Modelling a secure support vector machine classifier for private data

M. Sumana
Department of Information Science and Engineering,
MSRIT,
Bangalore, Karnataka, 560054, India
Email: sumana.a.a@gmail.com

K.S. Hareesha*
Department of Computer Applications,
Manipal Institute of Technology,
Manipal, Karnataka, 576104, India
Email: hareesh.ks@manipal.edu
*Corresponding Author

Abstract: Privacy preserving data mining engrosses in drawing out information from distributed data without disclosing sensitive information to collaborating sites. This paper aims on the construction of a vertically distributed privacy preserving support vector machine classifier. The learning model is build for datasets, where one of the collaborating parties comprises the dependent attribute. Furthermore, the amount of privacy, computation speed and the accuracy of our classifier outperform other benchmark algorithms. Privacy of the perceptive attributes values of the cooperating sites are retained while performing secure computations. Collaborative classification is performed using these attributes. The site with the dependent attribute is the master site that initiates the process of secure computation to identify support vectors. Homomorphic property is used to protectively compute the data matrix on records/tuples available at sites. The recommended nonlinear privacy preserving classifier provides an accuracy equivalent to the non-privacy undistributed SVM classifier which uses all the attributes directly.

Keywords: support vector machine classification; homomorphic encryption; vertically partitioned data; secure multiparty computation; privacy preserving data mining; PPDM; homomorphic addition; homomorphic multiplication; kernel function; computation cost; accuracy; receiver operating characteristics; Paillier cryptosystem.


Biographical notes: M. Sumana is an Assistant Professor in the Department of Information Science and Engineering. She received her BE in Computer Science and Engineering from Manipal Institute of Technology, Karnataka, India, in 2000 and MTech degree from VTU University in 2007, Karnataka, India. Currently, she is pursuing her PhD in Privacy Preserving Data Mining in Computer Science and Engineering from the Manipal University, Karnataka,
India. She has published papers in reputed journal and conference. Her research interests include data mining, cryptography and secure multiparty computations. She is a life member of the Indian Society for Technical Education (ISTE) and the System Society of India.

K.S. Hareesha is currently a Professor in the Department of Computer Applications, Manipal Institute of Technology, Manipal, Karnataka. He received his PhD in Computer Science and Engineering from Kuvempu University, Shankaraghatta, Karnataka, India in 2008. His major research interest is digital image processing, data mining, bioinformatics and artificial intelligence. Since 2008, he is working in Manipal University, Karnataka India as a professor, before this, he worked in Bapuji Institute of Technology, Davangere, Karnataka, India. He has published quite a good number of research papers in these areas. He has been a Life Member of ISTE, New Delhi, India and a member of IEEE Society.

1 Introduction

Two or more cooperating sites want to derive conclusions from the joined data available at their sites. Nevertheless, a good number of the attributes are sensitive and should not be revealed to the cooperating parties. A solution for modelling from the sensitive data and finding suitable conclusions with a better accuracy and privacy is the goal of this paper. Let us consider a hospital dataset which holds information of patients with sensitive attributes such as type of treatment, days of treatment, treatment success denoted as successful or not. Another dataset called insurance holds personal details such as annual income in family, number of dependents, number of insurance policies, number of residences, etc. about the same set of patients. Doctors from the hospital might be inquisitive to explore whether a treatment given to a new patient knowing his personal details would be successful or not. An insurance agent before availing an insurance policy, if aware of the type of treatment for a disease would be curious to know if the treatment is successful or not. To handle several similar situations in medical, finance, manufacturing sectors, data needs to be mined jointly. The need to maintain the privacy of the data is discussed in the report (Information: Standard for Privacy of Individually Identifiable Health, 2002). Due to the presence of sensitive attributes in these datasets the classification model needs to be built without disclosing any information about the features to one another. Parties collaborate with each other by not revealing their private data. The form of data mining performed by maintaining the privacy of the data at multiple sites is widely known as privacy preserving data mining (PPDM). The intention here is to run a data-mining algorithm on the union of the parties’ databases without allowing any party to view anyone else’s private data. Also the intermediate and the final results obtained during and after the construction of the final model should not disclose any information. The goal of using the privacy preservation measures is to secure access to confidential information while at the same time releases useful information to the collaborating parties.

Several techniques of PPDM have evolved. These approaches can be broadly categorised into randomisation or perturbed privacy preserving techniques, generalisation
Modelling a secure support vector machine classifier for private data

or anonymisation-based privacy preserving techniques and cryptographic privacy preserving techniques. Aggarwal and Philip (2008) propose randomisation techniques such as additive and multiplicative strategies, Bayes and EM reconstruction methods and the use of these methods in privacy preserving association mining, classification and clustering. But the key property of the randomisation method is that the original records are not used after the conversion and data mining algorithms need to use the cumulative distributions of the perturbed data in order to perform the mining process. Agrawal et al. (2006) discuss a symmetric perturbation approach and its reconstruction model that could be used for centralised association mining and classification. Agrawal and Srikant (2000) introduced the concept of perturbation in PPDM. Diverse algorithms are discussed to restructure distributions and learn a decision tree classifier from the perturbed data. Similar approaches of perturbation for privacy preserving association rule mining are converse in Evfimievski et al. (2002), Rizvi and Haritsa (2002) and Zhang et al. (2004). Oliveira and Zaiane (2003) illustrate a privacy preserving clustering approach that uses transformations.

Two main techniques have been proposed for enforcing anonymity on a private table: generalisation and suppression. k-anonymity was first introduced by Sweeney (2002), which required that each record in the anonymised table be indistinguishable with at least \( k - 1 \) other records with respect to quasi-identifier attributes. To solve the problems in k-anonymity approach, Machanavajjhala (2007) proposed \( \ell \)-diversity model, which enhanced the difficulty of linking sensitive values with their corresponding individuals. The diversity of sensitive values in each of equivalence class was also increased. Raymond et al. (2006) proposed simple \((\alpha, k)\)-anonymity model and extended it to general \((\alpha, k)\)-anonymity model which can resist inference attack by controlling frequencies of sensitive values in each equivalence class. Arik et al. (2006) discuss the construction of a decision tree classifier using k-anonymity technique. Some attacks on anonymisation-based privacy preserving are shown in Fung et al. (2010). Mielikainen (2004) also discusses the privacy problems encountered while using the anonymisation method.

Bertino et al. (2008) mentioned that cryptographical approaches provide high level of data privacy compared to the randomisation/anonymisation approach of PPDM. For Distributed systems involved in privacy preserving mining, it has been largely been observed that protecting privacy is by using cryptographic techniques. The hypothetical structure for all cryptographic protocols is secure multiparty computation. Yao (1986) first postulated the two-party comparison problem (Yao’s millionaire protocol) and developed a provably secure solution. This was extended to multiparty computations by Goldreich and Micali (1987). The key result in this field is that any function can be computed securely. However, the generic circuit evaluation technique does not work efficiently for large quantities of data. A detailed description of homomorphism is provided in Yao (1986). These homomorphic properties are used to perform secure computations (Jaideep et al., 2008; Zhang et al., 2004; Bansal et al., 2013). The additive and multiplicative homomorphic properties mentioned in Jaideep et al. (2008) and Zhong and Zang (2011) work well for our techniques. Blum and Goldwasser (1984) describe about an efficient probabilistic public-key encryption that hides all partial information. This property enhances the privacy of our algorithm. A concept of secret sharing in privacy preservation is discussed by Benaloh (1986).
Du and Zhan (2002) and Jaideep et al. (2008) discuss the construction of decision trees from data partitioned between k sites with the Pk holding the class attribute. However, the amount of communication and computations depend on the number of attributes, values of attribute and the complexity of the tree. Here, a secure naïve Bayesian classifier is constructed on the data partitioned on multiple sites. This technique works on vertically partitioned data with attributes categorical or numeric in nature and the class label attribute maintained at a single site. Random secure sum protocol is used to compute the sum. Another interesting privacy preserving classifier (Zhong and Zang, 2011) models a back propagation neural network using the probabilistic partial encryption/decryption property for secure computation of the activation function and product of two integers. On implementation, it is observed that the computation time and accuracy of the protocol largely depends on the number of attributes, number of epochs and number of hidden layers. The algorithm in Zhong and Zang (2011) is modified to include the class attribute in a single site called as master site. As the number of hidden layers increase, the computation time almost doubles making this method of classification comparatively sluggish. It is obvious that as the number of sites increases the training time also increases drastically.

As mentioned in Jaiwei and Micheline (2011), SVM is an efficient classifier for classifying data. The results obtained by these classifiers provide better accuracy compared to other classifiers such as decision trees, naïve Bayesian and artificial neural networks. They are also lesser prone to data overfitting. The concept of Support vectors was introduced by Vapnik (1998) followed by Christianini and Shawe-Taylor (2000) further discussing the different kernel methods. The support vectors provide a compact representation of the trained model. Our work concentrates on building distributed SVM models for non-linear data without revealing the sensitive data. A perturbation approach where data is locally perturbed by each of the parties using Gaussian distribution is used in Fung and Mangasarian (2001), Li et al. (2014), Mangasarian et al. (2008) and Vaidya et al. (2008a). These locally generated reduced kernels are shared among the various parties to obtain a global random reduced kernel. Since the class label attribute is maintained by all, the parties use optimisation methods to obtain the Lagrange multipliers, weights and bias. To predict new data, parties use their data, locally computed random reduced matrix, and share this with the remaining parties. Each party on receiving the other shares is capable of predicting new data. However, it is observed that this method is not suitable for vertically partitioned data where only a subset of the collaborating parties holds the class label attribute, because of which other parties cannot calculate weights and bias for its attributes.

In our protocol, site k is known as the master site as it holds the class attribute. It uses secure kernel function (for nonlinear SVM) to obtain the final data matrix. Master site uses its securely computed data matrix and the class label attribute solves the SMO optimisation problem to obtain the Lagrange multipliers for all the tuples. All tuples with Lagrange multiplier non-zero are support vectors. To obtain the final weights for the attributes a product of the Lagrange multipliers and the class label values \( \sum_{i=1}^{n} \lambda_i y_i \) is forwarded to all the sites. Sites compute the weights of each of its attributes and bias using the data received from the master site. The computed weights of the attributes and bias are forwarded to master site, which classifies a new tuple as positive or negative using the weights.
2 Overview

SVM classifier uses the statistical learning theory and has revealed excellent experimental results in several real-time applications. This method of classification is considered to work very well with high dimensional data and avoids the curse of dimensionality with its distinctive feature of representing the decision boundary through a subset of training samples called support vectors. The support vector machine (SVM) has been developed as a vigorous tool for classification in noisy and complex domains. The two key features of SVMs are generalisation theory, that provides a disciplined way to select a hypothesis and secondly kernel functions, which introduce non-linearity in the hypothesis space without unambiguously requiring a nonlinear algorithm.

2.1 SVM classifier methodology

SVM classifier searches for a hyperplane with the largest margin, which is known as the maximal margin classifier. For a binary classification problem consisting of \( N \) training examples, where each example is denoted by a tuple \((x_i, y_i)\) \((i = 1, 2, \ldots, N)\), where \(x_i = (x_{i1}, x_{i2}, \ldots, x_{id})\), \(x_{ij}\) is the \(j\)th attribute of the \(i\)th tuple and \(y_i \in \{ 1, -1 \}\) which denotes the class label. The decision boundary of SVM classifier can be written in the form \(wx + b = 0\), where \(w\) and \(b\) are parameters of the model. Learning the SVM model can be formalised as the following constrained optimisation problem:

\[
\min_w \frac{\|w\|^2}{2},
\]

subject to

\[y_i (wx_i + b) \geq 0, \text{ for } i = 1, 2, 3, \ldots, N\]

This objective function is quadratic and the constraints are linear for the parameters \(w\) and \(b\), hence this is known as a convex optimisation problem which can be solved using the Lagrange multiplier method. In order to proceed to the non-separable and nonlinear cases it is useful to consider the dual problem. Rewriting the objective function that follows the constraints on the solutions results in a new objective function known as the Lagrangian for the optimisation problem given as:

\[L_p = 0.5 \text{ square} \|w\| + \sum_{i=1}^{N} \lambda_i (y_i (w \cdot x_i + b) - 1)\]

where \(\lambda_i\) is called Lagrange multiplier and \(\text{square} \|x\| = x^2\).

Differentiating \(L_p\) with respect to \(w\) and \(b\) we obtain and set them to zero:

\[\frac{dl_p}{dw} = 0\] which gives \(w = \sum_{i=1}^{N} \lambda_i y_i x_i\)

\[\frac{dl_p}{db} = 0\] to give \(\sum_{i=1}^{N} \lambda_i y_i = 0\).
Solving the above optimisation problem is difficult because of the large number of parameters, i.e., $w$, $b$ and $\lambda$. This problem is resolved by transforming the Lagrangian into a function of the Lagrange multipliers only, which leads to the dual formation of the optimisation problem.

$$\text{Max } W(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j K(x_i, x_j), \quad 0 \leq \lambda_i \leq C$$

$$\sum_{i=1}^{N} y_i \lambda_i = 0.$$

The quadratic programming (QP) problem can be solved using the extended SMO algorithm suggested by Keerthi and Gilbert (2002) which is an improvement over Platt (1998) to further improve the training time. $K(x_i, x_j)$ is the radial basis function or any other kernel function for nonlinear classifiers. This computation is the only position where the training tuples would be referred. The values $y_i$ and $y_j$ are the class labels of the training tuples $x_i$ and $x_j$. The radial basis function of tuples $x_i$ and $x_j$,

$$K(x_i, x_j) = \exp \left( -\frac{\|x_i - x_j\|^2}{2 \sigma^2} \right)$$

where $\sigma$ is a free parameter.

Once the Lagrange multipliers are obtained, a tuple with $\lambda_i$ not equal to 0 are support vectors. These support vectors assist in computing the weights for each of the features and the final bias.

2.2 Extended sequential minimal optimisation

Extended sequential minimal optimisation (SMO) is a simple algorithm that solves the SVM QP problem without any extra matrix storage and without invoking an iterative numerical routine for each sub-problem. It decomposes the overall QP problem into QP sub-problems. At every step, this SMO chooses two Lagrange multipliers to jointly optimise and obtain an optimal value for these multipliers. These multipliers update the SVM to reflect the new optimal values. The components of this algorithm are to identify a logical method to solve for two Lagrange multipliers, a heuristic for choosing which multipliers to optimise and a method for choosing two threshold $b_{\text{high}}$ and $b_{\text{low}}$. Extended SMO is considered to be fast for SVM’s and sparse datasets.

2.3 Secure multiparty computation

Secure multiparty computations enable parties to perform distributed computing tasks in a secure manner. The essential requirements on any secure computation protocol are privacy and correctness. The privacy requirement states that parties should learn their output and nothing else. The correctness requirement states that each party should receive its correct output. To incorporate this, the adversary must not be able to cause the result of the computation to diverge from the function that the parties have set out to compute. The basic building blocks or primitives of secure multiparty computations are oblivious transfer, oblivious polynomial evaluation and homomorphic encryption. A homomorphic encryption scheme is an encryption scheme which allocates certain operations to be
carried out on the encrypted plaintext by applying an efficient operation to the corresponding ciphertext. In addition, most of encryption scheme which have this property are semantically secure.

3 Privacy preserving homomorphic support vector machine classification

Privacy preserving support vector machine classifier performs supervised learning on the data distributed at $k$ sites. Each of the sites, 1 to $k$ has a subset of features required for learning. Site $k$ is called as the training party as it holds the class attribute as well as some of the features required for training. The number of tuples ($T$) used for model building is same for all the sites. Figure 1, shows the flow diagram of the privacy preserving homomorphic support vector machine (PPHSVM) modelling. Data preprocessing is performed on the datasets are conversion of categorical to numeric and handling missing values. The master party $k$ that holds the class attribute initiates the classification process building by building a data matrix securely. It further uses the symmetric data matrix to generate Lagrange multipliers. Once the weights and bias for all the attributes are computed they are forwarded to master party which can use these weights and biases to classify new tuples. Algorithm 1 explains the working of the proposed privacy preserving SVM classifier. The secure kernel function computation is elaborated in Algorithm 2.

The privacy preserving nonlinear SVM approach privately computes the kernel function, radial basis function as shown in Figure 2. For secure computation, party $k$ holding the class attribute, Paillier encrypts its locally computed kernel value to obtain $E(v_k)$. It further transmits $E(v_k)$ to party 1 which performs $\text{pow}(E(v_k), v_1)$ where $v_1$ is the locally computed kernel value in party 1.

Figure 1 Flow diagram of privacy preserving nonlinear SVM classifiers

This party passes on the result obtained to its neighbouring party which repeats the process of evaluation $\text{pow}(\text{pow}(E(v_1), v_1), v_2)$ and forwards it to its neighbor. The $k-1$ party on computing $\text{pow}(\text{pow}(\text{pow}(E(v_1), v_1), v_2), v_3) \ldots, v_{k-1})$ forwards this result to the initiating party $k$. Party $k$ decrypts the value to obtain the final kernel value. Once the data matrix is obtained having the kernel function values, support vectors are identified.
Figure 2 Secure multiparty computation of kernel function from one tuple \((x)\) to another tuple \((x')\)

Algorithm 1: Privacy preserving homomorphic nonlinear support vector machine classification

**Input:** \(K\) parties, \(K^{th}\) party is the training (master) party

**Output:** Weights of each of the features/attributes and bias

**Procedure:**

**Begin**

Party \(k\) performs the following steps:

**Step 1**
1.1 Generate the keys for Paillier encryption private keys \((\lambda,\mu)\) and public keys \((n, g)\)
1.2 Broadcast the public key to all the parties.
1.3 Obtains the kernel matrix of size \((T\times(T-1))/2\) where \(T\) is the number of tuples in each party.

\[ k = 0 \]

\[ \text{for } i = 1 \text{ to } T \]

\[ \text{for } j = (i + 1) \text{ to } T \]

\[ \text{begin} \]

\[ \text{compute sym\_mat}[k++] = f(x_i, x_j) \text{ using multiparty secure kernel function protocol} \]

\[ \text{end} \]

**Step 2**

With the vector \(\text{sym\_mat}\) having \(f(x_i, x_j)\) generate the symmetric data matrix as \(f(x_i, x_j) = f(x_j, x_i)\).

Using the data matrix, a class value vector \(Y\), of size \(T\), the Lagrange multipliers vector \(\lambda\) for each of the tuples \(i\) is obtained using the extended SMO convex quadratic programming algorithm.

**Step 3**

3.1 Locally compute the vector \(\text{partial\_prod}(\lambda Y)\) as follows

\[ \text{for } i = 1 \text{ to } T \]

\[ \text{partial\_prodi} = \lambda_i Y_i \]

3.2 The vector \(\text{partial\_prod}(\lambda Y)\) is broadcasted to remaining parties.
Modelling a secure support vector machine classifier for private data

Step 4

4.1 The remaining parties on receiving the partial_prod from master site compute the weight and bias for each of its attributes as follows

Party $\ell$ where $\ell = 1$ to $k - 1$

Receive the vector $\text{partial}_\text{prod}(\lambda Y)$

for $i = 1$ to $p$, $p$ is the number of attributes in party $\ell$

$\text{weight}_i = \sum_{j=1}^{T} (x_{ij} \ast \text{partial}_\text{prod}(j))$, where $x_{ij}$ is the value of the attribute $i$ for record $j$.

for $h = 1$ to $T$

compute $\text{bias}_h = 1 - \sum_{i=1}^{p} (\text{weight}_i \ast x_{ih})$, where $x_{ih}$ is the value of the attribute $i$ for record $h$.

$\text{Bestbias}^\ell = \sum_{j=1}^{T} \text{bias}_j / T$

4.2 Circulate the weight vector, $\text{weight}^\ell$ (that indicates weight of all the attributes) and the $\text{Bestbias}^\ell$ to party $k$.

Step 5

5.1 Party $k$ also computes the weight and bias for each of its attributes as per Step 4.1.

$\text{Bestbias}^k = \sum_{j=1}^{T} \text{bias}_j / T$

5.2 Party $k$ receives the weight vector and $\text{Bestbias}$ from all the parties.

5.3 The final_bias is computed as $\sum_{j=1}^{k} \text{Bestbias}_j / k$

To classify a new tuple based on the learned SVM model

Given a tuple $x^T = (x_1, x_2, \ldots, x_n)$ with features $x_1, x_2, \ldots, x_n$

We obtain $\sum_{i=1}^{n} \text{weight}_i \ast x_i + \text{final_bias}$.

Class value is decided based on the sign of the result. As party $k$ has the weights of all attributes involved in the computation and bias the new tuple can be easily classified.

End

3.1 Multi-party secure computation of kernel function

Considering that $x$ and $x'$ are tuples represented as features vectors. The kernel function

$$F(x, x') = \exp\left[-\left(\sum (x_i - x'_i)^2 + (x_2 - x'_2)^2 + (x_3 - x'_3)^2 + \ldots + (x_n - x'_n)^2\right)/(2 \cdot \sigma^2)\right],$$

where $x_i$ and $x'_i$ are feature/attribute values of samples/tuples $x$ and $x'$ distributed at multiple sites. This function has to be computed in a distributed environment as the features/attributes are distributed at multiple sites.
Algorithm 2: Multiparty secure computation of kernel function from tuple \( x \) and \( x' \) whose features are distributed across \( K \) parties.

**Input:** \( K \) parties. Each party has their attributes. \( K \)th party is the training (master) party.

**Output:** Kernel function for all the distributed attributes \( F(x, x') \) from tuple \( x \) to \( x' \)

**Procedure:**

**Begin**

**Party \( k \)**

1. Computes \( E_{x_k} = \exp(-[(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \ldots + (x_m - x'_m)^2]/(2 * \sigma^2)) \) for all its features (1 to \( m \)) except for the class label attribute.
2. It encrypts its value \( E_{x_k} \) to obtain \( E(E_{x_k}) \) and forwards this to party 1.

**Party 1**

1. If party 1, receives the encrypted value \( E(E_{x_k}) \) from party \( k \).
2. Computes \( E_{x_1} = \exp(-[(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \ldots + (x_p - x'_p)^2]/(2 * \sigma^2)) \) for all its features (1 to \( p \)).
3. It then performs \( E(\text{Encrypt \_prod}(x, x')) = E(E_{x_1})^{\text{Encrypt \_prod}} \) mod \( n^2 \), and forwards this to its immediate next party, i.e., party 2.

**Party \( I \) \((I = 2 \text{ to } k - 1)\)**

1. Receives the value \( \text{Encrypt \_prod}(x, x') \) from its \( I-1 \)-th neighbour.
2. Computes \( E_{x_I} = \exp(-[(x_I - x'_I)^2 + (x_2 - x'_2)^2 + \ldots + (x_p - x'_p)^2]/(2 * \sigma^2)) \) for all its features (1 to \( q \)).
3. It then updates \( \text{Encrypt \_prod}(x, x') = \text{Encrypt \_prod}(x, x')^{\text{Encrypt \_prod}} \) mod \( n^2 \), and forwards this to its \( I+1 \)-th neighbour.

**Party \( k - 1 \)** forwards its \( \text{Encrypt \_prod}(x, x') \) to party \( k \). Party \( k \) obtains \( F(x, x') \) by decrypting \( \text{Encrypt \_prod}(x, x') \).

**End**

Method used is \( \exp(x_1 + x_2 + x_3 + \ldots + x_i) = \exp^{x_1} * \exp^{x_2} * \exp^{x_3} * \ldots * \exp^{x_i} \), \( \exp^{x_i} \) is computed by all parties for \( i = 1 \) to \( k \).

To securely perform this computation, homomorphic property of Paillier is used. The whole process of kernel computation starts from party \( k \) which has the class attribute. Party 1 to \( k \) locally compute the \( F(x, x') \) for each of its features. Party \( k \) encrypts it \( F_i(x, x') \) to get \( E(F_i(x, x')) \) and sends it to party 1. Party 1 performs \( E(F_1(x, x'))F_1(x, x') \) and forwards it to party 2 which computes \( (E(F_1(x, x'))F_1(x, x'))F_2(x, x') \) and sends it to the next party which repeats the process. Party \( k - 1 \) then forwards \( E(F_1(x, x'))F_1(x, x')) \) \( F_2(x, x'))F_3(x, x')) \ldots \) \( F_{k-1}(x, x')) \) to \( k \)-th party. Party \( k \) on decryption obtains

\[
F(x, x') = F_1(x, x') * F_2(x, x') * \ldots * F_{k-1}(x, x') * F_k(x, x')
\]

where \( x \) and \( x' \) are tuples with distributed attributes in sites 1 to \( k \). Figure 2 shows the working of the secure kernel function computation. The master party \( k \) initiates the kernel function computation by forwarding the \( \text{Encrypt \_prod} \) to its neighbour party 1.
3.2 Paillier cryptosystem

As per (Paillier, 1999) the following has been implemented.

- **Key generation**
  
  Randomly generate two large prime numbers $p$ and $q$, with the specified bitLength and certainty.
  
  Compute $\lambda = (p - 1) * (q - 1) / \gcd(p - 1, q - 1)$. 
  
  Obtain $n = p * q$ and $\text{nsquare} = n^2$.
  
  Generate a random integer $g \in Z_{\text{nsquare}}$, such that $\gcd(L(g^\lambda \mod \text{nsquare}), n) = 1$ where $L(u) = \frac{u - 1}{n}$.

- **Encryption**

  This function accepts plaintext $m$ and encrypts it to obtain a ciphertext $c = g^m * r^e \mod \text{nsquare}$.

- **Decryption**

  This function accepts a ciphertext $c$ as input and obtains a plaintext $m = L(c^e \mod \text{nsquare}) * u \mod n$,
  
  where $u = (L(g^\lambda \mod \text{nsquare}))^{(-1)} \mod n$.

3.3 Homomorphic property

An important property of the Paillier cryptosystem (Paillier, 1999) assists in performing definite types of computations on the ciphertext and produce an encrypted result which when decrypted equates the outcome of an operation carried out on the plaintext.

The two main homomorphic properties used are:

1. **Homomorphic addition**

   Let $E(v_i)$ indicate the encryption of a plaintext $v_i$ and $D(v_i)$ indicate decryption of value $v_i$ then,
   
   $D(E(v_1) * E(v_2) * E(v_3) * \ldots * E(v_n) \mod \text{nsquare}) = (v_1 + v_2 + v_3 + \ldots + v_n) \mod n$.

   In our case the values $v_1, v_2, \ldots, v_n$ are the locally computed values at each site. This property can be used to securely compute the dot product.

2. **Homomorphic multiplication**

   Let $E(v_i)$ indicate the encryption of a plaintext $v_i$ and $D(v_i)$ indicate decryption of value $v_i$ then,
   
   $D(\left(\left(\left(\left(E(v_1) \cdot v_2\right) \cdot v_3\right) \ldots v_n\right)\mod \text{nsquare}\right) = (v_1 \cdot v_2 \cdot v_3 \times \ldots \cdot v_n) \mod n$
In our case the values $v_1, v_2, \ldots, v_n$ are the locally computed values at each site.

This property is used to securely compute the kernel function.

4 Result analysis

The following evaluations have been done after performing for data mining on vertically partitioned data maintained at three sites. For the hospital dataset maintained at site 3 and the remaining sites 1 insurance detail and 2 holds the personal details. Only five attributes of the insurance dataset (number of car policies, number of accident insurance policies, average income, number of dependents, profession) are identified as sensitive. The sensitive attributes considered in the personal dataset are religion, education, total number of policies, and number of houses and region of residence. In the hospital dataset the recognised sensitive attributes are type of disease, type of treatment taken and the class label attribute success. For PPHSVM, all the attributes maintained in multiple parties are numeric in nature with the binary class label having values 1 or –1. Hence an essential prerequisite to this approach involves conversion of categorical values to numeric. Whereas in PPANN (Zhong and Zang, 2011) too all the attributes are numeric in nature with the class label value 0 or 1. Hence, an essential prerequisite to this approach also involves conversion of categorical values to numeric. For PPNBC (Vaidya et al., 2008b), the attributes are either numeric or categorical in nature. The data preprocessing performed on this dataset is approximation of the missing values.

The number of tuples in all the three sites is the similar and the datasets are built assuming that attributes age and name are known to all sites. This project is implemented in JAVA on eclipse IDE. The configuration of the systems where the modules have been implemented and executed includes Intel® Core™ i5 CPU M430 @2.27GHz, 3 GB RAM and 64-bit operating system.

Figure 3 Computation time of the PPHSVM with the other privacy classifiers such as PPNBC and PPANN (see online version for colours)

4.1 Computation time with nonlinear SVM classifier

Nonlinear PPHSVM classifier takes lesser execution time in model building compared to the other classifiers. The number of sensitive attributes in each participating sites do not
affect the computation time of the classifier. However as observed in Figure 3, the number of tuples influence the computation. It is relatively better than PPNBC and PPANN as shown in Figure 3.

Figure 4 shows that PPHSVM classifier is more accurate than other PPDM classifiers built using naive Bayesian approach and artificial neural network approach. The loss of the accuracy in the above privacy preserving classifiers is due to the data conversions performed before encrypting the values. Since the quantity of conversions is more in PPNBC and PPANN compared to PPHSVM, our classifier is comparably more accurate.

Another important measure for identifying the accuracy is the ROC curve. As observed in Figure 5, the probability of classifying a positive correctly as positive is highest with the PPHSVM model. Hence PPHSVM is an accurate classifier compared to the benchmark algorithms.

### Figure 4
Accuracy of PPHSVM over benchmark privacy preserving classifiers (see online version for colours)

4.2 Privacy preservation

The amount of the privacy in PPHSVM is also greater compared to PPNBC or PPANN approaches. PPNBC uses homomorphic encryption and random shares are maintained by the parties but calculation of mean and variance of each numeric attribute is complicated and involves larger computation time. The computation time of PPNBC largely depends on the number of attributes for modelling. PPANN uses the partial encryption and decryption approach of ElGamal to perform secure sigmoid and product computations. A range of values decisive by the server is encrypted for computing a single value. Privacy is limited as the server is aware that the expected value is within a particular range. Whereas in PPHSVM it can be observed that privacy of the data is retained without data leakage during computation or in any phase of model building. The encrypted data circulated between sites is semantically secure and hence identifying the data value of the sensitive attributes is almost impossible.
4.3 Security analysis

The proposed privacy preserving support vector machine classifier generates a model by not revealing the values of the private data at any point of classification process. The features and their values in a participating site are unknown to the others. In the process of model building, the local kernel computations are performed on the actual data, followed by global evaluations of the locally obtained results in their encrypted form. As Paillier cryptosystem is used, interpreting any kind of information from the data is not viable.

Also, as only the master site holds the private key it alone has the rights to decrypt the data. In order to hide the class label values (indicated as 0’s and 1’s) across each instance in the master site, they are encrypted and forwarded to the remaining participating sites. As the approach used is semantically secure the sites cannot deduce any information from the encrypted class label values, but can securely compute the essential support vectors that are essential for computing the weights of the features. Due to the above-mentioned inferences, it can be observed that the proposed PPHSVM approach is highly secure.

5 Conclusions

Data is often distributed and has to be mined collaboratively to identify useful information without disclosing sensitive information. PPDM essentially interprets the data without unveiling any sensitive information about the data. SVMs, are supervised learning models used for classification to analyse data and identify patterns. It gives empirically good performance compared to the other classifiers and accurate results. A privacy preserving SVM classifier for data that is vertically partitioned in multiple sites is built. A homomorphic approach of Paillier’s is used to enable secure computations such as kernel-RBF computation for nonlinear, while building a classification model. Based on the results obtained from the experiments, it is observed that the computation speeds of PPHSVM nonlinear classifiers are efficient.
This approach is also successful in retaining the privacy of the data. The classifications of test tuples on the model build provide better accuracies comparable to other privacy preserving classifiers. Our approach is also scalable to the increase in the number of attributes in any of the sites. Further, we plan to handle malicious adversaries during the construction of PPDM classifiers.

References


