Privacy-Preserving Evaluation of Generalization Error and Its Application to Model and Attribute Selection

Jun Sakuma[1] and Rebecca N. Wright[2]

[1] University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8577 Japan,
jun@cs.tsukuba.ac.jp
[2] Rutgers University, 96 Frelinghuysen Road Piscataway, NJ 08854 USA,
rebecca.wright@rutgers.edu

Abstract. Privacy-preserving classification is the task of learning or training a classifier on the union of privately distributed datasets without sharing the datasets. The emphasis of existing studies in privacy-preserving classification has primarily been put on the design of privacy-preserving versions of particular data mining algorithms, However, in classification problems, preprocessing and postprocessing— such as model selection or attribute selection—play a prominent role in achieving higher classification accuracy. In this paper, we show generalization error of classifiers in privacy-preserving classification can be securely evaluated without sharing prediction results. Our main technical contribution is a new generalized Hamming distance protocol that is universally applicable to preprocessing and postprocessing of various privacy-preserving classification problems, such as model selection in support vector machine and attribute selection in naive Bayes classification.

1 Introduction

Rapid growth of online services has increased the opportunities to store private or confidential information. Data mining enables to exploit valuable knowledge from data collections while dataminers are forced to treat such private datasets under prudent control in order to prevent the leakage or misuse. Considering such situations, privacy-preserving data mining (PPDM) provides a secure way to compute the output of a particular algorithm applied to the union of privately distributed datasets without sharing them. Privacy-preserving classification is defined as a problem to train a classifier with the union of privately distributed datasets without sharing them. Privacy-preserving classification was pioneered by Lindell and Pinkas, who considered ID3 decision trees [6]. Following this, various privacy-preserving classification methods have been presented, including naive Bayes classifiers [11], *k*-nearest neighbor [15], and support vector machines [14, 5].

These privacy-preserving classifiers have been designed as privacy-preserving versions of particular data mining algorithms. However, the dataminer's task rarely starts and ends with running a particular data mining algorithm. Rather, various preprocessing and postprocessing steps improve the capability of knowledge discovery with data mining. For example, real datasets often include noisy or useless attributes; including them in the classifer learning process reduces classification accuracy. Classification accuracy

Appears in *Proceedings of the 1st Asian Conference on Machine Learning*, Nanjing, China, November 2–4, 2009.

can be improved by eliminating these unnecessary attributes (i.e., attribute selection). Furthermore, when the target classifier has tunable parameters, adjustment of model parameters (i.e., model selection) can help minimize the generalization error.

The emphasis in privacy-preserving data mining has mostly not been placed on pre- and postprocessing. Earlier work of Yang et al. [12] begins to address privacy-preserving model selection. As we discuss below, our work extends their initial study in many ways.

In a distributed setting, there are many ways that data can be distributed. Two commonly considered cases are vertically partitioned data and horizontally partitioned data. In this paper, we consider data partitioned between two parties. In the vertically partitioned model, each party holds a subset of attributes of the data entries. In the horizontally partitioned model, each party holds a subset of data entries of all attributes [10]. Usually, these subsets are assumed to be disjoint; in the vertically partitioned case, both parties are assumed to know the identity of the complete attribute set.

Most distributed privacy-preserving classification algorithms produce as output an actual classifier known to one or more of the involved parties. In our work, we also consider "extending" the privacy to later stages of the process, sometimes keeping the classifier itself privacy (which we refer to as a *privately shared classifier*) or even the prediction results themselves (which we call *privately shared prediction*). If the classifier and the prediction are not privately shared, we refer to these as regular classifier and regular prediction, respectively. (See Section 2.2 for more detail.) Many combinations of partitioning models and representations are possible in privacy-preserving classification. The most privacy is obtained if both the classifer and the prediction are privately shared; this also enables a modular design of post-processing such as cross validation that requires using multiple generated models for prediction before deciding on the final output model. Table 1 summarizes representations of classifiers and predictions in existing privacy-preserving classification.

Yang et al. [12] presented a privacy-preserving k-fold cross validation that evaluates generalization error of privacy-preserving classifiers. A limitation of their protocol is that it is applicable only to privacy-preserving classifiers that return regular predictions in the vertically partitioned model.

We note that in principle, any private distributed computation can be carried out using secure function evaluation (SFE) [13, 4]. It follows that SFE technically allows us to evaluate generalization errors of privacy-preserving classifiers with any partitioning models and representations. However, as we demonstrate in Section 3.3 with experimental results, these computations can be inefficient for practical use, particularly when the input size is large.

Our contribution. We present a new Hamming distance protocol that allows us to evaluate generalization errors of privacy-preserving classifiers using k-fold cross validation (Section 3.2). Our protocol works with privacy-preserving classification with any representation and partitioning model. It is therefore is more general and more private than Yang et al.'s protocol [12]. Experimental results show that the computation load of our protocol is much smaller than that of Hamming distance computation implemented with SFE (Section 3.3). To demonstrate our protocol with pre- and postprocessing of privacy-preserving classification, our protocol is applied to model selection in support

Table 1. Data partitioning models and representations of classifiers and predictions in privacy-preserving classification and privacy-preserving cross valiadtion

	Horizontally partitioned model	Vertically partitioned model
Regular classifier Regular prediction	Naive Bayes [11] ID3 [2]	ID3 [6],SVM [14], cross validation [12]
Privately shared classifier Regular prediction	k-NN [15]	cross validation [12] (in restricted cases)
Regular classifier Privately shared prediction		
Privately shared classifier Privately shared prediction	SVM [14]	Naive Bayes [11] SVM [5]

vector machine with polynomial kernels (Section 4) and attribute selection in naive Bayes classification (Section 5). In both applications, the efficiency is examined with experimental analysis.

2 Preliminaries

2.1 Privacy-preserving Classification

Let a set of training examples be $X_{\mathrm{tr}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{1, ..., m\}$. In classification tasks, classifier $h : \mathcal{X} \mapsto \mathcal{Y}$ is trained on X_{tr}. Let $\{(\boldsymbol{x}, y)\}$ be test examples, which are unseen in the training phase. The classification problem is to find a classifier $h \in \mathcal{H}$ with a small generalization error $\epsilon = E_{(\boldsymbol{x},y)}([h(\boldsymbol{x}) \neq y])$ where $E_{(\boldsymbol{x},y)}$ is the expectation over $(\boldsymbol{x}, y)$. $[z]$ is 1 if predicate z holds and 0 otherwise.

In *privacy-preserving classification*, we assume a setting such that training and test examples are distributed over two or more parties and these examples are desired to be kept private mutually. We assume two parties, denoted as Alice and Bob. Let partitioned sets of training examples of Alice and Bob be X_{tr}^{A} and X_{tr}^{B}, respectively.

Typically two types of data partitioning models are considered. In *vertically partitioned model*, a partitioned dataset corresponds to a subset of attributes of all data entries. In *horizontally partitioned model*, a partitioned dataset corresponds to a subset of data entries of all attributes. Test examples are partitioned similarly. If not specifically mentioned, both partitioned models are considered together in this paper.

2.2 Representations of Classifier and Prediction

Privacy-preserving classification essentially includes two problems, privacy-preserving training and privacy-preserving prediction. In the training phase, classifier h is desired to be trained without violating the given partitioning model. Knowledge that Alice and Bob can obtain from the execution of the training phase is determined by the representation of the classifier. Given distributed training examples, if at least one party obtains a trained classifier after the training phase, we say the party learns a *regular classifier*. As long as the training protocol is designed correctly and securely, private inputs

will not be exposed to parties explicitly. However, note that the regular classifier itself might leak some private information to the party who receives the classifier. For example, naive Bayes classifier consists of frequency of attributes, which leaks statistics of attributes held by other parties.

The *privately shared classifier* provides enhanced privacy-preservation with the idea of secret sharing by *random shares*. Let $\mathbb{Z}_N = \{0, 1, ..., N-1\}$. We say Alice and Bob have random shares of secret $y \in \mathbb{Z}_N$ when Alice has $r^A \in \mathbb{Z}_N$ and Bob has $r^B \in \mathbb{Z}_N$ in which r^A and r^B are uniform randomly distributed in $\mathbb{Z}_N$ with satisfying $y = (r^A + r^B) \bmod N$. Assume that classifiers are characterized by a parameter vector $\boldsymbol{\alpha} \in \mathbb{Z}^q$. If two parties obtain random shares of $\boldsymbol{\alpha}$ after the training phase, we say two parties obtain a privately shared classifier. Note that no information associated with the classifier can be obtained from any single share of the classifier. For example, a hyperplane classifier is written as $h(\boldsymbol{x}; \boldsymbol{\alpha}, \beta) = \sum_{i=1}^{d} \alpha_i x_i + \beta$. Let $(\boldsymbol{\alpha}^A, \boldsymbol{\alpha}_i^B)$ and (β^A, β^B) be random shares of $\boldsymbol{\alpha}$ and β, respectively. Then, $(\boldsymbol{\alpha}^A, \beta^A)$ and $(\boldsymbol{\alpha}^B, \beta^B)$ forms a privately shared classifier.

Representations of predictions are stated similarly. Once a party obtains a regular classifier, the party can locally compute predictions of its own test examples. This is referred to as the *regular prediction*. When a privately shared classifier is trained, an additional protocol to obtain predictions is required. If the prediction protocol is designed such that it returns the prediction to at least one party, it is again called regular prediction. If the prediction protocol returns random shares of the prediction, it is referred to as *privately shared prediction*. See Table 1 for the summary of existing privacy-preserving classification with respect to partitioning models and representations.

2.3 Problem Statement

Let $\boldsymbol{y} = (y_1, ..., y_n)$ and $\hat{\boldsymbol{y}} = (\hat{y}_1, ..., \hat{y}_n)$ be a vector of labels of test examples and predicted labels, respectively. Assume binary classification for now, that is, $\boldsymbol{y}, \hat{\boldsymbol{y}} \in \{0, 1\}^n$. Then, the generalization error of binary classification is evaluated as Hamming distance $H(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i=1}^{n} (y_i \oplus \hat{y}_i)$. As shown in Table 1, there are many possible combinations of representations and partitioning models in privacy-preserving classifications, while the protocol for the generalization error evaluation in privacy-preserving classification is desired to be universally applicable to privacy-preserving classifiers with different representations and partitioning models.

First, assume that a prediction protocol which returns regular predictions is given to Alice to Bob. Then, in the vertically partitioning model, predictions can be represented as $\hat{\boldsymbol{y}}^A = (\hat{y}_1, ..., \hat{y}_n)$ and $\hat{\boldsymbol{y}}^B = (0, ..., 0)$. In the horizontally partitioning model, predictions can be represented as $\hat{\boldsymbol{y}}^A = (y_1, ..., y_c, 0, ..., 0)$ and $\hat{\boldsymbol{y}}^B = (0, ..., 0, \hat{y}_{c+1}, ..., \hat{y}_n)$ for some c without loss of generality. Thus, $\hat{\boldsymbol{y}} = \hat{\boldsymbol{y}}^A + \hat{\boldsymbol{y}}^B$ holds for both partitioning models.

Next, assume that a prediction protocol which returns privately shared predictions is given to Alice and Bob. Then, $\hat{y}_i = \hat{y}_i^A + \hat{y}_i^B \mod N$ holds for all i in both horizontally and vertically partitioning models. Thus, regardless of representations of predictions and partitioning models, private evaluation of generalization errors is reducible to the problem stated as follows:

Statement 1 *Let* $(\boldsymbol{y}^A, \boldsymbol{y}^B)$ *and* $(\hat{\boldsymbol{y}}^A, \hat{\boldsymbol{y}}^B)$ *be vectors of random shares of binary vector* $\boldsymbol{y}$ *and* $\hat{\boldsymbol{y}}$*, respectively. Assume that Alice and Bob takes* $(\boldsymbol{y}^A, \hat{\boldsymbol{y}}^A)$ *and* $(\boldsymbol{y}^B, \hat{\boldsymbol{y}}^B)$ *as inputs,*

respectively. After the execution of the Hamming distance protocol, Alice and Bob obtain random shares s^A and s^B such that $s^A + s^B = H(\boldsymbol{y}, \hat{\boldsymbol{y}}) \bmod N$. Furthermore, neither Alice nor Bob obtains anything else.

In Section 3.2, we describe our Hamming distance protocol, which is secure and correct in the sense of Statement 1. This protocol provides a way to evaluate classification errors of given privacy-preserving classifiers. Generalization errors of privacy-preserving classifiers can be evaluated by making use of k-fold cross validation with this Hamming distance protocol.

Next, a problem of model/attribute selection in privacy-preserving classification is stated. Let $\mathcal{H}$ be a set of candidate classifiers. Let $\ell : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \mapsto [0, 1]$ be an estimator of the generalization error. Then, the model selection problem in privacy-preserving classification is stated as follows:

Statement 2 *Let $H = \{h(\cdot; \boldsymbol{\alpha}_i^A, \boldsymbol{\alpha}_i^B) | i = 1, ..., \ell\}$ be a set of candidate privately shared classifiers. Alice takes X_{tr}^A and $(\boldsymbol{\alpha}_1^A, ..., \boldsymbol{\alpha}_\ell^A)$ as private inputs. Bob takes X_{tr}^B and $(\boldsymbol{\alpha}_1^B, ..., \boldsymbol{\alpha}_\ell^B)$ as private inputs. Let X_{ts}^A and X_{ts}^B be test examples of Alice and Bob, respectively. After the execution of model selection, Alice and Bob obtain privately shared classifier h^* such that $h^* = arg \min_{h \in H} \ell(X_{ts}^A \cup X_{ts}^B, h)$. Furthermore, neither of Alice nor Bob obtains anything else.*

Note that any regular classifiers are special cases of privately shared classifiers. This problem statement therefore includes any possible combinations of different representations listed in Table 1. Furthermore, if we regard $H = \{h(\cdot; \boldsymbol{\alpha}_i^A, \boldsymbol{\alpha}_i^B) | i = 1, ..., \ell\}$ as a set of privately shared classifiers which are trained with candidates of attribute subsets, the statement described above is readily rewritten as the statement of the attribute selection problem in privacy-preserving classification.

We present a protocol to solve problems defined as Statement 2 in Section 3. We then demonstrate the universal applicability of our protocol by taking model selection in privacy-preserving support vector machines (Section 4) and attribute selection in privacy-preserving Naive Bayes classifiers (Section 5) as examples.

3 Privacy-preserving Evaluation of Generalization Error

This section describes our Hamming distance protocol, which enables evaluation of classification errors in privacy-preserving classification. Then a model selection protocol for privacy-preserving classification is presented by using the Hamming distance protocol, in which the generalization errors are estimated with k-fold cross validation. Before going into the protocol description, we introduce necessary cryptographic tools.

3.1 Cryptographic Tools

Homomorphic Public-key Cryptosystem. Given a corresponding pair of $(\mathsf{sk}, \mathsf{pk})$ of private and public keys and a message m, then $c = e_{\mathsf{pk}}(m; \ell)$ denotes a (random) encryption of m, and $m = d_{\mathsf{sk}}(c)$ denotes decryption. The encrypted value c uniformly distributes over $\mathbb{Z}_N$ if ℓ is taken from $\mathbb{Z}_N$ randomly. An *additive homomorphic cryptosystem* allows addition computations on encrypted values without knowledge of the

secret key. Specifically, there is some operation $\cdot$ (not requiring knowledge of sk) such that for any plaintexts m_1 and m_2, $e_{\mathsf{pk}}(m_1 + m_2; \ell) = e_{\mathsf{pk}}(m_1; \ell_1) \cdot e_{\mathsf{pk}}(m_2; \ell_2)$, where ℓ is uniformly random provided that at least one of ℓ_1 and ℓ_2 is. Based on this property, it also follows that given a constant k and the encryption $e_{\mathsf{pk}}(m_1; \ell)$, we can compute multiplications by k via repeated application of $\cdot$, denoted as $e_{\mathsf{pk}}(km; k\ell) = e_{\mathsf{pk}}(m; \ell)^k$. In what follows, we omit the random number ℓ from our encrypt ions for simplicity.

Secure Function Evaluation. In principle, private distributed computations such as these can be carried out using Secure function evaluation (SFE) [13, 4] is a general and well studied methodology for evaluating any function privately. Technically, the entire computation of model selection itself can be implemented with SFE; however, these computations can be too inefficient for practical use, particular when the input size is large. Classification often takes large dataset as input. Therefore, we make use of existing SFE solutions for small portions of our computation for a more efficient overall solution. Specifically, we use SFE for a problem of private computation of argmax. Let Alice and Bob take $(s_1^A, ..., s_k^A)$ and $(s_1^B, ..., s_k^B)$ as private inputs. We use SFE for Alice and Bob to learn $i^* = \arg\min_i(s_i^A + s_i^B)$ without sharing their private inputs.

3.2 Hamming Distance Protocol

As discussed in Section 2.3, we can suppose Alice and Bob hold class label vectors $(\boldsymbol{y}^A, \boldsymbol{y}^B)$ and predictions of labels $(\hat{\boldsymbol{y}}^A, \hat{\boldsymbol{y}}^B)$ in the form of random shares regardless of partitioning models representations of classifiers/predictions. The classification error is then evaluated in the form of Hamming distance as

$$\begin{aligned} H(\boldsymbol{y}, \hat{\boldsymbol{y}}) &= \sum_{i=1}^{n} \left((y_i^A + y_i^B) \oplus (\hat{y}_i^A + \hat{y}_i^B) \right) = \sum_{i=1}^{n} \left((y_i^A + y_i^B) - (\hat{y}_i^A + \hat{y}_i^B) \right)^2 \\ &= (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A)^T \cdot (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) + (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B)^T \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B) + 2(\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B), \quad (1) \end{aligned}$$

where $\oplus$ is logical exclusive-or and other operations are all arithmetic. The first and the second term of RHS of eq. 1 can be locally and independently evaluated by Alice and Bob, respectively. Therefore, via the private evaluation of $2(\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B)$, random shares of $H(\boldsymbol{y}, \hat{\boldsymbol{y}})$ is privately evaluated . Based on this idea, the protocol to solve the problem of Statement 1 is shown in Fig. 1. In the protocol, operation $\in_r$ means choosing an element of the set uniform randomly. The correctness of the protocol is explained as follows. In Step 2, what Bob computes is rearranged as

$$c \leftarrow e_{\mathsf{pk}} \left(-r^B + 2 \sum_{i=1}^{n} (y_i^A - \hat{y}_i^A)(y_i^B - \hat{y}_i^B) \right) = e_{\mathsf{pk}} \left(-r^B + 2(\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B) \right).$$

Then, In Step 3, Alice obtains

$$s^A = d^A(c) + (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A)^T \cdot (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) = -r^B + 2(\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A) \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B) + (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A)^T \cdot (\boldsymbol{y}^A - \hat{\boldsymbol{y}}^A)$$

and Bob obtains $s^B = r^B + (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B)^T \cdot (\boldsymbol{y}^B - \hat{\boldsymbol{y}}^B)$. Thus, s^A and s^B are random shares of $H(\boldsymbol{y}, \hat{\boldsymbol{y}})$, which are the desired outputs. The security of this protocol is shown.

Lemma 1. *(Security of Hamming distance protocol) If Alice and Bob behave semi-honestly, Hamming distance protocol is secure in the sense of Statement 1.*

- Alice's input: $\boldsymbol{y}^A, \hat{\boldsymbol{y}}^A \in \mathbb{Z}_N^n$, key pair $(\mathsf{pk}, \mathsf{sk})$
- Bob's input: $\boldsymbol{y}^B, \hat{\boldsymbol{y}}^B \in \mathbb{Z}_N^n$, public key pk
- Alice's output: $s^A \in \mathbb{Z}_N$
- Bob's output: $s^B \in \mathbb{Z}_N$, where $s^A + s^B = H(\boldsymbol{y}^A + \boldsymbol{y}^B, \hat{\boldsymbol{y}}^A + \hat{\boldsymbol{y}}^B) \bmod N$

1. For $i = 1, ..., n$, Alice compute $e_{\mathsf{pk}}(y_i^A - \hat{y}_i^A)$ and send them to Bob
2. Bob computes $c \leftarrow e_{\mathsf{pk}}(-r^B) \cdot \prod_{i=1}^n e_{\mathsf{pk}}(y_i^A - \hat{y}_i^A)^{2(y_i^B - \hat{y}_i^B)}$, where $r^B \in_r \mathbb{Z}_N$. Then, Bob sends c to Alice
3. Alice outputs $s^A \leftarrow d^A(c) + \sum_{i=1}^n (y_i^A - \hat{y}_i^A)^2$ and Bob outputs $s^B \leftarrow r^B + \sum_{i=1}^n (y_i^B - \hat{y}_i^B)^2$

Fig. 1. Hamming distance protocol

Due to space limitations, we give only the intuition behind the security of this protocol. The message Alice receives during the protocol execution is c. Alice can decrypt c but this is randomized by Bob; nothing can be learned by Alice. The messages Bob receives are $e_{\mathsf{pk}}(y_i^A - \hat{y}_i^A)(i = 1, ..., n)$. Bob cannot decrypt this and nothing can be learned by Bob, either. Consequently, both learn nothing.

3.3 Preliminary Experiments: Hamming Distance Protocol

The scalability of Hamming distance protocol with randomly generated binary vectors were examined. Programs were implemented in Java 1.6.0. As the cryptosystem, [8] with 1024-bit keys was used. Experiments were carried out under Linux with Core2 2.0GHz (CPU), 2GB (RAM). The results were compared with the computation time of SFE in that the Hamming distance computation with exactly the same input and output is performed. Fairplay [7] was used for the implementation of SFE.

Fig. 2 shows the results of experiments. The results are the average of ten times execution with different binary vectors. A single execution of Hamming distance protocol with n-dimensional vectors includes n times modulo multiplication and n times modulo exponentiation. As expected, the change in the computation time with respect to the dimensionality of input vectors is linear. While the computation time of SFE is polynomially bounded, the results show that the computation time of SFE is more inefficient than that of our Hamming distance protocol.

3.4 Privacy-preserving Model Selection by means of k-fold Cross Validation

In k-fold cross validation, a set X of examples is randomly split into k mutually disjoint subsets $X_1, X_2, ..., X_k$. Classifier h_j is then trained on $X \setminus X_j$. Let ϵ_j be generalization error of h_j. The cross validation estimate of the generalization error is given as $\epsilon = \sum_{j=1}^k \frac{\epsilon_j}{k}$. For model selection, classifiers are trained with candidates of model parameters. For each trained classifier, the generalization error is evaluated with k-fold cross validation. The model that achieves the lowest generalization error is chosen as the output of the model selection.

Suppose that ℓ candidates of model parameters are prepared in the form of privately shared classifiers. Then, the protocol for privacy-preserving model selection by means of k-fold cross validation is described as shown in Fig. 3.

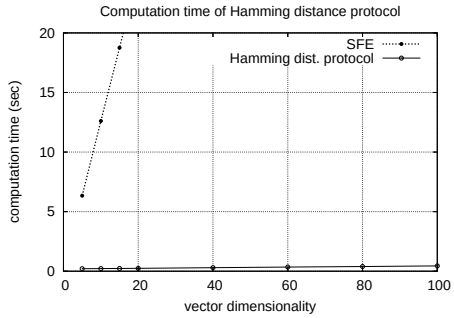

Fig. 2. Computation time of Hamming distance protocol: the dimensionality vs. time (sec)

In the protocol, $(X^A_{\mathrm{ts},j}, X^B_{\mathrm{ts},j})$ is the jth fold of training examples. $(\boldsymbol{\alpha}^A_{ij}, \boldsymbol{\alpha}^B_{ij})$ is the privately shared classifier trained with the ith candidate of model parameters and the jth fold of test examples. $(\hat{\boldsymbol{y}}^A_{ij}, \hat{\boldsymbol{y}}^B_{ij})$ is privately shared predictions of jth fold of test examples classified by the ith candidate model. Labels of test examples are evaluated as privately shared prediction in Step 1. Then, the number of misclassifications is evaluated as random shares of Hamming distance by Hamming distance protocol (Step 2). Finally, the model with the lowest generalization error is privately chosen by secure function evaluation (Step 3). The security of this protocol is as follows:

Theorem 1. *Assume Alice and Bob behave semi-honestly. Given a privacy-preserving protocol for prediction, privacy-preserving model selection is secure in the sense of Statement 2.*

A sketch of the proof is shown. In the protocol of Fig. 3, privacy-preserving subprotocols are composed in a way that randomized outputs of a subprotocol are taken as inputs of the next subprotocol. A composition theorem [6] guarantees that if subprotocols invoked at intermediate steps are privacy-preserving, then the resulting composition is privacy-preserving, too. Based on this fact, the proof of this theorem is readily derived via Lemma 1 and the security of SFE [13, 4].

Assuming $(\boldsymbol{\alpha}^A_{ij}, \boldsymbol{\alpha}^B_{ij})$ are privately shared classifiers trained from datasets with different attribute sets, the protocol in Fig. 3 is readily available for attribute selection in privacy-preserving classification. Exactly the same security proof is valid for this privacy-preserving attribute selection.

4 Model Selection in Privacy-Preserving Support Vector Machine

4.1 Privacy-preserving Support Vector Machines

This section demonstrates the application of our model selection protocol to privacy-preserving SVMs with polynomial kernels.

Let $(\boldsymbol{x}_i, y_i)_{1\le i\le n}$ be a set of training examples where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, -1\}$. Maximal margin classifiers of SVMs are trained in a high dimensional feature space. We

- Alice's input: test example X^A_{ts}, privately shared classifier $\boldsymbol{\alpha}^A_{ij}(i = 1, ..., \ell, j = 1, ..., k)$
- Bob's input: test example X^B_{ts}, privately shared classifier $\boldsymbol{\alpha}^B_{ij}(i = 1, ..., \ell, j = 1, ..., k)$
- Alice's output: privately shared classifier with the lowest generalization error $\boldsymbol{\alpha}^A_{i^* j}$
- Bob's output: privately shared classifier with the lowest generalization error $\boldsymbol{\alpha}^B_{i^* j}$

1. (Evaluation) For $i = 1, ..., \ell$, $j = 1, ..., k$, Alice and Bob jointly execute privacy-preserving evaluation of shared prediction by taking $(X^A_{\mathsf{ts},j}, \boldsymbol{\alpha}^A_{ij})$ and $(X^B_{\mathsf{ts},j}, \boldsymbol{\alpha}^B_{ij})$ as inputs, respectively. After the execution of the protocol, Alice and Bob obtain a vector of prediction shares $\hat{\boldsymbol{y}}^A_{ij}$ and $\hat{\boldsymbol{y}}^B_{ij}$, respectively.
2. (Prediction) For $i = 1, ..., \ell$, $j = 1, ..., k$, Alice and Bob jointly run Hamming distance protocol by taking $(\boldsymbol{y}^A_{\mathsf{ts},i}, \hat{\boldsymbol{y}}^A_{ij})$ and $(\boldsymbol{y}^B_{\mathsf{ts},i}, \hat{\boldsymbol{y}}^B_{ij})$, respectively. After the execution of the protocol, Alice and Bob obtain random shares s^A_{ij} and s^B_{ij} where $H(\boldsymbol{y}^A_{\mathsf{ts},i} + \boldsymbol{y}^B_{\mathsf{ts},i}, \hat{\boldsymbol{y}}^A_{ij} + \hat{\boldsymbol{y}}^B_{ij}) = (s^A_{ij} + s^B_{ij}) \bmod N$
3. (Model selection) Alice and Bob jointly run secure function evaluation for

$$i^* \leftarrow \arg\min_i \sum_{j=1}^{k} (s^A_{ij} + s^B_{ij} \mod N).$$

Alice and Bob respectively obtain shares of classifiers $(\alpha^A_{i^* j})$ and $(\alpha^B_{i^* j})$ which achieve the lowest generalization error

Fig. 3. Privacy-preserving model selection

substitute $\phi(\boldsymbol{x}_i)$ for each $\boldsymbol{x}_i$ where $\phi : \mathcal{X} \mapsto \mathcal{F}$ and $\mathcal{F}$ is a high-dimensional feature space. When features $\phi(\boldsymbol{x})$ only occurs in the form of dot products, Mercer kernels k for dot products, $\phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}) = k(\boldsymbol{x}_i, \boldsymbol{x})$, can be substituted.

SVMs take advantage of this trick to alleviate computation in the high-dimensional feature space. The linear maximal margin classifier is given as

$$h(\boldsymbol{x}) = \mathrm{sgn}\left(\sum_{i=1}^{n} \alpha_i \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x})\right) = \mathrm{sgn}\left(\sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x})\right), \tag{2}$$

which is obtained by solving a convex quadratic programming with respect to α [9].

Laur et al. have proposed a cryptographically private SVM of polynomial kernels for vertically partitioned private datasets [5]. In order to alleviate solving convex quadratic programming with satisfying privacy enforcement, their protocol adopts kernel adatron as the base algorithm.

4.2 Model Selection in Privacy-preserving Support Vector Machines

SVM with polynomial kernels includes polynomial dimension p and margin parameter c as model parameters. In this section, we demonstrate our model selection protocol by taking Laur et. al.'s privacy-preserving SVM as the base classifier algorithm.

As we addressed in Theorem 1, our protocol is designed assuming that the prediction protocol is privacy preserving—that is, nothing but prediction results are leaked

from the execution of the prediction protocol. Laur et. al.'s prediction protocol is designed assuming that $\boldsymbol{x} \cdot \boldsymbol{x}_i \in \mathbb{Z}_N^*$ holds[3] while this does not always necessarily hold. It follows that the security of the prediction protocol might be compromised if $\boldsymbol{x} \cdot \boldsymbol{x}_i \notin \mathbb{Z}_N^*$. In order for our model selection protocol to be secure in the sense of Statement 2, we introduce power-of-sum protocol as a new building block for prediction, which is privacy-preserving for any $\boldsymbol{x}, \boldsymbol{x}_i \in \mathbb{Z}_M$ if $(2M)^p < N$, and then present a new prediction protocol for polynomial kernels using power-of-sum.

Power-of-Sum Protocol In the prediction function of SVM, degree-p polynomials need to be privately evaluated. As a building block for prediction, we introduce a protocol, power-of-sum. Let $(2M)^p < N$. Let Alice and Bob have $x \in \mathbb{Z}_M$ and $y \in \mathbb{Z}_M$ as private inputs, respectively. Power-of-sum protocol allows two parties to compute random shares of $r^A + r^B = (x + y)^p \bmod N$ without sharing their private inputs. We assume that degree p is publicly known throughout the paper. Note that the knowledge of p does not violate any private information possessed by parties.

The protocol is shown in Fig. 4. By binomial theorem, $(x + y)^p = \sum_{i=0}^{p} \binom{p}{i} x^i y^{p-i}$ holds. So, the computation of Step 2 corresponds to

$$c^B \leftarrow e_{\mathsf{pk}}(-r^B) \cdot \prod_{i=0}^{p} (c^i)^{y^{p-i}} = e_{\mathsf{pk}}\left(-r^B + \sum_{i=0}^{p} \binom{p}{i} x^i y^{p-i}\right) = e_{\mathsf{pk}}((x+y)^p - r^B). \quad (3)$$

Thus, Alice obtains $s^A = -s^B + (x+y)^p$ and Bob obtains s^B, which are random shares of $(x+y)^p$. Note that both x and y are positive integers s.t. $(x+y)^p \leq (2M)^p < N$. Without this, numbers may be wrapped around in the computation of step 2, which does not induce desired outputs.

We show an intuitive explanation of the security. Messages received by Alice are randomized by Bob. Messages received by Bob are encrypted and the private key is possessed only by Alice. Thus, both learn nothing but the final result.

Private Prediction for Degree-p Polynomial Kernels

Our new prediction protocol is shown in Fig. 5. In the protocol, random shares

$$s^A + s^B = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i (\boldsymbol{x}_i \cdot \boldsymbol{x})^p \mod N \quad (4)$$

are privately evaluated in Step 3 and then the random shares of the prediction are obtained in Step 4 as $h^A(\boldsymbol{x}) + h^B(\boldsymbol{x}) = \mathrm{sgn}(s^A + s^B) \mod N$. Eq. 4 is derived as follows. Regardless of partitioning models, $\boldsymbol{x}_i = \boldsymbol{x}_i^A + \boldsymbol{x}_i^B$ and $\boldsymbol{x} = \boldsymbol{x}^A + \boldsymbol{x}^B$ hold. So, the scalar product of these is expanded as

$$\boldsymbol{x} \cdot \boldsymbol{x}_i = (\boldsymbol{x}^A + \boldsymbol{x}^B) \cdot (\boldsymbol{x}_i^A + \boldsymbol{x}_i^B) = \boldsymbol{x}^A \cdot \boldsymbol{x}_i^A + \boldsymbol{x}^A \cdot \boldsymbol{x}_i^B + \boldsymbol{x}^B \cdot \boldsymbol{x}_i^A + \boldsymbol{x}^B \cdot \boldsymbol{x}_i^B \mod N. \quad (5)$$

In Step 2-a, random shares of scalar products $\boldsymbol{x}^B \cdot \boldsymbol{x}_i^A$ and $\boldsymbol{x}^A \cdot \boldsymbol{x}_i^B$ are privately computed by scalar product protocol [3]. Note that the other two scalar products are locally computed. Thus, random shares of $\boldsymbol{x} \cdot \boldsymbol{x}_i$ are obtained by Alice and Bob in the form of $\boldsymbol{x} \cdot \boldsymbol{x}_i = (s^{A1} + s^{A2} + \boldsymbol{x}_i^A \cdot \boldsymbol{x}^A) + (s^{B1} + s^{B2} + \boldsymbol{x}_i^B \cdot \boldsymbol{x}^B) \mod N$. By taking these random

[3] integers less than N and coprime to N

shares as inputs of power-of-sum in Step 2-b, random shares of $k(\boldsymbol{x}, \boldsymbol{x}_i)$ are obtained (eq. 7). Since eq. 8 of Step 3-b can be rewritten as

$$
\begin{aligned}
c &\leftarrow \prod_{i=1}^{n} \left(e_{\mathsf{pk}}(\alpha_i^A k^A(\boldsymbol{x}_i, \boldsymbol{x})) \cdot e_{\mathsf{pk}}(\alpha_i^B k^B(\boldsymbol{x}_i, \boldsymbol{x})) \cdot e_{\mathsf{pk}}(\alpha_i^A)^{k^B(\boldsymbol{x}_i, \boldsymbol{x})} \cdot e_{\mathsf{pk}}(k^A(\boldsymbol{x}_i, \boldsymbol{x}))^{\alpha_i^B} \right) \cdot e_{\mathsf{pk}}(-s^B) \\
&= \prod_{i=1}^{n} e_{\mathsf{pk}}\left(\alpha_i^A k^A(\boldsymbol{x}_i, \boldsymbol{x}) + \alpha_i^B k^B(\boldsymbol{x}_i, \boldsymbol{x}) + \alpha_i^A k^B(\boldsymbol{x}_i, \boldsymbol{x}) + \alpha_i^B k^A(\boldsymbol{x}_i, \boldsymbol{x}) - s^B \right) \\
&= \prod_{i=1}^{n} e_{\mathsf{pk}}\left(\alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) - s^B \right) = e_{\mathsf{pk}}\left(\sum_{i=1}^{n} \left(\alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) \right) - s^B \right),
\end{aligned}
$$

Alice obtains $s^A = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) - s^B$ by decrypting c in Step 3-c. Finally, privately shared prediction with polynomial kernels is obtained by means of secure function evaluation in Step 4. Note that Step 2-a can be skipped in the vertically partitioned model because $\boldsymbol{x}^A \cdot \boldsymbol{x}_i^B = \boldsymbol{x}^B \cdot \boldsymbol{x}_i^A = 0$ holds.

Model Selection Using private prediction for polynomial kernels (Fig. 5) in the first step of the model selection protocol in Fig. 3, privacy-preserving model selection is readily achieved.

4.3 Experiments

Our privacy-preserving model selection produces the same final output as model selection without privacy preservation. Therefore, objectives of experiments are (1) to demonstrate the usability of our protocol in privacy-preserving classification with relatively large-size datasets and (2) to investigate the computational cost.

Setting. Two datasets (ionosphere and breast-cancer) were taken from the UCI machine learning repository [1].

We tuned two model parameters: polynomial degree $p = 1, 2, 3, 4$ and margin parameter $c = 2^{-8}, 2^{-6}, ..., 2^{6}$. In total, generalization errors of classifiers trained in 32 settings were evaluated as candidates using 10-fold cross validation. Two kinds of private information models are considered. In "type-I", training examples are not private but test examples are private. In this case, scalar product (Fig. 5, step 2-a) does not have to be performed privately. In "type-II", both training and test examples are private; Step 2-a must be done privately. Given privately shared classifiers, we measured the computational time spent for following five steps included in the model selection phase (the first three items are included in Step 1 of Fig. 3).

1. kernel sharing: scalar product and power-of-sum protocol (Step 2 of Fig. 5)
2. evaluation of $h(\boldsymbol{x})$: private prediction by polynomial kernels (Step 3 of Fig. 5)
3. prediction: secure function evaluation of comparison (Step 4 of Fig. 5)
4. evaluation of gen. err.: hamming distance protocol (HDP) or SFE (Step 2 Fig. 3)
5. model selection: secure function evaluation of argmin (Step 3 of Fig. 3)

Results. Fig 6 (left and right) illustrates generalization errors of SVM evaluated by 10-fold cross validation. As shown, the generalization error is significantly improved when model parameters are appropriately chosen. Note that these results are not revealed to the participants by the protocol, but only the parameters which achieves the lowest generalization error is revealed in privacy-preserving settings.

- Alice's input: $x \in \mathbb{Z}_M$, a valid key pair $(\mathsf{pk}, \mathsf{sk})$
- Bob's input: $y \in \mathbb{Z}_M$, Alice's public key pk
- Alice's output: random share s^A
- Bobs output: random share s^B where $s^A + s^B = (x + y)^p \bmod N$

1. Alice sends $c_i \leftarrow e_{\mathsf{pk}}\left(\binom{p}{i} x^i\right)$ to Bob for $i = 0, ..., p$.
2. Bob sends $c^B \leftarrow e_{\mathsf{pk}}(-s^B) \cdot \prod_{i=0}^{p} (c^i)^{y^{p-i}}$, where $s^B \in_r \mathbb{Z}_N$. Then, Bob sends this to Alice
3. Alice outputs $s^A \leftarrow d^A(c^B)$. Bob outputs s^B.

Fig. 4. Private sharing of degree-d polynomial kernels

- Alice's input: test example $\boldsymbol{x}^A$, training examples $(\boldsymbol{x}_1^A, ..., \boldsymbol{x}_n^A)$, share of classifier $\boldsymbol{\alpha}^A$ and a valid key pair $(\mathsf{pk}, \mathsf{sk})$
- Bob's input: test example $\boldsymbol{x}^B$, training examples $(\boldsymbol{x}_1^B, ..., \boldsymbol{x}_n^B)$, share of classifier $\boldsymbol{\alpha}^B$ and a public key pk
- Alice's output: shared prediction $h^A(\boldsymbol{x})$
- Bob's output: shared prediction $h^B(\boldsymbol{x})$ where $h(\boldsymbol{x}) = h^A(\boldsymbol{x}) + h^B(\boldsymbol{x}) \bmod N$

1. For $i = 1, ..., n$, Alice sends $e_{\mathsf{pk}}(\alpha_i^A)$ to Bob. Then Bob computes $e_{\mathsf{pk}}(\alpha_i) \leftarrow e_{\mathsf{pk}}(\alpha_i^A) \cdot e_{\mathsf{pk}}(\alpha_i^B)$.
2. Kernel sharing:
   (a) For $i = 1, ..., n$, Alice and Bob jointly do scalar product protocol to have shares

$$s^{A1} + s^{B1} = \boldsymbol{x}_i^A \cdot \boldsymbol{x}^B \bmod N, s^{A2} + s^{B2} = \boldsymbol{x}_i^B \cdot \boldsymbol{x}^A \bmod N \tag{6}$$

   where Alice obtains s^{A1}, s^{A2} and Bob obtains s^{B1}, s^{B2} as outputs
   (b) Alice and Bob jointly do power-of-sum protocol to have shares of kernels

$$k^A(\boldsymbol{x}, \boldsymbol{x}_i) + k^B(\boldsymbol{x}, \boldsymbol{x}_i) = k(\boldsymbol{x}, \boldsymbol{x}_i) = (\boldsymbol{x} \cdot \boldsymbol{x}_i)^p \bmod N \tag{7}$$

   where Alice's input is $s^{A1} + s^{A2} + \boldsymbol{x}_i^A \cdot \boldsymbol{x}^A$ and Bob's input is $s^{B1} + s^{B2} + \boldsymbol{x}_i^B \cdot \boldsymbol{x}^B$.
3. Evaluation:
   (a) For $i = 1, ..., n$, Alice sends $e_{\mathsf{pk}}(k^A(\boldsymbol{x}_i, \boldsymbol{x}))$ and $e_{\mathsf{pk}}(\alpha_i^A k^A(\boldsymbol{x}_i, \boldsymbol{x}))$ to Bob.
   (b) Bob generates $s_B \in_r \mathbb{Z}_N$, computes c as follows and sends c to Alice

$$c \leftarrow \prod_{i=1}^{n} \left(e_{\mathsf{pk}}(\alpha_i^A k^A(\boldsymbol{x}_i, \boldsymbol{x})) \cdot e_{\mathsf{pk}}(\alpha_i^B k^B(\boldsymbol{x}_i, \boldsymbol{x})) \cdot e_{\mathsf{pk}}(\alpha_i^A)^{k^B(\boldsymbol{x}_i, \boldsymbol{x})} \cdot e_{\mathsf{pk}}(k^A(\boldsymbol{x}_i, \boldsymbol{x}))^{\alpha_i^B} \right) \cdot e_{\mathsf{pk}}(-s^B) \tag{8}$$

   (c) Alice: Compute $s^A \leftarrow d^A(c)$
4. Prediction: Alice and Bob jointly compute $h^A(\boldsymbol{x}) + h^B(\boldsymbol{x}) = \mathrm{sgn}(s^A + s^B) \bmod N$ by secure function evaluation

Fig. 5. Private prediction by polynomial kernels

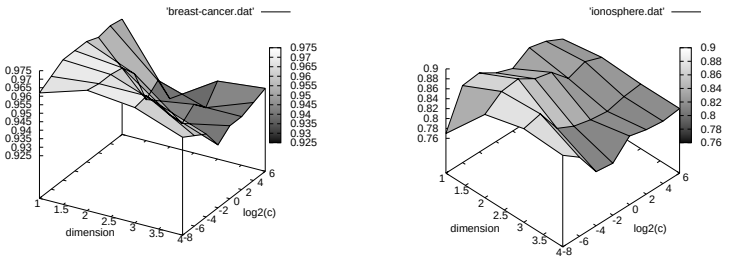

Fig. 6. Generalization error of SVM classifiers with varying polynomial-degree and margin parameter (left: ionosphere, right: breast cancer).

The computation time is summarized in Table 2. The number of steps including cryptographic operations in the first three steps, kernel sharing, evaluation and prediction, are $O(n^2d+np)$, $O(n^2\ell k)$, and $O(n\ell k)$ in type-II. In type-I, the number of steps with cryptographic operations of kernel sharing is decreased to $O(np)$. Private evaluation of $f(\boldsymbol{x})$ is the most time consuming because it includes large numbers of multiplication and exponentiation of ciphers. The computation complexity of private prediction is not large but this includes SFE. Therefore, the computation time of these steps takes a large portion of the entire computation, too. The complexity of Hamming distance protocol (gen. err.) and argmin by SFE (model sel.) is $O(n\ell k)$ and $O(k)$. The number of iterations of Hamming distance protocol is not small; however this protocol does not include SFE. The model selection step includes SFE; however the number of iteration k is usually not very large (in our case, $k = 32$). Thus, these are not time-consuming.

The computation time of SFE is 20 times more than that of Hamming distance protocol. However, kernel sharing, evaluation, and prediction take a large portion of the entire computation time. Therefore, the improvement in computation time by making use of Hamming distance protocol in this work is no more than 12 % in Type-I and 9 % in Type-II. From these, we can conclude that our Hamming distance protocol actually reduces the computation time of model selection while speeding up of the evaluation of the prediction function is essential to reduce the total amount of the computation time in this privacy-preserving classification.

5 Privacy-preserving Attribute Selection in Naive Bayes Classification

As another application of our protocol, privacy-preserving attribute selection in naive Bayes classification is demonstrated in this section. Vaidya et. al. have presented privacy-preserving naive Bayes classifiers in both horizontally and vertically partitioned datasets of nominal and numerical attributes [11]. Although our protocol can be combined with any possible variations of Vaidya et. al.'s classifiers, here we restrict our attention to vertically partitioned naive Bayes classifiers of nominal attribute, in which privately

Table 2. Computational time (second) consumed at each step of model selection in privacy-preserving SVM. In the generalization error evaluation step, computation time of Hamming distance protocol (HDP) and SFE are compared.

dataset			ionopshere			
computation			type-I/HDP	type-II/HDP	type-I/SFE	type-II/SFE
kernel sharing	$k(\boldsymbol{x}_i, \boldsymbol{x}_j)$	$d = 2$	2195	13574	2195	13574
		$d = 3$	2217	13596	2217	13596
		$d = 4$	2273	13652	2273	13652
eval.	$f(\boldsymbol{x}_j)$		52158			
pred.	$\hat{y}_j = \text{sgn}(f(\boldsymbol{x}_j))$		30576			
gen. err.	$\epsilon = \sum_j [y_j \neq \hat{y}_j]$		89.9		16643	
model sel.	$\ell^* = \min_\ell \epsilon^\ell$		127.6			
total			89996	140686	106189	157239

dataset			breast-cancer			
computation			type-I/HDP	type-II/HDP	type-I/SFE	type-II/SFE
kernel sharing	$k(\boldsymbol{x}_i, \boldsymbol{x}_j)$	$d = 2$	4264	42693	4264	42693
		$d = 3$	4308	47001	4308	47001
		$d = 4$	4417	51418	4417	51418
eval.	$f(\boldsymbol{x}_j)$		195840			
pred.	$\hat{y}_j = \text{sgn}(f(\boldsymbol{x}_j))$		59404			
gen. err.	$\epsilon = \sum_j [y_j \neq \hat{y}_j]$		125.1		32473.6	
model sel.	$\ell^* = \min_\ell \epsilon^\ell$		127.6			
total			268485	396878	300833	429227

shred predictions are returned to parties. See [11] for details of their training protocols and prediction protocol.

Setting. Two datasets (lenses and breast-cancer) were taken from the UCI machine learning repository [1]. The lenses dataset has three classes and the breast-cancer dataset has two classes. Since the lenses dataset is not binary classification, label vectors were transformed into binary vectors using indicator variables. For attribute selection, we enumerated subsets of attributes exhaustively and trained naive Bayes classifiers for each attribute subset. Lenses and breast-cancer dataset have four and nine attribute values, respectively, So we evaluated $\sum_{i=1}^{4} \binom{4}{i} = 15$ and $\sum_{i=1}^{9} \binom{9}{i} = 511$ attribute subsets in lenses and breast-cancer, respectively. The generalization error of the classifier trained is evaluated by 10-fold cross validation in each setting.

Given privately shared naive Bayes classifiers after the training phase, we measured the computational time spent for (1) evaluation and prediction (Step 1 of Fig. 3), (2) evaluation of generalization error (Hamming distance protocol or SFE, Step 2 of Fig. 3), (3) attribute selection (Step 3 of Fig. 3).

Results. The number of correctly classified examples (accuracy) was 70.83% in lenses dataset and 71.67% in breast-cancer dataset without attribute selection. With attribute selection, the accuracy was improved as 87.50% in lenses dataset (two of five

dataset	lenses		breast-cancer	
computation	w/ SFE	w/ HD ptcl.	w/ SFE	w/ HD ptcl.
eval. and pred.	179.27		43345	
gen.err.	58.35	3.0324	18329	136.47
att. sel.	5.981		203.76	
total	243.6	188.3	61877	43685

Table 3. Computational time (second) consumed at each step of attribute selection in privacy-preserving naive Bayes. In the generalization error step, computation time of Hamming distance protocol (HD ptcl.) and that of SFE was compared.

attributes were selected) and 75.17% in breast-cancer dataset (two of nine attributes were selected).

The computation time spent in each step is shown in Table 3. Again, we measured the time of computation in which SFE or Hamming distance protocol is used for the privacy-preserving evaluation of generalized errors. From the results, the computation time is reduced 23% in lenses dataset and 30% in breast-cancer dataset by making use of Hamming distance protocol for the evaluation of generalization error.

6 Conclusion

In this paper, we presented solutions for generalization error evaluation in privacy-preserving classification. We consider both vertically partitioned and horizontally partitioned data. In addition, our solutions can work with both regular prediction and privately shared prediction. Privacy-preserving classification can be designed (and have been designed by other researchers) for various combinations of these partitioning models and representations are possible. In this paper, we introduced a new Hamming distance protocol for generalized error evaluation that works with any of these representations and any data partitioning models.

To show the universal applicability of our protocol, we experimentally demonstrated our protocol with model selection in support vector machine and attribute selection in naive Bayes classification. The result showed that our privacy-preserving model selection and attribute selection could successfully improve the classification accuracy in privacy-preserving naive Bayes and in privacy-preserving SVM. Furthermore, the computation time for Hamming distance computation was reduced to nearly one-fiftieth by making use of our Hamming distance protocol in comparison with that of SFE, while the reduction rate of the total computation time in privacy-preserving classifiers including model/attribute selection with real-world datasets was around 10%–30%. This is because there is a computation bottleneck not only in the evaluation of generalization errors but also in the evaluation of the prediction function. From these observations, we conclude that speeding up of the evaluation of the prediction function is essential to further reducing the total amount of computation time in privacy-preserving classification.

Acknowledgement

This work was carried out partly while the first author was a visitor of the DIMACS Center. The second author is partially supported by the National Science Foundation under grant number CNS–0822269.

References

1. A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
2. W. Du and Z. Zhan. Building decision tree classifier on private data. In *Proceedings of the IEEE international conference on Privacy, security and data mining-Volume 14*, pages 1–8. Australian Computer Society, 2002.
3. B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. On private scalar product computation for privacy-preserving data mining. In *Proceedings of the The 7th Annual International Conference in Information Security and Cryptology (ICISC 2004)*, volume 3506, pages 104–120. Springer, 2004.
4. O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, 2004.
5. S. Laur, H. Lipmaa, and T. Mielikäinen. Cryptographically private support vector machines. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 618–624. ACM Press New York, NY, USA, 2006.
6. Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. *Journal of Cryptology*, 15(3):177–206, 2002.
7. D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. Fairplay: a secure two-party computation system. In *Proc. of the 13th USENIX Security Symposium*, pages 287–302, 2004.
8. P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Eurocrypt'99*, pages 223–238. Springer, 1999.
9. B. Schölkopf and A.J. Smola. *Learning with kernels*. MIT Press, 2002.
10. J. Vaidya, C. Clifton, and M. Zhu. Privacy Preserving Data Mining, volume 19 of Series: Advances in Information Security, 2006.
11. J. Vaidya, M. Kantarcıoğlu, and C. Clifton. Privacy-preserving Naïve Bayes classification. *The VLDB Journal The International Journal on Very Large Data Bases*, pages 1–20, 2007.
12. Z. Yang, S. Zhong, and R.N. Wright. Towards Privacy-Preserving Model Selection. In *First ACM SIGKDD International Workshop, PinKDD 2007, Lecture Notes In Computer Science*, volume 4980, pages 138–152. Springer, 2008.
13. A. C.-C. Yao. How to generate and exchange secrets. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 162–167, 1986.
14. H. Yu, X. Jiang, and J. Vaidya. Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data. In *Proceedings of the 2006 ACM symposium on Applied computing (SAC)*, pages 603–610. ACM Press New York, NY, USA, 2006.
15. J. Zhan, L.W. Chang, and S. Matwin. Privacy Preserving K-nearest Neighbor Classification. *International Journal of Network Security*, 1(1):46–51, 2005.