Privacy-Preserving Classification of Horizontally Partitioned Data via Random Kernels

Olvi L. Mangasarian* Edward W. Wild†

Abstract

We propose a novel privacy-preserving nonlinear support vector machine (SVM) classifier for a data matrix A whose columns represent input space features and whose individual rows are divided into groups of rows. Each group of rows belongs to an entity that is unwilling to share its rows or make them public. Our classifier is based on the concept of a reduced kernel $K(A, B')$ where B' is the transpose of a completely random matrix B. The proposed classifier, which is public but does not reveal the privately-held data, has accuracy comparable to that of an ordinary SVM classifier based on the entire data.

Keywords: privacy preserving classification, horizontally partitioned data, support vector machines

1 INTRODUCTION

Recently there has been wide interest in privacy-preserving support vector machine (SVM) classifiers. Basically, the problem revolves around generating a classifier based on data, parts of which are held by private entities who, for various reasons, are unwilling to make it public. For example, when each entity holds all the feature values for its own group of individuals while other entities hold similar data for other groups of individuals, the data is referred to as *horizontally partitioned*. This is so because feature values for each individual are stored as rows of a data matrix, while a specific feature values for all individuals are represented by columns of a data matrix. In [18, 19] *horizontally partitioned* privacy-preserving SVMs and induction tree classifiers were obtained for data where different entities hold the same features for different groups of individuals. Privacy-preserving SVM classifiers were obtained for vertically partitioned data by securely computing a kernel matrix using random perturbations in [20], and by using random kernels in [13]. Other privacy-preserving classifying techniques include cryptographically private SVMs [15], wavelet-based distortion [10] and rotation perturbation [2].

In this work we propose a highly efficient privacy-preserving SVM (PPSVM) classifier for horizontally partitioned data that is different from existing SVM classifiers for such partitioned data and which is based on the following two ideas. For a given data matrix $A \in R^{m\times n}$ instead of using the usual kernel function $K(A, A') : R^{m\times n} \times R^{n\times m} \longrightarrow R^{m\times m}$ we use a *reduced* kernel [9, 8] $K(A, B') : R^{m\times n} \times R^{n\times \bar{m}} \longrightarrow R^{m\times \bar{m}}$, $\bar{m} < n$, where B is a completely random matrix. The second idea is that each entity makes public only the matrix product of its privately held matrix of data rows multiplied by the transpose of the random matrix B for linear kernels, and a similar kernel function for nonlinear kernels. By employing these two ideas, we shall describe an algorithm that protects the privacy of each horizontal partition of the data matrix A, owned by a distinct entity, while generating an SVM classifier which has tenfold correctness comparable to that of an ordinary SVM classifier.

*Computer Sciences Department, University of Wisconsin, Madison, WI 53706 and Department of Mathematics, University of California at San Diego, La Jolla, CA 92093.*olvi@cs.wisc.edu*.

†Computer Sciences Department, University of Wisconsin, Madison, WI 53706. *wildt@cs.wisc.edu*.

We now briefly describe the contents of the paper. In Section 2 we describe our method for a privacy-preserving linear SVM classifier for horizontally partitioned data and show that what each entity reveals does not lead to the disclosure of its privately held data. In Section 3 we treat nonlinear SVM classifiers by a similar approach. In Section 4 we give computational results that show the accuracy of our approach is comparable to ordinary SVMs. Section 5 concludes the paper.

We describe our notation now. All vectors will be column vectors unless transposed to a row vector by a prime $'$. For a vector $x \in R^n$ the notation x_j will signify either the j-th component or j-th block of components. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x'y$. For $x \in R^n$, $\|x\|_1$ denotes the 1-norm: $(\sum_{i=1}^{n}|x_i|)$ while $\|x\|$ denotes the 2-norm: $(\sum_{i=1}^{n}(x_i)^2)^{\frac{1}{2}}$. The notation $A \in R^{m\times n}$ will signify a real $m \times n$ matrix. For such a matrix, A' will denote the transpose of A, A_i will denote the i-th row or i-th block of rows of A, $A_{\cdot j}$ the j-th column or the j-th block of columns of A, and A_{ir} the ir-th element of A or the r-th row of i-th block A_i of A. A vector of ones in a real space of arbitrary dimension will be denoted by e. Thus for $e \in R^m$ and $y \in R^m$ the notation $e'y$ will denote the sum of the components of y. A vector of zeros in a real space of arbitrary dimension will be denoted by 0. For $A \in R^{m\times n}$ and $B \in R^{k\times n}$, a *kernel* $K(A, B')$ maps $R^{m\times n} \times R^{n\times k}$ into $R^{m\times k}$. In particular, if x and y are column vectors in R^n then, $K(x', y)$ is a real number, $K(x', B')$ is a row vector in R^k and $K(A, B')$ is an $m \times k$ matrix. The base of the natural logarithm will be denoted by ε. A frequently used kernel in nonlinear classification is the Gaussian kernel [17, 3, 11] whose ij-th element, $i = 1, \ldots, m,\ j = 1, \ldots, k$, is given by: $(K(A, B'))_{ij} = \varepsilon^{-\mu\|A_i - B_{\cdot j}'\|^2}$, where $A \in R^{m\times n}$, $B \in R^{k\times n}$ and μ is a positive constant. We shall not assume that our kernels satisfy Mercer's positive definiteness condition [17, 16, 4] However, we shall assume that they are associative in the following sense:

$$K\left(\begin{bmatrix} E \\ F \end{bmatrix}, G'\right) = \begin{pmatrix} K(E, G') \\ K(F, G') \end{pmatrix}, \tag{1.1}$$

where $E \in R^{m_1\times n}$, $F \in R^{m_2\times n}$, $G \in R^{k\times n}$. It is straightforward to check that both a linear kernel $K(A, B') = AB'$ and a Gaussian kernel satisfy (1.1). For the k matrices $G^1, \ldots, G^k$, of the same dimensions, their *affine hull* is defined as the set $\{H \mid H = \sum_{j=1}^{j=k}\lambda^j G^j$, with $\sum_{j=1}^{j=k}\lambda^j = 1$. The abbreviation "s.t." stands for "subject to".

2 Privacy-Preserving Linear Classifier for Horizontally Partitioned Data

The dataset that we wish to obtain a classifier for consists of m points in R^n represented by the m rows of the matrix $A \in R^{m\times n}$. Each row contains values for n features associated with a specific individual, while each column contains m values of a specific feature associated with m different individuals. The matrix A is divided into q blocks of $m_1, m_2, \ldots m_q$ rows with $m_1 + m_2 + \ldots + m_q = m$, and each block of rows "owned" by an entity that is unwilling to make it public or share it with others. Furthermore, each row of A is labeled as belonging to the class $+1$ or -1 by a corresponding diagonal matrix $D \in R^{m\times m}$ of ± 1's. The linear kernel classifier to be generated based on this data will be a separating plane in R^n:

$$x'w - \gamma = x'B'u - \gamma = 0, \tag{2.2}$$

where $w = B'u$, $w \in R^n$ is the normal to the separating plane $x'w - \gamma = 0$, $\gamma \in R$ determines the distance of the plane from the origin and B is a completely random matrix in $R^{k\times n}$. The change of variables $w = B'u$ is employed in order to kernelize the data and is motivated by the fact that when

$B = A$ and hence $w = A'u$, the variable u is the dual variable for a 2-norm SVM [11]. The variables $u \in R^k$ and $\gamma \in R$ are to be determined by an optimization problem such that the labeled data A satisfy, to the extent possible, the separation condition:

$$D(AB'u - e\gamma) \geq 0. \tag{2.3}$$

This condition (2.3) places the $+1$ and -1 points represented by A on opposite sides of the separating plane (2.2). In general, the matrix B which determines a transformation of variables $w = B'u$, is set equal to A. However, in reduced support vector machines [9, 6] $B = \bar{A}$, where $\bar{A}$ is a submatrix of A whose rows are a small subset of the rows of A. In fact B can be a random matrix in $R^{\bar{m} \times n}$ with $\bar{m} < n$ for our application here. The random choice of B holds the key to our privacy-preserving classifier and has been used effectively in SVM classification problems [12]. Our computational results of Section 4 will show that there is no essential difference between using a random B or a random submatrix of $\bar{A}$ of the rows of A as in reduced SVMs [9, 8]. We shall partition our data matrix A into q row blocks $A_1, A_2, \ldots, A_q$ with each row block belonging to one of the q entities and held privately by it and never made public. However, what is made public by each entity i is the matrix product A_iB' which allows the public calculation of the the linear kernel $AB' \in R^{m \times \bar{m}}$ as follows:

$$\begin{bmatrix} A_1B' \\ A_2B' \\ \vdots \\ A_qB' \end{bmatrix} \tag{2.4}$$

We are now ready to state our algorithm which will provide a linear classifier for the data without revealing the private blocks of the privately held data blocks $A_1, A_2, \ldots, A_p$. The accuracy of this algorithm will be comparable to that of a linear SVM using a publicly available linear kernel AA' instead of merely the blocks $A_1B', A_2B', \ldots, A_pB'$ of (2.4) as is the case here.

Algorithm 2.1. **Linear PPSVM Algorithm**

(I) All q entities agree on the same random matrix $B \in R^{\bar{m} \times n}$ with $\bar{m} < n$ for security reasons as justified in the explanation immediately following this algorithm.

(II) All entities make public the class matrix D where $D_{\ell\ell} = \pm 1$, $\ell = 1, \ldots, m$, for the data matrices A_i, $i = 1, \ldots, m$, that they all hold.

(III) Each entity i makes public its linear kernel A_iB'. This does not reveal A_i but allows the public computation of the full linear kernel:

$$AB' = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_q \end{bmatrix} B' \tag{2.5}$$

(IV) A publicly calculated linear classifier $x'Bu - \gamma = 0$ is computed by some standard method such as a 1-norm SVM [11, 1]:

$$\begin{array}{rrcl} \displaystyle\min_{(u,\gamma,y)} & \nu\|y\|_1 + \|u\|_1 & & \\ s.t. & D(AB'u - e\gamma) + y & \geq & e, \\ & y & \geq & 0. \end{array} \tag{2.6}$$

(V) *For each* new *privately held* $x \in R^n$ *obtained by any entity, that entity privately computes* $x'B'$ *from which a linear classifier is computed as follows:*

$$x'B'u - \gamma = 0. \tag{2.7}$$

Note that *no* entity i reveals its data set A_i. This is so because each entity reveals only the $m_i\bar{m}$ numbers constituting the matrix $P_i = (A_iB') \in R^{m_i\times\bar{m}}$. When $\bar{m} < n$, there is an infinite number of matrices $A_i \in R^{m_i\times n}$ satisfying $A_iB' = P_i$, given B and P_i. We make this statement more precise by first showing that at least an exponential number of matrices A_i satisfy $A_iB' = P_i$ for a given B and P_i when $\bar{m} < n$. We then show that the infinite number of matrices that lie in the affine hull of these matrices also satisfy $A_iB' = P_i$. This obviously precludes the possibility of determining the dataset A_i held by entity i given only A_iB'.

Proposition 2.2. *Given the matrix product* $P_i{}' = A_iB' \in R^{m_i\times\bar{m}}$ *where* $A_i \in R^{m_i\times n}$ *is unknown and* B *is a known matrix in* $R^{\bar{m}\times n}$ *with* $\bar{m} < n$*, there are an infinite number of solutions, including:*

$$\binom{n}{\bar{m}}^{m_i} = \left(\frac{n!}{(n-\bar{m})!\bar{m}!}\right)^{m_i}, \tag{2.8}$$

possible solutions $A_i \in R^{m_i\times n}$ *to the equation* $BA_i{}' = P_i$*. Furthermore, the infinite number of matrices in the affine hull of these* $\binom{n}{\bar{m}}^{m_i}$ *matrices also satisfy* $BA_i{}' = P_i$*.*

Proof Consider the problem of solving for row r of A_i, that is $A_{ir} \in R^n$, $r \in \{1,\ldots,m_i\}$, from the r-th equation of $BA_i{}' = P_i$:

$$BA_{ir}{}' = P_{ir}. \tag{2.9}$$

Since $\bar{m} < n$, and B is a random matrix, it follows by [5] that each of the $\binom{n}{\bar{m}}$ random $\bar{m}\times\bar{m}$ square submatrices of B are of full rank and hence nonsingular. Consequently, Equation (2.9) has $\binom{n}{\bar{m}}$ solutions for each row r of A_i, that is A_{ir}, $r \in \{1,\ldots,m_i\}$. Hence there are $\binom{n}{\bar{m}}^{m_i} = \left(\frac{n!}{(n-\bar{m})!\bar{m}!}\right)^{m_i}$ solutions for the m_i rows of A_i.

To prove the last statement of the proposition, we note that if each of k matrices $A_i^1,\ldots,A_i^k$ solve $BA_i{}' = P_i$ for a given B and P_i, then so does $A_i = \sum_{j=1}^{j=k}\lambda^jA_i^j$ for $\sum_{j=1}^{j=k}\lambda^j = 1$. Hence any matrix in the affine hull of $A_i^1,\ldots,A_i^k$, $\{H \mid H = \sum_{j=1}^{j=k}\lambda^jA_i^j, \sum_{j=1}^{j=k}\lambda^j = 1\}$ also satisfies (2.9). $\square$

For the specific case of $\bar{m} = n-1$, which is used for our numerical results, we have that:

$$\binom{n}{\bar{m}}^{m_i} = \left(\frac{n!}{(n-\bar{m})!\bar{m}!}\right)^{m_i} = (n)^{m_i}. \tag{2.10}$$

This translates to $(30)^{20}$ for the typical case of $n = 30$, $\bar{m} = 29$ and $m_i = 20$.

We turn now to nonlinear classification.

3 Nonlinear SVM Classifier for Horizontally Partitioned Data

The approach to nonlinear classification is similar to that for the linear one, except that we make use of the associative property (1.1) of a nonlinear kernel which is satisfied by a Gaussian kernel. This kernel is one of the most commonly used nonlinear kernels. Otherwise, the approach is very similar to that of a linear kernel. We state that approach explicitly now.

ALGORITHM 3.1. **Nonlinear PPSVM Algorithm**

(I) All q entities agree on the same random matrix $B \in R^{\bar{m}\times n}$ with $\bar{m} < n$ for security reasons as justified in the explanation immediately following this algorithm.

(II) All entities make public the class matrix $D_{\ell\ell} = \pm 1$, $\ell = 1, \ldots, m$, for the data matrices A_i, $i = 1, \ldots, m$ that they all hold.

(III) Each entity i makes public its nonlinear kernel $K(A_i, B')$. This does not reveal A_i but allows the public computation of the full nonlinear kernel:

$$K(A, B') = K\left(\begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_q \end{bmatrix}, B'\right) = \begin{bmatrix} K(A_1, B') \\ K(A_2, B') \\ \vdots \\ K(A_q, B') \end{bmatrix} \tag{3.11}$$

(IV) A publicly calculated nonlinear classifier $K(x', B)u - \gamma = 0$ is computed by some standard method such as a 1*-norm SVM [11, 1]:*

$$\begin{array}{lrcl} \displaystyle\min_{(u,\gamma,y)} & \nu\|y\|_1 + \|u\|_1 & & \\ s.t. & D(K(A, B')u - e\gamma) + y & \geq & e, \\ & y & \geq & 0. \end{array} \tag{3.12}$$

(V) For each new *$x \in R^n$ obtained by an entity, that entity privately computes $K(x', B')$ from which a nonlinear classifier is computed as follows:*

$$K(x', B')u - \gamma = 0 \tag{3.13}$$

Note that in the above algorithm no entity i reveals its dataset A_i. This is so because it is impossible to compute unique $m_i n$ numbers constituting the matrix $A_i \in R^{m_i \times n}$ given only the $m_i\bar{m}$ numbers constituting the revealed kernel matrix $K(A_i, B') \in R^{m_i \times \bar{m}}$ with $\bar{m} < n$. However, all entities share the publicly computed nonlinear classifier (3.13) without revealing their individual datasets A_i, $i = 1, \ldots, q$, or any new x that they obtain. Thus, for example, if we wish to compute the r-th row A_{ir} of entity i's data matrix A_i from the given matrix $P_i = K(A_i, B') \in R^{m_i \times \bar{m}}$, we need to solve the $\bar{m}$ *nonlinear* equations $K(B, {A_{ir}}') = {P_{ir}}'$ for the n components of $A_{ir} \in R^n$. Because $n > \bar{m}$, this would in general generate a nonlinear surface in R^n containing an infinite number of solutions which makes it impossible to determine A_{ir} uniquely.

We turn now to our computational results.

4 Computational Results

We illustrate the effectiveness of our proposed privacy preserving SVM (PPSVM) in two ways. First, we demonstrate that by using our approach entities can obtain classifiers with lower misclassification error than classifiers obtained using only the examples of each entity alone. Second, we show that a random kernel $K(A, B')$ achieves comparable accuracy to the usual kernel $K(A, A')$ or the reduced kernel $K(A, \bar{A}')$, where $\bar{A}$ contains a subset of the rows of A. All experiments were run using both a linear kernel and the commonly used Gaussian kernel described in Section 1. In all of our results, $\bar{A}$ consisted of ten percent of the rows of A randomly selected, while B was a completely random matrix with the same number of columns as A. The number of rows of B was set to the minimum of $n-1$ and the number of rows of $\bar{A}$, where n is the number of features in the dataset. Thus, we ensure that the conditions discussed in the previous sections hold in order to guarantee the private data A_i cannot be recovered from $K(A_i, B')$. Each entry of B was selected independently from a uniform distribution on the interval $[0, 1]$. All datasets were normalized so that each feature was between zero and one. This normalization can be carried out if the entities disclose only the maximum and minimum of each feature in their datasets. When computing ten-fold cross validation, we first divided the data into folds and set up the training and testing sets in the usual way. Then each entity's dataset was formed from the training set of each fold. The accuracies of all classifiers were computed on the testing set.

4.1 Comparison of our approach to classifiers obtained using only each entity's examples

We investigate the benefit of using our PPSVM approach instead of using only the examples available to each entity using seven datasets from the UCI repository [14]. To simulate a situation in which each entity has only a subset of the available examples, we randomly distribute the examples among the entities such that each entity receives about the same number of examples. We chose arbitrarily to perform experiments using however many entities were needed so that each entity received about 25 examples. To save time, we computed results only for three entities, though when sharing we assumed the entire dataset was shared.

Figure 1 shows results comparing the ten-fold cross validation misclassification error of our linear kernel PPSVM with the average misclassification error of the 1-norm SVM classifiers learned using only the examples available to each of three entities. Points below the 45 degree line represent experiments in which our PPSVM has lower error rate than the average error rate of the classifiers learned with only each entity's subset of the examples. This indicates that the entities can expect improved performance using PPSVM instead of going it alone. The results shown in Figure 1 are detailed in Table 1. We note that PPSVM obtains classifiers with lower error than the average of the classifiers using only each entity's examples in six of the seven experiments. The parameter ν was selected from $\{10^i | i = -7, \ldots, 7\}$ for each dataset was selected using ten-fold cross validation on the training set when data was shared, and leave-one-out cross validation on the training set when data was not shared. We used leave-one-out cross validation when data was not shared because each entity only had about 25 examples in its training set.

Figure 2 shows similar results for experiments using Gaussian kernels, with details in Table 2. We used the same datasets as for the experiments described above. To save time, we used the tuning strategy described in [7]. In this Nested Uniform Design approach, rather than evaluating a classifier at each point of a grid in the parameter space, the classifier is evaluated only at a set of points which is designed to "cover" the original grid to the extent possible. The point from this smaller set on which the classifier does best is then made the center of a grid which covers a smaller range of parameter space, and the process is repeated. Huang et al. [7] demonstrate empirically that this approach finds classifiers with similar misclassification error as a brute-force search through the entire grid. We set the initial range of $\log_{10} \nu$ to $[-7, 7]$, and the initial range of $\log_{10} \mu$ as described in [7]. Note that we set the initial

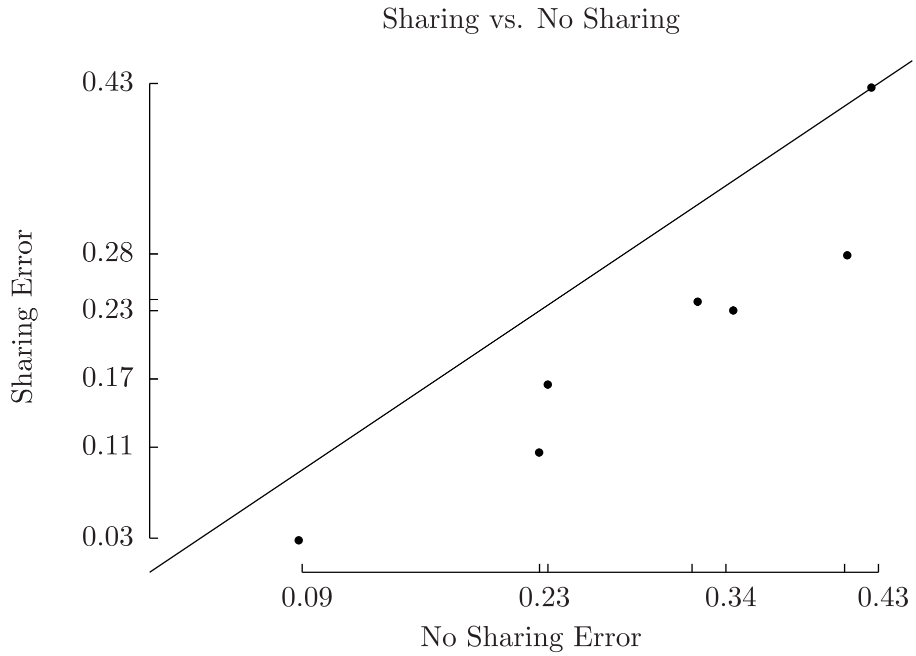

Figure 1: Error rate comparison for the seven datasets of Table 1 of a 1-norm linear SVM sharing A_iB' data for each entity versus a 1-norm linear SVM using only the examples A_i of each entity. Points below the diagonal represent situations in which the error rate for sharing is better than the error rate for not sharing. Results are given for each dataset with examples randomly distributed so that each entity has about 25 examples.

Dataset Examples $\times$ Input Features	No Sharing Error	Sharing Error
Cleveland Heart 297 $\times$ 13	0.2349	0.1652
Ionosphere 351 $\times$ 34	0.2298	0.1054
WDBC 569 $\times$ 30	0.0879	0.0281
Arrhythmia 452 $\times$ 279	0.4116	0.2788
Pima Indians 768 $\times$ 8	0.3443	0.2303
Bupa Liver 345 $\times$ 6	0.4259	0.4263
German Credit 1000 $\times$ 24	0.3233	0.2380

Table 1: Comparison of error rates for entities not sharing and sharing their datasets using a 1-norm linear SVM.

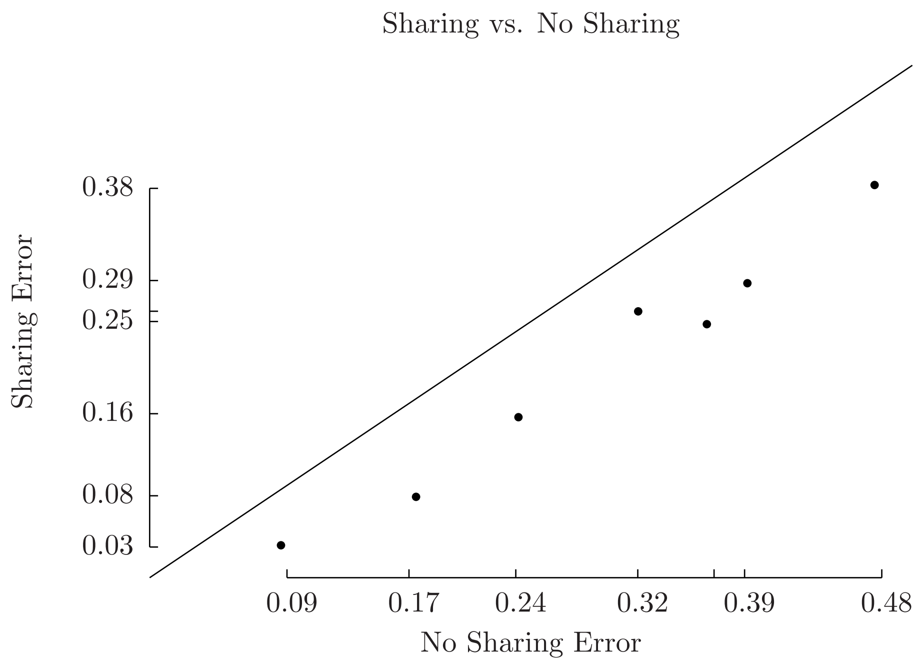

Figure 2: Error rate comparison for the seven datasets of Table 2 of a 1-norm nonlinear SVM sharing $K(A_i, B')$ data for each entity versus a 1-norm nonlinear SVM using only the examples A_i of each entity. Points below the diagonal represent situations in which the error rate for sharing is better than the error rate for not sharing.

Dataset Examples $\times$ Input Features	No Sharing Error	Sharing Error
Cleveland Heart 297×13	0.2418	0.1567
Ionosphere 351×34	0.1747	0.0790
WDBC 569×30	0.0861	0.0316
Arrhythmia 452×279	0.3919	0.2873
Pima Indians 768×8	0.3203	0.2599
Bupa Liver 345×6	0.4752	0.3832
German Credit 1000×24	0.3653	0.2473

Table 2: Comparison of error rates for entities not sharing and sharing their datasets using a 1-norm nonlinear Gaussian SVM.

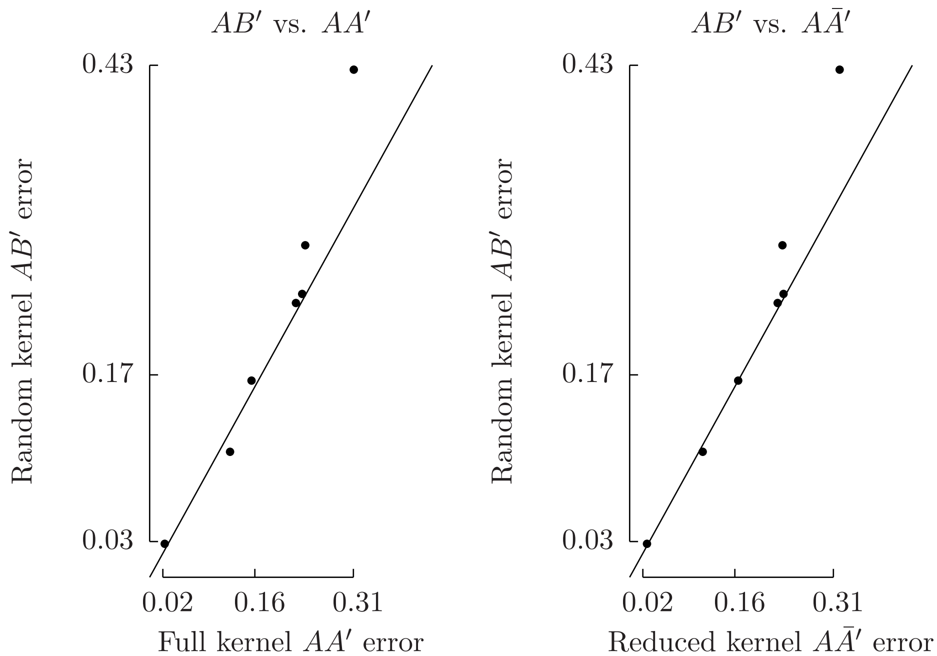

Figure 3: Error rate comparison of 1-norm linear SVMs for random kernel versus full and reduced kernels. For points below the diagonal, the random kernel has a better error rate. The diagonal line in each plot marks equal error rates. One result is given for each dataset listed in Table 1.

range of $\log_{10}\mu$ independently for each entity using only that entity's examples. We used a Uniform Design with thirty runs from `http://www.math.hkbu.edu.hk/UniformDesign` for both nestings, and used leave-one-out cross validation on the training set to evaluate each (ν, μ) pair when the entities did not share and five-fold cross validation on the training set they did. We chose to use leave-one-out cross validation when not sharing because only about 25 examples were available to each entity in that situation.

4.2 Comparison of a random kernel to full and reduced kernels To justify the use of a random kernel we compare the performance of our PPSVM (Algorithms 2.1 and 3.1) with both an ordinary 1-norm SVM using a full kernel matrix and a 1-norm SVM using a reduced kernel matrix (RSVM) [9]. Figure 3 shows scatterplots comparing the error rates of our PPSVM with 1-norm SVM and PPSVM with RSVM, all using linear kernels. Note that points close to the 45 degree line represent datasets for which the classifiers being compared have similar error rates. All of the error rates were obtained using the procedure described above for linear kernels, and the datasets used are those in Table 1. Figure 4 shows similar results using the Gaussian kernel. All of the error rates were obtained using the same procedures and datasets as those in Table 2, described above. We note that the misclassification error for our PPSVM approach is comparable with that of 1-norm SVM and RSVM using both linear and Gaussian kernels.

5 Conclusion and Outlook

We have proposed a linear and nonlinear privacy-preserving SVM classifier based on a privately generated and privately held random matrix by each entity. Each entity possesses a different set of

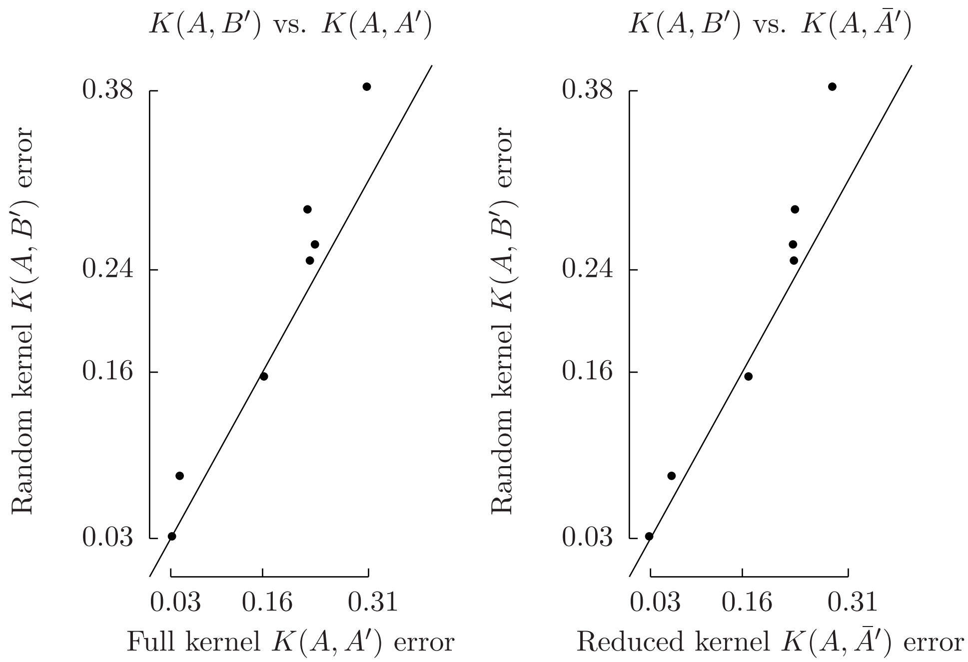

Figure 4: Error rate comparison of 1-norm nonlinear SVMs for random kernel versus full and reduced kernels. For points below the diagonal, the random kernel has a better error rate. The diagonal line in each plot marks equal error rates. One result is given for each dataset listed in Table 2.

examples used collectively to generate the SVM classifier. The proposed approach uses all the privately held data, yet does not reveal the data. Computational comparisons indicate that the accuracy of our proposed approach is comparable to full and reduced data classifiers. Furthermore, a the accuracy obtained by the privacy-preserving SVM is markedly better than the accuracy of SVM classifiers generated by each entity using only its own data.

Acknowledgments The research described in this Data Mining Institute Report 07-03, October 2007, was supported by National Science Foundation Grants CCR-0138308 and IIS-0511905, the Microsoft Corporation and ExxonMobil.

References

[1] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proceedings 15th International Conference on Machine Learning*, pages 82–90, San Francisco, California, 1998. Morgan Kaufmann. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-03.ps.

[2] K. Chen and L. Liu. Privacy preserving data classification with rotation perturbation. In *Proceedings of the Fifth International Conference of Data Mining (ICDM'05)*, pages 589–592. IEEE, 2005.

[3] V. Cherkassky and F. Mulier. *Learning from Data - Concepts, Theory and Methods.* John Wiley & Sons, New York, 1998.

[4] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, Cambridge, 2000.

[5] X. Feng and Z. Zhang. The rank of a random matrix. *Applied Mathematics and Computation*, 185:689–694, 2007.

[6] S.Y. Huang and Y.-J. Lee. Theoretical study on reduced support vector machines. Technical report, National Taiwan University of Science and Technology, Taipei, Taiwan, 2004. yuh-jye@mail.ntust.edu.tw.

[7] C.-H. Hunag, Y.-J. Lee, D.K.J. Lin, and S.-Y. Huang. Model selection for support vector machines via uniform design. In *Machine Learning and Robust Data Mining of Computational Statistics and Data Analysis*, Amsterdam, 2007. Elsevier Publishing Company. http://dmlab1.csie.ntust.edu.tw/downloads/papers/UD4SVM013006.pdf.

[8] Y.-J. Lee and S.Y. Huang. Reduced support vector machines: A statistical theory. *IEEE Transactions on Neural Networks*, 18:1–13, 2007.

[9] Y.-J. Lee and O. L. Mangasarian. RSVM: Reduced support vector machines. In *Proceedings First SIAM International Conference on Data Mining, Chicago, April 5-7, 2001, CD-ROM*, 2001. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/00-07.pdf.

[10] L. Liu, J. Wang, Z. Lin, and J. Zhang. Wavelet-based data distortion for privacy-preserving collaborative analysis. Technical Report 482-07, Department of Computer Science, University of Kentucky, Lexington, KY 40506, 2007. http://www.cs.uky.edu/ jzhang/pub/MINING/lianliu1.pdf.

[11] O. L. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146, Cambridge, MA, 2000. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.

[12] O. L. Mangasarian and M. E. Thompson. Massive data classification via unconstrained support vector machines. *Journal of Optimization Theory and Applications*, 131:315–325, 2006. ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/06-01.pdf.

[13] O. L. Mangasarian, E. W. Wild, and G. M. Fung. Privacy-preserving classification of vertically partitioned data via random kernels. Technical Report 07-02, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, September 2007.

[14] P. M. Murphy and D. W. Aha. UCI machine learning repository, 1992. www.ics.uci.edu/∼mlearn/MLRepository.html.

[15] H. Lipmaa S. Laur and T. Mielikäinen. Cryptographically private support vector machines. In D. Gunopulos L. Ungar, M. Craven and T. Eliassi-Rad, editors, *Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2006, Philadelphia, August 20–23, 2006. ACM*, pages 618–624, 2006. http://eprints.pascal-network.org/archive/00002133/01/cpsvm.pdf.

[16] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[17] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, second edition, 2000.

[18] M.-J. Xiao, L.-S. Huang, H. Shen, and Y.-L. Luo. Privacy preserving id3 algorithm over horizontally partitioned data. In *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*, pages 239–243. IEEE Computer Society, 2005.

[19] H. Yu, X. Jiang, and J. Vaidya. Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 603–610, New York, NY, USA, 2006. ACM Press.

[20] H. Yu, J. Vaidya, and X. Jiang. Privacy-preserving svm classification on vertically partitioned data. In *Proceedings of PAKDD '06*, volume 3918 of *Lecture Notes in Computer Science*, pages 647 – 656. Springer-Verlag, January 2006.