Privacy Preserving PCA for Multiparty Modeling

Yingting Liu[1,2] , Chaochao Chen[2], Longfei Zheng[2], Li Wang[2], Jun Zhou[2], Guiquan Liu[1]*
[1]University of Science and Technology of China
[2]Ant Financial Services Group
sa517218@mail.ustc.edu.cn, gqliu@ustc.edu.cn
{chaochao.ccc, zlf206411, raymond.wang, jun.zhoujun}@antfin.com

Abstract

In this paper, we present a general multiparty modeling paradigm with Privacy Preserving Principal Component Analysis (PPPCA) for horizontally partitioned data. PPPCA can accomplish multi-party cooperative execution of PCA under the premise of keeping plaintext data locally. We also propose implementations using two techniques, i.e., homomorphic encryption and secret sharing. The output of PPPCA can be sent directly to data consumer to build any machine learning models. We conduct experiments on three UCI benchmark datasets and a real-world fraud detection dataset. Results show that the accuracy of the model built upon PPPCA is the same as the model with PCA that is built based on centralized plaintext data.

Introduction

Although big data concepts and technologies have become popular, a single organization is likely to have limited data, which makes it difficult to meet the requirements of machine learning models. Meanwhile, different organizations are often unwilling or unable to share data because of the competition or privacy regulations. *Isolated data islands* situation has become a serious problem weakening the performance of artificial intelligence in the real world.

Existing researches adopt cryptographic techniques, e.g., homomorphic encryption (Hardy et al. 2017) or secure Multi-Party Computation (MPC) (Mohassel and Zhang 2017), to solve these problems. However, they have two weaknesses. First, their running time is much longer than the models built on aggregated data due to high computation complexity and communication complexity of the cryptographic techniques. It gets even worse in practice because of the large scale data. Second, they are usually model-specific, i.e., the models need to be built case by case. However, in real-world applications such as fraud detection and risk control, machine learning models usually need to be rebuilt periodically. If a general modeling or preprocessing method can be designed, the deployment of privacy preserving machine learning models can be greatly accelerated and the development costs can be significantly reduced.

*Guiquan Liu is the corresponding author.

Furthermore, Principal Component Analysis (PCA) is popularly used to transform the original data from one space to another (Pearson 1901). Under isolated data island situation, how to perform PCA becomes a hot research problem. For example, Al-Rubaie *et al* (Al-Rubaie et al. 2017) proposed to use additive homomorphic encryption and garbled circuits for PCA with a single data user. With the increase in feature dimension, the computing time increases rapidly. Grammenos *et al* (Grammenos et al. 2019) adaptively estimated the rank of PCA with unknown distribution in a streaming setting. Liu *et al* (Liu et al. 2018) proposed a huge digital data outsourcing scheme. However, none of them focus on building privacy preserving PCA for multiparty modeling.

For these reasons, we propose a general modeling paradigm with PPPCA for horizontally partitioned data. It involves two methods using homomorphic encryption and secret sharing respectively. Results of PCA can protect data privacy to a certain extent, and can be further improved by using differential privacy (Wei et al. 2016). Under the honest-but-curious setting, the results of PPPCA can be sent directly to any data consumer to build any machine learning models. Our contributions are two-folds:

- We propose a privacy preserving modeling framework using PPPCA, where the plaintext data are hold by participants locally. The result of PPPCA can be carried out for building any machine learning models, which significantly improves the generality and efficiency compared with the models built using time-consuming cryptographical techniques case by case.
- We propose two PPPCA approaches using secret sharing and homomorphic encryption respectively. Experiments on three UCI benchmark datasets and a real-world fraud detection dataset demonstrate that our approaches are lossless comparing with the approach that aggregates plaintext data.

Preliminaries

In this section, we present problem statement and some preliminary concepts of our proposal, including secret sharing and homomorphic encryption.

Problem Statement

Data Setting. Data provided by different participants are partitioned horizontally in the data space. That is, datasets share the same feature space but different sample space.

Input. The plaintext data that are provided by M ($M > 1, M \in Z^+$) participants. We use $\boldsymbol{X_i}$ ($1 \leq i \leq M$) to denote the corresponding plaintext data of i-th participant, where each column represents an attribute, and each raw represents a sample. In horizontally partitioned situation, the joint data can be formalized as $\boldsymbol{X} = (\ \boldsymbol{x_1}^T\ \boldsymbol{x_2}^T\ \cdots\ \boldsymbol{x_M}^T\ )^T$. We use d_i and n_i to denote the feature dimension and the sample dimension of participant i, and use d and n to denote the feature dimension and sample dimension of the joint data, where $n = \sum_{i=1}^{M} n_i, d = d_1 = \cdots = d_M$.

Output. The dimension reduced matrix X_i' from each participant whose dimension is $n_i \times k$ ($k < d$). It can be carried out for building any machine learning models.

Constraint. The lossless & privacy preserving ability. The performance of PPPCA must be comparable to the non-private solution that brings all data in one place, while protecting the data privacy.

Additive Homomorphic Encryption

Additive homomorphic encryption, e.g., paillier (Paillier 1999), is a method that supports secure addition of numbers given only the encryptions of them. The ciphertext operation on untrusted participant is secure. The result of the operation needs to be sent to the private key holder for decryption to get the plaintext result. We denote the ciphertext of a number u as $[\![u]\!]$ and overload '$+$' as a homomorphic addition operation on ciphertext space. Additive homomorphic encryption satisfy that for any plaintext u and v,

$$[\![u]\!] + [\![v]\!] = [\![u + v]\!]. \tag{1}$$

However, plaintext in this arithmetic must be the element of finite field. In order to support floating point operations, we adopt the encoding scheme (Hardy et al. 2017). This approach is based on a floating-point representation, and ultimately encode a number as a pair consisting of an encrypted significant and an unencrypted exponent. Moreover, these operations can be extended to work with vectors and matrices component-wise.

Secret Sharing

Secret sharing is a technique of secure Multi-Party Computation (MPC) (Mohassel and Zhang 2017). In our case, all data providers participate in the calculation together, and the final result is reconstructed from their calculation results. Therefore, we choose n-out-of-n secret sharing (Singh et al. 2017), which is an efficient scheme.

In this paper, we mainly focus on additive secret sharing. To additively share l-bit secret value s owned by party M, party M first generates M shares $\{r_1, \cdots, r_M\}$, where $r_i \in \mathbb{Z}_{2^l}$ ($1 \leq i \leq M-1$) are random numbers generated using Pseudo-Random Generator (PRG) and the last share $r_M = s - \sum_{i=1}^{M-1} r_i \bmod 2^l$. Then party M distributes these shares to M data providers (including itself). All intermediate values are secret-shared between data providers. We denote the share owned by party i as $\langle s \rangle_i$. Given secret values $\{s_1, \cdots, s_M\}$ from M data providers, to calculate the sum of them, each data provider needs to calculate the sum of their local shares (e.g., party i calculates intermediate values $\langle v_i \rangle = \sum_{j=1}^{M} \langle s_j \rangle_i \bmod 2^l$). Finally, the result is the sum of these local results. During this process, all participants are invisible to others' inputs.

The above secret sharing schema works in finite field, and can be extended to real number field following the encode and denote scheme in (Hardy et al. 2017). It is trivial to generalize the single number operations to matrices.

The Proposed Method

In this section, we propose PPPCA framework and its two implementations.

Overview

PPPCA aims to perform private PCA for multiple data providers under server-aided setting. It refers to three roles: data provider, server, and data consumer. Data provider owns the data and participates in the calculation. Server does not have any input, participates in some computation but does not receive any privacy output. It can be untrusted but cannot collude with data providers. Data consumer use the result of PPPCA for further modeling. It takes three steps to execute PPPCA under horizontally partitioned case:

- Firstly, all data providers jointly normalize the columns (attributes) to zero with secure operation. For example, each value of attribute t owned by different data providers should be divided by the mean of them, which is denoted as $\overline{x_t} = \frac{\sum_{i=1}^{M} s_{it}}{n} = \frac{\sum_{i=1}^{M} \sum_{j \in I_i} x_{jt}}{\sum_{i=1}^{M} n_i}$, where s_{it} is the sum of observations of feature t owned by data provider i and can be computed locally. $\overline{x_t}$ can be computed by all participants using secure additive operation.
- Secondly, all data providers jointly compute the the covariance matrix $\boldsymbol{C}$:

$$\begin{aligned}\boldsymbol{C} &= \frac{1}{n-1}\boldsymbol{X}^T\boldsymbol{X} \\ &= \frac{1}{n-1}\begin{pmatrix}\boldsymbol{X_1}^T & \boldsymbol{X_2}^T & \cdots & \boldsymbol{X_M}^T\end{pmatrix}\begin{pmatrix}\boldsymbol{X_1} \\ \boldsymbol{X_2} \\ \vdots \\ \boldsymbol{X_M}\end{pmatrix} \\ &= \frac{1}{n-1}(\boldsymbol{X_1^T X_1} + \boldsymbol{X_2^T X_2} + \cdots + \boldsymbol{X_M^T X_M}),\end{aligned} \tag{2}$$

  where $\boldsymbol{X_i^2} = \boldsymbol{X_i}^T\boldsymbol{X_i}$ can be computed locally. Hence, $\boldsymbol{C}$ can be computed by all data providers and server collaboratively using secure additive operation. At the end of this step, the plaintext of $\boldsymbol{C}$ is owned by server.
- Thirdly, server calculates the eigenvalues and corresponding eigenvectors of $\boldsymbol{C}$ locally, chooses k ($k < d$) eigenvectors to form the matrix $\boldsymbol{T}$ according to the size of the

Algorithm 1: HE based PPPCA framework

Input : Data $\{\boldsymbol{X_1}, \cdots, \boldsymbol{X_M}\}$; dimension after PCA (k)
Output: Dimension reduced matrix $\boldsymbol{X'}$

1 Server generates a key pair and sends public key to all data providers;
2 Data providers compute and encrypt the mean of their features locally as $[\![\mathbf{S_i}]\!]$ and send it to party p;
3 Party p receives $\{[\![\mathbf{S_1}]\!], \cdots, [\![\mathbf{S_M}]\!]\}$, computes $[\![\overline{x}]\!] = \sum_{i=1}^{M} [\![\mathbf{S_i}]\!]$, and sends it to server;
4 Server receives $[\![\overline{x}]\!]$, decrypts it, and sends it to all data providers;
5 Data providers receive $\overline{x}$ and normalize columns to zero locally;
6 Data providers compute and encrypt $[\![\boldsymbol{C_i}]\!] = [\![\frac{1}{n-1}\boldsymbol{X_i}^T\boldsymbol{X_i}]\!]$ locally and send it to party p;
7 Party p receives $\{[\![\mathbf{C_1}]\!], \cdots, [\![\mathbf{C_M}]\!]\}$, computes $[\![\boldsymbol{C}]\!] = \sum_{i=1}^{M} [\![\mathbf{C_i}]\!]$, and sends it to server;
8 Server decrypts $[\![\boldsymbol{C}]\!]$, computes transfer matrix $\boldsymbol{T}$, then sends it to all data providers;
9 Data providers receive matrix $\boldsymbol{T}$, calculate $\boldsymbol{X_i'} = \boldsymbol{X_i}\boldsymbol{T}$ locally, then send it to data consumer;
10 Data consumer receives dimension reduced matrix $\boldsymbol{X'} = (\boldsymbol{X_1'}^{\boldsymbol{T}}, \cdots, \boldsymbol{X_M'}^{T})^T$ for modeling;
11 **return** *Dimension reduced matrix* $\boldsymbol{X'}$;

eigenvalues, and broadcasts it to data providers. Each data provider calculates $\boldsymbol{X_i'} = \boldsymbol{X_i}\boldsymbol{T}$ locally and sends it to data consumer for further modeling.

Step 1 and step 2 will use secure operation, which can be boiled down to secure matrix addition operations. We will present how to implement these two steps with homomorphic encryption and secret sharing.

Additive Homomorphic Encryption Based Method

Secure additive operation is done by Homomorphic Encryption (HE). The server is responsible for generating a key pair, sharing public key with all participants, and holding the private key for decryption. For secure matrix additions in step 1 and 2, all data providers firstly encrypt matrices that need to be added (e.g., $\boldsymbol{X_i^2}$) and send them to one of the data provider (denoted as p, $1 \leq p \leq M-1$). Party p receives these encrypted matrices and does the homomorphic additive operation in ciphertext space. Finally, party p sends the result to server for decryption. We present the whole HE based PPPCA framework in Algorithm 1.

Secret Sharing Based Method

In this method, secure additive operation is done by secret sharing. In the last step of secret sharing, all of the data providers will send their local calculation results on shares to server. Then server computes the sum of these shares to reconstruct the plaintext result. We present the whole Secret Sharing (SS) based PPPCA framework in Algorithm 2.

Algorithm 2: SS based PPPCA framework

Input : Data $\{\boldsymbol{X_1}, \cdots, \boldsymbol{X_M}\}$; dimension after PCA (k); plaintext space $\mathbb{Z}_{2^l}$
Output: Dimension reduced matrix $\boldsymbol{X'}$

1 Data providers compute features' mean $\mathbf{S_i}$ locally, generate M shares $\{\langle\mathbf{S_i}\rangle_1, \cdots, \langle\mathbf{S_i}\rangle_M\}$, and distribute them;
2 Data providers receive $\{\langle\mathbf{S_1}\rangle_i, \cdots, \langle\mathbf{S_M}\rangle_i\}$ from each other, compute $\langle\mathbf{V_i}\rangle = \sum_{j=1}^{M}\langle\mathbf{S_j}\rangle_i \bmod 2^l$, and send it to server ;
3 Sever receives $\{\langle\mathbf{V_1}\rangle, \cdots, \langle\mathbf{V_M}\rangle\}$, computes $\overline{x} = \sum_{i=1}^{M}\langle\mathbf{V_i}\rangle \bmod 2^l$, and sends it to all data providers;
4 Data providers receive $\overline{x}$ and normalize data columns to zero locally;
5 Data providers compute $\boldsymbol{C_i} = \frac{1}{n-1}\boldsymbol{X_i}^T\boldsymbol{X_i}$, generate M shares $\{\langle\mathbf{C_i}\rangle_1, \cdots, \langle\mathbf{C_i}\rangle_M\}$, and distribute them;
6 Data providers receive $\{\langle\mathbf{C_1}\rangle_i, \cdots, \langle\mathbf{C_M}\rangle_i\}$ from each other, compute $\langle\mathbf{V_i'}\rangle = \sum_{j=1}^{M}\langle\mathbf{C_j}\rangle_i \bmod 2^l$ locally, and send it to sever;
7 Server receives $\{\langle\mathbf{V_1'}\rangle, \cdots, \langle\mathbf{V_M'}\rangle\}$ and computes global covariance matrix $\mathbf{C} = \sum_{i=1}^{M}\langle\mathbf{V_i'}\rangle \bmod 2^l$;
8 Server computes eigenvectors, chooses k of them to form the matrix $\boldsymbol{T}$, and sends it to data providers;
9 Data providers receive matrix $\boldsymbol{T}$, calculate $\boldsymbol{X_i'} = \boldsymbol{X_i}\boldsymbol{T}$ locally, and send it to data consumer;
10 Data consumer receives dimension reduced matrix $\boldsymbol{X'} = (\boldsymbol{X_1'}^{\boldsymbol{T}}, \cdots, \boldsymbol{X_M'}^{T})^T$ for modeling;
11 **return** *dimension reduced matrix* $\boldsymbol{X'}$;

Security Discussion

Our protocols are secure against honest-but-curious adversaries. That is, data providers, server, and data consumer will strictly follow the protocol, but they will keep all intermediate computation results and try to infer as much information as possible. We also assume that the server does not collude with any data providers.

In PPPCA, data providers only receive the mean of each feature which does not contain the details of each feature. As for data provider p in HE based method, all it receives from others are encrypted matrices. The server only has the covariance matrix which does not contain private information neither. The data consumer gets the result of PPPCA, which can not infer private raw data. Therefore, the security of our proposed PPPCA framework is satisfied.

Experimental Studies

In this section, we focus on answering the following research questions. Q1: is the accuracy of the model built upon PPPCA the same as the model with PCA? Q2: what is the difference of efficiency between homomorphic encryption based method and secret sharing based method?

Table 1: AUC and RMSE results of logistic regression and linear regression on different datasets

Dataset	PCA (Centralized)	PCA (Separately)	PPPCA	
Secure Op	-	-	HE	SS
Fraud	0.9663	0.8978	0.9692	0.9613
APS	0.9915	0.9863	0.9918	0.9906
Wine	0.7122	0.7498	0.7103	0.7126
Online	8.0484	8.0801	8.0517	8.0493

Experimental Setup

We conduct experiments on 4 datasets, including one real-world fraud detection dataset (Fraud) (Group April 25 2018) and 3 public datasets from UCI, i.e., APS Failure dataset (APS) (Tony Lindgren September 2016), Wine-quality dataset (Wine) (Cortez et al. 1998), and Online News Popularity dataset (Online) (Kelwin Fernandes January 8 2015). The first 2 datasets are for classification problems and the others are for regression tasks. The fraud detection dataset has 298 features and 792,004 transactions. We focus on horizontally partitioned case, therefore we assume these data are hold by different parties and each of them has roughly equal partial samples.

We use five-fold cross validation during experiments. After PPPCA, we adopt the most popular three kinds of models, i.e., linear model, tree based ensemble model, and deep model for classification and regression problem. Note that it can be generalized to any other machine learning models. We adopt Area Under the receiver operating characteristic Curve (AUC) as the evaluation metric for binary classification tasks and Root Mean Squared Error (RMSE) for regression tasks.

Comparison Results

Answer to Q1: accuracy. We assume there are two parties and conduct the following comparison to answer Q1.

First, we compare the model performance built on (1) PCA using centralized plaintext data, (2) PCA using decentralized plaintext data, and (3) PPPCA using Homomorphic Encryption (HE) and Secret Sharing (SS). We choose logistic regression model for fraud and APS, since they are binary classification tasks, and choose linear regression model for Wine and Online dataset, since they are regression tasks. We report the comparison results in Table 1. From it, we can see that the model built upon PPPCA is lossless comparing with the model build on centralized plaintext PCA, and models build on local PCA separately have worse performance. This experiment indicates the effectiveness of PPPCA.

Second, we compare the performance of different models, i.e., Logistic Regression (LR), Gradient Boosting Decision Tree (GBDT), and Deep Neural Networks (DNN) on APS dataset. We summarize the results in Table 2, where we find similar results as in Table 1. This experiment indicates the PPPCA can be used to build any privacy preserving machine learning models for multi-parites.

Answer to Q2: efficiency. We now study the time efficiency of our two proposed methods. To do this, we vary the number of parties and compare the running time of PPPCA on Wine dataset. We report the results (in seconds) in Table 3. We can see that with the increase of data providers, the running time of HE based method grows slower than SS based method. This is because the time of HE is mainly spent on encryption, which can be done by each party in parallel. In contrast, as the number of parties increases, the running time of SS based method increases dramatically.

Table 2: AUC results of different models on APS dataset

Model	PCA (Centralized)	PCA (Separately)	PPPCA	
Secure Op	-	-	HE	SS
LR	0.9915	0.9863	0.9918	0.9906
GBDT	0.9941	0.9857	0.9943	0.9948
DNN	0.9937	0.9913	0.9928	0.9931

Table 3: Running time (in seconds) of PPPCA with different number of parties

Runing time / Num of parties / Method	2	3	4
HE based method	3.16	3.23	3.36
SS based method	2.91	3.22	3.42

Conclusion

In this paper, we presented a general modeling paradigm with Privacy Preserving Principal Component Analysis (PPPCA) for horizontally partitioned data under server-aided setting. It involves two methods using homomorphic encryption and secret sharing respectively. Experiments on three UCI datasets and a real-world dataset demonstrated the efficiency and effectiveness of PPPCA. In the future, we would like to deploy PPPCA in real-world applications in Ant Financial.

Acknowledgments

We would like to thank all the anonymous reviewers for their valuable suggestions. Y. Liu and G. Liu were supported in part by the Anhui Sun Create Electronics Company Ltd., under Grant KD1809300321, the National Key R&D Program of China under Grant 2018YFC0832101, and STCSM18*DZ*2270700.

References

Al-Rubaie, M.; Wu, P.-y.; Chang, J. M.; and Kung, S.-Y. 2017. Privacy-preserving pca on horizontally-partitioned data. In *2017 IEEE Conference on Dependable and Secure Computing*, 280–287. IEEE.

Cortez, P.; Cerdeira, A.; Almeida, F.; Matos, T.; and Reis, J. 1998. Modeling wine preferences by data mining from physicochemical properties. *DSS* 47(4):547–553.

Grammenos, A.; Mendoza-Smith, R.; Mascolo, C.; and Crowcroft, J. 2019. Federated pca with adaptive rank estimation. *arXiv preprint arXiv:1907.08059*.

Group, A. F. S. April 25, 2018. Payment risk identification.

Hardy, S.; Henecka, W.; Ivey-Law, H.; Nock, R.; Patrini, G.; Smith, G.; and Thorne, B. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*.

Kelwin Fernandes, Kelwin Fernandes, P. C. P. S. January 8, 2015. Online news popularity data set.

Liu, X.; Lin, Y.; Liu, Q.; and Yao, X. 2018. A privacy-preserving principal component analysis outsourcing framework. In *TrustCom*, 1354–1359. IEEE.

Mohassel, P., and Zhang, Y. 2017. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*, 19–38. IEEE.

Paillier, P. 1999. Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt*, 223–238. Springer.

Pearson, K. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2(11):559–572.

Singh, P.; Raman, B.; Agarwal, N.; and Atrey, P. K. 2017. Secure cloud-based image tampering detection and localization using pob number system. *TOMM* 13(3):23.

Tony Lindgren, J. B. September, 2016. Aps failure and operational data for scania trucks.

Wei, L.; Sarwate, A. D.; Corander, J.; Hero, A.; and Tarokh, V. 2016. Analysis of a privacy-preserving pca algorithm using random matrix theory. In *GlobalSIP*, 1335–1339. IEEE.