FEAST: A Communication-efficient Federated Feature Selection Framework for Relational Data

RUI FU, Beijing Institute of Technology, China
YUNCHENG WU, National University of Singapore, Singapore
QUANQING XU, OceanBase, Ant Group, China
MEIHUI ZHANG*, Beijing Institute of Technology, China

Vertical federated learning (VFL) is an emerging paradigm for cross-silo organizations to build more accurate machine learning (ML) models. In this setting, multiple organizations (i.e., parties) hold the same set of samples with different features. However, different parties may have redundant or highly correlated features, leading to inefficient and ineffective VFL model training. Effective feature selection in VFL is therefore essential to mitigate such a problem and improve model effectiveness, as well as computation and communication efficiency. To this end, in this paper, we propose a federated feature selection framework, called FEAST, which leverages conditional mutual information (CMI) to select more informative features while having low redundancy. Furthermore, we design a communication-efficient method to reduce the information exchanged among the parties while protecting the parties' raw data. Extensive experiments on four real-world datasets demonstrate that the proposed framework achieves state-of-the-art performance in terms of accuracy, communication and computation costs.

CCS Concepts: • **Computing methodologies** → **Feature selection**; *Cooperation and coordination*; Supervised learning by classification; • **Mathematics of computing** → *Information theory*.

Additional Key Words and Phrases: feature selection, vertical federated learning, communication-efficient, conditional mutual information

ACM Reference Format:
Rui Fu, Yuncheng Wu, Quanqing Xu, and Meihui Zhang. 2023. FEAST: A Communication-efficient Federated Feature Selection Framework for Relational Data. *Proc. ACM Manag. Data* 1, 1, Article 107 (May 2023), 28 pages. https://doi.org/10.1145/3588961

1 INTRODUCTION

Recent years have witnessed a growing interest in exploiting data from cross-silo organizations to design more accurate machine learning (ML) [46, 59] models and provide better customer services [16, 39, 66]. However, the raw data held by the distributed organizations cannot be shared with each other due to privacy concerns. To this end, the federated learning (FL) [6, 41, 62] paradigm is proposed, which enables cross-silo organizations to collaboratively build ML models without disclosing their raw data. FL can be categorized into different settings based on the data partitioning. In this paper, we consider the vertically-partitioned setting (aka. VFL), where the organizations

*Meihui Zhang is the corresponding author.

Authors' addresses: Rui Fu, Beijing Institute of Technology, Beijing, China, 3120201016@bit.edu.cn; Yuncheng Wu, National University of Singapore, Singapore, Singapore, wuyc@comp.nus.edu.sg; Quanqing Xu, OceanBase, Ant Group, Hangzhou, China, xuquanqing.xqq@antgroup.com; Meihui Zhang, Beijing Institute of Technology, Beijing, China, meihui_zhang@bit.edu.cn.

2836-6573/2023/5-ART107 $15.00
https://doi.org/10.1145/3588961

Bank

ID	Gender	Age	Income	Label
1	Male	18	2K	No
2	Male	30	8K	Yes
3	Female	24	6K	Yes

Insurance Company

ID	Gender	Birth	Package
1	Male	2004	35K
2	Male	1992	115K
3	Female	1998	95K

Fig. 1. An illustration of VFL and overlapping features.

(aka. parties) hold the same set of samples but with different features, and only one party owns the labels. We call the party which holds the labels as *active party* and the other parties *passive parties*. VFL targets feature-level collaborative learning among parties; therefore, it is especially useful for structured data analytics [10, 23, 44, 63], which has gained growing interests in the database community, and can be adopted in a wide spectrum of applications, such as healthcare and economics analytics.

Figure 1 illustrates a VFL example, where a bank (i.e., the active party) aims to build a model for predicting whether it should approve a customer's loan application by consolidating more features from an insurance company (i.e., the passive party). Feature selection is particularly important in VFL because the participating organizations may collect similar or highly correlated customer information and process the information differently. For example, we can observe three types of overlapping features in Figure 1. First, the 'gender' features of the two parties are duplicated. Second, the 'age' feature at the bank and the 'birth' feature at the insurance company enfold the same information. Third, the 'income' feature reflects a customer's monthly salary, and the 'package' feature reflects the annual earnings. Although they are not the same, they are highly correlated. These overlapping features are likely to contribute less useful information in totality and may affect the model effectiveness, and computation and communication efficiency [9, 28, 58].

Feature selection in VFL has special requirements in two aspects. The first is privacy concerns. For organizations such as the bank and insurance company in the above example, their data are related to user privacy (e.g., income), and companies cannot undertake the risk of information leakage. Therefore, the parties typically are not willing to share their raw data, leading the centralized feature selection methods hardly applicable in VFL. The second is communication and computation efficiency. VFL naturally expands the feature dimensionality in the analytical tasks as each organization nowadays may have hundreds of features. This poses a challenge to efficiency since both the communication cost and computation time will be significant with more parties and features.

While most recent works in VFL [12, 14, 18, 19, 36, 38] focus on model training and model prediction, several open-source FL systems (e.g., FATE [35]) support several feature selection algorithms for VFL, such as information value [26] and Pearson correlation coefficient [53]. These algorithms however only consider the relationship between each feature and the label, without examining the correlation among the features. As a result, they may select overlapping features from different parties. One way to remove overlapping features in the VFL setting is to apply secure schema matching techniques [15, 52]. However, they are still inadequate to identify correlated features (e.g., income vs. package) and select informative features. Although it is possible to utilize

advanced methods designed for the centralized scenario to identify the correlation among the features, such as iteratively training models by adding or removing a subset of features and selecting the features with the best performance [1, 2, 43], efficiency is a great concern since each training requires intensive computation and incurs high communication overhead.

In this paper, we propose a communication-efficient Federated fEAture SelecTion (FEAST) framework under the VFL setting. Our framework considers conditional mutual information (CMI) based feature selection, which utilizes CMI to identify features that are highly correlated with the label while having low redundancy between each other. However, enumerating all possible feature combinations is inefficient. Therefore, we present a multi-round selection approach, where a subset of features on a candidate party with the highest average CMI score is selected in each round. Subsequently, the selected party transmits only necessary feature information to other candidate parties for the CMI score calculation in the next round. To reduce the communication cost and protect the selected party's raw data, we devise a method that merges several features into a statistical variable for transmission. Importantly, the statistical variables are sufficient for the other parties to compute the CMI scores for feature selection. This leads to a solution that is not only accurate but also efficient for feature selection in VFL.

Specifically, we make the following contributions in this paper.

- We formulate the federated feature selection problem based on conditional mutual information (CMI) in VFL, which enables the parties to select more informative features with less redundancy.
- We design a federated feature selection framework FEAST to solve the proposed problem. FEAST generates statistical variables exchanged among parties instead of the selected features to protect raw data. We propose a feature score normalization method to address the biased selection trends introduced by statistical variables to improve accuracy.
- For computation and communication efficiency, we propose optimization strategies to reduce the number of samples and features involved in feature selection, which significantly decrease the computation and communication costs. We further provide a theoretical analysis of cost trade-offs.
- We implement FEAST and conduct extensive experiments on four real-world datasets, namely MIMIC III, PhysioNet Challenge 2012, Census-Income, and Nomao datasets, to evaluate its performance. The results demonstrate that FEAST offers comparable accuracy to a centralized algorithm and outperforms state-of-the-art filter-based baseline FATE-IV by up to 8.71% in terms of the ROC_AUC score (with RF classifier), and is much more efficient than FATE in terms of communication and computation costs.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 reviews the preliminaries and formalizes the feature selection problem in VFL. Section 4 presents the FEAST framework in detail. We evaluate the proposed framework in Section 5 and conclude in Section 6.

2 RELATED WORK

In this section, we summarize related work from three aspects: centralized feature selection, federated feature selection and privacy-preserving schema matching.

Centralized feature selection. In centralized settings, data is collected together and can be accessed easily. Existing feature selection works can be classified into the filter, wrapper, and embedded methods. The filter methods score each feature by dispersion [65], relevance [30, 53] or fairness [20, 51], and select features that exceed a pre-defined threshold. The wrapper methods [1, 2, 43] train different models by iteratively adding or removing features from a subset of

features and find the optimal subset of features based on the model performance. The embedded methods [24, 42] first implement feature embedding, and then regard it as a layer of the model and obtain the coefficients of each feature during the model training. Specifically, filter methods are efficient as they do not rely on heavy machine learning model training, but they are relatively less accurate. In contrast, wrapper and embedded methods have better performance but are with higher computational costs.

The mutual information (MI) based feature selection algorithm is a kind of filter method that selects the features highly correlated with labels while having low redundancy between features according to the MI scores. Specifically, the MI method measures the amount of information introduced by a feature based on the entropy difference. MIM [30] is an early MI-based method, which can select label-related features quickly. It however does not consider feature redundancy. MIFS [3] introduces feature redundancy as a metric for feature evaluation, and its variant mRMR [47], as well as other methods CIFE [33], JMI [61], JMIM [5], CMIM [17], and CFR [21], attempt to find a better expression of the relevance and redundancy relationship. RCDFS [13] considers the correlation between features and labels, redundancy between features, as well as the complementarity between features, as a means to improve the performance of the feature selection. However, these centralized methods cannot be directly applied in VFL setting as the features are held by different parties and may be too sensitive for sharing.

Federated feature selection. Although VFL has gained traction in recent years, most works focus on model training and prediction [12, 27, 36, 37, 62] or privacy preservation [14, 32, 34, 60], instead of feature selection. There are few VFL feature selection approaches [31, 45] have been proposed, which pay more attention to implementing privacy-preserving selection algorithms. However, their communication and computation performance is less desirable. Note that the wrapper and embedded methods may be too time-consuming and impractical for feature selection in VFL, due to the heavy computation and communication costs. Li et al. [31] introduce a Secure Multiparty Computation (MPC) based protocol for private feature selection based on the filter method. In their protocol, the feature and label matrices are encrypted by secret sharing, and thus computing scores under ciphertext is inefficient. Sakamoto et al. [45] design a secure algorithm based on fully homomorphic encryption, which provides high privacy guarantees. Still, the strict encryption leads to a huge overhead in practical tasks.

There are also several federated learning frameworks that have been proposed in recent years. For example, Google has developed Tensorflow Federated [7], which builds the FL framework based on Tensorflow for research purposes. Apache SystemDS [4] is a flexible and scalable ML system, aiming to provide system infrastructure for this exploratory data science process on federated and heterogeneous data sources. However, these frameworks mainly focus on horizontal FL, which are not applicable to the VFL setting. PySyft [50] proposes a privacy preserving deep learning framework that considers VFL, but the conventional ML algorithms like LR and GBDT are not supported. Recently, WeBank releases an industrial-grade framework FATE for federated learning [35], which supports feature selection with simple filter methods, such as information value [26] and Pearson correlation coefficient [53] under VFL setting. However, these filter methods do not consider the feature correlations among multiple parties. Thus, the selected features may be ineffective and lead to undesirable performance. Additionally, FATE supports Lasso regression [42], which can perform feature selection, but it requires multiple rounds of training, thus the communication and computation cost are much higher.

Privacy-preserving schema matching. Schema matching is a well-studied problem to find attributes or features that are semantically related. Scannapieco et al. [52] propose a protocol for privacy-preserving schema matching between two parties that can have different privacy

requirements by embedding the records of each of the two parties in a vector space. Cruz et al. [15] develop an efficient privacy-preserving schema matching protocol using mutual information of pair-wise attributes. These methods are able to identify overlapping features between parties with the same or similar semantics, and can be used as a complementary pre-processing step before feature selection. However, they have not been designed to identify correlated features and select informative features for model training. Further, existing schema matching works mainly consider the two-party setting, which is inefficient to handle the multi-party setting in VFL.

3 PRELIMINARY AND PROBLEM STATEMENT

In this section, we first introduce some preliminaries of the conditional mutual information (CMI)-based feature selection in Section 3.1. Next, we formulate the research problem in Section 3.2.

3.1 Preliminary

3.1.1 Entropy and Conditional Entropy. In information theory, entropy is used to represent the uncertainty of variables. In general, entropy increases when the state of uncertainty goes high. From the perspective of information contents, the larger the entropy value is, the less information the variable contains. The definition of information entropy is as follows [29]:

$$H(Y) = -\sum_{y \in Y} p(y) \log_2 p(y), \tag{1}$$

where $p(y)$ is the probability of occurrence of the y-th possible value of the variable Y. Generally speaking, the uncertainty of a variable can be affected by many factors. If all factors are determined, the variable becomes a fixed value. That is, the uncertainty is equals to 0. In some cases, we expect to obtain the uncertainty of a variable under certain conditions, which can be defined by conditional entropy:

$$H(Y|F) = \sum_{f \in F} p(f) \Big[-\sum_{y \in Y} p(y|f) \log_2 p(y|f) \Big], \tag{2}$$

where $p(y|f)$ is the probability of y under the condition $F = f$ while f is a possible value of the conditional variable F.

3.1.2 Mutual Information (MI) and Conditional MI. Given entropy $H(Y)$ and conditional entropy $H(Y|F)$, mutual information (MI), namely $I(Y;F)$, can be used to represent the uncertainty reduction of one variable by knowing another variable, i.e.,

$$I(Y;F) = H(Y) - H(Y|F). \tag{3}$$

Note that mutual information can only express the relationship between a single factor and the variable. However, as mentioned above, a variable is typically affected by multiple factors, and there are also repeated, complementary, and promoting relationships among factors' information contents. To better deal with the relationship between multiple factors and variables, conditional mutual information (CMI) is proposed. It represents the MI value under a given condition, which can be defined as follows:

$$\begin{aligned} I(Y;F_i|F_j) &= H(Y|F_j) - H(Y|F_i,F_j) \\ &= \sum_{y \in Y} \sum_{f_i \in F_i} \sum_{f_j \in F_j} p(y,f_i,f_j) \log_2 \frac{p(y,f_i,f_j)p(f_j)}{p(f_i,f_j)p(y,f_j)}, \end{aligned} \tag{4}$$

where F_i and F_j are two conditions for the variable Y. As a result, CMI can get the information contents that F_i provides to variable Y under the condition of F_j.

3.1.3 CMI-based Feature Selection. The feature selection algorithm based on CMI is used to find features that are highly correlated with label and yet, have low redundancy among the features. These methods first calculate the CMI score of each feature, then rank and select the top-ranked features. One of the state-of-the-art CMI algorithms is the JMI method [61]. Specifically, the score of a candidate feature F_i in the JMI method is defined as:

$$\begin{aligned} J_{JMI}(F_i) &= I(Y;F_i) - \frac{1}{|S|}\sum_{F_j\in S}\left[I(F_i;F_j) - I(F_i;F_j|Y)\right] \\ &= \frac{1}{|S|}\sum_{F_j\in S} I(Y;F_i|F_j), \end{aligned} \tag{5}$$

where S is the set of selected features and F_j is an element in S.

3.2 Problem Statement

We now formulate our federated feature selection problem. We consider a set of M parties $P = \{p_0, p_1, \cdots, p_{M-1}\}$ who aim to select a set of N features by consolidating their respective datasets. Specifically, we assume that each party p_k owns a dataset $\mathcal{D}^k \in \mathbb{R}^{L\times d_k}$ where $k \in \{0, \cdots, M-1\}$, and L and d_k denote the number of samples and features held by p_k, respectively. Note that L is the same for all parties in VFL, and we assume that the parties' samples are aligned beforehand [12, 14]. We denote the feature set of k-th party by $\mathcal{F}^k = \{F_1, \cdots, F_{d_k}\}$. Without loss of generality, we assume p_0 is the active party who also holds the label of the target task (i.e., Y) and is the initiator of federated feature selection. The remaining parties, $\{p_1, \cdots, p_{M-1}\}$, are the passive parties who only hold their corresponding features.

The main goal of federated feature selection is to select N features that carry the most information from all parties. Towards this end, the ultimate objective is to minimize the conditional entropy with respect to the label information as follows:

$$\underset{S^i\subseteq\mathcal{F}^i}{\operatorname{argmin}} H(Y|\bigcup_{i=0}^{M-1} S^i) \quad \text{s.t.} \quad \sum_{i=0}^{M-1}|S^i| = N, \tag{6}$$

where S^i is the selected features from party p_i (i.e., a subset of $\mathcal{F}^i$, which can be empty). However, because there are many combinations for $\cup_{i=0}^{M-1} S^i$, Equation 6 would be computationally intractable in real-world scenario. In order to make this problem computationally feasible, we first transform Equation 4 as follows by adjusting the positions of its terms.

$$H(Y|F_i, F_j) = H(Y|F_j) - I(Y;F_i|F_j). \tag{7}$$

According to Equation 7, we then transform our objective function Equation 6 by expanding the conditional entropy as follows:

$$\begin{aligned} H(Y|\bigcup_{i=0}^{M-1} S^i) &= H(Y|S^0, ..., S^{M-1}) \\ &= H(Y|S^0, ..., S^{M-2}) - I(Y;S^{M-1}|S^0, ..., S^{M-2}) \\ &= H(Y|S^0) - I(Y;S^1|S^0) - ... - I(Y;S^{M-1}|\bigcup_{i=0}^{M-2} S^i) \\ &= H(Y) - I(Y;S^0) - I(Y;S^1|S^0) - ... - I(Y;S^{M-1}|\bigcup_{i=0}^{M-2} S^i). \end{aligned} \tag{8}$$

Since $H(Y)$ is constant, the entropy $H(Y|\cup_{i=0}^{M-1} S^i)$ can be viewed as the accumulation of negative values of conditional mutual information. Although Equation 8 is still computationally intractable with the constrain of $\sum_{i=0}^{M-1}|S^i| = N$, it divides the ultimate goal into a polynomial, where each term

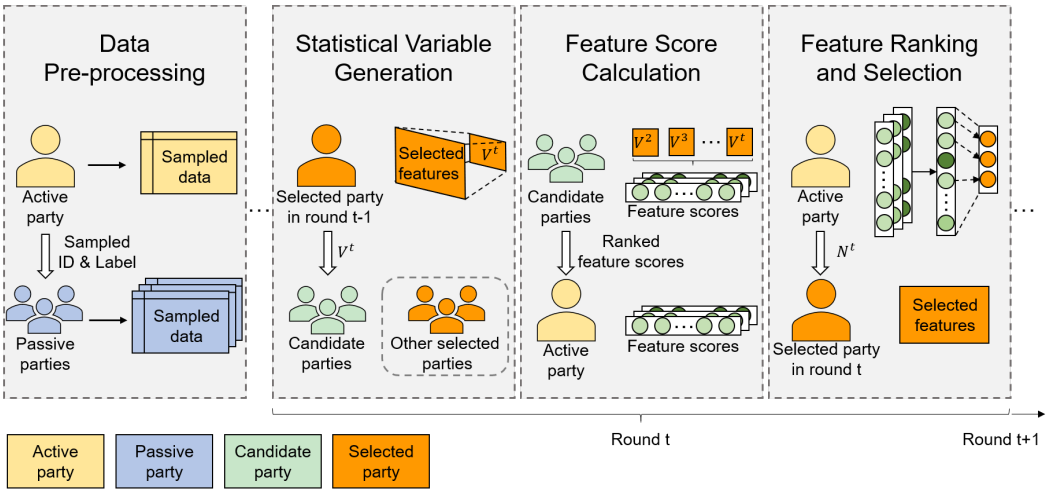


Fig. 2. An overview of the FEAST framework.

represents the MI values of the selected features. Based on this, we propose an iterative multi-round feature selection solution to find an approximation of our objective (i.e., Equation 6) in a greedy manner. Our method picks a party each time (without repetition) and selects a subset of features from this party based on the features selected previously. We consider a party to be selected when its (subset of) features have the largest average conditional mutual information under currently selected features (i.e., maximizing each term in the polynomial). Formally, we denote the iterative steps as follows:

$$v(t+1) = \underset{n}{\operatorname{argmax}} \left\{ \frac{I(Y; S^n | \cup_{i=1}^{t} S^{v(i)})}{|S^n|} \right\} \quad \text{s.t.} \quad \sum_{i=1}^{M} |S^{v(i)}| = N, \tag{9}$$

where $I(Y; S^n | \cup_{i=1}^{t} S^{v(i)})$ is the term in Equation 8 and $v(t)$ is the id of the party selected at t-th time.

4 FEDERATED FEATURE SELECTION

In this section, we shall present our federated feature selection framework FEAST.

4.1 Overview

Figure 2 illustrates an overview of FEAST, which consists of a data pre-processing step and an iterative feature selection step. During the selection process, we classify both parties and features into two states: *selected* and *candidate*. We initialize all the parties and their features as the candidate state. When a feature is selected, the states of the feature and the party where the feature is located will be changed from the candidate state to the selected state. It is worth noting that the feature state of different parties is known only to themselves. The detailed workflow is as follows.

Data pre-processing. In order to use the CMI methods to capture the information of selected features, we transform each party's continuous features into discrete features by binning [22, 57], where the maximum number of bins is denoted as B. We support a variety of discretization methods in FEAST, including both supervised and unsupervised learning-based approaches, such as the

minimum description length principle (MDLP), ChiMerge, equal-frequency binning, and equal-width binning [22, 57]. Meanwhile, we adopt stratified sampling, which aims to reduce the data size while keeping the data distribution, so that both the size of transmitted data and computation time in the iterative step can be reduced. Specifically, p_0 conducts the sampling and passes the sampled IDs and labels to other parties. Given that the subsequent processes are based on discrete features and statistics, the original distribution of the dataset can be maintained to a large extent as long as the stratified sampling is performed within a reasonable range, then it can retain a satisfactory accuracy of the subsequent results. We will analyze the required number of samples in Section 4.6 and evaluate the impact of stratified sampling on the accuracy in Section 5.

Recall that our objective is to select N features from all parties. In the iterative selection step, the parties will collaboratively execute the selection in T rounds till N features are selected. The workflow in each round includes three stages: statistical variable generation, feature score calculation, and feature ranking and selection.

Statistical variable generation. The selected party p_k in the previous round[1] computes some statistical information based on its selected features and forwards it to the existing candidate parties. Given the statistical information, the candidate parties can calculate their feature scores in next stage for feature selection. To reduce the communication cost and avoid transmitting the raw feature values, we let p_k merge its selected features and generate a statistical variable V^t. In particular, V^t is a combination of selected features' values, reflecting the distribution of combined features. We shall present the feature merging in the next section.

Feature score calculation. Each candidate party receives the latest statistical variable V^t and combines it with the statistical variables $V^2 \sim V^{t-1}$ in the previous rounds[2] to calculate the scores of all candidate features it holds. We will describe the calculation in Section 4.3. Finally, each candidate party sends the top-ranked local feature scores to the active party p_0.

Feature ranking and selection. After receiving the feature scores from all candidate parties, the active party decides on which candidate party to be picked and selects a subset of its top-ranked features in each round t. In addition, the active party notifies the candidate parties to delete some tail features, reducing the overall cost. We will elaborate on the selection strategy in Section 4.4.

4.2 Statistical Variable Generation

Recall that in each round t, each candidate party needs to compute the feature scores given the selected features in previous rounds using the CMI method. A naive way is to let the selected party transmit the selected features directly. However, this will result in high communication cost and reveal the selected party's sensitive raw data, which are unacceptable. To mitigate this problem, we propose a method that merges the selected features into a new variable V^t, which contains all the information that the candidate parties need in CMI calculation. The statistical variable will be transmitted to the candidate parties instead of the raw data.

Assume that the selected party in the previous round is p_k, who initiates the feature selection process in the current round t. Let $|S^k|$ be the number of selected features in p_k and m be the merge parameter. The selected party p_k randomly divides the selected features in p_k into $\left\lceil \frac{|S^k|}{m} \right\rceil$ feature groups, each of which consists of at most m features. After that, p_k takes each feature group as a unit and re-organizes each feature group to obtain $\left\lceil \frac{|S^k|}{m} \right\rceil$ merged tables, and each merged table has at most m factors. Figure 3 gives an example, where UID denotes user ID. Figure 3(a) shows the

[1]This stage starts from $t = 2$.

[2]When $t = 1$, there are no selected features, so each candidate party only uses labels to calculate the feature score based on mutual information (Equation 3).

UID	F_1	F_2	F_3	F_4	F_5
1	4	3	8	4	1
2	5	3	8	5	2
3	6	2	7	6	1
4	5	2	6	7	2
5	5	3	8	7	2
6	4	3	7	6	1
7	5	3	7	5	2
8	6	2	7	4	1

(a) Original selected features

	G_1			G_2		
GID	F_1	F_2	F_3	GID	F_4	F_5
0	4	3	8	0	4	1
1	5	3	8	1	5	2
2	6	2	7	2	6	1
3	5	2	6	3	7	2
4	4	3	7			
5	5	3	7			

(b) Merged tables

UID	V_1^t	V_2^t
1	0	0
2	1	1
3	2	2
4	3	3
5	1	3
6	4	2
7	5	1
8	2	0

(c) Formed V^t

Fig. 3. An illustration of statistical variable V^t generation.

original selected features with $|S^k| = 5$. Given $m = 3$, the selected features are randomly divided into $\lceil \frac{5}{3} \rceil = 2$ feature groups, say $G_1 = \{F_1, F_2, F_3\}$ and $G_2 = \{F_4, F_5\}$. The merged table in each group is computed by taking projection (with duplicate elimination) from the original table as shown in Figure 3(b). Take G_1 as an example. We project on F_1, F_2, F_3 from the table in Figure 3(a) and obtain 6 valid combinations associated with a sequential GID column as the elements in the resulting merged table. Similarly, we can obtain the merged table for G_2. We treat each feature group as a new combined feature and transform the original table, e.g., Figure 3(a), to a new table with $\left\lceil \frac{|S^k|}{m} \right\rceil$ features, e.g., Figure 3(c), according to the association between merged features and the original features. We denote the newly transformed table as the statistical variable V^t. An important aspect is that V^t is sufficient for the candidate parties to calculate the feature scores, which will be presented next.

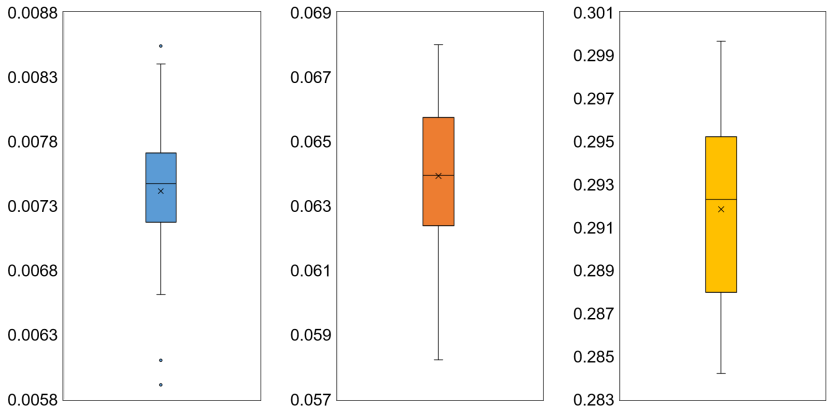
4.3 Feature Score Calculation

Now we present how each candidate party uses the statistical variables to calculate the scores of their local features. Let V be the union of received statistical variables, i.e., $V = \cup_{\alpha=2}^{t} V^{\alpha}$, which contains all the selected features in t rounds. We use $V_j \in V$ to denote the j-th feature in V. Formally, we calculate the feature score as:

$$J_{FEAST}(F_i) = \frac{1}{|V|} \sum_{V_j \in V} I(Y; F_i | V_j). \tag{10}$$

We use the average feature scores to approximate the CMI values of the subset expressed in Equation 9. In other words, we choose the party with the largest average feature score as the selected party.

Intuitively, we expect the score to measure the candidate features in terms of the correlation with label and the redundancy with existing information, regardless of the size (i.e., the number of elements) of selected features. However, the score in Equation 10 is not independent of the selected feature size. Indeed, as noted in [40, 48, 49], estimating the mutual information naively with empirical probabilities can lead to a bias toward large domain sizes. Since each V_j is generated by several features, and there are many potential combinations among features, the domain size of V_j often varies widely. Equation 10 favors the feature V_j with a large number of elements. In other words, it weighs more on the V_j with a large size when computing the feature score and thus affects the feature selection. We give an example as follows to illustrate this problem.



(a) $|V_j| = 10$ (b) $|V_j| = 100$ (c) $|V_j| = 1000$

Fig. 4. Box plots of $I(Y; F_i|V_j)$ with different size of V_j.

Example 1. *Given a target variable $Y \in \{0, 1\}$ and a variable $F_i \in [0, 1]$. We randomly construct three variables V_j with the number of elements 10, 100, and 1000 to compute $I(Y; F_i|V_j)$. Through one hundred experiments, we plot the box plots of three $I(Y; F_i|V_j)$, as shown in Figure 4. The mean values of $I(Y; F_i|V_j)$ with different sizes of V_j are 0.0074, 0.0639 and 0.2923 respectively. Intuitively, since V_j is constructed randomly and independently of label Y and variable F_i, the values of $I(Y; F_i|V_j)$ should be close. However, in practice, the values of $I(Y; F_i|V_j)$ vary by orders of magnitude. This suggests that the bias introduced by the large domain also occurs in our scenario.*

Feature score normalization. To reduce the effect of size of V_j on the feature scoring, we propose to normalize Equation 10 with symmetric uncertainty (SU) [54, 64]. Specifically, we first convert the CMI formula into an expression for information entropy:

$$\begin{aligned} I(Y; F_i|V_j) &= I(Y, F_i; V_j) - I(V_j; F_i) \\ &= H(Y, V_j) + H(F_i) - H(Y, F_i, V_j) - I(V_j; F_i) \\ &= H(Y, V_j) + H(V_j, F_i) - H(V_j) - H(Y, F_i, V_j). \end{aligned} \tag{11}$$

Then, we re-define part of our feature score as follows:

$$SU(Y; F_i|V_j) = \frac{I(Y; F_i|V_j)}{\alpha \cdot \left[H(Y, V_j) + H(V_j, F_i) - H(V_j)\right]}, \tag{12}$$

where $[H(Y, V_j) + H(V_j, F_i) - H(V_j)]$ is the guarantee of the normalization and $\alpha = \sqrt{|Y, F_i, V_j|}$ is a factor to further restrict the effect of $|V_j|$. Equation 12 normalizes the value of $I(Y; F_i|V_j)$ to range [0,1], mitigating the bias. A larger $SU(Y; F_i|V_j)$ indicates higher correlation with Y and less redundancy with V_j.

Finally, the normalized feature score is defined as:

$$J_{FEAST}(F_i) = \frac{1}{|V|} \sum_{V_j \in V} SU(Y; F_i|V_j). \tag{13}$$

Note that in the first round, there are no selected features, at which point we use the MI method (See Equation 3) for the initial selection. In subsequent round t, each candidate party only needs to

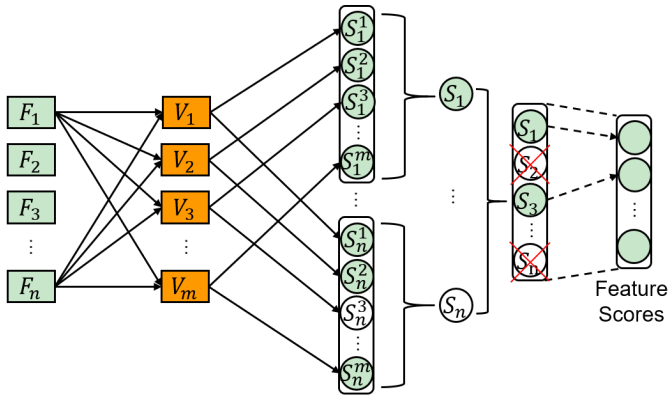


Fig. 5. The process of generating feature scores. The solid line represents the computation process and the dashed line represents the selection process. We use white circles to represent the ones with values close to 0 (or equal to 0).

compute $J_{FEAST}(F_i)$ of local features with respect to the selected features in V. Finally, it ranks the candidate features based on the scores and sends the ranked feature scores to the active party.

Overlapping feature deletion. According to Equation 13, we find that the each feature's score is the average of $|V|$ SU values. Therefore, when the number of selected features is large, there is likely to be a feature that contains a lot of information but overlaps with the previous statistical variables. Theoretically, we should not select the feature that is highly similar or even identical to the already selected features. However, since it contains a large amount of information, the SU values obtained from the calculation with other statistical variables are large. Thus, the value of J_{FEAST} is large, leading to the feature being mis-selected.

To prevent such a situation, we design an overlapping feature deletion strategy in this phase. As shown in Figure 5, since S_3^n is very close to 0, we regard the feature F_n as an overlapping feature that is very similar to V_3. We will delete feature F_n and exclude its corresponding S_n (i.e., J_{FEAST}) from being transmitted to the active party. To this end, we identify two types of overlapping features, i.e., identical features and similar features. Identical features are easy to discover as they have at least one SU value equal to 0. For similar features, their SU values are close to 0, and differ from other SU values by orders of magnitude. In this paper, we define the features whose SU values are less than one-tenth of the median as similar features. As will be shown in experiments, deleting overlapping features can reduce the computation time of the subsequent rounds without degrading the feature selection accuracy.

4.4 Feature Ranking and Selection

In this stage, the active party aims to select $\hat{N}^t$ features in round t (as illustrated in Figure 6). Note that $\hat{N}^1 = N$, which is the total number of features to be selected during the whole process. Let N_{sum}^t be the total number of ranked feature scores received from all candidate parties. Given N_{sum}^t feature scores, the active party p_0 first ranks them globally. For the top features in the ranking list, i.e., feature's rank in $[1, \hat{N}^t]$, these are the features that may be selected in this round. For the tail features (the feature size is determined by the active party), the active party can (optionally) notify the candidate parties to prune them out. The rationale is two-fold. On the one hand, these features are often poorly correlated with the label or too similar to the previously selected features, which can hardly provide useful information. On the other hand, the candidate parties can reduce the computation cost in subsequent rounds.

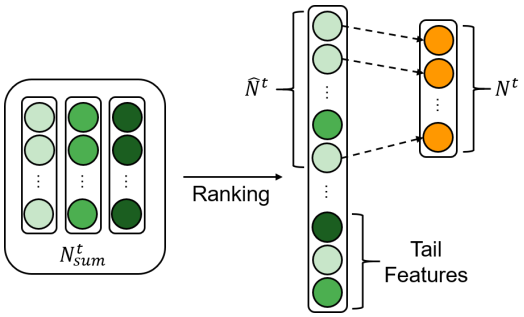


Fig. 6. The process of feature ranking and selection. The solid line represents the ranking process and the dashed line represents the selection process. We use different shades of green circles to indicate the features from different parties, and orange circles to denote the features that are selected.

Note that the globally top-ranked $\hat{N}^t$ features usually belong to multiple parties, i.e., $\hat{N}^t = \sum_{k \in P_c} N_k^t$, where N_k^t is the number of top-ranked features held by party p_k and P_c is the set of candidate parties. Therefore, we let the active party only select features from one candidate party in each round t. More specifically, the active party p_0 first groups the top-ranked $\hat{N}^t$ features by the party; then, p_0 calculates the average feature score of the features on each candidate party p_k; and finally selects the candidate party who has the largest average score as the selected party in this round. We denote the number of top-ranked features the selected party holds as N^t. These features are also marked as selected features in round t. Finally, the selected party sets $\hat{N}^{t+1} = \hat{N}^t - N^t$, i.e., the remaining features to be selected.

4.5 Privacy Protection Analysis

In the FEAST framework, the information transmitted among the parties is mainly the statistical variable set $V = \cup_{\alpha=2}^{t} V^\alpha$, which consists of the selected features in t rounds. Notice that $t \leq M$ because each party is selected at most once in FEAST, where M is the number of parties. Assume that there is a curious party p_x, who receives a statistical variable $V^\alpha = \{V_1^\alpha, \cdots, V_a^\alpha\}$ from p_k, where $a = \left\lceil \frac{|S^k|}{m} \right\rceil$ is the number of feature groups divided by p_k during the statistical variable generation stage. p_x aims to infer p_k's sensitive feature information from V^α. Since the statistical variable elements $V_1^\alpha, \cdots, V_a^\alpha$ are calculated from different sets of features held by p_k, the elements are independent. Without loss of generality, we focus on a single element $V_i^\alpha (i \in [1, a])$ and analyze the probability of finding the correct mapping between V_i^α and the original features after binning on p_k.

Theorem 4.1. *Given a statistical variable element V_i^α that consists of information from m selected features, where each feature is categorized with B bins, the probability of inferring the correct mapping between V_i^α and the original feature values is $\frac{1}{A(B^m,d)}$, where d is the number of distinct values in V_i^α.*

Proof. Since V_i^α consists of m features and each feature is composed of B bins, there could be B^m possible combinations or possible values of the m features. Note that there are only d distinct values in V_i^α. From the perspective of V_i^α, the d values can be selected from the B^m values. Thus, the number of possible solutions can be expressed as:

$$N_{comb} = B^m \cdot (B^m - 1) \cdot \ldots \cdot (B^m - d + 1) = A(B^m, d), \tag{14}$$

where A denotes permutation. Consequently, the inference probability of the mapping between V_i^α and the original feature values is $\frac{1}{A(B^m,d)}$. □

Notice that the features held by p_k are invisible to p_x; thus, p_x does not know the number of bins each feature is split into, though the number of bins of some categorical features is less than B. Nonetheless, even if the number of bins is known to p_x, the inference probability is still extremely small. Take V_1^t in Figure 3 as an example. It is generated from F_1, F_2, F_3, where $F_1 = \{4, 5, 6\}$, $F_2 = \{2, 3\}$, $F_3 = \{6, 7, 8\}$. Thus, there are $3 \times 2 \times 3 = 18$ combinations of the three features if the number of bins for each feature is known to p_x. Moreover, since there are 6 distinct values in V_1^t, the number of possible solutions $N_{comb} = A(18, 6) = 13366080$. In consequence, the probability of correctly inferring the original feature values is small even for this toy example with very few values.

In general, the larger B or m, the more number of possible combinations in Equation 14 and the less the inference probability are, and therefore the better privacy protection each party can obtain. However, as will be analyzed in Section 4.6, when B or m becomes large, the communication cost also increases greatly (see Equation 18). We shall discuss the choice of B and m in the next subsection.

4.6 Communication Cost Analysis

We now discuss how the maximum number of bins B and the merge parameter m affects the communication cost. Notice that if we do not apply feature merging or stratified sampling, the communication cost of transmitting the selected features for p_k is:

$$Cost(S^k) = L \cdot |S^k|, \tag{15}$$

where L is the number of original rows. With FEAST, we manage to reduce the communication cost to:

$$Cost(V^t) = L' \cdot \left\lceil \frac{|S^k|}{m} \right\rceil, \tag{16}$$

where L' is the number of rows after sampling. In FEAST, we dynamically decide the required number of rows L' as follows. To compute the feature score, we need to calculate the joint probability of Y, F_i, V_j. Recall that with feature merging, we generate a group of combined features, where each combined feature V_j contains at most m original features. Meanwhile, there are at most B values for a candidate feature F_i and each of the m selected feature in V_j, and $|Y|$ values for the label Y. Then, the maximum number of possible combinations of Y, F_i and V_j is $B^{m+1} \cdot |Y|$. As a result, to have a sufficient amount of data for calculating the feature scores in the following stage, it is required that L' in Equation 16 to satisfy the following inequation:

$$L' \geq B^{m+1} \cdot |Y|. \tag{17}$$

As will be shown in Section 5.4, this dynamic stratified sampling method can significantly reduce the communication cost without degrading accuracy. Besides, L' should be no larger than the original number of samples in the datasets, i.e., $L' \leq L$. Therefore, the lower bound of $Cost(V^t)$ is :

$$Cost(V^t) = \min \left\{ B^{m+1} \cdot |Y| \cdot \left\lceil \frac{|S^k|}{m} \right\rceil, \quad L \cdot \left\lceil \frac{|S^k|}{m} \right\rceil \right\}. \tag{18}$$

Case 1. *The first case is $L' \leq L$, where $Cost(V^t) = B^{m+1} \cdot |Y| \cdot \left\lceil \frac{|S^k|}{m} \right\rceil$. Since B, $|Y|$ and $|S^k|$ are constants, it can be regarded as a function of m:*

$$f(m) = B^{m+1} \cdot |Y| \cdot \left\lceil \frac{|S^k|}{m} \right\rceil, \tag{19}$$

and the extreme point is at:

$$m = \frac{1}{\ln B}. \tag{20}$$

That is, $Cost(V^t)$ will gradually increase when m is greater than $\frac{1}{\ln B}$. The smaller the m, the smaller the $Cost(V^t)$.

Case 2. *The second case is $L' > L$, where $Cost(V^t) = L \cdot \left\lceil \frac{|S^k|}{m} \right\rceil$. In this case, all the samples in the original datasets will be used for statistical variable generation. Moreover, the larger the m, the smaller the $Cost(V^t)$ because fewer merged feature groups are generated.*

Choice of B and m. By analyzing Theorem 4.1 and Equation 18, we notice that there is a trade-off between the $Cost(V^t)$ and the inference probability for privacy protection. When the value of B or m increases, the level of privacy protection increases, but the communication cost also grows. Note that B determines the maximum number of bins. Usually, the larger the B, the more information can be preserved about the original feature. In practice, we set $B \geq 8$ so that we can extract effective statistical information. Similarly, although smaller m will generate more merged feature groups (i.e., $\left\lceil \frac{|S^k|}{m} \right\rceil$), resulting a transformed table with more columns. The sampling number L' is also determined by m (Equation 17). Therefore, smaller m will reduce L' to a large extent. However, m cannot be too small. If m is too small, other candidate parties may easily infer the original feature values through the transmitted statistical variable. For example, if $m = 1$, it is equivalent to sending the original table, which is apparently unacceptable. In this paper, we set $m = 3$ by default and will experimentally evaluate the impact of B and m on the communication cost in Section 5.

5 EXPERIMENTS

In this section, we shall present the experimental evaluation of FEAST and analyze its performance against state-of-the-art baselines on four real-world datasets.

5.1 Experiment Setup

We implement the proposed framework in Python[3] and conduct the experiments using 4 servers on a cloud platform. Each cloud server is equipped with a 3rd Gen Intel (R) Xeon (R) scalable processor@2.7GHz (2 vCPU), 8GB RAM, and 1Mbps bandwidth. Specifically, we set the number of parties to 4, and use one server as the active party while the remaining servers as the passive parties.

Datasets and models. We use four real-world datasets, MIMIC III[4], PhysioNet Challenge 2012[5], Census-Income[6], and Nomao[7] for the evaluation.

- MIMIC III. The dataset contains more than 40,000 patients' medical data. It consists of 26 tables, and we use the data from the Admissions table to predict whether a patient in the Medical Intensive Care Unit (MICU) would be readmitted to the hospital. The resulting dataset contains 8,523 instances with 38 features.
- PhysioNet Challenge 2012. The dataset contains records of patients in Intensive Care Units (ICU). We use the medical features in this dataset, such as respiratory rate, glucose etc., to predict whether the patient will survive. The resulting dataset contains 12,000 instances with 41 features.

[3]The source code of FEAST is available at https://github.com/furuifr/FEAST.
[4]https://mimic.mit.edu/
[5]https://paperswithcode.com/dataset/physionet-challenge-2012
[6]https://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD)
[7]https://archive.ics.uci.edu/ml/datasets/Nomao

- Census-Income. The dataset contains census data extracted from the 1994 and 1995 Current Population Surveys. We use the demographic and employment related features to predict whether an adult's income is above $50K. The resulting dataset contains 199,523 instances with 41 features.
- Nomao. The dataset contains 34,465 instances and 120 features, which consists of 89 continuous and 31 categorical features (including the attributes 'label' and 'id').

We randomly split the datasets vertically into multiple partitions as the data held in different parties, and only the active party have labels. Moreover, to simulate the scenario with feature redundancy illustrated in Figure 1, we randomly add 4-16 overlapping features from other parties to each party.

To evaluate the performance of feature selection, we utilize five classifiers, logistic regression (LR) [25], support vector machines (SVM) [56], random forest (RF) [8], XGBoost [11], and neural network (NN) [55] to test the selected features.

Baselines. We compare our method with six baselines, where the first two baselines are from FATE [35], an industrial-grade open-source framework for federated learning developed by WeBank (a digital bank). FATE provides a distributed secure computing framework that supports various feature selection algorithms. We describe the baselines as follows.

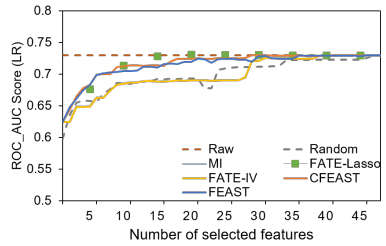
- FATE-IV. We adopt FATE's information value (IV) [26] feature selection algorithm, which is the most commonly used filter method in FATE, as the first baseline. It calculates the IV scores that are only related to the label information and selects the corresponding top-N features.
- FATE-Lasso. We use the lasso regression algorithm in FATE as the second baseline. It enables automatic feature selection by training a lasso regression model, where the informative features tend to have non-zero coefficients. In the experiments, we adjust the hyper-parameters of the model to obtain a specific number of features with non-zero coefficients for a fair comparison.
- CFEAST. The third baseline is FEAST in centralized setting. In this setting, we transfer all datasets directly to the active party and select features using a centralized method, i.e., using a variant of Equation 13 (replacing V_j with the selected feature F_j) to get the feature scores. We adopt this baseline to illustrate that the accuracy of FEAST in VFL is very close to that in centralized cases, demonstrating FEAST's effectiveness.
- MI. In the fourth baseline, we use the MI method (Equation 3), which selects features with top-N MI scores among all features.
- Random. We randomly select a party and randomly select one feature from the remaining features in this party at each round as the fifth baseline.
- Raw. The sixth baseline is based on the original feature set, which does not contain overlapping features, and we do not use any feature selection method.

Metrics. We employ three metrics to measure the performance:

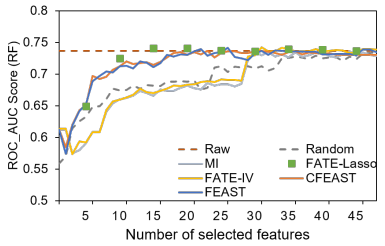
- Accuracy. We aggregate the selected features in one party after feature selection, train centralized classifiers on these features, and use ROC_AUC to measure the accuracy of the classifiers.
- Computation cost. We evaluate the time required to complete the whole federated feature selection process for all the approaches.
- Communication cost. We examine the total amount of data transferred during the whole federated feature selection process.

5.2 Comparison with the Baselines

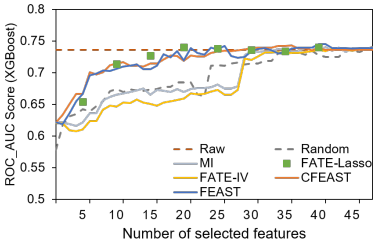
We evaluate FEAST and compare it with the baselines using the three metrics on accuracy and efficiency. We conduct experiments using four parties for accuracy and two parties for cost, and set $m = 3$ and $B = 8$ (equal-frequency binning) by default.



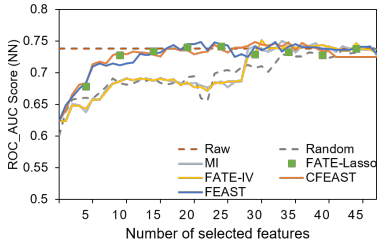
(a) Logistic Regression



(b) Random Forest

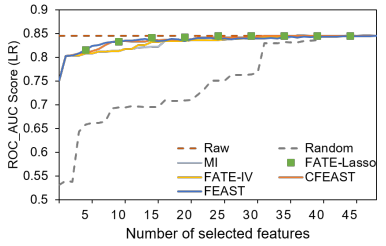


(c) XGBoost

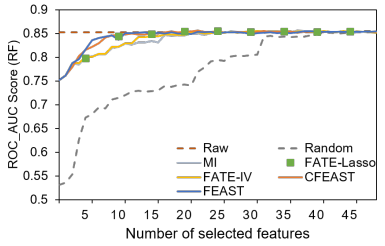


(d) Neural Network

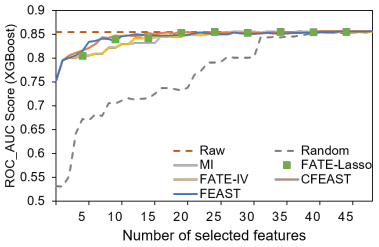
Fig. 7. Comparison with baseline with respect to accuracy on MIMIC.



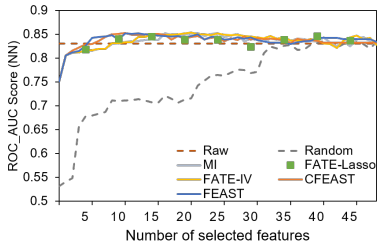
(a) Logistic Regression



(b) Random Forest



(c) XGBoost

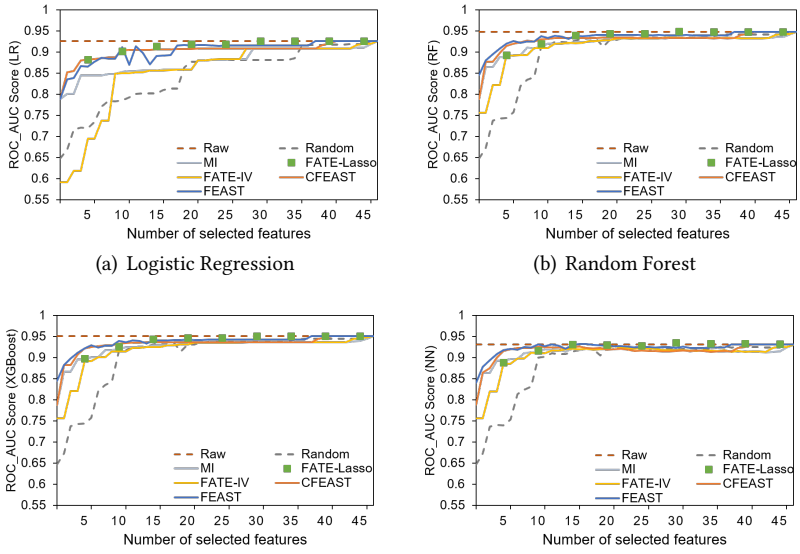


(d) Neural Network

Fig. 8. Comparison with baseline with respect to accuracy on PhysioNet.

Accuracy. We evaluate the classifier accuracy on the four datasets, as shown in Figure 7, Figure 8, Figure 9 and Figure 10, respectively. We only present the results of LR, RF, XGBoost and NN due to the space constraint. The results of SVM exhibit a similar trend.

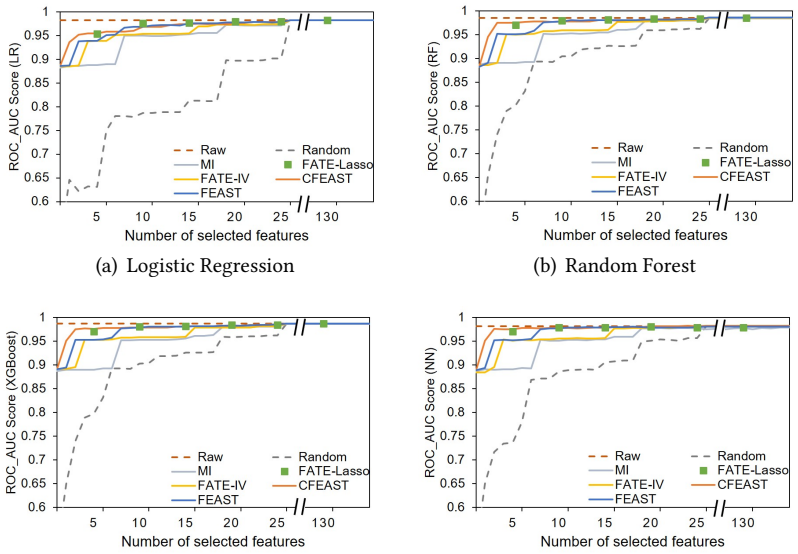
We make the following four key observations. *First*, FEAST outperforms the FATE-IV and MI baselines significantly. The reason is that the two filter-based baselines fall into repeated selection, resulting in low classification accuracy. Take the MIMIC dataset for example. FEAST is superior to FATE-IV by up to 3.7% (with LR classifier at $N = 7$), 8.71% (with RF classifier at $N = 7$), 6.93% (with XGBoost classifier at $N = 20$) and 6.3% (with NN classifier at $N = 24$), respectively. A noteworthy



(a) Logistic Regression (b) Random Forest

(c) XGBoost (d) Neural Network

Fig. 9. Comparison with baseline with respect to accuracy on Census.



(a) Logistic Regression (b) Random Forest

(c) XGBoost (d) Neural Network

Fig. 10. Comparison with baseline with respect to accuracy on Nomao.

aspect is that, as the number of selected features N increases, these feature selection methods would eventually reach their maximum model accuracies. However, the numbers of selected features that reach the maximum accuracy of these methods are very different. For instance, regarding the RF model on the MIMIC dataset, FEAST can reach 74.13% accuracy with 26 features; while the numbers for FATE-IV and MI need 31 features for 74.12% accuracy and 44 features for 73.61% accuracy, respectively. In practice, federated feature selection aims to select fewer important features while achieving high model accuracy. In addition, when N is large (i.e., $N \geq 30$ on MIMIC), the accuracy

gap between FEAST and the FATE-IV (or MI) method is small. This is expected because most of the informative features are already selected, leading to similar accuracy. An extreme case is that N equals the number of total features in all parties, where the accuracy of the two methods will be the same. However, when N is small, FEAST can select more informative features than FATE-IV and MI. This is especially useful in VFL because the number of total features is expanded, and the parties prefer to include only informative features while keeping the model concise.

Second, regarding the comparison to the Lasso baseline, since it is impractical to obtain all possible N by adjusting the hyper-parameters, we only sampled a few N to show the trend of Lasso. We observe that Lasso achieves comparable accuracy to FEAST or even higher accuracy because it can learn feature correlations in the regression model. Nevertheless, it requires multiple rounds of federated learning training, incurring much higher computation and communication costs.

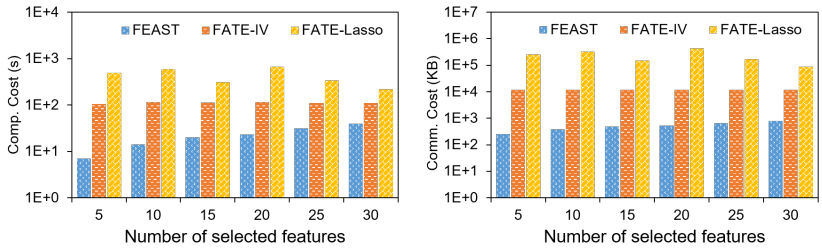
Third, FEAST is comparable to the centralized method CFEAST. It is worth noting that CFEAST is superior to FEAST in most cases, but there are some cases where the accuracy of FEAST exceeds those of CFEAST. We attribute this situation to two reasons. For one thing, our workflow is based on a greedy algorithm, and each step is locally optimal, but not necessarily globally optimal. For another, the CMI methods only measure the increment of information entropy, which cannot fully represent the real importance of the feature. We can find out that, the classifier's accuracy improves steadily as the number of selected features increases. Especially in the early stage, each newly selected feature can provide more information gain, which improves the classifier accuracy rapidly. However, as the selected features further increase, the growth rate of accuracy slows down gradually and eventually plateaus.

Fourth, we can see that Random performs the worst in most cases because it has no guarantee to select informative features. Conversely, the Raw baseline achieves the highest accuracy under most settings since it utilizes all the features for training the model. However, we notice that when training with some models (especially NN model), there are several cases in which its accuracy is lower than the feature selection methods (e.g., FEAST). For example, on the PhysioNet dataset with NN model, the raw baseline's accuracy is 83.07%, while FEAST achieves up to 85.17% when $N = 13$.

Computation cost. Figure 11(a) and Figure 12(a) compare the computation cost of FEAST, FATE-IV, FATE-Lasso with log scale on the MIMIC and Nomao datasets, respectively. We find that FATE-IV takes the same amount of time regardless of how many features are selected, which is because the IV-based algorithm calculates the IV values for all features and selects the top-N features. However, FATE-Lasso consumes different time with regard to different N. The reason is that in order to get different numbers of non-zero parameters, we need to set different hyper-parameters, which will cause the lasso regression model to reach convergence in different rounds. Meanwhile, the time of FEAST increases as the number of selected features increases. The reason is that it iteratively selects features and calculates on the selected features.

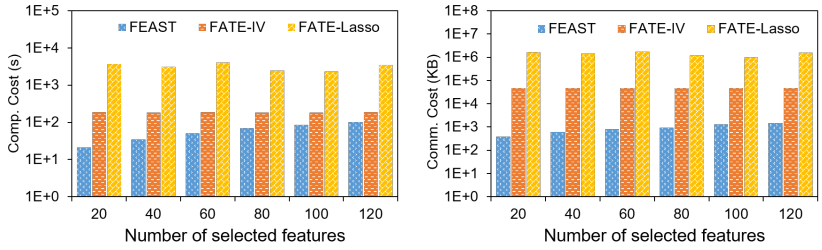
We note that FATE-Lasso's computation time is much higher than FATE-IV and FEAST. This is because the model training in FATE-Lasso needs multiple rounds of calculation, and it adopts cryptographic techniques in the calculation. For FATE-IV, though it does not utilize any cryptographic techniques, its computational cost is still higher than FEAST. This is because we employ two feature deletion strategies that effectively remove non-informative features and utilize stratified sampling in FEAST, which greatly reduces the computational cost while achieving higher model accuracy.

Communication cost. Figure 11(b) and Figure 12(b) report the communication cost of the three methods with log scale on the MIMIC and Nomao datasets, respectively. Similar to the computational cost analysis, FATE-IV has constant communication costs regarding different N because the transmitted data is independent of the selected features. Also, Lasso incurs a much higher communication cost as it requires model training with multiple rounds and uses cryptographic



(a) Computation cost (b) Communication cost

Fig. 11. Comparison with baselines on MIMIC.



(a) Computation cost (b) Communication cost

Fig. 12. Comparison with baselines on Nomao.

schemes to protect the transmitted information. Moreover, FEAST consumes less communication cost than FATE-IV (which does not use cryptographic techniques) from two aspects. First, FEAST combines multiple features into statistical variables, resulting in less transmitted information among parties. Second, FEAST requires fewer data samples to calculate the feature scores with the dynamic stratified sampling method, which further reduces the communication costs.

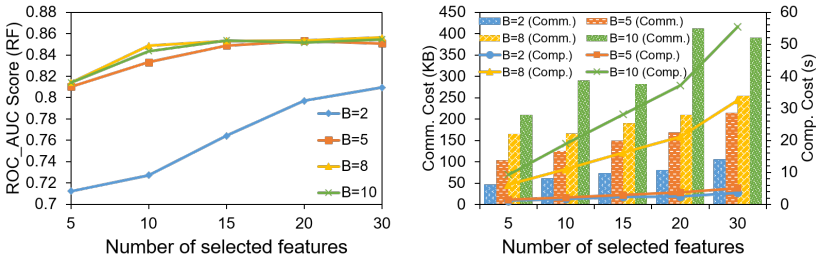
5.3 Effects of Tuning Parameters

Now we analyze the impacts of different parameter configurations on the performance of our framework. In this set of experiments, we vertically split the dataset into two partitions for two parties. Again, the features in the two partitions are partially overlapped.

Varying the number of bins (B). B is a parameter for discretizing variables, which determines the granularity of features after binning. We set $m = 3$ by default and vary B in $\{2, 5, 8, 10\}$ to evaluate the impact of B. Figure 13 shows the results.

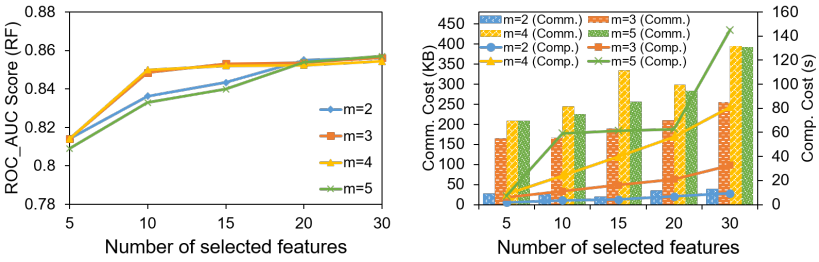
We can see from Figure 13(a) that the accuracy of the RF classifier is undesirable when $B = 2$. The reason is that, given a small B, each bin has a large range of values and a lot of information will be lost. For example, people have ranged in age from 1 to 80, and age are only discretized into two groups $[1, 40]$ and $[41, 80]$ when $B = 2$, which means that a five-year-old child falls into the same category with a 35-year-old adult. In contrast, the classifier's accuracy increases significantly when $B > 2$, because more bins can capture more fine-grained feature information. We note that, the accuracy with $B = 8$ and $B = 10$ are very similar (e.g., with only 0.001 level of fluctuation). The reason is that 8 bins are sufficient to capture the feature information, and the accuracy fluctuation is caused by the randomness introduced in the training process.

In terms of the computation cost and communication cost, they also increase as B goes up, as shown in Figure 13(b). This is expected, as the larger the number of bins, the more combinations of features, and thus the bigger the statistical variables V^t, resulting in more communication cost. In addition, as mentioned in Section 4.6, B affects the number of required samples (see Equation 17). That is, the size of V^t will become larger as B increases, which further increases the communication



(a) Accuracy of RF (b) Cost Comparison

Fig. 13. Impacts of B on PhysioNet.



(a) Accuracy of RF (b) Cost Comparison

Fig. 14. Impacts of m on PhysioNet.

cost. The computation cost shows a similar trend. The reason is that, when the statistical variable increases with B, not only the communication process takes more time, but also more samples need to be traversed for calculating the feature scores, resulting in a higher computation cost.

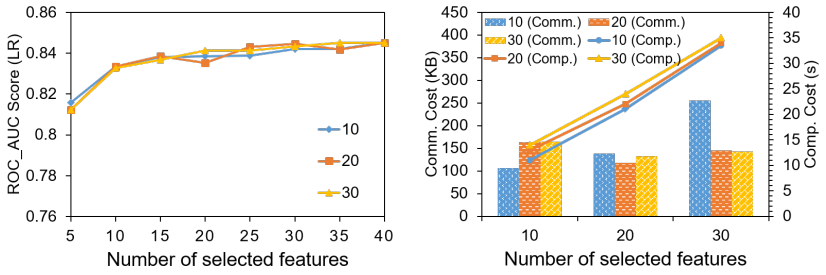
Varying the merge parameter (m). As discussed in Section 4.2, the merge parameter can affect the cost and the protection of raw data. In this set of experiments, we set $B = 8$, and vary m in {2, 3, 4, 5} to evaluate its impact. The results are illustrated in Figure 14.

In terms of classifier accuracy, we can see from Figure 14(a) that the ROC_AUC scores are stable under different m. This is because our framework essentially considers the relationship between each candidate feature and all selected features. Thus, the way we group the features has little impact on the classifiers' accuracy.

Regarding the computation cost (see Figure 14(b)), it increases as m increases. The reason is twofold. First, the number of required samples L' increases when m becomes large. Second, the number of possible combinations in V^t also increases, making the time for generating V^t increase. Besides, we observe that when $m = 5$, the time required to select 10, 15, and 20 features is similar. This is because there are two parties in this experiment, and the number of selected features at the first party is roughly the same, resulting in a similar time for generating V^t.

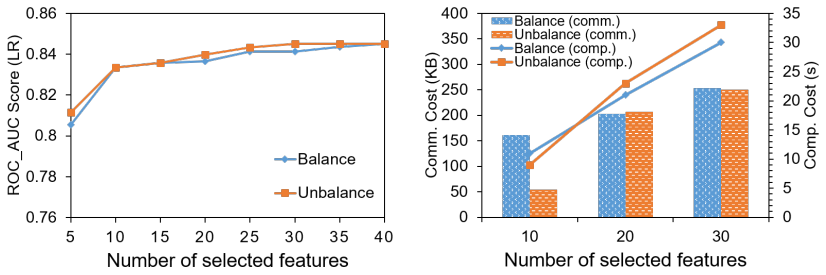
For the communication cost shown in Figure 14(b), we can see that it gradually increases until $m = 4$ and starts to decrease when $m = 5$. This trend is consistent with our analysis in Section 4.6. Note that when $m < 4$, L' is smaller than L, which means that the $Cost(V^t)$ is monotonically increasing with m. When $m \geq 4$, L' is greater than L, so $Cost(V^t) = L \cdot \lceil \frac{|S^k|}{m} \rceil$. Thus, given the same L, a larger m results in lower cost.

Varying the number of overlapping features. We notice that overlapping features can affect the accuracy of some filter-based feature selection methods (e.g., FATE-IV and MI). For example, in Figure 9(a), FATE-IV appears as a stair-wise phenomenon, such as when $N = 3$ and $N = 4$, the



(a) Accuracy of LR (b) Cost Comparison

Fig. 15. Impacts of overlapping features on PhysioNet.



(a) Accuracy of LR (b) Cost Comparison

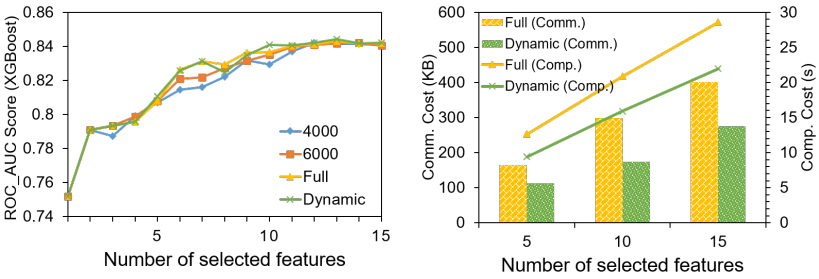
Fig. 16. Impacts of feature distribution on PhysioNet.

accuracy is the same. The reason is that it selects overlapping features, leading to lower accuracy. In contrast, FEAST can mitigate this problem by considering the correlation among features.

In this set of experiments, we further study the effects of the number of overlapping features on FEAST. Specifically, we divide the features equally into two parts for the two parties, and add different numbers of overlapping features to each party (i.e., $\{10, 20, 30\}$) that are owned by the other party. Figure 15 shows the comparison results with respect to accuracy, computation cost, and communication cost. We can observe that the accuracy is in a stable range with $10, 20, 30$ overlapping features, which indicates that FEAST is insensitive to the number of overlapping features, demonstrating its superior performance. For the computation cost, as the number of overlapping features increases, it also increases linearly, because each party holds more features, which needs more calculations. Regarding the communication cost, it mainly depends on the size of the statistical variable V^t, which varies according to the number of selected features in each round.

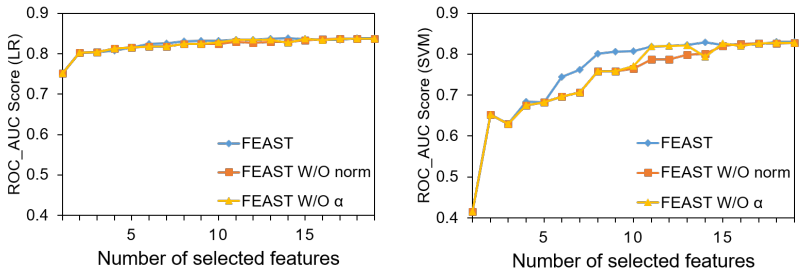
Varying the distribution of features. We study the effect of the distribution of features among parties on FEAST's performance. In particular, we use two partitioning methods. One method is to split the features randomly (i.e., balanced distribution). The other is to split the features according to feature importance (i.e., unbalanced distribution); that is, we assign the important features to one party and the rest of the features to another party.

Figure 16(a) shows the accuracy comparison of the two partitioning methods. We observe that the unbalanced distribution achieves slightly better accuracy than the balanced distribution. The rationale is that the unbalanced distribution allows the party to select more important features in the first round, which can better guide the selection afterward. In terms of the cost in Figure 16(b), when $N = 10$, the communication cost of unbalanced distribution is much smaller because all the 10 selected features are in the same party. Thus, the feature selection is complete in the first round without the need to transmit statistical variables V^t.



(a) Accuracy of XGBoost (b) Cost Comparison

Fig. 17. Impacts of stratified sampling on PhysioNet.



(a) Accuracy of LR (b) Accuracy of SVM

Fig. 18. Impacts of normalization on PhysioNet.

5.4 Ablation Study

Next, we conduct a series of ablation studies on the PhysioNet Challenge 2012 dataset to validate the effectiveness of dynamic stratified sampling, feature score normalization, and feature deletion optimizations in the FEAST framework. To highlight the reduction in computation, we use the overall computation time of the parties instead of the overall running time of the framework to represent the computation cost in this ablation study.

5.4.1 Stratified sampling. We first demonstrate that the proposed dynamic stratified sampling method achieves comparable accuracy to that using the full dataset while being more efficient in terms of communication and computation costs. Figure 17(a) shows the accuracy comparison. We report the classification accuracy of the XGBoost classifier with regard to dynamic stratified sampling (with 8192 samples), the full dataset (with 12000 samples), and two normal stratified sampling settings (with 4000 and 6000 samples, respectively). We observe that the accuracy of the dynamic stratified sampling and full dataset methods are almost the same w.r.t. different numbers of selected features, indicating that subsequent data binning and statistical variable generation stages are hardly affected. However, when the sample size decreases to 6000 or 4000, we can find a significant drop in the accuracy, showing that the selected dataset has produced a deviation. These results suggest that our dynamic stratified sampling method can effectively reduce the amount of data without degrading accuracy.

Further, we compare the communication cost (i.e., bars) and computation cost (i.e., lines) of dynamic stratified sampling to those using the full dataset, as illustrated in Figure 17(b). When N is equal to 5, 10, and 15, the communication cost of dynamic stratified sampling is 67.9%, 58.1%, and 68.4% of that using the full dataset, respectively. For the computation cost, the percentages are 67.9%, 58.1%, and 68.4%, respectively. As a consequence, we achieve about a 25% reduction in computation cost and 30% reduction in communication cost, with almost no accuracy loss.

5.4.2 Feature score normalization. Then, we validate the feature score normalization method presented in Section 4.3, which aims to alleviate the bias toward large domain sizes in CMI-based methods. We compare our method (i.e., Equation 12) in FEAST with two methods: one is Equation 12 without the factor α, and the other is feature score without normalization (i.e., Equation 4).

Figure 18(a) and 18(b) summarize the accuracy of the three methods on the PhysioNet dataset using the LR and SVM classifiers, respectively. The accuracy trends are similar on the two classifiers w.r.t. the number of selected features N. We analyze the results in three stages. First, when $N < 3$, the features selected by the three formulas are the same, due to the fact that all the features are selected in the first party using the MI method (i.e., Equation 3), given the small N. Thus, there is no bias towards large domains in this stage because no features are merged. Second, when $3 < N \leq 15$, we observe that FEAST without normalization performs the worst. This is expected because of the large domain bias introduced. Also, the accuracy of FEAST is higher than that of FEAST without the factor α, because α is the square root of the domain size, which can further reduce the effect of domain size on the feature score. Taking the SVM classifier as an example, the accuracy of FEAST is always the highest when $5 < N < 15$, and it achieves up to 5.58% accuracy improvement compared to that without normalization or without the factor α. Third, when $N > 15$, most of the highly informative features have already been selected; therefore, the accuracy difference of the three methods is insignificant. The results demonstrate that our normalization method can effectively mitigate the bias especially when selecting highly informative features.

Table 1. Impacts of the feature deletion strategies on the PhysioNet dataset.

Delete Strategy		ROC_AUC Score (LR)			Computation Time		
		N=5	N=10	N=20	N=5	N=10	N=20
Overlapping features	Non-del	0.809	0.8299	0.8359	5.34s (100%)	10.62s (100%)	13.39s (100%)
	Olp-del	0.809	0.8336	0.8359	5.19s (98.16%)	9.36s (88.16%)	10.58s (79.05%)
Tail features	Non-del	0.809	0.8336	0.8359	5.84s (100%)	10.04s (100%)	13.30s (100%)
	5%	0.809	0.8336	0.8359	5.62s (96.22%)	9.82s (97.74%)	12.58s (94.56%)
	10%	0.809	0.8336	0.8359	5.28s (90.48%)	9.2s (91.58%)	11.92s (89.57%)
	15%	0.809	0.8336	0.8354	4.94s (84.66%)	8.44s (84.08%)	10.91s (81.98%)
	20%	0.809	0.8336	0.8354	4.5s (77.00%)	7.96s (79.24%)	10.22s (76.85%)

5.4.3 Feature deletion. In the following, we validate the effectiveness of the two feature deletion strategies in FEAST: overlapping feature deletion and tail feature deletion. Table 1 summarizes the accuracy and computation time of the logistic regression classifier on the PhysioNet dataset. In order to compare the computation time more clearly, we calculate the fraction of the computation time using the proposed deletion strategy over that of the non-deletion strategy (which is always 100%, regardless of N).

Overlapping feature deletion. We observe that deleting overlapping features before the feature ranking and selection stage prevents features that provide highly similar information from being selected simultaneously and reduces the computation time of feature scores in subsequent rounds. Specifically, the accuracy of the non-deletion and overlapping-deletion strategies are the same when $N = 5$ (0.809) and $N = 20$ (0.8359), indicating that the same features are selected. In some cases, the overlapping-deletion strategy even slightly outperforms the non-deletion strategy (e.g., when $N = 10$, the accuracy is 0.8299 and 0.8336, respectively). This demonstrates that our strategy has high reliability and does not mistakenly delete useful features but may delete features that affect the accuracy negatively in subsequent rounds.

In terms of the computation cost, as N increases, the percentage of computation time for overlapping-deletion versus non-deletion decreases from 98.16% to 88.16% and to 79.05%. Such a

substantial decrease in computation time is because: the more features selected, the more features will be highly similar or even duplicated among other parties, resulting in more features being deleted and, therefore, reducing the computation time. If there are no overlapping features among the parties, the overlapping-deletion strategy will be equivalent to the non-deletion strategy.

Tail feature deletion. In the feature ranking and selection stage (see Section 4.4), we delete the tail features because they can hardly provide more useful information in the subsequent rounds. Now we evaluate this strategy by comparing it to the non-deletion strategy, i.e., without deleting any tail features. We further vary the deletion percentage with 5%, 10%, 15%, and 20%, to investigate its impact on the accuracy and computation cost. As shown in Table 1, the ROC_AUC score does not change when the last 5% and 10% of the tail features are removed, compared to the non-deletion strategy. Besides, when the deletion percentage increases to 15% or even 20%, the ROC_AUC score only decreases by 0.0005 at $N = 20$.

With regard to the computation cost, we see that the larger the deletion percentage, the less the overall computation time, which is in accordance with our expectations. For example, when $N = 20$, the computation time is 12.58s, 11.92s, 10.91s, and 10.22s for deletion percentages 5%, 10%, 15%, and 20%, respectively. In addition, we find that given the same deletion percentage, the ratio of computation time over the non-deletion strategy is similar, even with different values of N. For instance, when the deletion percentage is 10%, the computation time for selecting 5, 10, and 20 features is 90.48%, 91.58%, and 89.57% of the original time, respectively. This is because the percentage of deleted features is fixed in each round.

6 CONCLUSION

In this paper, we study the problem of feature selection in vertical FL. We present a federated feature selection framework FEAST based on conditional mutual information (CMI) scores as a mechanism to enable multi-parties to collectively select informative features. To reduce the communication cost exchanged among the parties and protect the parties' raw data, we devise a communication-efficient method that merges several features before transmission. Our extensive experimental study using MIMIC III, PhysioNet Challenge 2012, Census-Income, and Nomao datasets shows that FEAST achieves comparable accuracy to a centralized algorithm and outperforms state-of-the-art baselines by up to 8.71% in terms of ROC_AUC score. Moreover, FEAST has an order of magnitude improvement in communication and computation costs compared to the baselines.

ACKNOWLEDGMENTS

This work is supported by National Key Research and Development Program of China (2020YFB1708100), National Natural Science Foundation of China (62072033) and the Ant Group Research Fund. Yuncheng Wu's work is supported by the National Research Foundation, Singapore under its Emerging Areas Research Projects (EARP) Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

REFERENCES

[1] Naoual El Aboudi and Laila Benhlima. 2016. Review on wrapper feature selection approaches. *2016 International Conference on Engineering & MIS (ICEMIS)* (2016), 1–5.

[2] Javier Apolloni, Guillermo Leguizamón, and Enrique Alba. 2016. Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments. *Appl. Soft Comput.* 38 (2016), 922–932. https://doi.org/10.1016/j.asoc.2015.10.037

[3] Roberto Battiti. 1994. Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. Neural Networks* 5, 4 (1994), 537–550. https://doi.org/10.1109/72.298224

[4] Sebastian Baunsgaard, Matthias Boehm, Ankit Chaudhary, Behrouz Derakhshan, Stefan Geißelsöder, Philipp M. Grulich, Michael Hildebrand, Kevin Innerebner, Volker Markl, Claus Neubauer, Sarah Osterburg, Olga Ovcharenko, Sergey Redyuk, Tobias Rieger, Alireza Rezaei Mahdiraji, Sebastian Benjamin Wrede, and Steffen Zeuch. 2021. ExDRa: Exploratory Data Science on Federated Raw Data. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, Guoliang Li, Zhanhuai Li, Stratos Idreos, and Divesh Srivastava (Eds.). ACM, 2450–2463. https://doi.org/10.1145/3448016.3457549

[5] Mohamed Bennasar, Yulia Hicks, and Rossitza Setchi. 2015. Feature selection using Joint Mutual Information Maximisation. *Expert Syst. Appl.* 42, 22 (2015), 8520–8532. https://doi.org/10.1016/j.eswa.2015.07.007

[6] Akash Bharadwaj and Graham Cormode. 2022. An Introduction to Federated Computation. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 2448–2451. https://doi.org/10.1145/3514221.3522561

[7] Kallista A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. In *Proceedings of Machine Learning and Systems 2019, MLSys 2019, Stanford, CA, USA, March 31 - April 2, 2019*, Ameet Talwalkar, Virginia Smith, and Matei Zaharia (Eds.). mlsys.org. https://proceedings.mlsys.org/book/271.pdf

[8] Leo Breiman. 2001. Random Forests. *Mach. Learn.* 45, 1 (2001), 5–32. https://doi.org/10.1023/A:1010933404324

[9] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. 2018. Feature selection in machine learning: A new perspective. *Neurocomputing* 300 (2018), 70–79. https://doi.org/10.1016/j.neucom.2017.11.077

[10] Shaofeng Cai, Kaiping Zheng, Gang Chen, H. V. Jagadish, Beng Chin Ooi, and Meihui Zhang. 2021. ARM-Net: Adaptive Relation Modeling Network for Structured Data. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*. 207–220.

[11] 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, San Francisco, CA, USA, August 13-17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 785–794. https://doi.org/10.1145/2939672.2939785

[12] Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. 2020. VAFL: a Method of Vertical Asynchronous Federated Learning. *CoRR* abs/2007.06081 (2020). arXiv:2007.06081 https://arxiv.org/abs/2007.06081

[13] Zhijun Chen, Chaozhong Wu, Yishi Zhang, Zhen Huang, Bin Ran, Ming Zhong, and Nengchao Lyu. 2015. Feature selection with redundancy-complementariness dispersion. *Knowl. Based Syst.* 89 (2015), 203–217. https://doi.org/10.1016/j.knosys.2015.07.004

[14] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. 2021. SecureBoost: A Lossless Federated Learning Framework. *IEEE Intell. Syst.* 36, 6 (2021), 87–98. https://doi.org/10.1109/MIS.2021.3082561

[15] Isabel F. Cruz, Roberto Tamassia, and Danfeng Yao. 2007. Privacy-Preserving Schema Matching Using Mutual Information. In *Data and Applications Security XXI, 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Redondo Beach, CA, USA, July 8-11, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4602)*, Steve Barker and Gail-Joon Ahn (Eds.). Springer, 93–94. https://doi.org/10.1007/978-3-540-73538-0_7

[16] Jian Dai, Meihui Zhang, Gang Chen, Ju Fan, Kee Yuan Ngiam, and Beng Chin Ooi. 2018. Fine-grained Concept Linking using Neural Networks in Healthcare. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, Gautam Das, Christopher M. Jermaine, and Philip A. Bernstein (Eds.). ACM, 51–66. https://doi.org/10.1145/3183713.3196907

[17] François Fleuret. 2004. Fast Binary Feature Selection with Conditional Mutual Information. *J. Mach. Learn. Res.* 5 (2004), 1531–1555. http://jmlr.org/papers/volume5/fleuret04a/fleuret04a.pdf

[18] Fangcheng Fu, Yingxia Shao, Lele Yu, Jiawei Jiang, Huanran Xue, Yangyu Tao, and Bin Cui. 2021. VF^{2}Boost: Very Fast Vertical Federated Gradient Boosting for Cross-Enterprise Learning. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, Guoliang Li, Zhanhuai Li, Stratos Idreos, and Divesh Srivastava (Eds.). ACM, 563–576. https://doi.org/10.1145/3448016.3457241

[19] Fangcheng Fu, Huanran Xue, Yong Cheng, Yangyu Tao, and Bin Cui. 2022. BlindFL: Vertical Federated Machine Learning without Peeking into Your Data. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 1316–1330. https://doi.org/10.1145/3514221.3526127

[20] Sainyam Galhotra, Karthikeyan Shanmugam, Prasanna Sattigeri, and Kush R. Varshney. 2022. Causal Feature Selection for Algorithmic Fairness. In *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Zachary Ives, Angela Bonifati, and Amr El Abbadi (Eds.). ACM, 276–285. https://doi.org/10.1145/3514221.3517909

[21] Wanfu Gao, Liang Hu, Ping Zhang, and Jialong He. 2018. Feature selection considering the composition of feature relevancy. *Pattern Recognit. Lett.* 112 (2018), 70–74. https://doi.org/10.1016/j.patrec.2018.06.005

[22] Salvador García, Julián Luengo, José Antonio Sáez, Victoria López, and Francisco Herrera. 2013. A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning. *IEEE Trans. Knowl. Data Eng.* 25, 4 (2013), 734–750. https://doi.org/10.1109/TKDE.2012.35

[23] Muon Ha and Yulia A. Shichkina. 2022. Translating a Distributed Relational Database to a Document Database. *Data Sci. Eng.* 7, 2 (2022), 136–155. https://doi.org/10.1007/s41019-022-00181-9

[24] Xiaofei He, Ming Ji, Chiyuan Zhang, and Hujun Bao. 2011. A Variance Minimization Criterion to Feature Selection Using Laplacian Regularization. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 10 (2011), 2013–2025. https://doi.org/10.1109/TPAMI.2011.44

[25] David W. Hosmer and Stanley Lemeshow. 2000. *Applied Logistic Regression, Second Edition.* Wiley. https://doi.org/10.1002/0471722146

[26] Ronald A. Howard. 1966. Information Value Theory. *IEEE Trans. Syst. Sci. Cybern.* 2, 1 (1966), 22–26. https://doi.org/10.1109/TSSC.1966.300074

[27] Yaochen Hu, Di Niu, Jianming Yang, and Shengping Zhou. 2019. FDML: A Collaborative Machine Learning Framework for Distributed Features. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 2232–2240. https://doi.org/10.1145/3292500.3330765

[28] Samina Khalid, Tehmina Khalil, and Shamila Nasreen. 2014. A survey of feature selection and feature extraction techniques in machine learning. *2014 Science and Information Conference* (2014), 372–378.

[29] John Kieffer. 1994. Elements of Information Theory (Thomas M. Cover and Joy A. Thomas). *SIAM Rev.* 36, 3 (1994), 509–511. https://doi.org/10.1137/1036124

[30] David D. Lewis. 1992. Feature Selection and Feature Extraction for Text Categorization. In *Proceedings of the Workshop on Speech and Natural Language.* Association for Computational Linguistics, 212–217.

[31] Xiling Li, Rafael Dowsley, and Martine De Cock. 2021. Privacy-Preserving Feature Selection with Secure Multiparty Computation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 6326–6336. http://proceedings.mlr.press/v139/li21e.html

[32] Zitao Li, Bolin Ding, Ce Zhang, Ninghui Li, and Jingren Zhou. 2021. Federated Matrix Factorization with Privacy Guarantee. *Proc. VLDB Endow.* 15, 4 (2021), 900–913. https://doi.org/10.14778/3503585.3503598

[33] Dahua Lin and Xiaoou Tang. 2006. Conditional Infomax Learning: An Integrated Framework for Feature Extraction and Fusion. In *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 3951)*, Ales Leonardis, Horst Bischof, and Axel Pinz (Eds.). Springer, 68–82. https://doi.org/10.1007/11744023_6

[34] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. 2021. Projected Federated Averaging with Heterogeneous Differential Privacy. *Proc. VLDB Endow.* 15, 4 (2021), 828–840. https://doi.org/10.14778/3503585.3503592

[35] Yang Liu, Tao Fan, Tianjian Chen, Qian Xu, and Qiang Yang. 2021. FATE: An Industrial Grade Platform for Collaborative Learning With Data Protection. *J. Mach. Learn. Res.* 22 (2021), 226:1–226:6. http://jmlr.org/papers/v22/20-815.html

[36] Yang Liu, Yingting Liu, Zhijie Liu, Yuxuan Liang, Chuishi Meng, Junbo Zhang, and Yu Zheng. 2022. Federated Forest. *IEEE Trans. Big Data* 8, 3 (2022), 843–854. https://doi.org/10.1109/TBDATA.2020.2992755

[37] Yejia Liu, Weiyuan Wu, Lampros Flokas, Jiannan Wang, and Eugene Wu. 2021. Enabling SQL-based Training Data Debugging for Federated Learning. *Proc. VLDB Endow.* 15, 3 (2021), 388–400. https://doi.org/10.14778/3494124.3494125

[38] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. 2021. Feature Inference Attack on Model Predictions in Vertical Federated Learning. In *ICDE*. 181–192.

[39] Zhaojing Luo, Sai Ho Yeung, Meihui Zhang, Kaiping Zheng, Lei Zhu, Gang Chen, Feiyi Fan, Qian Lin, Kee Yuan Ngiam, and Beng Chin Ooi. 2021. MLCask: Efficient Management of Component Evolution in Collaborative Data Analytics Pipelines. In *ICDE*. 1655–1666.

[40] Panagiotis Mandros, Mario Boley, and Jilles Vreeken. 2020. Discovering dependencies with reliable mutual information. *Knowl. Inf. Syst.* 62, 11 (2020), 4223–4253. https://doi.org/10.1007/s10115-020-01494-9

[41] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Xiaojin (Jerry) Zhu (Eds.). PMLR, 1273–1282. http://proceedings.mlr.press/v54/mcmahan17a.html

[42] Ramakrishnan Muthukrishnan and R. Rohini. 2016. LASSO: A feature selection technique in predictive modeling for machine learning. *2016 IEEE International Conference on Advances in Computer Applications (ICACA)* (2016), 18–20.

[43] Bach Hoai Nguyen, Bing Xue, Ivy Liu, and Mengjie Zhang. 2014. Filter based backward elimination in wrapper based PSO for feature selection in classification. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014*. IEEE, 3111–3118. https://doi.org/10.1109/CEC.2014.6900657

[44] Milos Nikolic, Haozhe Zhang, Ahmet Kara, and Dan Olteanu. 2020. F-IVM: Learning over Fast-Evolving Relational Data. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*. 2773–2776.

[45] Shinji Ono, Jun Takata, Masaharu Kataoka, Tomohiro I, Kilho Shin, and Hiroshi Sakamoto. 2022. Privacy-Preserving Feature Selection with Fully Homomorphic Encryption. *Algorithms* (2022).

[46] Beng Chin Ooi, Kian-Lee Tan, Sheng Wang, Wei Wang, Qingchao Cai, Gang Chen, Jinyang Gao, Zhaojing Luo, Anthony K. H. Tung, Yuan Wang, Zhongle Xie, Meihui Zhang, and Kaiping Zheng. 2015. SINGA: A Distributed Deep Learning Platform. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM*. 685–688.

[47] Hanchuan Peng, Fuhui Long, and Chris H. Q. Ding. 2005. Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 8 (2005), 1226–1238. https://doi.org/10.1109/TPAMI.2005.159

[48] Frédéric Pennerath, Panagiotis Mandros, and Jilles Vreeken. 2020. Discovering Approximate Functional Dependencies using Smoothed Mutual Information. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1254–1264. https://doi.org/10.1145/3394486.3403178

[49] Simone Romano, Xuan Vinh Nguyen, James Bailey, and Karin Verspoor. 2016. A Framework to Adjust Dependency Measure Estimates for Chance. In *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*, Sanjay Chawla Venkatasubramanian and Wagner Meira Jr. (Eds.). SIAM, 423–431. https://doi.org/10.1137/1.9781611974348.48

[50] Théo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *CoRR* abs/1811.04017 (2018). arXiv:1811.04017 http://arxiv.org/abs/1811.04017

[51] Ricardo Salazar, Felix Neutatz, and Ziawasch Abedjan. 2021. Automated Feature Engineering for Algorithmic Fairness. *Proc. VLDB Endow.* 14, 9 (2021), 1694–1702. https://doi.org/10.14778/3461535.3463474

[52] Monica Scannapieco, Ilya Figotin, Elisa Bertino, and Ahmed K. Elmagarmid. 2007. Privacy preserving schema and data matching. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, June 12-14, 2007*, Chee Yong Chan, Beng Chin Ooi, and Aoying Zhou (Eds.). ACM, 653–664. https://doi.org/10.1145/1247480.1247553

[53] Patrick Schober, Christa Boer, and Lothar A. Schwarte. 2018. Correlation Coefficients: Appropriate Use and Interpretation. *Anesthesia & Analgesia* 126 (2018), 1763–1768.

[54] Bharat Singh, Nidhi Kushwaha, and Om Prakash Vyas. 2014. A Feature Subset Selection Technique for High Dimensional Data Using Symmetric Uncertainty.

[55] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958. https://doi.org/10.5555/2627435.2670313

[56] Ingo Steinwart and Andreas Christmann. 2008. *Support Vector Machines*. Springer.

[57] Chih-Fong Tsai and Yu-Chi Chen. 2019. The optimal combination of feature selection and data discretization: An empirical study. *Inf. Sci.* 505 (2019), 282–293. https://doi.org/10.1016/j.ins.2019.07.091

[58] Sonal Tuteja and Rajeev Kumar. 2022. A Unification of Heterogeneous Data Sources into a Graph Model in E-commerce. *Data Sci. Eng.* 7, 1 (2022), 57–70. https://doi.org/10.1007/s41019-021-00174-0

[59] Wei Wang, Jinyang Gao, Meihui Zhang, Sheng Wang, Gang Chen, Teck Khim Ng, Beng Chin Ooi, Jie Shao, and Moaz Reyad. 2018. Rafiki: Machine Learning as an Analytics Service System. *Proc. VLDB Endow.* 12, 2 (2018), 128–140.

[60] Yuncheng Wu, Shaofeng Cai, Xiaokui Xiao, Gang Chen, and Beng Chin Ooi. 2020. Privacy Preserving Vertical Federated Learning for Tree-based Models. *Proc. VLDB Endow.* 13, 11 (2020), 2090–2103. http://www.vldb.org/pvldb/vol13/p2090-wu.pdf

[61] Howard Hua Yang and John E. Moody. 1999. Feature Selection Based on Joint Mutual Information.

[62] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* 10, 2 (2019), 12:1–12:19. https://doi.org/10.1145/3298981

[63] Zhenkun Yang, Chuanhui Yang, Fusheng Han, Mingqiang Zhuang, Bing Yang, Zhifeng Yang, Xiaojun Cheng, Yuzhong Zhao, Wenhui Shi, Huafeng Xi, Huang Yu, Bin Liu, Yi Pan, Boxue Yin, Junquan Chen, and Quanqing Xu. 2022. OceanBase: A 707 Million tpmC Distributed Relational Database System. *Proceedings of the VLDB Endowment* 15, 12 (2022), 3385–3397.

[64] Li Zhang and Xiaobo Chen. 2021. Feature Selection Methods Based on Symmetric Uncertainty Coefficients and Independent Classification Information. *IEEE Access* 9 (2021), 13845–13856. https://doi.org/10.1109/ACCESS.2021.3049815

[65] Zheng Zhao, Ruiwen Zhang, James Cox, David Duling, and Warren Sarle. 2013. Massively parallel feature selection: an approach based on variance preservation. *Mach. Learn.* 92, 1 (2013), 195–220. https://doi.org/10.1007/s10994-013-5373-4
[66] Kaiping Zheng, Shaofeng Cai, Horng Ruey Chua, Melanie Herschel, Meihui Zhang, and Beng Chin Ooi. 2022. DyHealth: Making Neural Networks Dynamic for Effective Healthcare Analytics. *Proc. VLDB Endow.* 15, 12 (2022), 3445–3458.

Received July 2022; revised October 2022; accepted November 2022