OpBoost: A Vertical Federated Tree Boosting Framework Based on Order-Preserving Desensitization

Xiaochen Li[*]
Zhejiang University
xiaochenli@zju.edu.cn

Yuke Hu[*]
Zhejiang University
yukehu@zju.edu.cn

Weiran Liu
Alibaba Group
weiran.lwr@alibaba-inc.com

Hanwen Feng
Alibaba Group
fenghanwen.fhw@alibaba-inc.com

Li Peng
Alibaba Group
jerry.pl@alibaba-inc.com

Yuan Hong
University of Connecticut
yuan.hong@uconn.edu

Kui Ren
Zhejiang University
kuiren@zju.edu.cn

Zhan Qin[†]
Zhejiang University
qinzhan@zju.edu.cn

ABSTRACT

Vertical Federated Learning (FL) is a new paradigm that enables users with non-overlapping attributes of the same data samples to jointly train a model without directly sharing the raw data. Nevertheless, recent works show that it's still not sufficient to prevent privacy leakage from the training process or the trained model. This paper focuses on studying the privacy-preserving tree boosting algorithms under the vertical FL. The existing solutions based on cryptography involve heavy computation and communication overhead and are vulnerable to inference attacks. Although the solution based on Local Differential Privacy (LDP) addresses the above problems, it leads to the low accuracy of the trained model.

This paper explores to improve the accuracy of the widely deployed tree boosting algorithms satisfying differential privacy under vertical FL. Specifically, we introduce a framework called OpBoost. Three order-preserving desensitization algorithms satisfying a variant of LDP called distance-based LDP (dLDP) are designed to desensitize the training data. In particular, we optimize the dLDP definition and study efficient sampling distributions to further improve the accuracy and efficiency of the proposed algorithms. The proposed algorithms provide a trade-off between the privacy of pairs with large distance and the utility of desensitized values. Comprehensive evaluations show that OpBoost has a better performance on prediction accuracy of trained models compared with existing LDP approaches on reasonable settings. Our code is open source.

[*]Both authors contributed equally to this work.
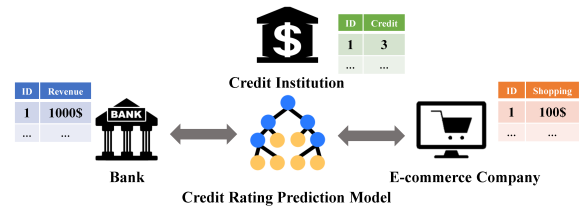[†]Corresponding author.

Figure 1: An example of Vertical FL.

1 INTRODUCTION

Federated Learning (FL) [44] is an emerging paradigm that enables multiple parties to jointly train a machine learning model without revealing their private data to each other. According to the way of data partitioning, FL can be classified into two categories: Horizontal FL and Vertical FL [62]. *Horizontal* FL considers the scenarios where different data samples with the same features are distributed among different parties. *Vertical* FL works when different parties hold the same data samples with disjoint features. Vertically distributed datasets are very common in real-world scenarios. One typical example of vertical FL is shown in Figure 1. A credit institution collaborates with an E-commerce company and a bank to train a model to predict the labels, i.e., users' credit ratings, based on features, i.e., shopping records and revenue.

Tree boosting algorithms (e.g., GBDT [33]) are popular supervised ML algorithms that enjoy high efficiency and interpretability. They are widely used in prediction tasks based on heterogeneous features, i.e., revenue, age, in the scenarios, i.e., credit, price forecast [17, 21, 25, 40]. Recent work [35] shows that when the training data mainly includes heterogeneous features, GBDT outperforms state-of-the-art deep learning solutions. However, most existing tree boosting algorithms are proposed in the centralized setting, which train the model with direct access to the whole training data. With the increasing public privacy concerns and the promulgation of privacy regulations (e.g., GDPR), this centralized setting limits the widespread deployment of tree boosting algorithms. Therefore, it becomes an essential problem to design practical privacy-preserving vertical federated tree boosting algorithms.

To address this problem, there are some solutions based on two different technologies: *Cryptography* and *Local Differential Privacy (LDP)*. Framework SecureBoost [22] and its subsequent work VF^{2}Boost [34] are based on additive *homomorphic encryption*.

Although well-designed engineering optimization is performed, a large number of homomorphic operations still inevitably cause participants to suffer a prohibitively computational overhead. Besides, each party shares the true order of their feature values for training the model. To our knowledge, some attacks have been proposed to use auxiliary information to infer the distribution of original values based on the order of values [14, 15, 28]. Abspoel et al. [10] present a simple vertical federated boosting framework based on Multi-Party Computation (MPC) protocols. Nevertheless, MPC protocols cause a heavy communication overhead. Although the order of feature values is not visible to any party in MPC protocol, the trained model is not a privacy-preserving model, which is vulnerable to inference attack [41, 48, 51]. Tian et al. [52] propose FederBoost, which provides *LDP* privacy protection for each party's feature values and can resist all the aforementioned attacks. FederBoost is shown to be much more efficient than the MPC-based and Encryption-based solutions, which is favorable in real-world applications. However, the introduction of randomness leads to a serious loss of the relative order information, and results in low accuracy of the trained model.

Observation. The process of building the decision tree in tree boosting algorithms is to constantly find the split points of the features, and this process only depends on the order of feature values rather than the exact values. In existing solutions, each party is essentially sending the order of feature values to the party holding the label for training the model. It's necessary to provide privacy protection for the order of feature values since training the model with the true order of the feature values can leak the original private values. However, the mechanisms satisfying LDP are usually designed to provide privacy protection for values without order (e.g., enumeration values). They perturb the private values to achieve the same degree of indistinguishability for any pair of values in the data domain. Meanwhile, this causes desensitized values to lose too much order information, which seriously reduces the accuracy of the trained model. In fact, people require different degrees of indistinguishability for pairs of values with different distances. For example, an employee doesn't mind being revealed that his income is less than his boss, but he minds being known by others to be less than his colleagues. Therefore, it is more suitable to provide different degrees of indistinguishability for pairs with different distances. Meanwhile, the relative order of value pairs, especially those that are far apart, can be preserved with a high probability.

Another observation is that the existing distance-based LDP (dLDP) definition has limitations. There is only one privacy parameter ϵ in the existing dLDP definition, which allocates privacy budgets based on l_1 distance between two values. Given the total privacy budget ϵ, one might want its private value to be as indistinguishable as possible from its nearby values, and not mind weakening the indistinguishability from the values farther away. However, the existing dLDP definition cannot achieve this privacy requirement. Specifically, increasing ϵ can increase the probability of the desensitized value falling near the true value, but at the same time its distribution near the true value is more concentrated, and vice versa. If we can increase the probability of the desensitized value falling in a specific area around the true value, but flatten the probability distribution in this area, an optimized output probability distribution of desensitized value can be obtained.

Contribution. Our contributions are summarized below.

Proposal of OpBoost: We propose a novel framework called OpBoost for privacy-preserving vertical federated tree boosting. Within the framework, we design three order-preserving desensitization algorithms for desensitizing the training data. Different from the existing LDP-based solution, the desensitized training data satisfy a variant of LDP called dLDP. It can preserve more relative order information of desensitized values than LDP while providing privacy protection. When strong indistinguishability is required for close values, i.e., $\epsilon = 0.08$ for value pairs with distance $t = 1$, OpBoost can still achieve accuracy close to that without protection for both classification and regression tasks. For example, for a classification task, OpBoost achieves 60% when no protection is 87%, while the LDP-based solution is close to 10%. Meanwhile, OpBoost also retains the advantages of LDP-based solutions over Cryptography-based solutions. The total communication overhead of each party is about $O(rn \log n)$ bits, whereas $O(rnk \log^2 n)$ is required in the MPC-based solution (n, k, r are the number of samples, values' bits, and features, respectively). Moreover, we replace the exponential mechanism with the (bounded) Laplace mechanism to reduce the computational complexity of desensitizing a sensitive value to $O(1)$.

Optimizing existing dLDP definition: We also optimize the existing dLDP definition in order to break through its limitations. We divide the data domain into several partitions with the length of θ. Then we introduce two privacy parameters ϵ_{prt} and ϵ_{ner} to adjust the probability distribution of desensitized value falling in different partitions and the probability distribution within one partition, respectively. We prove that the existing definition is just a special case where ϵ_{prt} and ϵ_{ner} satisfy a fixed proportional relationship. We can always get higher accuracy than the existing dLDP under the same privacy guarantee by adjusting the ratio of ϵ_{prt} and ϵ_{ner}.

Introducing new order-preserving metrics: In addition to quantifying the privacy of the order-preserving desensitization algorithms with dLDP, we also introduce new theoretical and experimental metrics to quantify the order preserved by desensitized values. We define that the proposed desensitization algorithms are probabilistic order-preserving in theory. The probability of any pair of desensitized values preserving the original relative order is at least γ. Besides, we introduce the weighted-Kendall coefficient weighted by distance to evaluate the order of desensitized values experimentally.

Comprehensive Evaluation: We conduct comprehensive theoretical and experimental evaluations to analyze the performance of OpBoost, including all the designed order-preserving desensitization algorithms. We evaluate the order preservation of the desensitized values using all introduced metrics. We also conduct the experiments on public datasets used for *Binary Classification*, *Multiple Classification*, and *Regression* tasks. Both GBDT and XGBoost are implemented in OpBoost. The experimental results show that OpBoost achieves the prediction accuracy close to and even higher than plain models, i.e.,1.0003× improvement over plain model of XGBoost, which is superior to the existing LDP approaches.

2 PRELIMINARIES

2.1 Differential Privacy

Differential privacy [29] is the *de facto* privacy definition of data disclosure, preventing attempts from learning private information about any individual in a data release. In this work, we are interested

in *local* differential privacy [39], which allows each user to perturb his sensitive data using a randomization mechanism $\mathcal{M}$ such that the perturbed results from different data values will be "close".

Definition 2.1. (*Local Differential Privacy, LDP*). An algorithm $\mathcal{M}$ satisfies ϵ-LDP, where $\epsilon \geq 0$, if and only if for any input $v, v' \in \mathbb{D}$, and any output $y \in Range(\mathcal{M})$, we have

$$\Pr\left[\mathcal{M}(v) = y\right] \leq e^{\epsilon} \Pr\left[\mathcal{M}(v') = y\right].$$

The parameter ϵ above is called the *privacy budget*; the smaller ϵ means stronger privacy protection is provided. Since all pairs of sensitive data shall satisfy ϵ-privacy guarantee for the same ϵ, it may hide too much information about a dataset, such that utility might be insufficient for certain applications. The distance-based LDP [12, 19, 37] is proposed to improve the utility, which measures the level of privacy guarantee between any pair of sensitive data based on their distance. We use l_1 distance in this paper, and the definition of distance-based LDP is defined as follows.

Definition 2.2. (*Distance-based Local Differential Privacy, dLDP*). An algorithm $\mathcal{M}$ satisfies ϵ-dLDP, if and only if for any input $x, x' \in \mathbb{D}$ such that $|x - x'| \leq t$, and any output $y \in Range(\mathcal{M})$, we have

$$\Pr\left[\mathcal{M}(x) = y\right] \leq e^{t\epsilon} \cdot \Pr\left[\mathcal{M}(x') = y\right],$$

where $t\epsilon$ controls the level of indistinguishability between outputs of $\mathcal{M}(x)$ and $\mathcal{M}(x')$. The indistinguishability decreases as the distance t between x and x' increases.

2.2 Order-Preserving Desensitization Algorithm

In some application scenarios (e.g., recommender system, range query), the accuracy of the algorithm mainly depends on the order of the dataset. It would be desirable that the numerical order of sensitive data is somehow preserved after desensitizing. A lot of order-preserving desensitization algorithms are proposed in cryptographic studies [11, 15, 42, 43], in which the order is rigorously preserved after desensitization. The formal definition of the order-preserving desensitization algorithm is as follows.

Definition 2.3. (*Order-Preserving Desensitization Algorithm*). Denote $X = x_1, x_2, ..., x_n$ ($\forall i.x_i \in \mathbb{N}$) as the sensitive sequence, and $Y = y_1, y_2, ..., y_n$ ($\forall i.y_i \in \mathbb{N}$) be the noisy sequence output by a desensitization algorithm $\mathcal{R}$, where $y_i = \mathcal{R}(x_i)$. The algorithm $\mathcal{R}$ is order-preserving if and only if the following conditions are satisfied:

$$\forall i, j. \ x_i > x_j \Rightarrow y_i > y_j, \quad and$$
$$\forall i, j. \ y_i > y_j \Rightarrow x_i \geq x_j.$$

However, rigorous order itself could be leveraged by attackers to perform attacks (e.g., big-jump attack[15], inter-column correlation-based attack [28], multinomial attack [14], inference attack [41, 48]). These attacks use auxiliary information to estimate the distribution of the original values and then correlate them with the desensitized values based on their order. Besides, there is a lack of widely accepted cryptography tools to quantify how much privacy is compromised through attacks. The notion of differential privacy can help with these predicaments. It provides a rigorous upper bound for information disclosure and turns deterministic output into probabilistic results. Hence, we extend a relaxed version of the order-preserving notion called *Probabilistic Order-Preserving* in Definition 2.4.

Definition 2.4. (*Probabilistic Order-Preserving Desensitization Algorithm*). Denote $X = x_1, x_2, ..., x_n$ ($\forall i.x_i \in \mathbb{N}$) as the sensitive sequence, and $Y = y_1, y_2, ..., y_n$ ($\forall i.y_i \in \mathbb{N}$) be the noisy sequence output by a desensitization algorithm $\mathcal{R}$, where $y_i = \mathcal{R}(x_i)$. The algorithm $\mathcal{R}$ is probabilistic order-preserving if and only if the following conditions are satisfied:

$$\forall i, j. \ x_i > x_j \Rightarrow Pr[y_i > y_j] \geq \gamma(t),$$

where $\gamma(t) \in [0, 1]$ and $|x_i - x_j| \leq t$.

Here, γ is a function related to the distance between x_i and x_j. The definition satisfies rigorous order-preserving desensitization in Definition 2.3 when $\gamma(t) = 1$ for any t. The algorithms satisfying probabilistic order-preserving preserve the relative order of partial pairs of values rather than all pairs with $\gamma(t) < 1$. Specifically, the probabilistic order-preserving desensitization algorithms can be achieved by adding carefully selected random noise to sensitive values. Meanwhile, randomness can provide provable privacy guarantee to resist all aforementioned attacks based on auxiliary information. All proposed algorithms are probabilistic order-preserving. Moreover, we prove that these algorithms all satisfy dLDP.

2.3 Gradient Tree Boosting

The term "gradient tree boosting" originates from the paper by Friedman et al. [32]. Each iteration of training involves incrementally adjusting the gradient to fit the residual, with the goal of minimizing the loss function. There are some gradient tree boosting algorithms have been widely used such as GBDT [33], XGBoost [21], where XGBoost is an efficient implementation of GBDT.

We analyze the process of XGBoost building decision trees to help understand that only the order of features' values is necessary during the process of gradient tree boosting. The algorithm continually finds the split point with the greatest gain after splitting. We denote that I_L and I_R are the sample sets of left and right nodes after splitting, g_i and h_i are gradients, λ and ω are regularization parameters, the gain of the split is given by

$$G_{split} = \frac{1}{2}\Big[\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda}\Big] - \omega.$$

Note that all the variables we need for calculating G_{split} can be derived only from the order of features' values. Thus we can build the boosting tree without knowing the exact values of each feature. We can decide which node should a sample fall in based on this tree, but can not know the value that this node represents, which made it a "partial tree". The accurate predictions can be achieved with the assistance of the parties holding corresponding features.

3 SYSTEM OVERVIEW

3.1 Architecture

The training dataset is vertically partitioned and is distributed among different users' devices. Each user holds different features of samples but overlapping sample IDs and only one user holds labels. Since the label is essential for supervised learning, the user holding labels is generally the central node responsible for aggregating information and updating the model. Therefore, our framework focuses on guiding the information exchange between other users and the user holding labels, rather than sharing labels among all users. For the sake of simplicity, we divide users into two parties.
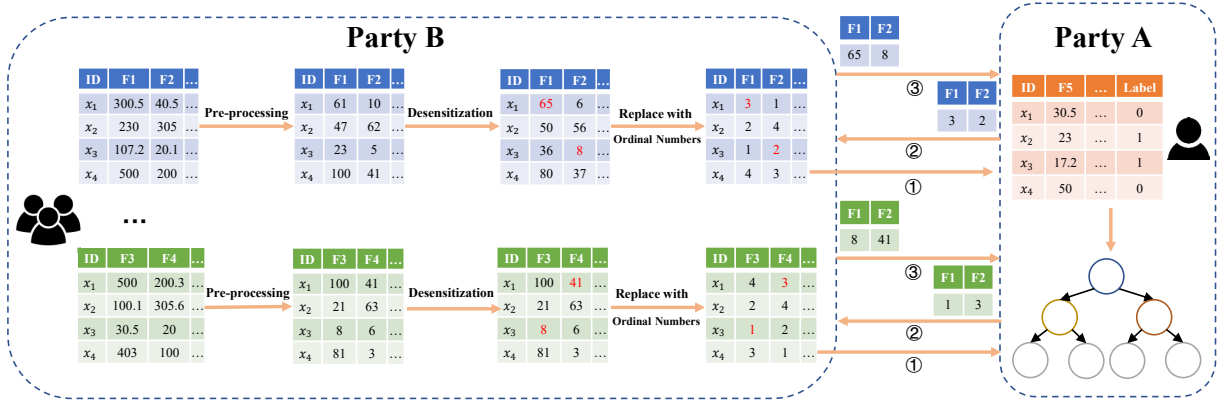
Figure 2: Training process of OpBoost.

Party A. Party A refers to the user who holds the label of the training samples. It may also hold several features. In the training process, Party A acts as a central server to exchange information with Party B and train the model. The trained model is only stored in Party A, and Party A is responsible for using it to predict the new samples.

Party B. We define Party B as the set of users who only hold several features of the samples. Party B acts as a client that participates in training by exchanging necessary information with Party A.

3.2 Execution Workflow

In the following, we describe the process of training decision trees with OpBoost in detail, and also explain how to predict with these trained decision trees.

Training Process. The overall process can be summarized into three steps, which are shown in Figure 2.

First, Party B desensitizes the local features before communicating with Party A, which is specified as follows.

- *Pre-processing for Feature Values.* We focus on numerical features and categorical features that have natural ordering between categories. The categorical features with no distance between values are not included, and the tree boosting algorithms handle them differently. These features are usually encoded by one-hot encoding, and the encoded values can be desensitized by existing LDP mechanisms [31, 56]. As the ordinal categorical values can be mapped to discrete numerical values, w.l.o.g., we assume that all features are numerical values, i.e., continuous or discrete numerical values. Besides, since the specific values do not affect the structure of the decision tree, it suffices to remap features coming with diverse distance metrics to a unified discrete data domain for the subsequent distance-based privacy-preserving algorithm to work with.
- *Desensitize Values with Order-preserving Desensitization Algorithm.* We design several order-preserving desensitization algorithms that satisfy dLDP, and give guidance to help Party B choose algorithm to desensitize features' values.
- *Replace the values with serial numbers.* Party B replaces all the desensitized values of features with their corresponding ordinal numbers and then sends them to Party A.

Second, Party A finds the best split points over the features after collecting all features' information from Party B. Specifically, Party A does not know the values of split points for features stored in Party B. It records the ordinal numbers as the split points.

Finally, Party A sends all order numbers of split points to Party B to get their desensitized values. Then Party B sends the specific desensitized values of corresponding split points back to Party A, and Party A updates the tree models.

Prediction. After the above training steps, Party A can obtain a complete decision tree model for predicting new samples. Party A can independently predict the new samples stored locally (non-private), or continue to cooperate with Party B to predict new samples (private). All the new samples need to do the same pre-processing as the training samples before desensitization or being input into the model.

Note that Party B is not required to be online all the time in both training and prediction procedures. It can go offline after sending all features' information and values of split points to Party A. In addition, Party A can utilize the trained decision tree to independently perform the prediction tasks with non-private samples or desensitized private samples.

4 PROPOSED ALGORITHMS

4.1 Pre-Processing for Feature Values

Since the training samples come from multiple parties, the tree boosting algorithms usually preprocess the values of all features before training, i.e., fulling the missing values, handling wrong values. In addition, we present an additional preprocessing step to improve the privacy and utility of desensitization algorithms. Firstly, there are some existing works that propose that implementing differential privacy mechanisms over floating-point numerical values is vulnerable to privacy attacks [38, 47]. Secondly, note that the privacy guarantee provided by OpBoost satisfies distance-based LDP. It's necessary to normalize the values of different features in a unified distance unit. To address these issues, we map numerical values of all different features into a unified discrete value domain. We show the details in the following.

We remap discrete numerical values by *Mapping Function $\mathcal{B}$* to a unified discrete value domain $\mathbb{D}_{\perp}$. Denote $\mathcal{X}_c \in \mathbb{D}$ as the set of

Table 1: Important Notations

Variable	Description
$\mathbb{D}$	Finite and numerical input data domain
$\mathbb{D}_\perp$	Finite and discrete data domain after mapping
L/R	Lower/Upper bound of $\mathbb{D}_\perp$
t	Distance between values in $\mathbb{D}_\perp$
θ	Length of a partition
$\mathcal{P}_m$	m^{th} partition of $\mathbb{D}_\perp$
$\epsilon_{prt}/\epsilon_{ner}$	Parameter for Adjusting the privacy budget between different partitions/within one partition
α	Ratio of ϵ_{prt} and $\theta\epsilon_{ner}$
γ	Lower bound of order-preserving probability

numerical values of a feature. Party B maps each value $\mathcal{X}_c^i$ as follows

$$\mathcal{X}_{int}^i = \lceil L + \frac{\mathcal{X}_c^i - lower}{upper - lower} \cdot (R - L) \rceil.$$

where $lower$ and $upper$ are lower bound and upper bound of $\mathbb{D}$, L and R are lower bound and upper bound of $\mathbb{D}_\perp$, respectively. The larger the domain $\mathbb{D}_\perp$ is, the more original relative orders the mapped values preserved. We take an example to explain why we need to map values of different features into a unified value domain. The values of *age* are usually in the range of (0, 100] years, while values of *salary* are usually in the range of (0, 100, 000] dollars/year. It is easy to see that the sensitivity of changing 10 years old to 30 years old is not the same as that from 2000 dollars/year to 2020 dollars/year, though the differences are the same. Therefore, it is necessary to map all values with different meanings to the same value domain to facilitate evaluating the privacy guarantee and the utility of the desensitized training dataset.

4.2 Order-preserving Desensitization Algorithms

Since preserving all order information of values is vulnerable to inference attacks, it is necessary to introduce randomness into the orders of desensitized values. Although desensitizing values with LDP mechanisms can resist privacy attacks, a lot of order information is lost, leading to the trained model's poor performance. Instead of achieving the same indistinguishability of any pair of values as LDP, the distance-based LDP (dLDP) provides different indistinguishability for pairs of values with different distances. The pairs of values with smaller distance is harder to be distinguished. While in the training of decision tree, the inverse of a pair of values with larger distance has a more significant impact on the trained model (split point of decision tree changes with higher probability). Besides, the public does not mind the indistinguishability between values with large distance in practice (e.g., an office worker does not mind the fact that he spends less time doing sports than an athlete being revealed). Therefore, the dLDP definition is more suitable for order-preserving desensitization than LDP.

In this subsection, we first take a variant of exponential mechanism [23, 36] satisfying dLDP as the preliminary algorithm, called Global-map. The probability that each value in the data domain is output as a desensitized value is inversely proportional to its distance from the sensitive value. We then do an in-depth study of Global-map from the definition of dLDP, and propose optimization algorithms for it. Note that the input values of all algorithms are

pre-processed by the algorithms proposed in Section 4.1. We denote $\mathbb{D}_\perp$ as the data domain of both input values and output values, and denote t as the distance of a pair of input values. The important notations are shown in Table 1.

Global-Map: Global-map is built upon a variant of exponential mechanism, which assigns the output probability to each value according to a score function. The score function can be defined as the distance between the value and sensitive value so that a value will be desensitized to a nearer value with a higher probability. The details of Global-map are shown in Algorithm 1. The privacy guarantee provided by Global-map is shown in Theorem 4.1.

Algorithm 1 Global-map

Require: $x \in \mathbb{D}_\perp$, parameter $\epsilon > 0$
Ensure: $o \in \mathbb{D}_\perp$
1: **for** $i \in \mathbb{D}_\perp$ **do**
2:      $p_{x,i} = Pr[o = i] = \frac{e^{-|x-i|\cdot\epsilon/2}}{\sum_{j\in\mathbb{D}_\perp} e^{-|x-j|\cdot\epsilon/2}}$
3: **end for**
4: Sample $o \sim p_x = \{p_{x,L}, \cdots, p_{x,R}\}$
5: **return** o

THEOREM 4.1. *Global-map provides ϵ-dLDP privacy guarantee for any pair of values $x, x' \in \mathbb{D}_\perp$, where $|x' - x| \le t$, and $t, \epsilon > 0$.*

PROOF. See Appendix A.1 in [9]. □

Then we analyze the utility of Global-map. We theoretically analyze the order-preserving degree of values after being desensitized by Global-map. According to the probabilistic order-preserving definition (Definition 2.4), we calculate the order-preserving probability γ of any pair of desensitized values. The result is shown in Theorem 4.2.

THEOREM 4.2. *Global-map is a probabilistic order-preserving desensitization algorithm with $\gamma(t) \ge 1 - \frac{(1-q^2)\cdot t+1}{(1+q-q^{t+1}-q^{|\mathbb{D}_\perp|-t})(1+q)} \cdot q^t$, where $q = e^{-\epsilon/2}$.*

PROOF. See Appendix A.2 in [9]. □

The LDP mechanism Generalized Random Response (GRR) [56] used in [52] can also be regarded as an probabilistic order-preserving desensitization algorithm. To show the advantage of dLDP mechanism in ordinal preservation compared with LDP mechanism, we also calculate the order-preserving probability γ of GRR. The result is shown in Theorem 4.3. In addition, we also provide an intuitive comparison and theoretical analysis in Appendix D.2 in [9].

THEOREM 4.3. *GRR is a probabilistic order-preserving desensitization algorithm with $\gamma(t) = p_1^2 + p_1p_2 \cdot (|\mathbb{D}_\perp|-3) + p_2^2 \cdot (\frac{1}{2}|\mathbb{D}_\perp|(|\mathbb{D}_\perp|-3) + 2) + p_2(p_1 - p_2)t$, where $p_1 = \frac{e^\epsilon}{|\mathbb{D}_\perp|+e^\epsilon-1}, p_2 = \frac{1}{|\mathbb{D}_\perp|+e^\epsilon-1}$.*

PROOF. See Appendix A.3 in [9]. □

Here, $p_2(p_1 - p_2)t < 1/((\mathbb{D}_\perp|+e^\epsilon-1)(\frac{1}{e^\epsilon-1} + \frac{1}{(|\mathbb{D}_\perp|)}))$, is usually small enough to be negligible, especially when $|\mathbb{D}_\perp|$ is large.

Adj-Map: We analyzed that it is more reasonable to assign the probability of mapping other values in domain $\mathbb{D}_\perp$ based on distance when desensitizing an ordered numerical value. However, we found that the existing dLDP definition allocates privacy budgets based
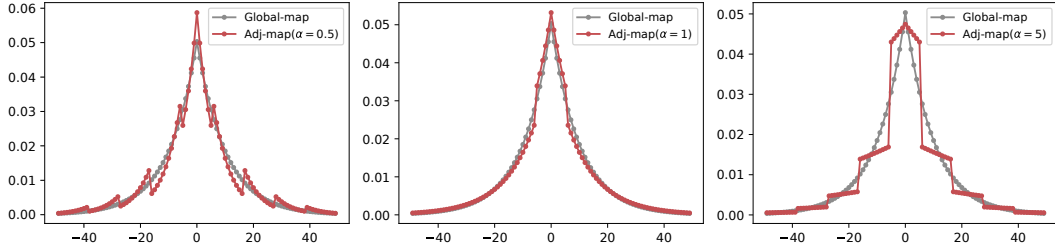
Figure 3: Adjust the ratio of ϵ_{prt} and ϵ_{ner}, where $\mathbb{D}_\perp$ is limited to $[-50, 50]$, $\epsilon = 0.1$, $\theta = 10$. The probability distribution of output values of Global-map achieved by exponential mechanism and Adj-map are almost fitted when $\alpha = 1$ ($\epsilon_{prt} = \theta \epsilon_{ner}$).

on l_1 distance between two values is not sufficient for satisfying all privacy requirements of feature values with rich semantics. Taking feature *age* as an example, people usually think that ages in the partition of $[1, 30]$ are young, in the partition of $[30, 60]$ are middle-aged, and those over 60 years old are elderly. Therefore, although the l_1 distances of value pairs $\{20, 25\}$ and $\{28, 33\}$ are all 5, 28 and 33 span two partitions. In other words, if a person's age is desensitized from 28 to 33, his identity changes from a young to a middle-aged person, while desensitization from 20 to 25 will not. If people don't care about which partition their ages belong to, but are more concerned about the indistinguishability of values from the same partition. The privacy definition should allocate more privacy budgets to value pairs located in different partitions, and allocate less privacy budget for values pairs located in the same partition. Obviously, the existing dLDP definition cannot achieve the above privacy budget allocation since l_1 distance cannot distinguish whether value pairs are in different partitions.

To address the above problem, we replace the privacy budget ϵ defined by existing dLDP with two parameters ϵ_{prt} and ϵ_{ner} to adjust the privacy budget between different partitions and within the same partition, respectively. We give a new distance-based local differential privacy definition *partition-dLDP*, which defines the protection strength based on the l_1 distance of value pairs and the number of partitions between them.

Definition 4.4. (Partition-dLDP). An algorithm $\mathcal{M}$ satisfies $(\epsilon_{prt}, \epsilon_{ner})$-partition-dLDP, if and only if for any input $x, x' \in \mathbb{D}_\perp$ such that $|x - x'| \le t$, $\mathbb{D}_\perp$ is equally divided into several partitions of length θ, and any output $y \in Range(\mathcal{M})$, we have

$$\Pr\left[\mathcal{M}(x) = y\right] \le e^{\lceil \frac{t}{\theta} \rceil \epsilon_{prt} + \theta \epsilon_{ner}} \cdot \Pr\left[\mathcal{M}(x') = y\right].$$

Compared with the existing dLDP definition, partition-dLDP can allocate privacy budget more finely based on both l_1 distance and partition distance. We find that the existing dLDP definition is just a special case when ϵ_{prt} and ϵ_{ner} satisfy the following relationship

$$\begin{cases} \epsilon_{prt} = \theta \epsilon_{ner} \\ \epsilon_{ner} = \frac{\epsilon}{1 + \theta / |\mathbb{D}_\perp|} \end{cases} \quad (1)$$

By adjusting ϵ_{prt} and ϵ_{ner}, we can get more kinds of probability distributions of desensitized values than the existing dLDP mechanisms. Therefore, it can meet a wider range of privacy policies and seek for achieving better utility. Note that the partition-dLDP in Definition 4.4 assumes that the feature can be evenly divided according to its semantic information. Some features may cannot

---

Algorithm 2 Mapping Partition of Adj-Map

Require: $x \in \mathbb{D}_\perp$, parameter $\theta \in \mathbb{N}^+$, $\theta \le |\mathbb{D}_\perp|$, $\epsilon_{prt} > 0$
Ensure: $\mathcal{P}_{\dot{m}} \in \{\mathcal{P}_1, ..., \mathcal{P}_k\}$
1: Partition $\mathbb{D}_\perp$ into k partitions: $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_k$, and x is located in the partition $\mathcal{P}_m$.
2: **for** $i \in [k]$ **do**
3: $\quad p_{x,i} = Pr[\dot{m} = i] = \frac{e^{-|m-i| \cdot \epsilon_{prt}/2}}{\sum_{j \in [k]} e^{-|m-j| \cdot \epsilon_{prt}/2}}$
4: **end for**
5: Sample $\dot{m} \sim p_x = \{p_{x,1}, p_{x,2}, ..., p_{x,k}\}$
6: **return** $\mathcal{P}_{\dot{m}}$

---

be evenly divided into intervals according to its semantics, i.e., the range of $[0, 60]$ is unqualified, $[60, 80]$ is qualified, and $[80, 100]$ is excellent for *score* feature. It suffices to only switch the partition strategy from evenly to unevenly, while the remaining processing of the proposed mechanisms in the paper can then be applied without any change. However, it requires further generalizing the privacy budget assignment strategy to account for the extended uneven partition, which we leave for future work.

In order to verify our theoretical analysis, we design a mechanism that satisfies partition-dLDP, called Adj-map. Instead of randomly mapping sensitive values based on distance in the whole domain $\mathbb{D}_\perp$, Adj-map first randomly selects a partition using Global-map satisfying ϵ_{prt}-dLDP as the output domain. The details are described in Algorithm 2. Then the sensitive value is randomly mapped to a value in this partition using Global-map satisfying ϵ_{ner}-dLDP. We then prove that Adj-map satisfies partition-dLDP in Theorem 4.5.

THEOREM 4.5. *Adj-map satisfies partition-dLDP.*

PROOF. See Appendix A.4 in [9]. □

In Figure 3, we show the output probability distribution of a sensitive value desensitized by Adj-map and Global-map when providing the approximately same privacy protection for any pair of values. The distribution of output probabilities between partitions and within partitions can be changed by adjusting the ratio of ϵ_{prt} and ϵ_{ner}. When $\epsilon_{prt} = \alpha \theta \epsilon_{ner}$, $\epsilon_{ner} = \frac{\epsilon}{\alpha + \theta / |\mathbb{D}_\perp|}$. Increasing α can make the desensitized value remains in the original partition with a greater probability. Meanwhile, the distribution of probabilities in each partition is more uniform. Thus, the probability of reverse order of value pair with large distance is reduced and the indistinguishability of the closed values is increased. Figure 3 also confirms

that when Equation 1 is satisfied, the probability distribution of output values of Adj-map and Global-map is almost entirely fitted. The Global-map satisfying existing dLDP is just a special case when ϵ_{prt} and ϵ_{ner} in Adj-map meet Equation 1. Then we theoretically show the order-preserving probability of Adj-map in Theorem 4.6.

Theorem 4.6. *Adj-map is a probabilistic order-preserving desensitization algorithm with* $\gamma(t) \geq 1 - q^T \left(\frac{(1-q^2)\cdot T+1}{(1+q-q^{T+1}-q^{k-T})(1+q)} - \frac{(1-q)^2(T+1)}{2(1+q)^2} \right)$, *where* $q = e^{-\epsilon_{prt}/2}, T = \lfloor \frac{t}{\theta} \rfloor$.

Proof. See Appendix A.5 in [9]. □

Local-Map: The sensitive value can be mapped to the partition where it is located with the greater probability when ϵ_{prt} increases. When $\epsilon_{prt} = \infty$, the output domain of the desensitized value is reduced from $\mathbb{D}_\perp$ to the partition where the sensitive value is located. Therefore, the sensitive value pairs in different partitions always remain the order after desensitization. Specifically, each sensitive value is desensitized by Global-map satisfying ϵ_{ner}-dLDP in the partition that it is located. We show the privacy guarantee of Local-map in Theorem 4.7.

Theorem 4.7. *The privacy guarantee provided by Local-map satisfies the following: (1) For any pair of values x, x' are in different partitions, x, x' can be distinguished.*
$$\exists o \in \mathbb{D}_\perp, Pr[O = o|x] \neq 0, Pr[O = o|x'] = 0.$$
(2) For any pair of values x, x' are in the same partition, x, x' satisfies ϵ_{ner}-dLDP, where $|x' - x| \leq t \leq \theta, \epsilon_{ner} > 0$.
$$\forall o \in \mathbb{D}_\perp, Pr[O = o|x] \leq e^{t \cdot \epsilon_{ner}} \cdot Pr[O = o|x'].$$

The order-preserving probability $\gamma(t) = 1$ for value pairs located in different partitions, while $\gamma(t)$ is the same as that of Global-map stated in Theorem 4.2 when value pairs are in the same partition. When providing the same privacy guarantee for value pairs in the same partition, Local-map can preserve more order information than Adj-map and Global-map. The finer the granularity of the partition, the more order information the desensitized values retain.

4.3 Improve Efficiency with (Bounded) Discrete Laplace

We have utilized a variant of the exponential mechanism to construct all our aforementioned order-preserving desensitization algorithms. Such constructions have two limitations. First, the computational complexity of sampling in an exponential mechanism is proportional to the size of the output domain. If the output domain is large, the sampling procedure becomes inefficient. Second, the exponential mechanisms require bounded input/output domains, and the bounds of the domain need to be known in advance. Although all values of a feature are held by one party, and the value domain is bounded and known in OpBoost, the proposed algorithms are difficult to desensitize distributed values in other complex scenarios. On the other hand, if the input domain is unknown in advance, one needs to collect all input values *before* determining desensitization parameters, introducing additional deployment requirements. Since releasing the output domain may violate privacy, other differentially private mechanisms would be introduced to determine the output domain (e.g., [58]) in a privacy-preserving manner.

In this section, we focus on the situations where the input domain is unknown in advance, or the output domain is large. We introduce (bounded) discrete Laplace mechanism in our order-preserving desensitization algorithms as an alternative to the exponential mechanism. The discrete Laplace mechanism supports unbounded input/output sampling so that the input/output domain can be treated as infinity. Instead of leveraging the Inverse Cumulative Distribution Function (Inverse CDF) sampling method as in the exponential mechanism, the sampling procedure for discrete Laplace mechanism is independent of the domain size and more efficient [18]. It remains to show that replacing the exponential mechanism with the discrete Laplace mechanism can still provide the privacy guarantee. We first consider the infinite input/output domain setting. We show the definition of discrete Laplace distribution in Definition 4.8.

Definition 4.8. (Discrete Laplace Distribution). The discrete Laplace distribution with scale parameter λ is denoted $Lap_{\mathbb{Z}}(\lambda)$, where $\lambda = 1/\epsilon$. Its probability distribution can be defined as
$$\forall z \in \mathbb{Z}, \; Pr[Z = z] = \frac{e^{1/\lambda} - 1}{e^{1/\lambda} + 1} \cdot e^{-|z|/\lambda}.$$

Then we show in Theorem 4.9 that adding noise following discrete Laplace distribution satisfies dLDP.

Theorem 4.9. *Any pair of values x_1 and x_2 with $|x_1 - x_2| \leq t$, satisfies ϵ-dLDP after adding the noise sampling from discrete Laplace distribution $Lap_{\mathbb{Z}}(\frac{1}{\epsilon})$.*

Proof. The probability ratio of x_1 and x_2 being randomized to the same output value o is
$$\frac{Pr[x_1 + N_1 = o]}{Pr[x_2 + N_2 = o]} = \frac{Pr[N_1 = o - x_1]}{Pr[N_2 = o - x_2]} = \frac{e^{-|o-x_1|/\lambda}}{e^{-|o-x_2|/\lambda}} \leq e^{t \cdot \epsilon}.$$
□

Theorem 4.9 shows that, in the infinite input/output domain setting, Global-map can still satisfy ϵ-dLDP when sampling the desensitized output values from the discrete Laplace distribution. Randomizing the partition in Adj-map using the discrete Laplace distribution also does not change the privacy guarantee of partition algorithms proved in Subsection 4.2.

We next consider the large (but finite) input/output domain setting, where the discrete Laplace mechanism can also be utilized as an alternative to the exponential mechanism. However, due to its unbounded output domain, we need to consider the case where the discrete Laplace mechanism samples a value outside the desired output domain. We solve this by resampling, and we call the mechanism as *bounded* discrete Laplace mechanism. We first give the probability distribution of the bounded discrete Laplace distribution obtained by resampling in Lemma 4.10.

Lemma 4.10. *Given the sampling range $[l, u]$, the probability distribution of the bounded discrete Laplace distribution is*
$$\forall z \in [l, u], \; Pr[Z = z] = \tau \cdot \frac{e^{1/\lambda} - 1}{e^{1/\lambda} + 1} \cdot e^{-|z|/\lambda},$$
where $\lambda = 1/\epsilon$, and
$$\tau = \begin{cases} \frac{2}{e^{u/\lambda}(1 - e^{-(u-l+1)/\lambda})}, & l < u < 0, \\ \frac{2}{1 - e^{-(-l+1)/\lambda} - e^{-(u+1)/\lambda} + e^{-1/\lambda}}, & l < 0 < u, \\ \frac{2}{e^{-l/\lambda}(1 - e^{-(u-l+1)/\lambda})}, & 0 < l < u. \end{cases}$$

PROOF. See Appendix A.7 in [9]. □

Next, we analyze the privacy guarantee of bounded discrete Laplace mechanism in Theorem 4.11.

THEOREM 4.11. *For any output range $[l, u]$, any pair of input values x_1 and x_2 with $|x_1 - x_2| \le t$ satisfies 2ϵ-dLDP after adding the noise sampling from bounded discrete Laplace distribution $Lap_{\mathbb{Z}}(\frac{1}{\epsilon})$.*

PROOF. See Appendix A.6 in [9]. □

Theorem 4.11 shows that for Global-map, sampling the output value from the bounded discrete Laplace distribution $Lap_{\mathbb{Z}}(\frac{2}{\epsilon})$ can provide ϵ-dLDP privacy guarantee. Also, for Adj-map, and Local-map, replacing the exponential distribution $Exp(\frac{1}{\epsilon})$ with bounded discrete Laplace distribution $Lap_{\mathbb{Z}}(\frac{2}{\epsilon})$ when mapping the partition and sampling output values in the partition does not affect the privacy guarantee, as shown in Corollary 4.12.

COROLLARY 4.12. *Sampling from the bounded discrete Laplace distribution $Lap_{\mathbb{Z}}(\frac{2}{\epsilon})$ instead of exponential mechanism has no effect on privacy guarantee provided by Adj-map and Local-map.*

One may also wonder if additional privacy budgets should be consumed when resampling occurs. Observe that if the output distribution satisfies the privacy guarantee, only the time consumption for sampling reflects if resampling happens. In the situation where the adversary has a very strong capability of carrying out the side-channel attack, we recommend considering extra privacy budget consumptions. In addition, the probability of resampling increases as the domain decreases. Therefore, we still recommend using the exponential mechanism for tasks with a small input/output domain.

5 THEORETICAL ANALYSIS

5.1 Utility Analysis of OpBoost

Here, we provide theoretical evidence for the utility of OpBoost.

THEOREM 5.1. *The probability that no desensitized values of a feature crosses any potential split point $x^{\dagger} \in [L, R]$ after sorting is at least β. where*

$$\beta = \sum_{k \in \mathcal{I}_l} \sum_{x^{\dagger} \in [L,R]} (\mathcal{M}(k, x^{\dagger}) \cdot \prod_{j \ne k, j \in \mathcal{I}_l} \sum_{x_l \in [L,x^{\dagger}]} \mathcal{M}(j, x_l)$$
$$\cdot \prod_{j \in \mathcal{I}_r} \sum_{x_r \in [x^{\dagger}+1,R]} \mathcal{M}(j, x_r)).$$

Here, $\mathcal{I}_l$ (resp. $\mathcal{I}_r$) is the set of feature values on the left (resp. right) of the split point. The function $\mathcal{M}(x, y)$ outputs the probability of desensitizing the value x to y with a desensitization algorithm. β contains the probability that any value k in $\mathcal{I}_l$ is desensitized as the left maximum value $x^{\dagger}$ and the other values $x_l \ne k$ in $\mathcal{I}_l$ are all lower than $x^{\dagger}$, the values x_r in $\mathcal{I}_r$ are all greater than $x^{\dagger}$ after desensitizing.

In Theorem 5.1, the distribution and density of the feature values and the location of the split point are intertwined to influence the value of β. To provide theoretical evidence, we have to analyze β under the condition that all these factors are controllable. We choose two commonly used distributions to calculate β when the split point is located at 25% and 50% quantiles. Figure 4 shows the comparison of β between the proposed order-preserving desensitization algorithms and the LDP mechanism. The denser or more
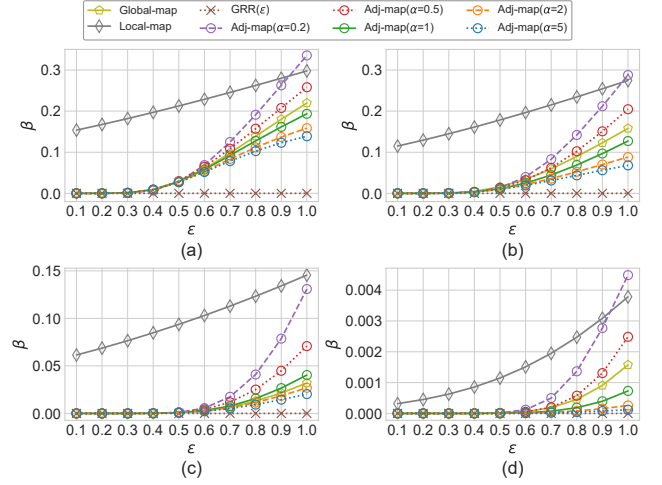

Figure 4: Comparison of β calculated by the algorithms on a uniformly and a normal distributed datasets with 100 values, where $\theta = 4$, $|\mathbb{D}_\perp| = 100$. (a)(b) are on uniform datasets, and (c)(d) are on normal datasets. (a)(c) and (b)(d) are the split points at the 25% and 50% quantiles, respectively.

centralized the distribution of the datasets, the higher the number of disordered value pairs after desensitization, thus reducing the utility of the desensitized values. Besides, we observe that β decreases as the split point approaches the median of the feature values. When the split point at 25% of the feature values, β can reach 10 times that of the split point at 50% of the feature values for the normal dataset. Although many factors have a great impact on β, the proposed algorithms are always better than the LDP method.

COROLLARY 5.2. *Let G be the maximum gain when splitting a feature, and $\hat{G}$ be the maximum gain after desensitizing feature's values. We have*

$$Pr[\hat{G} \ge G] \ge \beta.$$

When the optimal split point with maximum gain G is chosen in Theorem 5.1, we can give a theoretical lower bound of the probability that the maximum gain $\hat{G}$ is not less than G after desensitizing as shown in Corollary 5.2. Given the fact that the exhaustive split point searching will traverse all potential split points, it is guaranteed that the current split point obtained with desensitizing always has the largest maximum gain, because otherwise, we can find another split point with an even greater maximum gain during the exhaustive search. Intuitively, we provide a probability lower bound that the accuracy of the model trained based on desensitized features is not lower than that of the plain model.

5.2 Efficiency Analysis of OpBoost

We analyze the computational and communicational complexities of OpBoost to provide the overhead in theory. Without loss of generality, we consider the setting consisting of m-1 users in party B and one user in Party A, where each user holds at most r features with N total training samples.

THEOREM 5.3. *The total computation overhead for each user in party B is $O(Nr)$, and $O(NT + (2^L − 1)T)$ for the user in Party A, where T is the number of trees, L is the number of tree layers.*

PROOF. See Section 5.2 in [9]. □

To show the efficiency of the proposed desensitization algorithms, we compare them with an Order-Preserving Encryption (OPE) scheme. Our proposed algorithms are 40-200 times faster than OPE desensitizing the same value set (See Appendix D.1 in [9]). The training process is the same as that of the non-private tree boosting algorithm [21, 33].

THEOREM 5.4. *The communication channel of OpBoost needs to transmit $O(mrN \log N + (2^L − 1)T(\log N + \log |\mathbb{D}_\perp|))$ bits in total, where T is the number of trees, L is the number of tree layers.*

PROOF. See Section 5.2 in [9]. □

6 EXPERIMENTAL EVALUATION

In this section, we empirically evaluate the performance of OpBoost. In experiments, we measure (1) the order preservation of a numerical dataset after desensitizing by the proposed order-preserving desensitization algorithms, i.e., how much relative order information between values is preserved, the impact of the ratio of privacy budget parameters on the order preservation of desensitized results. (2) the accuracy of the proposed order-preserving desensitization algorithms for order-dependent statistical applications, i.e., how do they compare with the state-of-the-art LDP-based range query method in accuracy. (3) the performance of decision tree models trained by OpBoost, i.e., how much does desensitization of features' values affect the accuracy of trained models; What is the computation and communication overhead of OpBoost during the training. Towards these goals, we run the GBDT and XGBoost training algorithm in OpBoost for both classification and regression tasks. We let $T = 80$ (number of trees), $\eta = 0.1$ (learning rate), and $L = 3$ (tree layers) throughout the experiment.

6.1 Setup

Datasets. We conduct range query evaluation using a real-world dataset *Salaries* [6] and a synthetic dataset. *Salaries* [6] is also considered in our competitor AHEAD [26], which contains $148, 654$ records. We follow the same practice with AHEAD to map values into the range of $[1, 1024]$ for the fairness. We conduct GBDT and XGBoost on four public datasets and a large-scale industrial dataset, which are listed in Table 2. For datasets with all samples in one file, we randomly select 80% and 20% samples for training and testing.

Table 2: Description of datasets, where N and E refer to Numeric and Enumeration, respectively.

Dataset	#Instances	#Features(N/E)	Tasks
Adult[1]	32.6k	4/10	2-Cls.
Pen-digits[2]	11k	16/0	M-Cls.
Powerplant[4]	9.5k	4/0	Reg.
CASP[3]	45.7k	9/0	Reg.
Industrial	293.6k	38/262	Reg.

Metrics. The metrics we use are as follows.

Weighted-Kendall. We use weighted-Kendall as the metric to evaluate the order preservation of the desensitized values. The formal definition of weighted-Kendall is given by Vigna [53].

Definition 6.1. (Weighted-Kendall). For two real-valued vectors r and s, weighted-kendall $\tau_w(r, s)$ is defined as

$$\frac{\langle r, s \rangle_w}{\sqrt{\langle r, r \rangle_w}\sqrt{\langle s, s \rangle_w}}, where \; sgn(x) := \begin{cases} 1 & if \; x > 0; \\ 0 & if \; x = 0; \\ -1 & if \; x < 0. \end{cases}$$

$\langle r, s \rangle_w = \sum_{i<j} sgn(r_i − r_j)sgn(s_i − s_j)w(i, j)$,

$\tau_w(r, s) = 1$ means that the vector is strictly order-preserving, while $\tau_w(r, s) = −1$ means that the vector is completely reversed. In experiments, we consider the vector r as the raw sensitive values and the vector s as the desensitized values. We define the weight function $w(i, j)$ as the distance of ordinal numbers between r_i and r_j after sorting the vector r.

Accuracy. We evaluate the accuracy of the prediction results of the model for *classification* as following

$$Accuracy = \frac{\#Correct \; Predictions}{\#Correct \; Predictions + \#Wrong \; Predictions}\%.$$

Mean Square Error. We use MSE as the metric to evaluate the statistical results of range query and the prediction results of the model for *regression*. For n testing samples, we calculate the squared difference between each prediction result $\tilde{y}_i$ from OpBoost and the corresponding results y_i from model without privacy protection.

$$MSE = \frac{1}{n} \sum_{i \in [n]} (\tilde{y}_i − y_i)^2.$$

The results of range query tasks are averaged with 100 repeats, and the results of other tasks are all averaged with 10 repeats.

Environment. We do our experiments on a single Intel Core i9-9900K with 3.6GHz and 4×32GB RAM, running Ubuntu 20.04.2 LTS. We execute our protocols on two progresses, one for Party A and the other for Party B. The network connection between the two progresses are built via the local network. We emulate the LAN network setting with latency 0.02ms and bandwidth 10Gbps using the Linux tc command. Our schemes are implemented in Java with the multi-thread support by the Fork-Join concurrency technique. While the source code is based on Java 8, we run experiments on Java HotSpot(TM) with higher version 17.0.2.

6.2 Experimental Details

Range Query. Our algorithms are not optimized by any data structures, while our competitor AHEAD uses a complex tree structure to improve the utility. We randomly generate $10k$ queries in the value range, and calculate the MSE of the frequency of real values and desensitized values falling within the query ranges.

GBDT and XGBoost. All experiments are conducted with two parties' participation (any number of parties is supported), Party A cooperates with Party B. We assign the enumeration features and label to Party A and distribute all the numerical features to Party B. Except for the range query tasks, all preprocessed values are mapped to the range of $[1, 10]$. We use the following libraries in our implementations.

GBDT. We use the codebase in Smile[8] to implement our federated GBDT training. Smile provides data abstraction 'DataFrame' to
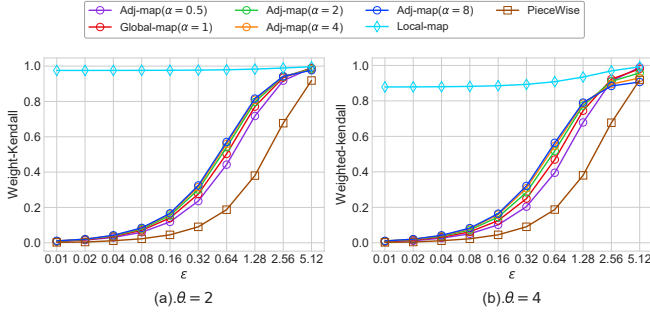
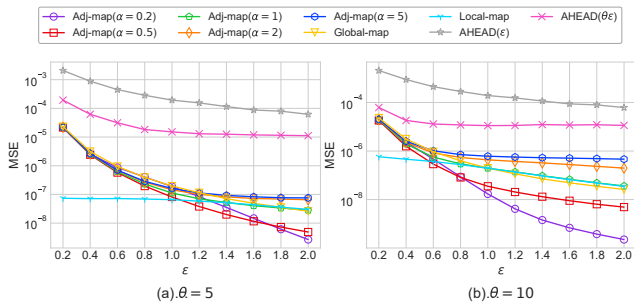Figure 5: Weighted-Kendall on *age* of Adult Dataset Containing $32.6k$ Age Values.

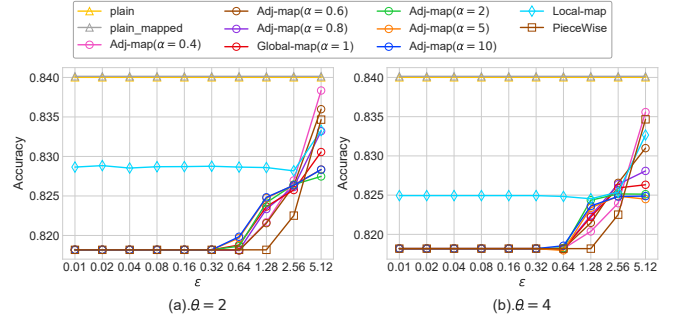Figure 6: The MSE of Range Query on Salaries Dataset.

Figure 7: Prediction Precision of GBDT Models for Classification Trained on Adult Dataset.

Figure 8: Prediction Precision of GBDT Models for Classification Trained on Pen-digits Dataset.

allow us easily read data from files, add noises, encode/decode the randomized data for communication, and do the training.
XGBoost. We also do the implementation based on the well-known XGBoost library[7]. We use the XGBoost JVM package to invoke XGBoost from Java. XGBoost model modification is implemented using xgboost-predictor[5]. Besides, we adjust the code 'Node' in 'RegTreeImpl.java' to allow split condition replacement.

6.3 Experimental Results

We evaluate the proposed algorithms from three perspectives, including 1) the order preservation capability under the given privacy requirements, 2) the utility for range query tasks, and 3) the utility for tree boosting algorithms. We not only compare all the proposed algorithms under the different parameter settings, but also compare them with the state-of-the-art LDP mechanism in the corresponding tasks. Since the privacy definitions are different, it is impossible to compare the LDP algorithms and the proposed dLDP algorithms under the same privacy guarantee, which are the same case for the existing dLDP-related works [12, 37, 55]. We unify Global-map and Adj-map under the same ϵ-dLDP privacy guarantee. Specifically, we correspondingly set ϵ_{ner} to $\epsilon/(\alpha + \theta/|\mathbb{D}_\perp|)$ when we increase ϵ_{prt} to $\alpha\theta\epsilon_{ner}$ to ensure that Adj-map algorithms with different parameters are all provide ϵ-dLDP privacy guarantee. Moreover, Adj-map can be approximated as Global-map when $\alpha = 1$. Since Local-map does not provide privacy guarantee for values located in different partitions, it only provides ϵ-dLDP privacy guarantee for values in the same partitions.

Order Preservation of Desensitized Values. We first evaluate the order preservation capability of the proposed order-preserving desensitization algorithms. Although we evaluate the order preservation of a pair of values separated by different distances after desensitization in the previous section, it's not enough to reflect the overall order preservation of the entire desensitized value set. Because the order-dependent applications generally rely on the order of the entire value set, and the farther away the values are out of order, the greater the impact on the results. We selected values of a feature *age* from the Adult dataset, and generate a uniformly distributed dataset with $10k$ values to evaluate the desensitized value sets by calculating the weighted-Kendall. Note that the reason why a uniformly distributed value set is generated here is to facilitate the comparison of the performance of different algorithms on order preservation. Since most of the value pairs in the unevenly distributed value set are not easily out of order after desensitization, it cannot reflect the improvement of the order preservation brought by optimizing desensitized value distribution.

Figure 5 and Figure 11 (See Appendix D.3 in [9]) show that the order preservation capability of Local-map is always better than that of other algorithms since it ensures that the orders of values in different partitions are strictly preserved. In addition, we observe that when ϵ is small, i.e., $\epsilon < 2.56$, Adj-map with a larger α can achieve better order preservation, and the opposite when $\epsilon > 2.56$. The reason is that the desensitized values fall far away from the raw values with a high probability when ϵ is relatively
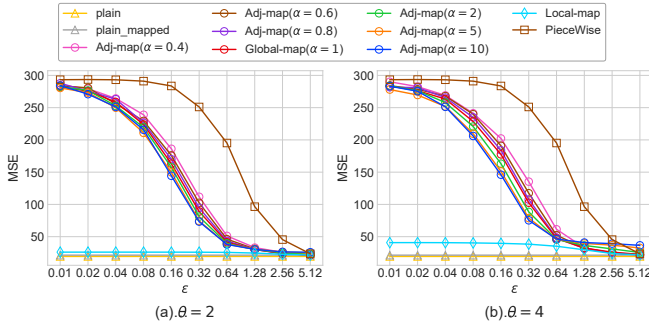
Figure 9: Prediction MSE of GBDT Models for Regression Trained on Powerplant Dataset.

Table 3: Performance of XGBoost models trained by OpBoost on industrial dataset with $\theta = 2$. We show the MSE ratio compared to the plain model.

ϵ	Method	MSE	Time(s)	Communication(B)	
				Party A	Party B
0.64	Local-map	0.957×	64.929	2961	202M
	Adj-map($\alpha = 0.5$)	5.341×	99.183	2921	202M
	Global-map($\alpha = 1$)	5.273×	98.448	2981	202M
	Adj-map($\alpha = 2$)	5.122×	97.956	3113	202M
	Piecewise	5.773×	115.771	3097	202M
1.28	Local-map	0.951×	64.207	2945	202M
	Adj-map($\alpha = 0.5$)	4.803×	92.915	3137	202M
	Global-map($\alpha = 1$)	4.792×	92.108	3413	202M
	Adj-map($\alpha = 2$)	4.295×	91.495	3373	202M
	Piecewise	6.150×	110.488	3717	202M
2.56	Local-map	0.952×	63.300	2901	202M
	Adj-map($\alpha = 0.5$)	1.063×	86.247	3681	202M
	Global-map($\alpha = 1$)	1.002×	86.278	3473	202M
	Adj-map($\alpha = 2$)	1.015×	86.387	3525	202M
	Piecewise	6.150×	103.826	3053	202M

small. A large α allocates more privacy budget to ϵ_{prt} so that the desensitized value is preserved in the partition where the raw value is located with a greater probability. When ϵ is relatively large, the desensitized values are preserved in the raw values' partition with a high probability. At this time, allocating more privacy budget to ϵ_{ner} to increase the order-preserving probability between values in the same partition is more helpful. All the proposed algorithms are superior to the Piecewise mechanism [54], which is a widely accepted LDP mechanism for randomizing numerical values.

Utility Comparison for Range Query Tasks. Figure 6 and Figure 12 (See Appendix D.4 in [9]) show the results of the proposed algorithms are applied to the range query tasks. We compare with the state-of-the-art LDP-based algorithm AHEAD to demonstrate the utility improvement of the proposed algorithms on order-dependent statistical tasks by relaxing the privacy definition. We use the same real-world dataset and data mapping method as AHEAD for the fairness of the comparison. Consistent with the results of weighted-Kendall, $\alpha < 1$ can get a smaller error than $\alpha > 1$ when ϵ is large. This advantage is obvious when the length of the partition is large. Besides, we observe that $\alpha < 1$ dominates earlier when $\theta = 10$ than when $\theta = 5$. The reason is that when the partition is relatively large,

it is difficult for the randomly generated query ranges to cover the complete partitions, which causes the order preservation of the values in the partition to have a greater impact on the results.

Performance of OpBoost for Tree Boosting Tasks. We then evaluate the performance of OpBoost for tree boosting tasks. We run both GBDT and XGBoost in OpBoost utilizing different order-preserving desensitization algorithms. Binary classification tasks, multi-classification tasks, and regression tasks are all covered. As before, the Piecewise mechanism is also conducted for comparison. Considering that data pre-processing can also cause some loss of prediction accuracy to the trained model, we also compare the prediction accuracy of models trained by the pre-processed plaintext dataset and the plaintext dataset without any processing.

Prediction Accuracy of GBDT and XGBoost. Figure 7 shows the results of GBDT running in OpBoost for binary classification task on Adult datasets. Since more than 75% samples are labeled *Negative*, thus high prediction accuracy can be obtained as long as the model tends to output *Negative*. Therefore, the results in Figure 7 show that adding noise to the dataset does not significantly reduce the prediction accuracy of the model. Compared with other algorithms, Local-map always maintains obvious advantages when ϵ is small, i.e., $\epsilon < 2.56$. We think it is because the values desensitized by other algorithms are more likely to fall far away from raw values when ϵ is small, while Local-map keeps the desensitized values within the partitions where the raw value are located. The desensitized values are kept in the partitions of raw values with a high probability when ϵ is large, so Adj-map that allocates more privacy budget to the ϵ_{ner} can be more dominant. This also explains why a large α is better for Adj-map when $\epsilon < 2.56$, and a small α is better when $\epsilon > 2.56$. Besides, we observe that the effect of θ on the results is not obvious, and θ and α both affect ϵ_{ner} and ϵ_{prt} in the same direction according to Equation 1. Similar observations are also obtained from the results of the multi-classification and regression tasks. Figure 8 shows the results of GBDT running in OpBoost for multi-classification task on Pen-digits dataset. Figure 9 and Figure 13 (See Appendix D.5 in [9]) show the results of regression task on Powerplant dataset and CASP dataset. All proposed order-preserving desensitization algorithms outperform the Piecewise mechanism in accuracy in all tasks.

We also conduct all tasks on XGBoost with the same datasets as GBDT (See Appendix D.6 in [9]). Besides, we test OpBoost with XG-Boost on an industrial large-scale regression dataset. Different from other previous benchmark datasets, this industrial dataset contains a large number of categorical features with no order between values. We use one-hot encoding to encode all categorical values and then randomize them with Unary Encoding (UE) [56]. The results are summarized in Table 3. We show the ratio of the MSE of models trained with the proposed algorithms and the model trained by the unprocessed plaintext dataset when $\theta = 2$. The observations of results are all consistent with GBDT. Note that the prediction accuracy of the model trained by dataset desensitized by Local-map is even higher than that of the plain model. Such phenomenon is also observed in other tree boosting frameworks [45, 46, 52]. A possible reason is that the desensitization introduces a certain amount of randomness, which improves the model generalization. Theorem 5.1 also show that the maximum gain of the desensitized feature is larger than the raw feature with a certain probability.

Communication and Computation overhead of GBDT and XGBoost.
We record the training time and communication overhead of running GBDT and XGBoost in OpBoost on each dataset. The results of large-scale dataset are shown in Table 3, and the results of other datasets are shown in Appendix D.7 in [9]. We find that the training time increases with the decrease of ϵ, because smaller ϵ incurs more times of resampling the bounded discrete Laplacian noise. To verify such a trend, we further compare the training time of sampling with exponential mechanism and bounded discrete Laplace in the partitions when θ is small. Moreover, we observe that as ϵ decreases, the number of split points that Party A needs to request from Party B decreases. We think the reason is that desensitizing a feature may reduce the maximum gain of its optimal split point. Therefore, the model is inclined to find the split point on features on the Party A side when features on Party B are desensitized with a small ϵ.

7 DISCUSSION

7.1 Multiple Features Desensitization

Although we focus on desensitizing a single numerical feature when presenting the proposed order-preserving desensitization algorithms for easier exposition, these algorithms can also be extended to the multiple features case. For one example, we can split the privacy budget among the feature and apply the proposed algorithms for each feature with its share of the privacy budget. In existing LDP studies dealing with multidimensional numerical values, evenly splitting the privacy budget can achieve satisfactory performance [24, 27, 49, 54, 57]. For the other example, we can also randomly sample some features and only allocate the privacy budget to the sampled features, while the un-sampled features are not disclosed to the aggregator. The second exemplary solution can obtain higher utility than the first one [49, 54, 57].

After applying the proposed algorithms for each feature, it suffices to apply the composition theorem to ensure that the overall processing for multidimensional values satisfies dLDP with a given privacy budget. Although dLDP and partition-dLDP relaxes the definition of LDP based on the distance, they still satisfy the sequential composition theorem [30] (See Appendix B in [9]).

7.2 Setting of privacy parameters

In summary, we propose three order-preserving desensitization algorithms: Global-map, Local-map, and Adj-map. Local-map and Adj-map support setting different α and θ when given a privacy budget ϵ. All three algorithms are contained in OpBoost. Next, we give some guidelines on choosing these algorithms when using OpBoost in actual training tasks.

(1) *Utility prioritized users.* For all tasks, Local-map is always the best choice for users with less concern about the indistinguishability of values in different partitions. (2) *Privacy prioritized users.* The users without extra domain knowledge and who do not pursue high accuracy of the trained model could directly choose Global-map to avoid considering parameters' settings other than ϵ. For users with extra domain knowledge and higher utility pursuit, they can choose Adj-map and set α and θ to obtain higher model accuracy than Global-map. The length of the partition θ is usually determined by the semantic information of the features in practice. We show in subsection 5.1 that the distribution and density of feature

values and the values of labels all affect the maximum gain when finding the split point. Besides, the training dataset usually contains multiple features, which together affect the trained model. The distribution and density of each feature are different, and the influence on the trained model is also different. Therefore, it's difficult to theoretically give an optimal parameter setting. But we can empirically give some suggestions on settings of α based on our experimental evaluation. In all our experimental datasets, $\alpha < 1$ is dominant only when ϵ is large, i.e., $\epsilon > 2.56$. As ϵ should usually be set less than 1 or even 0.1 in practical applications, users can always set $\alpha > 1$. Besides, the results show that when $\alpha = 10$, there is no obvious improvement, so it is not necessary to set α too large.

8 RELATED WORKS

The extended version of related works is in Appendix C in [9].
Distance-based LDP (dLDP). The formal dLDP definition is first proposed and applied in Location-Based Systems to guarantee location privacy within a specific distance [13, 61]. Following the intuition of d_χ-privacy in [19], Alvim et al. [12] define Metric-LDP, a variant of dLDP. Afterward, dLDP shows its broad applicability in vast scenarios [16, 20, 36, 50, 55, 60]. However, the potential capabilities of dLDP in ordinal information preserving remain undiscussed.
Privacy-Preserving Tree Boosting on Vertical FL. Traditional tree boosting algorithms have drawn privacy concerns for their direct access to raw datasets. In cryptography-based schemes [22, 34], the parties exchange gradients and hessians encrypted with HE, which is extremely time-consuming. Although the MPC-based schemes[10, 59] avoid complex cryptographic operations, the massive communication overhead caused by MPC is unbearable. To solve this problem, Tian et al. [52] propose a scheme based on Local Differential Privacy (LDP). Since the randomness introduced by LDP, the accuracy of the trained model is not satisfying.

9 CONCLUSION

In this paper, a novel framework called OpBoost is proposed for privacy-preserving vertical federated tree boosting. The privacy notion of dLDP is firstly applied in vertical federated tree boosting tasks. It is shown that the prediction accuracy of the model trained by OpBoost is much higher than that of the LDP-based scheme. Meanwhile, the computational and communication overheads of OpBoost are significantly lower than cryptography-based schemes. Specifically, we optimize the existing dLDP definition and instantiate three order-preserving desensitization algorithms for OpBoost. We also study and apply (bounded) discrete Laplace distribution as an alternative sampling distribution, which further reduces the computational overhead. Finally, we conduct a comprehensive evaluation to show the effectiveness and efficiency of OpBoost.

REFERENCES

[1] 1996. Adult Data Set. https://archive.ics.uci.edu/ml/datasets/Adult.
[2] 1998. Pen-Based Recognition of Handwritten Digits Data Set. https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits.
[3] 2013. Combined Cycle Power Plant Data Set. https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure.
[4] 2014. Combined Cycle Power Plant Data Set. https://archive.ics.uci.edu/ml/datasets/combined+cycle+power+plant.
[5] 2018. Xgboost-Predictor-JAVA. https://github.com/h2oai/xgboost-predictor.
[6] 2019. SF Salaries Data Set. https://www.kaggle.com/datasets/kaggle/sf-salaries.
[7] 2021. Scalable, Portable and Distributed Gradient Boosting (GBDT, GBRT or GBM) Library, for Python, R, Java, Scala, C++ and more. Runs on single machine, Hadoop, Spark, Dask, Flink and DataFlow). https://github.com/dmlc/xgboost.
[8] 2021. Smile (Statistical Machine Intelligence and Learning Engine). https://github.com/haifengl/smile.
[9] 2022. OpBoost: A Vertical Federated Tree Boosting Framework Based on Order-Preserving Desensitization (full version). https://arxiv.org/abs/2210.01318.
[10] Mark Abspoel, Daniel Escudero, and Nikolaj Volgushev. 2021. Secure training of decision trees with continuous attributes. *Privacy Enhancing Technologies Symposium (PETS)* 2021, 1 (2021), 167–187.
[11] Rakesh Agrawal, Jerry Kiernan, Ramakrishnan Srikant, and Yirong Xu. 2004. Order preserving encryption for numeric data. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data(SIGMOD)*. 563–574.
[12] Mário Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. 2018. Local differential privacy on metric spaces: optimizing the trade-off with utility. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 262–267.
[13] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security (CCS)*. 901–914.
[14] Vincent Bindschaedler, Paul Grubbs, David Cash, Thomas Ristenpart, and Vitaly Shmatikov. 2018. The tao of inference in privacy-protected databases. *Proceedings of the VLDB Endowment (VLDB)* 11, 11 (2018), 1715–1728.
[15] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'neill. 2009. Order-preserving symmetric encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt)*. Springer, 224–241.
[16] Christian Borgs, Jennifer Chayes, Adam Smith, and Ilias Zadik. 2018. Revealing network structure, confidentially: Improved rates for node-private graphon estimation. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 533–543.
[17] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.
[18] Clément Canonne, Gautam Kamath, and Thomas Steinke. 2020. The discrete gaussian for differential privacy. *arXiv preprint arXiv:2004.00010* (2020).
[19] Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the scope of differential privacy using metrics. In *International Symposium on Privacy Enhancing Technologies Symposium (PETS)*. Springer, 82–102.
[20] Konstantinos Chatzikokolakis, Ehab Elsalamouny, and Catuscia Palamidessi. 2017. Efficient utility improvement for location privacy. *Proceedings on Privacy Enhancing Technologies (PoPETs)* 2017, 4 (2017), 308–328.
[21] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (SIGKDD)*. 785–794.
[22] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. 2021. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems* (2021).
[23] Amrita Roy Chowdhury, Bolin Ding, Somesh Jha, Weiran Liu, and Jingren Zhou. 2020. Intertwining Order Preserving Encryption and Differential Privacy. *arXiv preprint arXiv:2009.05679* (2020).
[24] Jean-François Couchot, Héber Hwang Arcolezi, Bechara Al Bouna, and Xiaokui Xiao. 2021. Random Sampling Plus Fake Data: Multidimensional Frequency Estimates With Local Differential Privacy. In *International Conference on Information and Knowledge Management (CIKM)*.
[25] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. 2018. CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363* (2018).
[26] Linkang Du, Zhikun Zhang, Shaojie Bai, Changchang Liu, Shouling Ji, Peng Cheng, and Jiming Chen. 2021. AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1266–1288.
[27] John C Duchi, Michael I Jordan, and Martin J Wainwright. 2018. Minimax optimal procedures for locally private estimation. *J. Amer. Statist. Assoc.* 113, 521 (2018), 182–201.

[28] F Betül Durak, Thomas M DuBuisson, and David Cash. 2016. What else is revealed by order-revealing encryption?. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1155–1166.
[29] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation (TAM)*. Springer, 1–19.
[30] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science (TCS)* 9, 3–4 (2014), 211–407.
[31] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security (CCS)*. 1054–1067.
[32] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2000. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics (ANN STAT)* 28, 2 (2000), 337–407.
[33] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics (ANN STAT)* (2001), 1189–1232.
[34] Fangcheng Fu, Yingxia Shao, Lele Yu, Jiawei Jiang, Huanran Xue, Yangyu Tao, and Bin Cui. 2021. VF2Boost: Very Fast Vertical Federated Gradient Boosting for Cross-Enterprise Learning. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD)*. 563–576.
[35] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems (NIPS)* 34 (2021).
[36] Mehmet Emre Gursoy, Acar Tamersoy, Stacey Truex, Wenqi Wei, and Ling Liu. 2019. Secure and utility-aware data collection with condensed local differential privacy. *IEEE Transactions on Dependable and Secure Computing (TDSC)* (2019).
[37] Xi He, Ashwin Machanavajjhala, and Bolin Ding. 2014. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data (SIGMOD)*. 1447–1458.
[38] Christina Ilvento. 2020. Implementing the exponential mechanism with base-2 differential privacy. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 717–742.
[39] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What can we learn privately? *SIAM J. Comput.* 40, 3 (2011), 793–826.
[40] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems (NIPS)* 30 (2017), 3146–3154.
[41] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'neill. 2016. Generic attacks on secure outsourced databases. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1329–1340.
[42] Florian Kerschbaum. 2015. Frequency-hiding order-preserving encryption. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 656–667.
[43] Florian Kerschbaum and Axel Schröpfer. 2014. Optimal average-complexity ideal-security order-preserving encryption. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 275–286.
[44] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).
[45] Qinbin Li, Zeyi Wen, and Bingsheng He. 2020. Practical federated gradient boosting decision trees. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, Vol. 34. 4642–4649.
[46] Qinbin Li, Zhaomin Wu, Zeyi Wen, and Bingsheng He. 2020. Privacy-preserving gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 34. 784–791.
[47] Ilya Mironov. 2012. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security (CCS)*. 650–661.
[48] Muhammad Naveed, Seny Kamara, and Charles V Wright. 2015. Inference attacks on property-preserving encrypted databases. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 644–655.
[49] Thông T Nguyên, Xiaokui Xiao, Yin Yang, Siu Cheung Hui, Hyejin Shin, and Junbum Shin. 2016. Collecting and analyzing data from smart device users with local differential privacy. *arXiv preprint arXiv:1606.05053* (2016).
[50] Reza Shokri. 2014. Privacy games: Optimal user-centric data obfuscation. *arXiv preprint arXiv:1402.3426* (2014).
[51] Liwei Song, Reza Shokri, and Prateek Mittal. 2019. Membership inference attacks against adversarially robust deep learning models. In *2019 IEEE Security and Privacy Workshops (SPW)*. IEEE, 50–56.
[52] Zhihua Tian, Rui Zhang, Xiaoyang Hou, Jian Liu, and Kui Ren. 2020. Federboost: Private federated learning for gbdt. *arXiv preprint arXiv:2011.02796* (2020).
[53] Sebastiano Vigna. 2015. A weighted correlation index for rankings with ties. In *Proceedings of the 24th international conference on World Wide Web (WWW)*. 1166–1176.

[54] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. 2019. Collecting and analyzing multidimensional data with local differential privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 638–649.

[55] Shaowei Wang, Yiwen Nie, Pengzhan Wang, Hongli Xu, Wei Yang, and Liusheng Huang. 2017. Local private ordinal data distribution estimation. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 1–9.

[56] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*. 729–745.

[57] Teng Wang, Jun Zhao, Zhi Hu, Xinyu Yang, Xuebin Ren, and Kwok-Yan Lam. 2021. Local Differential Privacy for data collection and analysis. *Neurocomputing* 426 (2021), 114–133.

[58] Royce J Wilson, Celia Yuxin Zhang, William Lam, Damien Desfontaines, Daniel Simmons-Marengo, and Bryant Gipson. 2020. Differentially Private SQL with Bounded User Contribution. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, Vol. 2. 230–250.

[59] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. 2020. Privacy preserving vertical federated learning for tree-based models. *arXiv preprint arXiv:2008.06170* (2020).

[60] Zhuolun Xiang, Bolin Ding, Xi He, and Jingren Zhou. 2020. Linear and range counting under metric-based local differential privacy. In *2020 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 908–913.

[61] Yonghui Xiao and Li Xiong. 2015. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1298–1309.

[62] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–19.