Privacy Preserving Machine Learning with Limited Information Leakage

Wenyi Tang[1], Bo Qin[1], Suyun Zhao[1(✉)], Boning Zhao[1], Yunzhi Xue[2], and Hong Chen[1]

[1] Renmin University of China, Beijing, China
zhaosuyun@ruc.edu.cn
[2] Institution of Software, Chinese Academy of Sciences, Beijing, China

Abstract. Machine learning is now playing important roles in daily lives, however, the privacy leakages are increasingly getting serious in the meantime. Current solutions to the privacy issues in machine learning, like differential privacy or homomorphic encryption either could only be applied to some specific scenarios or bring huge modification to the model construction, not to mention massive efficiency loss. In this paper, we consider addressing the privacy issue in machine learning from another perspective, without modification to models or severe efficiency loss. We proposed a straightforward privacy preserving machine learning scheme, training machine learning models directly over encrypted data. Ideally, this scheme could provide privacy protection to both training data and test data. We gave it a try by applying order preserving encryption (OPE) to the scheme. We discussed the possibility of using OPE to reveal the order information confidentially for model training. Several OPE algorithms were chosen to utilize the proposed method. Finally, comprehensive experiments were deployed on both synthetic and real datasets. The experiments on real datasets show that the learning performance of several well-known classifiers on before and after encryption changes slightly. The experiments on synthetic datasets show the classifier performance could be ranked according to fidelity and reliability.

Keywords: Machine learninig · Privacy preserving · Order preserving encryption

1 Introduction

Machine learning has become a major component of numerous applications. Companies and organizations use machine learning algorithms as essential components of their service, and sometimes the model training itself is another kind of service. Generally, the generalization ability of the trained model depends on the quantity and quality of the training data. Therefore, to provide better service, a mass of original data are collected for model training, which brings serious privacy leakage in practice.

J. K. Liu and X. Huang (Eds.): NSS 2019, LNCS 11928, pp. 352–370, 2019.
https://doi.org/10.1007/978-3-030-36938-5_21

The privacy issue in machine learning has been considered by academic community and government for a long time. With the European Union announcing the General Data Protection Regulation (GDPR), people started paying more attention to privacy protection.

The first try is anonymity, which includes deleting some real information, perturbing the original data or other ways, whereas these anonymity methods were turned out to be not safe enough in the Netflix prize challenge [4], in which the users' private information were extracted by differential attacks even all sensitive information were removed. To protect against such attacks, differential privacy has been carefully studied for providing the strongest privacy protection, as well as a framework to quantify the privacy loss [12]. It limits the probability of extracting individual information from an aggregated database, focusing on preserving privacy from public data. However, typical differential privacy machine learning frameworks do not protect the data itself, which means they either do not share the data [1] or only add noise in the specific procedure [24]. Such specialty makes the differential privacy framework could not fit in the service provider scenario arbitrarily. Moreover, if the dataset itself is private, the differential framework could be unapplicable as well.

Another perspective is preserving the privacy through data confidentially, which is mostly implemented by using some cryptographic primitives like Homomorphic Encryption (HE) and Secure Multiparty Computation (SMC). Compared with differential privacy solutions, the cryptography-based solutions are more intuitive. In fact, it is remarkably challenging to locate what kind of information is 'private' and selectively preserve such information from being revealed from a large dataset. Therefore, the cryptographic frameworks could provide more reliable security guarantee, which may have great prospect. However, current cryptography-based schemes are often caught in trouble for 2 main reasons: 1. they bring huge modification to target models; 2. the frequent parameter sharing brings extra interaction load to model training. Such reasons are stumbling blocks to the cryptography-based privacy preserving machine learning using in service providing.

For supervised learning, the algorithm tries to discover the relationship between a feature vector and a label to construct a classifier. Generally, the data would be preprocessed before being fitted into the model. Common preprocess methods include min-max normalization, z-score normalization. Such preprocess methods inevitably change the values of the feature vectors, which indicates the precise data are not necessary for the learning procedure of a classifier and only some information contained in the dataset is critical for learning. If we can selectively expose such information instead of private data itself, it would be a great solution for privacy preserving machine learning applications. Therefore, consider a straightforward idea: is it possible to train models over encrypted data and still obtain similar result? We believe it is possible for the developing of property preserving encryption.

The notion of property preserving encryption [23] allows anyone to check a property on plaintexts by running a public test over the corresponding ciphertexts, suggesting that selective and secure exposure of critical information is possible. In fact, the homomorphic encryption systems could be regarded as a property preserving encryption, which preserve the property of polynomial calculation. The property considered in this paper is *order information* and the method to reveal order information safely is OPE. It is a functional encryption scheme developed for efficient encrypted data query. Therefore, the ciphertext of OPE implicitly holds some distribution information of the original data, which is likely to be learnable, and if it is, with its cryptographic level security property, OPE could be a great choice in privacy preserving machine learning.

In this paper, we propose a simple and universal used model to consider the privacy protection, machine learning over encrypted data, which achieves two major purposes:

1. the data used in model training is private;
2. the fully trained model is private.

Along with OPE, these purposes could be achieved easily and efficiently. The considered model is a high-level one so that it could be applied to many scenarios. This model also presents an open issue in privacy preserving data mining: how much information contained in dataset is enough to train a model? And how do we securely dispose such information without any privacy loss?

The main contribution of this paper contains:

- we firstly consider to use the limited information leakage of the dataset to perform privacy-preserving model training;
- we firstly combine OPE with machine learning, and analyze the information leakage of OPE to show that it may fit in current machine learning models;
- we use special criteria to assess how an OPE algorithm is appropriate for model training, and choose 3 typical implementations of OPE to evaluate.

The rest of this paper is organized as follows. Section 2 presents the recent work about OPE and privacy preserving machine learning. Section 3 gives a general view and security analysis of our model. Section 4 discusses the revealing of order information by order preserving information, and why the model training could be performed over OPE ciphertexts. Section 5 comprehensively demonstrates and analyzes the experimental results of our methods. In Sect. 6 we conclude this paper.

2 Related Work

We firstly review some work of Order-Preserving Encryption, then we review the progress in privacy preserving machine learning.

2.1 Order-Preserving Encryption

OPE provides efficient range query over ciphertext, which is usually used for building secure database. Several methods have been proposed, such schemes hold great practical value in practice. The first OPE [2] was proposed in 2004, which is based on bucket partitioning and was proven insecure sooner. In 2016, Boneh et al. proposed the notion of Order-Revealing Encryption [7], which introduces an extra public evaluation algorithm to perform comparison instead of revealing order information directly in numeric field. Current researches about OPE fall into two categories, the stateful solutions and the stateless solutions. The stateful solutions require the algorithm along with an extra state parameter, which represents the current state of all ciphertext and changes the pre-existing ciphertext while the state parameter changes [25] [16]. Such methods are not appropriate for the proposed purpose since the updating of ciphertext invalidates the fully-trained models. A more reasonable choice is stateless solutions, which does not require prior changes for existing ciphertext. To increase security, some stateless solutions like [5,6,17] are based on s partition, and they were designed to be non-deterministic encryptions, meaning that the same plaintext might be encrypted into different ciphertext for frequency hiding. Chenette et al. presented another kind of solution [9]. They treated plaintext as fixed-length binary string, and every ciphertext bit was the modular value of a encrypted result produced by a symmetric encryption (like AES) with prefix sub-plaintext string of the corresponding plaintext bit. The procedure of comparing two ciphertexts was actually finding the first different bit (most significant different bit).

2.2 Privacy Preserving Machine Learning

There were two groups of researchers ([3,18]) putted forward the notion of privacy preserving data mining (PPDM) in the same year of 2000, after that, a mount of different methods were applied to meet the need in data mining or machine learning. Differential privacy based methods were widely studied and was usually applied to public data training, which could prevent personal data from being extracted. Researchers from Google Brain introduced the differential privacy into non-convex objectives deep neural networks with noisy stochastic gradient descent (SGD), and proposed moments accountant to track precise privacy loss in model training [1]. They then proposed another strategy, using an ensemble of teachers trained on disjoint subsets of sensitive data to label the public data [24], so that the non-sensitive knowledge could be transferred to the student model. Reza Shokri and Vitaly Shmatikov presented distributed SGD and designed a system for collaborative deep learning among multiple participants [26]. The participants would firstly train their own model on private data, and then selectively shared model parameters with differential privacy, which brought attractive trade-off between utility and privacy. Meng et al. addressed the problem of privacy-preserving social recommendation under personalized privacy settings with differential privacy [20].

Apart from differential privacy, other frameworks combining several cryptographic primitives were proposed to deal with different private part of machine learning. Homomorphic encryption and secure multi-party computation are the most widely used primitives to build PPDM schemes.

The first fully homomorphic encryption scheme [13] was proposed in 2009. Since then numerous researches have been done in improving the efficiency of HE schemes so that it would be applied in practical. Secure multi-party computation was firstly introduced in 1982 [27], which also suffered limitation in efficiency. Ideally, secure multi-party computation enables jointly computing a function from the private inputs of each party, who would not have to reveal such inputs to others, so that the privacy is preserved. Many progresses have been made in decades, and now SMC and HE have been used to construct various machine learning models. The basic idea of such solutions is replacing the arithmetic operation in models with homomorphic/SMC ones, so that the computation could be executed over encrypted data. In [8], the authors extracted some basic operations in the model computing as building blocks, applying several homomorphic cryptosystems to implement and constructed classifiers over encrypted data, which could be applied to build multiple models. Another group of researchers also presented a privacy preserving ridge regression system combining HE and garbled circuits [22]. CryptoNets [14] was another significant work, which used HE to apply neural networks to encrypted data, which brought high throughput. After that MiniONN [19] was proposed in 2017, which presented a framework to transfer a well-trained model into an obvious version by using several SMC protocols. Payman Mohassel and Yupeng Zhang proposed a two-server model with SMC protocols for linear regression, logistic regression and neural network training, which achieved great efficiency [21].

3 Scheme Description

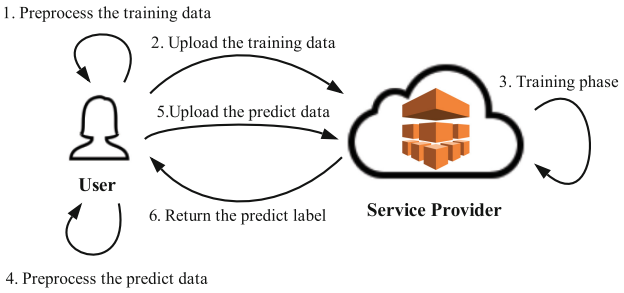

Fig. 1. Dataflow of proposed scheme

3.1 Scheme Overview

Our scheme is quite straightforward – applying any model training procedure directly to the OPE-encrypted data set. Figure 1 shows a general data flow of our scheme. The key component of the scheme is the preprocess procedure in step 1 and 4 from Fig. 1, which contains:

- the metadata of the raw dataset should be erased;
- the labels for classification should be replaced by meaningless identifiers;
- the name of features should be substituted by meaningless strings.
- encrypt every feature with OPE independently (vertical encryption).

Encryption are applied feature by feature independently, which is referenced from the common normalization methods such as min-max normalization and z-score normalization, since the OPE could be treated as another kind of 'normalization'.

As a result, 2 parts of data are protected in this architecture: one is the data from step 1 in Fig. 1, which are some historical data or pre-existing data with tagged labels (for supervised learning); the other is the data in step 4, which could be new data generated recently, and needs to be imported to the trained model for predictions. Therefore the scheme could achieve both privacy preserving training and classification.

3.2 Security Analysis

We define the adversary model first. It is clear, without regard to the transmission failure in network, the only possible leakage in the scheme is the service provider. Therefore, in this paper, we consider the service provider as *Honest but Curious*, which means the service provider would execute the protocol correctly, meanwhile, tries to extract private information from the uploaded data. In the worst case, the service provider may be attacked and comprised, as a result, the service provider would become only *Curious*, with the attacker having complete access to the data and trained model.

Considering the adversary model defined before, the possible leakage points are the transmission procedure and service provider, therefore, the solution is cryptographic level protection. In this case, the minimum level of the processed data is resisting cipher-only attack (COA security) since we protect privacy by preserving data confidentiality. It is true that COA security is a rather weak security model, whereas in this particular situation, the servers in the proposed model are unable to obtain any other information other than ciphertext, the privacy could be preserved by the data confidentiality that COA security provides. According to the formal definition of COA security, the attacker could not retrieve any information of the plaintext from the ciphertext, which could be considered an ideal situation of privacy protection. If the encryption algorithm is COA security, as long as the security parameters remain confidential, neither service provider nor the attacker could retrieve meaningful plaintext from the uploaded data. The encryption procedure also brings native access control

to the trained model because the raw data has to be encrypted as the training data did, otherwise the fitting and prediction would make no sense.

Note that even the order preserving encryption is a practical and efficient method in some database, but it is also a risky one in privacy preserving. On one hand, it has been proven that database protected by order preserving encryption is highly possible to be attacked with some auxiliary information [11,15], so it is important that the released data does not carry any information that may lead to privacy leakage (as described in step 1 and 4). On the other hand, Some particular kinds of data are suggested to avoid using order preserving encryptions. For example, a typical image is a matrix of several pixel points, each point consists of values of three color channel (RGB). The value of each color channel ranges from 0 to 255. For a dataset of images, it is very likely that every possible values of a color channel would appear multiple times. If we deal such data with naive order preserving mapping, the processed data would be very much likely to be themselves, which turns out to be in vain. So, for a dense and bespread feature dataset, it is suggested to avoid using order preserving mapping (at least deterministic order preserving) for privacy preserving purpose.

4 Learning with OPE Ciphertexts

In this section, we give a brief introduction about OPE and discuss the different way to reveal order information.

4.1 Order Preserving Encryption

As the name implies, the order information indicates the comparison relationship in a certain range. The formal definition is given below, for any mapping function $F : D \rightarrow R$, in which D denotes domain and R denotes range, we say F is an order preserving mapping if:

$$\forall x_i, x_j \in D,\ if\ x_i < x_j,\ then\ F(x_i) < F(x_j)$$

For example, a list with 5 values (2, 5, 23, 0, 13), a function that maps these values into (2, 3, 5, 1, 4) is an order preserving mapping. Note that equality relationship is not mentioned in the definition because, for confidentiality concern, a much more secure choice of order preserving function is order preserving encryption, which applies different ways of handling equal elements in different cryptographic algorithms. The equality-handling methods roughly divide into two types: deterministic and non-deterministic. The deterministic algorithms encrypt one plaintext to a certain ciphertext, no matter how many times the same plaintext appears. In such methods, the revealed information contains not only order, but also frequency. The non-deterministic algorithms, on the contrast, encrypt one plaintext to several different ciphertext if the plaintext appears multiple times, which makes such algorithms one-to-many mapping functions. Note that since the comparison relationship between two unequal values still holds, all possible ciphertext of one plaintext e.g. $x_i \in (x_i - 1, x_i + 1)$ lie within a limited range e.g. $(F(x_i - 1), F(x_i + 1))$.

4.2 Model Training over OPE Ciphertexts

We discuss the principle of model training over OPE ciphertexts from two perspectives.

Information Preserved by OPE. In information theory, information entropy is a concept describing the average rate at which information is produced by a stochastic source of data. For a dataset D with $|Y|$ different classes, the information entropy of is computed as:

$$Ent(D) = -\sum_{k=1}^{|Y|} p_k \log_2 p_k$$

in which p_k represents the frequency of the k_{th} sample. In our vertical encryption, it is clear that $Ent(D)$ remains invariant after encrypted by deterministic OPE. When it comes to non-deterministic OPE, randomness would be involved. Ideally, the distribution of non-deterministic OPE ciphertexts is closed to uniform, which leads to that every ciphertexts would only appear once, making the dataset barely contains information entropy. This is a similar situation to dealing with continuously distributed values. In such cases, the common solution is to discretize the continuous feature values using bi-partition technique, which would get an approximate result of the plaintext result, making the entropy of ciphertexts closer to the original one.

Model Training over Data. From a high level perspective, all of the machine learning models can be divided into two categories based on whether they assume the data obeys a specific distribution or not – parametric models and non-parametric models.

Parametric models, such as linear regression, logistic regress, would assume all the data come from a specific distribution first, then search the whole parametric space of the distribution to find the best match for the given training set. For the OPE ciphertext data, the marginal distribution of ciphertext could be regarded as asymmetric strech or compress of the original marginal distribution, which means the best position in parametric space would be altered. However, the universal distribution would remain mostly invariant, which means the same model may still be functional over ciphertexts.

Non-parametric models, such as decision tree, naive bayes, k-nearest neighbor and so on, would try to find a best match in the training data, since they do not assume the distribution of data, they may sometimes obtain generalization ability to the data that never appear in training set. The non-parametric models works with only one requirement: the data could be ranked, which perfectly fits the property of OPE. Theoretically, the model training could work on ciphertexts as well as on plaintexts.

5 Evaluation

In this section, we firstly describe some configuration of our experiments, including the OPE we choose to evaluate, the datasets and machine learning models we tested, and the special criteria we use to evaluate the performance of different models over specific OPE algorithm. Then we give comprehensive demonstration and analysis to results of our experimental results.

5.1 Experiments Configuration

OPE Algorithms. We give brief introductions to the OPE algorithms evaluated in our experiment, for more details about each algorithm, please refer to the cited paper.

OPEA [17] is a non-deterministic OPE based on cipher space division. OPEA would firstly discretize the integer s into sequential-partitions randomly, in which the interval between two adjacent partitions are non-empty. Then for a specific integer plaintext value b, it would be encrypted to a random integer in partition $[L_b, U_b]$, where the L_b and U_b denote the lower bound and upper bound of the b_{th} partition. Therefore, OPEA is an one-to-many mapping encryption (Fig. 2).

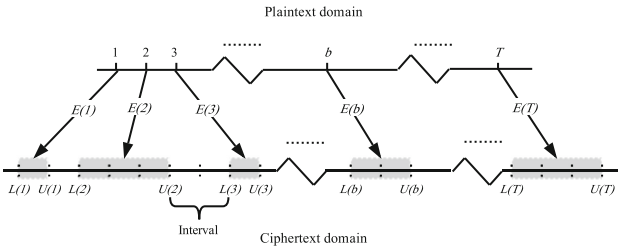

Fig. 2. The one-to-many mapping in OPEA

Hypergeometric OPE [5] is also a cipher space division-based OPE. Different with OPEA, it uses a pseudo random function to divide the ciphertext space according to hypergeometric distribution, and mapping the same plaintext to one ciphertext. The authors notice that an order preserving function f from $[M]$ to $[N]$ corresponds to a unique M-out-of-N random sequence without repetition (the right-hand of 1), which indicates that the generation of an OPF could be considered as an experiment of x success (random draws for which the object drawn has a specified feature) in y draws, without replacement, from a finite population of size M contains exactly N objects with such feature (Hypergeometric distribution). The equality of both probabilities means the generation of an OPE could be accomplished by recursively calling a pseudo Hypergeometric

sample algorithm with secret key. And this is exactly how the Hypergeometric OPE scheme was constructed. For more details about Hypergeometric OPE, please refer to [5].

$$Pr[f(x) \leqq y < f(x+1) : f \leftarrow OPF_{[M],[N]}] = \frac{\binom{y}{x}\binom{N-y}{M-x}}{\binom{N}{M}} \tag{1}$$

mOPE [25] is a sort tree-based OPE which leaks no more information than order. In the original scheme, the encryption procedure is designed with 2 parts: one is building a binary sort tree upon the plaintexts, and tagging each left branch with "0", right branch with "1", then generates encode for every node by concating all bits on the path from the root to the current location in the tree, padding it with postfix "1...0" to fixed length. The other part is generating semantically secure ciphertext using common symmetric encryption scheme. As shown in Fig. 3, by transforming the fixed length binary codes into decimal form, the numbers preserve the order information of all plaintext, such that the user could request range query using ordered encode and gets the semantically secure ciphertexts. In our scenario, the symmetric encryption could be omitted, and we also do not consider the situation that the mOPE tree needs to be balanced. We simply use all existed plaintexts to build the mOPE tree and then partitioning the training and testing sets.

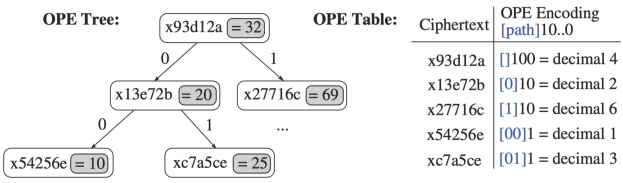

Fig. 3. Overview of mOPE

We also involve AES algorithm as a comparison, and to avoid decimal overflow, we reduce the length of AES ciphertexts by module 2^{16}. As for the parameter settings for OPE, OPEA use random number generator ranges from 0 to 999, ciphertext space of Hypergeometric OPE is limited in $(0, 2^{31} - 1)$, mOPE does not require parameter setting.

Datasets and Models. We deploy 2 groups of experiments to evaluate the performance of the mentioned OPEs fitting in the proposed model, using several datasets for different purposes. The first one is the Wisconsin Diagnostic Breast Cancer dataset (WDBC) from UCI Machine Learning Repository [10], which is used to test the compatibility of different machine learning models over the encrypted data. The other group with randomly generated datasets are used

to assess the impact of some extreme cases, in which the expecting learning accuracy is not significantly better than random guess. We limit the upper bound of the learning accuracy by change the number of the classes, which ranges from 2 to 7 (corresponding to D2–D7). Both the features and labels are generated randomly. The details about these datasets are listed in Table 1. All the datasets are splitted by 75% & 25% as training set and testing set respectively. The data could be retrieved on https://github.com/Tomfortemp/tested-data.

Table 1. Information of the datasets

Group	Data name	Records	Features	Classes
WDBC	D1	569	30	2
Randomly synthetic	D2–7	2000	100	2–7

We also consider some generally-used machine learning models including: linear regression, logistic regression, k-nearest neighbor, support vector machine, naive bayes, decision tree, random forest and gradient boost decision tree. All the models are implemented by using the scikit-learn library with all parameters set as default.

Special Criteria. Most previous work consider the training accuracy as crucial and only evaluation indicator. The experiments were usually tested over some widely applied dataset, achieving highly learning accuracy for most models. However, for some datasets resulting in lower learning accuracy, accuracy might be a misguidance. Some models may obtain the frequency distribution as a minimum margin of learning accuracy, even with cross-validation, the new coming data may still cause the plaintexts-trained model and the ciphertexts-trained model to produce quite another answers since they are different in inner logic, making it less meaningless in accuracy value as well as less convincing in learnability. Therefore, we propose Match Rate (MR) to evaluate the fidelity of the ciphertext-trained model, which is defined as the proportion of records on which the plaintext-trained model and the ciphertext-trained model produce the same prediction in a testing set.

5.2 Demonstration and Analysis

The OPE algorithms we choose to evaluate preserve some iconic properties of typical OPE ciphertexts. Firstly, all of three algorithms reveal the arithmetic order information directly on integer field, which is friendly to the current machine learning models. The learning algorithms could execute on the ciphertexts without any changes, which is quite different from other cryptography-based solutions. Secondly, the distribution properties of three OPE ciphertexts vary from each other. Hypergeometric OPE maps the plaintexts into a larger

ciphertext space, acting like a random order preserving function, with the frequency information still remains. OPEA ciphertexts look like a uniform distribution on ciphertext space, with almost no duplicated records. mOPE ciphertexts are more 'tighter' compared with others, with the frequency information preserves as well.

Group 1. In the first group of experiments we take a quick glance over the proposed model, which is shown in Fig. 4 and Table 2. Compared with plaintexts-trained model, the ciphertext-trained models perfectly preserve the learning accuracy no matter which OPE is applied. It is also showed in the result that the fidelity level of model remains pretty high in all models.

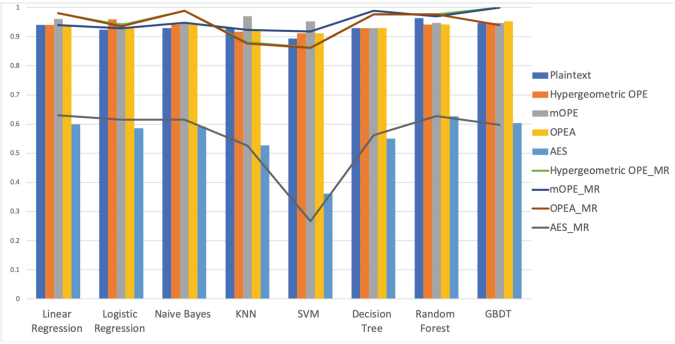

Fig. 4. Experiment on group 1

Table 2. Results of WDBC dataset

Models	Raw accuracy	Hypergeometric		mOPE		OPEA		AES	
		Acc	MR	Acc	MR	Acc	MR	Acc	MR
Linear regression	0.94	0.94	0.98	0.96	0.94	0.94	0.98	0.60	0.63
Logistic regression	0.923	0.959	0.940	0.947	0.929	0.929	0.935	0.586	0.615
Naive bayes	0.929	0.941	0.988	0.947	0.947	0.941	0.988	0.592	0.615
KNN	0.929	0.917	0.881	0.970	0.923	0.923	0.876	0.527	0.526
SVM	0.893	0.911	0.863	0.953	0.917	0.911	0.862	0.361	0.266
Decision tree	0.929	0.929	0.976	0.929	0.988	0.929	0.976	0.550	0.562
GBDT	0.947	0.947	1.0	0.947	1.0	0.953	0.94	0.604	0.597

Non-Tree-Based Models. For the non-tree-based models tested in this paper (including logistic regression, knn, naive bayes, and svm), even though different models achieve different accuracy, the ciphertext-trained models still have similar performance with the plaintexts-trained models. The great performance

in high MR and low accuracy difference shows that, these models are able to produce highly similar learning results relying on only the information that OPE algorithms provide. The causes of such performance might due to that these models concern the changing properties of data rather than precise data. It may also be connected with the impact from randomness introduced by OPEA to the generalization ability of models. The match rate of KNN and SVM are slightly lower than others, which may indicates that OPE encryption may influence the decision making of both models. We will take a closer look in the experiments of Group 2.

Tree-Based Models and Boosting. As is well-known, tree-based models are easily get overfitting, which is also considered as unstable models – with little changes on data comes great changes over the trained models. The tree-based models we evaluated, even for the basic decision tree model shows great effect on both accuracy and fidelity.

As a control group, learning accuracy and fidelity over AES-encrypted data show significant decreasing compared with others. Apart from a pseudo-random projection, AES could also be considered as a cryptographic primitive which leaks only the relation of equality vertically. Therefore, this experiment also indicates that order information is more suitable in machine learning than equality.

Group 2. Figure 5 shows the performance of the mentioned models over our synthetic data, showing the how models' fidelity change with the accuracy declining. With uniform random dataset, both the plaintext-trained and ciphertext-trained models fail to achieve significantly high accuracy than random choosing.

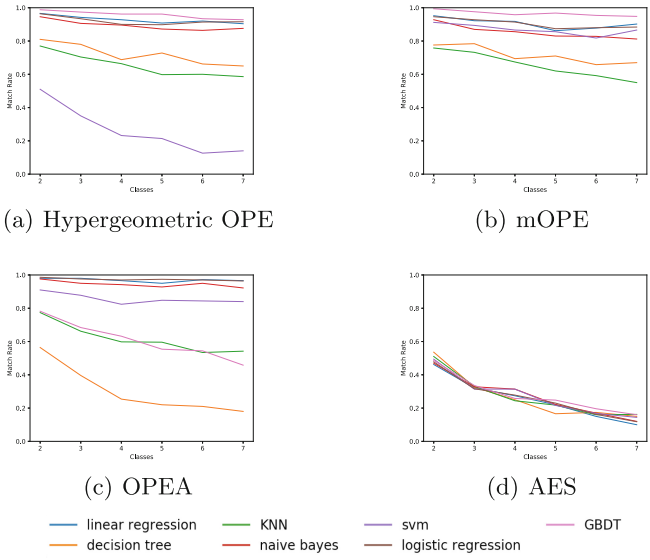

Fig. 5. MR curves on experiments of the second group

We repeated the training for several times and concluded that the average accuracy of every model were approximately close to $1/2, 1/3...1/7$. There are mainly 2 possible reasons: for almost uncorrelated datasets, the models could at least learn the frequency distribution to keep the accuracy above prescribed minimum; even in extremely low accuracy situations, the OPE could still keep the reliability of the plaintexts-trained models.

According to the trend of MR curves, the reliability and fidelity over OPE-processed data depend on different models and different OPE algorithms. As analyzed before, the tree-based models show extremely volatile, because decision tree model is easy to get overfitting, which explains its poor effect over OPEA ciphertexts. The reason why the accuracy difference stay low in both models due to the prescribed minimum learning accuracy, so, for multi-class problems, the decision tree are not good choice to perform privacy preserving machine learning over OPEA encrypted data. With weaker classifiers assembled, boosting can improve such performance to a certain extent. The former stated non-tree-based models remain stable, MR tend to flatten out with the increasing of classes. Logistic regression and linear regression perform well over every OPE algorithms, which show great compatibility to the machine learning over OPE encrypted data. KNN and SVM are both models that make decisions based on distance. Take euclidean distance as example, the process of OPE changes a record position in the whole feature space, however, since the order information is preserved, the relative position between each record is roughly preserved, which makes such models could obtain similar result over OPE ciphertexts. For tree-based models, the fluctuations in MR should due to the structural change of trees and overfitting by the OPE encryption. Therefore, the aggregation and boosting could make some improvement since they increase the stability and generalization ability of models. On one hand the experiments show that boosting brings better performance to simple tree models, it also indicates that the boosting models are more suitable to fit encrypted data on the other hand.

The performance of various models is usually affected by the datasets. So, for datasets that most machine learning models could achieve high accuracy, it would obtain highly reliable training result no matter what model was chosen to train over encrypted data. For other datasets, if the various models perform closely in accuracy, one should also take the reliability into consideration when deciding which model to choose. Even if the tree-based models show slightly advance in accuracy, they also tend to produce unreliable results.

5.3 Efficiency

The proposed method is quite different from the most popular homomorphic-based or secure multi-party computation based solutions. Instead of changing the construction of machine learning models to adapt the properties of homomorphic ciphertext, we choose to reveal the important information that contained in the data itself confidentially, so that the models could learn similarly to the plaintexts-trained models. The proposed method treat all of the applied models

as black boxes, therefore it does not involve interactive load like homomorphic-based or SMC-based solutions. Theoretically, the extra load of the proposed model comes from 2 aspects: one is the time consumed by the encryption procedure of the training datasets, the other is the extra model training time over encrypted datasets compared with that over the raw datasets.

Table 3. Time consumption of encryption

Procedure	AES	Hypergeometric	OPEA	mOPE
KeyGeneration	–	–	O(n)	O(nlogn)
Encryption	O(1)	O(logM)	O(1)	O(logn)

Encryption Load. Compared to common symmetric encryption algorithms like AES, it is fairly to say that most order-preserving encryption schemes are not so efficient. Because in most cases, the OPE considers the whole state of a dataset in encryption procedure, which is more complicate than common symmetric encryption. In our model, the datasets is encrypted with OPE vertically, which means for each feature, OPE is applied to all of values of n instances. For Hypergeometric OPE, once the ciphertext space is determined, the encryption algorithm would recursively call a pseudo-Hypergeometric sample function to run a binary search over the ciphertext space, so the average time complexity is $O(logM)$ (M is the size of ciphertext space). For OPEA, the KeyGeneration procedure consumes a linear time complexity with the size of ciphertext space M. The encryption after the KeyGeneration is quite efficient since such procedure is an indexed sequential search, which is a constant level of time complexity. For mOPE, the encryption procedure relies on the balancing binary search tree built upon all n values in a feature, which consumes a $O(nlogn)$ time complexity. After the tree has been built, the encryption of each plaintext is a binary search over the built tree which consumes $O(logn)$. The time-consuming results are listed in Table 3, with both theoretical analysis and empirical results. We also give the AES time-consuming data as a comparison. Note that even the time consuming of the encryption scheme is significant, we still consider it worthwhile since it brings enhancements to the security level of the outsourced data. Furthermore, the large batch of encryption over the dataset is actually an one-time-consumption. When the model was fully trained, the data in actual use are much less. Table 3 shows the encryption average time load for encrypting 1000 plaintext. Hypergeometric OPE and AES use a random string as secret key, so the key generation procedure for these 2 schemes are omitted. OPEA needs to pre-divide the whole ciphertext space by generating a series of random numbers as secret key. mOPE needs to rank all the plaintext into a binary balanced sort tree as secret key. So key generation procedure for these 2 schemes are considered.

Training Load. The other load is caused by the ciphertext expansion invoked by OPE encryption. In fact, in our experiments, only OPEA and Hypergeometric OPE would bring ciphertext expansion for sure. As for mOPE, there is no clear relationship between the size of original data and the length of the final encode.

Table 4. Expansion of ciphertext

OPE	D1	D7–12
Raw data	124 KB	585 KB
Hypergeometric OPE	219 KB	1.6 MB
mOPE	115 KB	719 KB
OPEA	190 KB	1.3 MB
AES	139 KB	1.2 MB

Table 4 shows the expansion of the datasets in our experiments. Note that for data length consideration, we compress the ciphertexts of AES encryption by modulo each one with 2^{16} to avoid float overflow. Mostly, how much the ciphertext expands mainly depends on the possible range of plaintext value in linear dependency, which is obviously superior that the exponential expansion of homomorphic encryption. Besides, in training phase, usually the data would be preprocessed before fitting into the models. In other word, the extra load in training is negligible. Without the extra interactive load, the proposed method achieves much better efficiency.

6 Conclusion

This work applies a well-known functional encryption to machine learning applications. In our simple scheme, the data privacy could be protected by OPE under only COA security notion in cryptography. As a result, once the data is released after encrypted by OPE, the training procedure is completely non-interactive and supports private prediction based on the natural 'access control' of an encryption algorithm, which means. Furthermore, in evaluation, we proposed an extra criteria to evaluate whether a certain data processing scheme fits the privacy preserving machine learning framework or not, which has potential to be applied to more encryption schemes. Compared with the evaluation methods applied before, our proposal is more rigorous and pays more attention to the fidelity of the ciphertext-trained model, which is a critical factor of producing reliable predictions. Three order preserving encryption schemes were applied to the model, plentiful of experiments suggest that, with only the information OPE reveals, most machine learning models are able to obtain knowledge as much as those from raw data. We also evaluated the efficiency of our scheme. Experiments show that the time consumption of our scheme mainly increases in the initial encryption, while bringing negligible overhead in model training and

using, which distinguishes our schemes from most cryptography-based schemes. We believe this scheme could be a new direction of privacy protection in machine learning. This work is also a potential solution to the open issues proposed in this paper.

Acknowledgment. This paper is supported by the National Key Research & Develop Program of China through project 2017YFB1400700, 2017YFB0802500, 2018YFB1004401 and 2016YFB1000702, by the Natural Science Foundation of China through project 61772538, 61672083, 61370190, 61532021, 61472429, 61772536, 61772537, 61732006, 61702522, 91646203 and 61402029, by the National Cryptography Development Fund through project MMJJ20170106.

References

1. Abadi, M., et al.: Deep learning with differential privacy. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016, pp. 308–318. ACM (2016). https://doi.org/10.1145/2976749.2978318, http://doi.acm.org/10.1145/2976749.2978318
2. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Order-preserving encryption for numeric data. In: Weikum, G., König, A.C., Deßloch, S. (eds.) Proceedings of the ACM SIGMOD International Conference on Management of Data, Paris, France, 13–18 June 2004, pp. 563–574. ACM (2004). https://doi.org/10.1145/1007568.1007632, http://doi.acm.org/10.1145/1007568.1007632
3. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Chen, W., Naughton, J.F., Bernstein, P.A. (eds.) Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 16–18 May 2000, Dallas, Texas, USA, pp. 439–450. ACM (2000). https://doi.org/10.1145/342009.335438, http://doi.acm.org/10.1145/342009.335438
4. Bell, R.M., Koren, Y.: Lessons from the Netflix prize challenge. SIGKDD Explor. **9**(2), 75–79 (2007). https://doi.org/10.1145/1345448.1345465. http://doi.acm.org/10.1145/1345448.1345465
5. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-preserving symmetric encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 224–241. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_13
6. Boldyreva, A., Chenette, N., O'Neill, A.: Order-preserving encryption revisited: improved security analysis and alternative solutions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 578–595. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22792-9_33
7. Boneh, D., Lewi, K., Raykova, M., Sahai, A., Zhandry, M., Zimmerman, J.: Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 563–594. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_19
8. Bost, R., Popa, R.A., Tu, S., Goldwasser, S.: Machine learning classification over encrypted data. In: 22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, 8–11 February 2015. The Internet Society (2015). https://www.ndss-symposium.org/ndss2015/machine-learning-classification-over-encrypted-data

9. Chenette, N., Lewi, K., Weis, S.A., Wu, D.J.: Practical order-revealing encryption with limited leakage. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 474–493. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-52993-5_24
10. Dua, D., Graff, C.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
11. Durak, F.B., DuBuisson, T.M., Cash, D.: What else is revealed by order-revealing encryption? In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016, pp. 1155–1166. ACM (2016). https://doi.org/10.1145/2976749.2978379
12. Dwork, C.: Differential privacy: a survey of results. In: Agrawal, M., Du, D., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 1–19. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79228-4_1
13. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, 31 May–2 June, 2009, pp. 169–178. ACM (2009). https://doi.org/10.1145/1536414.1536440, http://doi.acm.org/10.1145/1536414.1536440
14. Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K.E., Naehrig, M., Wernsing, J.: CryptoNets: applying neural networks to encrypted data with high throughput and accuracy. In: Balcan, M., Weinberger, K.Q. (eds.) Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, 19–24 June 2016. JMLR Workshop and Conference Proceedings, vol. 48, pp. 201–210 (2016). JMLR.org, http://jmlr.org/proceedings/papers/v48/gilad-bachrach16.html
15. Grubbs, P., Sekniqi, K., Bindschaedler, V., Naveed, M., Ristenpart, T.: Leakage-abuse attacks against order-revealing encryption. In: 2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, 22–26 May 2017, pp. 655–672. IEEE Computer Society (2017). https://doi.org/10.1109/SP.2017.44
16. Kerschbaum, F.: Frequency-hiding order-preserving encryption. In: Ray, I., Li, N., Kruegel, C. (eds.) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015, pp. 656–667. ACM (2015). https://doi.org/10.1145/2810103.2813629
17. Li, Y.-N., Wu, Q., Tang, W., Qin, B., Wang, Q., Miao, M.: Outsourcing encrypted excel files. In: Liu, J.K., Samarati, P. (eds.) ISPEC 2017. LNCS, vol. 10701, pp. 506–524. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-72359-4_30
18. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_3
19. Liu, J., Juuti, M., Lu, Y., Asokan, N.: Oblivious neural network predictions via MiniONN transformations. In: Thuraisingham, B.M., Evans, D., Malkin, T., Xu, D. (eds.) Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, 30 October–03 November 2017, pp. 619–631. ACM (2017). https://doi.org/10.1145/3133956.3134056
20. Meng, X., et al.: Personalized privacy-preserving social recommendation. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, 2–7 February 2018. AAAI Press (2018). https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16768

21. Mohassel, P., Zhang, Y.: SecureML: a system for scalable privacy-preserving machine learning. In: 2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, 22–26 May 2017, pp. 19–38. IEEE Computer Society (2017). https://doi.org/10.1109/SP.2017.12
22. Nikolaenko, V., Weinsberg, U., Ioannidis, S., Joye, M., Boneh, D., Taft, N.: Privacy-preserving ridge regression on hundreds of millions of records. In: 2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, 19–22 May 2013, pp. 334–348. IEEE Computer Society (2013). https://doi.org/10.1109/SP.2013.30
23. Pandey, O., Rouselakis, Y.: Property preserving symmetric encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 375–391. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_23
24. Papernot, N., Abadi, M., Erlingsson, Ú., Goodfellow, I.J., Talwar, K.: Semi-supervised knowledge transfer for deep learning from private training data. CoRR abs/1610.05755 (2016). http://arxiv.org/abs/1610.05755
25. Popa, R.A., Li, F.H., Zeldovich, N.: An ideal-security protocol for order-preserving encoding. In: 2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, 19–22 May 2013, pp. 463–477. IEEE Computer Society (2013). https://doi.org/10.1109/SP.2013.38
26. Shokri, R., Shmatikov, V.: Privacy-preserving deep learning. In: Ray, I., Li, N., Kruegel, C. (eds.) Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015, pp. 1310–1321. ACM (2015). https://doi.org/10.1145/2810103.2813687, http://doi.acm.org/10.1145/2810103.2813687
27. Yao, A.C.: Protocols for secure computations (extended abstract). In: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3–5 November 1982, pp. 160–164. IEEE Computer Society (1982). https://doi.org/10.1109/SFCS.1982.38