and only disclose the randomized values. The model is then built over the randomized data, after first compensating for the randomization (at the aggregate level). This approach is potentially vulnerable to privacy breaches; based on the distribution of the data, one may be able to learn with high confidence that some of the randomized records satisfy a specified property, even though privacy is preserved on average.

The classification is part of PPDM, where decision trees can be used well, since this is based on aggregate values of a data set, rather than individual data items; they are able to classify unknown data based on the tree model built during

the training phase. There is an algorithm, Random Forest [7], which originally contains random feature in creating many decision trees, and on the other hand its accuracy is better than only a single decision tree. The combination of Random Forest and perturbation algorithms as a new idea is the main contribution of this paper. Randomized Response technique, as perturbation algorithms, has been used for this work, and this has been developed further.

Random Forests [7] are a combination of tree predictors 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 forests converges to a limit as the number of trees in the forest becomes large. This error depends on the strength of the individual trees in the forest and the correlation among them.

In the Random forest procedure for the kth tree, a random vector R_k is generated, independent of the past random vectors $R_1, \ldots, R_{k-1}$, but with the same distribution. Using the training set R_k tree is grown resulting in a classifier $h(\mathbf{x}, R_k)$ where $\mathbf{x}$ is an input vector. After a large number of trees is generated, each tree casts a unit vote for the most popular class at input $\mathbf{x}$, and the vote which possesses the largest occurrence will be the decision of the Random Forest.

At the learning phase, the Random Forest uses bagging to produce a randomly sampled set of training data for each of the trees. After that, Random Forest builds the decision trees as can be seen on its pseudocode below.

(1) Choose a training set by choosing N times with replacement from all N available training cases (i.e., take a bootstrap sample).

(2) For each bootstrap sample: Build an unpruned classification tree for bootstrap sample.

(3) For each node in the classification tree

  (a) randomly select a subset from all available predicting variables,

  (b) choose the best split from among those variables in the training set.

(4) End for (until each tree is fully built, as may be done in constructing a normal tree classifier).

(5) End for

2. Randomized Response Technique

The idea of Randomized Response (RR) was the modification of data in the data set in such a way that nobody could know—with larger probability than a predefined threshold—whether the human's answer of a question in the data set is true or false. Randomized Response technique was first introduced by Warner [8] as a technique to solve an estimation problem, where the percentage of people possessing attribute A should have been estimated in a population. The humans may have decided to reply with correct or with incorrect answers. The RR technique considers only one attribute; however, in data mining the data sets usually consist of multiple attributes.

Multivariate Randomized Response (MRR) technique [9] is able to use Randomized Response technique for multiple-attribute data set. Let E represent any logical expression based on attributes. Let $P^*(E)$ be the proportion of the records in the whole disguised data set (where the persons' privacy is preserved) that satisfy E = true. Let $P(E)$ be the proportion of the records in the whole undisguised data set that satisfy E = true (the undisguised data set contains the true data, but it does not exist). $P^*(E)$ can be observed directly from the disguised data, but $P(E)$, the actual proportion that we are interested in is unknown, and we have to estimate this.

At binary attributes, the values of attributes A_i can be 0 or 1. In the following example, (dealing with only AND operators) we are interested in the probability of $(A_1 = 1)$ AND $(A_2 = 1)$ AND $(A_3 = 0)$, so we want to know $P(110)$, where this notation represents the $P(A_1 = 1$ AND $A_2 = 1$ AND $A_3 = 0)$. It can be imagined that respondents give true (and false) answers to the privacy questions with the probability θ (and $1 - \theta$, resp.). In the model, there is a randomization procedure, which sends the original data into the disguised data set with the probability θ and sends the complementary data (changes all 0 bits to 1 and all 1 bits to 0) with the probability $1 - \theta$. So the complement is a unary operation that performs logical negation on each bit. The probability $P(110)$ is based on only two types of records in the disguised data set: 110 and 001 (the other types of records like 111 etc., cannot be derived by original 110 because they are not complement of 110 and they are not equal to 110). In the disguised data set, we do not know whether a record is true or false.

Let $R(E)$ be the subset of records which satisfy the E expression, and let $\neg R(E)$ be the subset of complementary records which satisfy the E expression. Because the contributions to $P^*(R(E))$ and $P^*(\neg R(E))$ partially come from $P(R(E))$ and $P(\neg R(E))$, correspondingly, it can be derived that

$$\begin{aligned} P^*(R(E)) &= P(R(E)) \cdot \theta + P(\neg R(E)) \cdot (1-\theta), \\ P^*(\neg R(E)) &= P(\neg R(E)) \cdot \theta + P(R(E)) \cdot (1-\theta). \end{aligned} \tag{1}$$

By solving the above equations, we can get:

$$\begin{aligned} P(R(E)) &= \frac{\left(P^*(R(E)) \cdot \theta + P^*(\neg R(E)) \cdot (\theta - 1)\right)}{2 \cdot \theta - 1}, \\ P(\neg R(E)) &= P^*(R(E)) + P^*(\neg R(E)) - P(R(E)). \end{aligned} \tag{2}$$

These equations can be used for constructing decision trees as can be seen in later section.

3. Anonymity Metric

Many definitions can be found for anonymity metric in the literature. ILoss [10] is a metric proposed to capture the information loss of generalizing a specific value to a general value. The discernibility metric [11] addresses notion of loss by charging a penalty to each record for being indistinguishable from other records. The k-anonymization methods [12, 13] solve this indistinguishable problem by suppressing or generalizing attributes until each row is identical

with at least $k - 1$ other rows. Fung et al. [14] proposed another metric based on the principle of information/privacy trade-off. Suppose that the anonymous table is searched by iteratively specializing a general value into child values. Each specialization operation splits each group containing the general value into a number of groups, one for each child value. Each specialization operation gains some information and loses some privacy. This metric prefers the specialization that maximizes the information gained per each loss of privacy. These metrics belong to generalization methods, but these cannot be used for RR.

3.1. Binary Anonymity. In this paper, a new anonymity metric is introduced, the binary anonymity (BA) indicator. In MRR, there are pairs of instance sets, which are mixed during randomization. The aim of randomization is to make them undistinguishable from each other. The binary anonymity, BA_i (for the ith pair), measures this undistinguishability based on information entropy. The BA is the weighted average of these pairs in the whole data set, where the ith weight is the ratio of the occurrences of the ith pair to the number of all instances. These can be seen in (3), where p_1 and p_2 are the relative frequencies of instances of the pair in each set:

$$BA_{i,1} = -\theta \cdot p_1 \cdot \log\left(\theta \cdot p_1\right) - (1-\theta) \cdot p_2 \cdot \log\left((1-\theta) \cdot p_2\right),$$

$$BA_{i,2} = -\theta \cdot p_2 \cdot \log\left(\theta \cdot p_2\right) - (1-\theta) \cdot p_1 \cdot \log\left((1-\theta) \cdot p_1\right),$$

$$BA_i = \frac{BA_{i,1} + BA_{i,2}}{2},$$

$$BA = \text{weighted average } \left(BA_i\right). \tag{3}$$

The binary anonymity is investigated from θ point of view. In Figure 1, the examples for BA functions at two pairs are presented. The solid line corresponds to a pair where the original probabilities were equal to each other (0.5 and 0.5). The dashed line correspond to another pair where the original probabilities were different (0.8 and 0.2), and at the lowest dotted line the difference between the probabilities was the largest (between 0.95 and 0.05, the difference is 0.9). As can be seen, the largest binary anonymity in both cases can be found at $\theta = 0.5$ and the maximal value, which is 1, can be reached by $p_1 = p_2 = 0.5$ probabilities. Furthermore, a short conclusion can be taken based on Figure 1. Larger binary anonymity can be reached by more symmetric probability pairs, that is, the larger difference between probabilities the lower binary anonymity.

In BA formula, the entropy values can be calculated easily with positive probabilities, but the zero probability cannot be substituted into entropy formula. In order for the problem to be solved, limit theorem of probability theory was applied and (4) was derived; therefore $BA_i = 0$. Consider the following:

$$\lim_{p \to 0} p \cdot \log p = 0. \tag{4}$$

So BA measures the anonymity for the whole randomized data set; this is in the range of $(0, 1)$; the larger values represent larger certainty for preserving privacy.

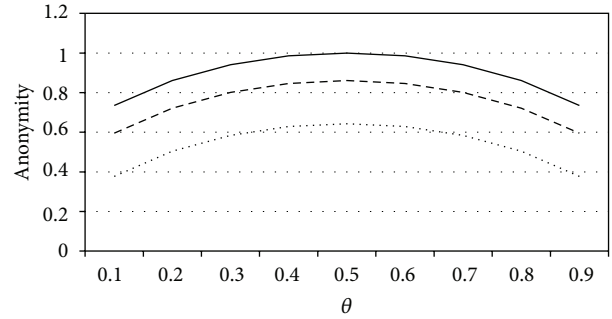

Figure 1: Binary anonymity against the θ.

3.2. Transformation of Different Attribute Types Based on Binary Anonymity. The binary data set can be well used for Response Technique and thus privacy preserved decision trees can be built, but these trees contain only binary attributes, so other types of attributes should be transformed into binary. A splitting point can be defined for each continuous attribute as the median point of the range of the attribute [9]. The values below this splitting point can be transformed to zero and the values above can be transformed to 1. At ordinal attributes (where the items can be ordered) the transformation can be also executed by using the median, as splitting point.

The transformation of nominal attributes is a problem, because there is no median point. It would be possible to transform the mode (the most frequent data) to 1 and the others to zero, but this solution merges quite different items, and the information loss may be large. At better solution, each nominal value is coded by more binary attributes. So the number of attributes is increased by new attributes (dummy attributes) in order to avoid large information loss.

The first idea for the number of dummy attributes is $k - 1$ similar to regression methodology by the following coding: each symbol (except one) of attributes will get a dummy binary attribute, where 1 represents that actual value is equal to this symbol and 0 represents that it is not equal to the symbol. The last symbol will not get dummy attribute, because all zeros represent that the actual value is equal to this last symbol. This solution would be suitable for normal data mining, but for PPDM this is not adequate from a binary anonymity point of view. Since the complements of records would contain many 1 values, and there is no regular record with more than two 1 values, therefore after randomization an undistinguishable pair of a regular and a complementary record will not occur, and this causes empty set. An empty set means that its probability is zero, and based on (4) we can conclude that in this case the binary anonymity will be zero. At zero anonymity, the records are totally revealed; thus, a new, better solution is required.

The elaborated method for transforming nominal attributes to binary is based on binary coding. $[\text{Log}_2 k]$ gives the minimal upper rounded integer number that is needed for representing k different symbols. So the number of dummy binary attributes will be $[\text{Log}_2 k]$, and the different symbols will be transformed to different binary figures automatically. If the $\text{Log}_2 k$ is fortunately an integer number

(so k is the power of 2), then each instance has a pair of instance sets. But if this is not an integer number, then probability of some codes will be zero (because of lack of pair). If the k is even, then coding can be created in such a way that complementary code of each k value should be also a code of other k value. In this case, every original instance will have a pair. If the k is odd, then coding can be created in such a way that complementary code of each k value except one should be also a code of other k value. In this case, only one code will be alone; the others will possess a pair.

What will be the optimal coding from binary anonymity point of view? Let us take the k values in order based on the relative frequency (probability) of corresponding instances. Denote their relative frequencies by $p_{u_1}, p_{u_2}, \ldots, p_{u_k}$. Previously, it has been shown that larger difference between the probabilities induces smaller binary anonymity. The optimal coding gives complementary codes for the neighborhoods of ordered k values. Formally (5)–(8) present the optimal coding for k values in binary numeral system, where b depends on the number of dummy variables, the k values are in descending order, and i goes from 1 to k. Consider the following:

$$b = 2^{[\log_2 k]}, \tag{5}$$

$$p_{u_1} \geq p_{u_2} \geq \cdots \geq p_{u_i} \geq \cdots \geq p_{u_k}. \tag{6}$$

If k is even, then the optimal coding can be seen in

$$u_i = \begin{cases} \text{binary}\left(\dfrac{i-1}{2}\right) & \text{if } i \text{ is odd,} \\ \text{binary}\left(b - \dfrac{i}{2}\right) & \text{if } i \text{ is even.} \end{cases} \tag{7}$$

If k is odd, then the first u_1 or the last u_k will remain to be alone, so for the optimal coding two alternatives should be compared. The first alternative (where the last one will remain alone) is described in (7), and the second alternative is formulized in (8). The alternative that gives larger BA is better, so the optimal coding can be solved by comparing these two alternatives and selecting the larger one. Consider the following:

$$u_i' = \begin{cases} \text{binary}\left(\dfrac{i}{2}\right) & \text{if } i \text{ is odd,} \\ \text{binary}\left(b - \dfrac{i+1}{2}\right) & \text{if } i \text{ is even.} \end{cases} \tag{8}$$

The whole optimal coding procedure is executed automatically. In this case, there is a positive probability that complementary records may be coincided with regular records, and there is no possibility to reveal the truth, so the privacy will be preserved.

Similar to nominal attributes, a better solution for continuous attributes can be found as well. The splitting at the median is not sufficient for information preserving, because lots of information is lost by merging data with large differences. Instead of using only median (Q_2), other quartiles (Q_1 and Q_3) can be also used, and the coding of the continuous attribute is similar to the elaborated method for nominal attributes. Thus, the problem of all types of attributes has been solved perfectly, and different types of attributes can be used in common data set by transforming them into binary (sometimes dummy) attributes.

4. Random Response Forest

4.1. Preparation Phase. Random Forests are able to classify in standard data mining problems, but for PPDM, a randomization technique is needed as well. Random Response method described in Section 3 can be used in decision tree building, so based on this a new method, Random Response Forest, is introduced. It concerns three procedures: preparation procedure, training the decision trees, and classification.

The preparation procedure is as follows:

(i) Continuous attributes are converted into binary attributes using the quartiles of distribution.

(ii) Nominal attributes are converted into dummy binary attributes automatically using the coding rules outlined in the previous section.

(iii) Some records (the ratio is the predefined θ) are replaced to complementary records. In undisguised class label case, the values of class labels will remain unchanged during the complementation. In disguised class label case the values of class labels will change (from 0 to 1 and vice versa) during the complementation.

(iv) The output of the preparation procedure is the disguised data set.

4.2. Training Phase of Random Response Forest. Training phase contains the following steps.

(a) Draw K bootstrap samples from the training data.

(b) For each of the bootstrap samples, grow an unpruned decision tree with the following steps: (i) at each node, rather than choosing the best split among all predictors, choose randomly sample m of the predictors, (ii) calculate and estimate the probabilities for splitting examination at only m selected variables, (iii) calculate Gini index as can be seen at (9), the least Gini index will be the best for splitting, and (iv) choose the best split from among those m variables, where only the second step is the modification of the original one.

Consider the following:

$$\text{Gini}(S) = 1 - \sum_{j=1}^{C_n} p_j^2. \tag{9}$$

In (9), p_j is the relative frequency (proportion) of class j in S, and C_n is the number of different classes. Pseudocode of Random Response Forest shows the details of the algorithm as follows.

$N_{\min}$ is defined as minimum number of instances in a node,

N represents a node in the tree,

D represents training data,

for $i = 1$ to K (i.e., number of trees).

{ Randomly sample the training data D with replacement to produce D_i: this is a bootstrap sample

Create a root node, N_i containing D_i

Call BuildTree(N_i)

}

BuildTree(N):

if (N contains instances with only same class)

OR (number of instances is less than $N_{\min}$) then return

else

{ Randomly select m of the possible splitting features in N

for $i = 1$ to m do

{ Estimate $P(R(E))$ and $P(\neg R(E))$ probabilities for SF_i using (2), where SF_i is the ith selected splitting feature.

Calculate Gini index for SF_i using estimated probabilities,

}

Select the feature F with the least Gini index to split on

Create 2 child nodes of N: N_1 and N_2 according to possible F values $F_0 = 0$ and $F_1 = 1$.

for $i = 1$ to 2 do

{ Set the contents of N_i to D_i, where D_i is all instances in N that match F_i

Call BuildTree(N_i)

}

return

}.

4.3. Classification by Random Response Forest. The Random Response Forest classifies x by taking the most popular voted class (i.e., majority vote) from all the tree predictors in the forest $h(x; R_i)$, where i goes from 1 to the number of the trees.

An estimate of the error rate can be obtained, based on the training data, by the following. At each bootstrap iteration, predict the data that is not in the bootstrap sample, called "out-of-bag" data, using the tree which is grown with the bootstrap sample. At second phase the out-of-bag predictions should be aggregated. On the average, each data point would be out-of-bag around 36.8% of the times, so aggregate these predictions. Calculate the error rate, and call it the "out-of-bag" estimate of error rate. The out-of-bag estimate of error rate (briefly OOB error rate) is quite accurate, given that enough trees have been grown, otherwise the out-of-bag estimate can bias upward [15].

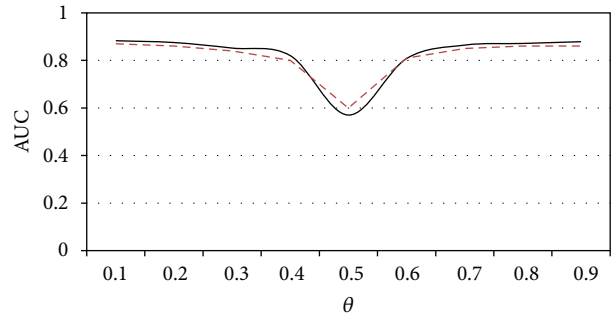

Figure 2: Accuracies of Random Response Forest against the θ.

4.4. Investigation with Examples. Random Response Forest has been implemented in Matlab. In order to investigate the elaborated methods, two data sets have been used from the UCI Machine Learning Repository [16], (the original owner of data sets was US Census Bureau). The first data set ("Adult Data Set") contains 48842 instances (records) with 14 attributes (6 continuous and 8 nominal) and a label describing the salary level. Prediction task is a binary classification, where the goal is to determine whether a person's income exceeds \$50k/year based on data. The second data set, (Census-Income (KDD)) contains 299285 instances (records) with 40 attributes including the salary as well. The instance weight in the second data set indicates the number of people in the population that each record represents due to stratified sampling, but this has not been used in the classification.

All attributes have been transformed into binary attributes. For example, the possible values of attribute "relationship" are: "Wife", "Own-child", "Husband", "Not-in-family", "Other-relative", and "Unmarried". Three dummy attributes have been used for coding.

Choosing the appropriate θ is a task in the preparation phase. For testing, a ROC curve is defined in ROC space by FPR and TPR and the area under the ROC curve, so called AUC has been calculated as the most frequent used accuracy indicator. The AUC value of decision tree without any randomization and privacy preserving was 0.89 at the "Adult Data Set" and 0.88 at the data set of Census-Income.

The randomization has been executed several times for different θ values. After each randomization, Random Response Forest has built model (decision trees), and the models has been tested. The AUC values, as the results of testing, can be seen in Figure 2 plotted against the θ values. The solid line corresponds to "Adult Data Set", and the dashed line corresponds to the other data set. The curves (and curves of binary anonymity in Figure 1) present that the anonymity and the accuracy are strongly dependant on each other; increasing one of them can be reached by decreasing the other; the modeler should find the trade-off between them.

The classical Random Forest has been also used for learning the salary variable in these two original data sets in order to compare the classical method with Random Response Forest. In the original data sets the data contain the personal information without any anonymization technique, and the classical method can be executed on only these data

sets, because Random Forest cannot handle disguised data. Thus, the comparison has been achieved at only one situation, where the θ is zero (at undisguised data). The experiments have shown that the AUC values of Random Response Forest and classical one have been the same at "Adult Data Set" and at "Census-Income" data sets as well. Thus, Random Response Forest can be considered as the extension of Random Forest.

5. Conclusion

The paper has focused on classification, particularly decision, tree algorithm considering the privacy problems as well. The major difference between a privacy-preserving algorithm and the original decision tree algorithm is how $P(E)$ is computed. In the original algorithm the data are not disguised, and $P(E)$ can be computed by simply counting how many records in the database satisfy E. In privacy-preserving algorithm, such counting (on the disguised data) only gives $P^*(E)$, which can be considered as the "disguised" $P(E)$, because this counts the records in the disguised database, not in the actual (but inaccessible) database. Randomized Response technique has been used for randomization, which allows us to estimate $P(E)$ from $P^*(E)$, but this has been appropriate for only binary attributes. A contribution of this paper is a technique, which is able to use any type of attributes by optimal transforming of the continuous and nominal attributes into binary ones. For the randomization a new metric, so called binary anonymity, is defined; the attribute transforming (coding) is optimal from a binary anonymity point of view. Random Response Forest is the largest contribution consisting of decision trees and randomization techniques. Random Response Forest is an extension of the classical Random Forest to handle privacy-preserving data mining problems. Examples have demonstrated present experience of theoretical results on real data. It has shown that the binary anonymity and the accuracy of classification are strongly dependant on each other, increasing one of them can be reached by decreasing the other; the modeler should find the trade-off between them. Around half value of θ, the classification would be worse besides the excellent anonymity. If the value of θ is near to zero or near to 1, then the classification would be accurate, but there would not preserve the anonymity at all.

Acknowledgments

The paper was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union and cofinanced by the European Social Fund.

References

[1] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '00)*, vol. 29, pp. 439–450, ACM, New York, NY, USA, May 2000.

[2] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu, "Privacy-preserving data publishing: a survey of recent developments," *ACM Computing Surveys*, vol. 42, no. 4, article 14, pp. 1–53, 2010.

[3] X. Qi and M. Zong, "An overview of privacy preserving data mining," *Procedia Environmental Sciences*, vol. 12, pp. 1341–1347, 2012.

[4] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in privacy preserving data mining," *SIGMOD Record*, vol. 33, no. 1, pp. 50–57, 2004.

[5] C. C. Aggarwal and P. S. Yu, *Privacy-Preserving Data Mining: Models and Algorithms*, vol. 34 of *The Kluwer International Series on Advances in Database Systems*, Springer, New York, NY, USA, 2008.

[6] N. Zhang, W. Zhao, and J. Chen, "Performance measurements for privacy preserving data mining," in *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD '05)*, T. B. Ho, D. Cheung, and H. Liu, Eds., vol. 3518 of *Lecture Notes in Computer Science*, pp. 43–49, Hanoi, Vietnam, May 2005.

[7] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[8] S. L. Warner, "Randomized response: a survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–66, 1965.

[9] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*, L. Getoor, T. Senator, P. Domingos, and C. Faloutsos, Eds., pp. 505–510, ACM, New York, NY, USA, August 2003.

[10] X. Xiao and Y. Tao, "Personalized privacy preservation," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '06)*, pp. 229–240, ACM, New York, NY, USA, June 2006.

[11] A. Skowron and C. Rauszer, *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Set Theory*, vol. 11, Springer, 1992.

[12] R. J. Bayardo and R. Agrawal, "Data privacy through optimal k-anonymization," in *Proceedings of the 21st IEEE International Conference on Data Engineering (ICDE '05)*, pp. 217–228, Tokoyo, Japan, April 2005.

[13] K. LeFevre, D. J. Dewitt, and R. Ramakrishnan, "Mondrian multidimensional k-anonymity," in *Proceedings of the 22nd IEEE International Conference on Data Engineering (ICDE '06)*, p. 25, Atlanta, Ga, USA, April 2006.

[14] B. C. M. Fung, K. Wang, and P. S. Yu, "Anonymizing classification data for privacy preservation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 5, pp. 711–725, 2007.

[15] T. Bylander, "Estimating generalization error on two-class datasets using out-of-bag estimates," *Machine Learning*, vol. 48, no. 1–3, pp. 287–297, 2002.

[16] Census Income Data Set, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/datasets.