Privacy-Preserving Encoding and Scaling of Tabular Data in Horizontal Federated Learning Systems

Tim Piotrowski[1(✉)], Zoltán Nochta[1], Manuel Karl[2], and Martin Johns[2]

[1] Karlsruhe University of Applied Sciences, Karlsruhe, Germany
{tim.piotrowski,zoltan.nochta}@h-ka.de,
[2] TU Braunschweig, Braunschweig, Germany
{m.karl,m.johns}@tu-braunschweig.de

Abstract. Federated Machine Learning (FML) enables multiple entities to collaboratively train a shared ML model while keeping their local datasets private. By combining data and knowledge across participants, FML can produce better models than those trained on isolated datasets. The preprocessing of training data, such as scaling numerical features and encoding categorical variables, is essential to ensure data uniformity and thus improve model quality. In horizontal FML systems, participants have data with the same features, but own different samples. Achieving a uniform preprocessing among participants usually requires revealing statistics about training data or even the exchange of sensitive categorical values, which undermines privacy. To avoid it, privacy-preserving preprocessing methods are necessary. This work addresses the secure scaling of numerical features and the consistent encoding of categorical data in FML systems. We propose a privacy-preserving approach for Z-Score normalization that allows participants to scale numerical features collaboratively without revealing local statistics. We evaluate eight methods for a uniform encoding of categorical values across the system and assess their impact on model quality, dimensionality, scalability, and privacy. Our experimental evaluation on five datasets demonstrates that system-wide consistent preprocessing improves model performance compared to isolated approaches. However, we observe a trade-off between privacy preservation and efficiency. Based on our findings, we provide practical recommendations for selecting suitable encoding strategies depending on the use case, aiming to balance privacy, data consistency, and model performance in FML settings.

Keywords: Federated Learning · Preprocessing · Tabular Data · Encoding · Scaling · Secure Multiparty Computation · Security

M. Dalla Preda et al. (Eds.): ARES 2025, LNCS 15992, pp. 402–424, 2025.
https://doi.org/10.1007/978-3-032-00624-0_20

1 Introduction

In the recent decade, Machine Learning (ML) has become increasingly capable of tackling complex problems such as credit card fraud, or detect cancer based on images. ML models are typically created and managed by a single entity that stores and utilizes large volumes of data, including images, text files, and structured tabular data, for model training and testing purposes. However, storing and processing the data of multiple data owners raise security and privacy concerns and can also violate legal regulations. In sectors, such as finance or healthcare, sensitive data must not leave the respective institution.

Horizontal FML enables multiple independent data owners to collaboratively train ML models to solve a common business problem. In FML, participants train models locally on their datasets and share only the calculated model weights (or gradients) across multiple rounds to build a common global model. During the training, no training or test data is exchanged between participants.

In centralized FML systems, a server collects the locally calculated weights from connected participants and computes the resulting global model [12]. In decentralized FML settings, participants train the model without involving a central entity or server [16,26]. In both setups, all participants receive the same (initially untrained) ML model, use the same optimization algorithm, and agree on specific training parameters, including learning rates, target accuracy, and the maximum number of iteration rounds. In addition, they must prepare their training datasets before the training can begin.

Pre-processing is particularly crucial when using structured (tabular) data stored in relational databases for model training. It consists of multiple tasks, such as feature selection, data cleaning and data transformation, which encompasses encoding categorical feature values and scaling numerical features. Previous work on tabular data in FML has focused mainly on feature selection, with limited consideration of data transformation steps. Consequently, existing FML research implicitly assumes that data transformation is carried out independently by each participant or performed by a central server. Both assumptions lead to problems: The isolated preprocessing of local training data at each site can lead to inconsistent feature representations in the overall system. These inconsistencies can degrade the quality of the jointly trained model. Conversely, centralized preprocessing ensures high model quality, but requires the transfer of sensitive data to the server. For example, system-wide consistent encoding of categorical features requires the transfer of all distinct values of all categorical features in plain text, e.g., patient names, medical records, which is infeasible when dealing with sensitive information. Many studies bypass these challenges entirely by preprocessing the entire dataset before splitting it into simulated federated datasets, neglecting the complexities and constraints of real-world FML settings.

So far, limited attention has been paid to the challenges posed by data transformation in FML. The specific information required to be transmitted to a central server for consistent encoding and scaling across all datasets in FML has not yet been thoroughly examined.

Our paper addresses these gaps, examines the limitations of local preprocessing in FML, determines the required information exchange between participants to perform uniform preprocessing, and introduces a federated approach to run data transformation tasks that enable consistency while preserving data confidentiality. We propose and evaluate specific protocols and cryptographic techniques that prevent the exchange of sensitive data during encoding and scaling. Our main contributions are as follows:

1. The presentation of a federated data transformation schema consisting of federated scaling and federated encoding applicable in horizontal FML environments.
2. A privacy-preserving protocol for Z-Score normalization, allowing numerical feature scaling within federations without exposing local statistical data, thus maintaining the confidentiality of participant information while ensuring effective preprocessing.
3. Introduction of eight protocols, that partly leverage cryptographic methods, to uniformly encode categorical features in FML systems without sharing confidential or sensitive data.
4. A comprehensive evaluation of the approaches through experiments, focusing on model quality, federated scalability, the impact of input dimensionality, and the extent of shared information, followed by a discussion and selection of appropriate algorithms for different use cases.

2 Background

2.1 Federated Machine Learning

In a FML system, n independent entities jointly train the ML model to address a common challenge without revealing training data D_i [12]. Conventionally, each participant P_i exchanges only locally computed model updates, such as gradients $\nabla\mathcal{L}_i$ or model weights θ_i^t with a central server. The server creates and refines the aggregated model using Federated Stochastic Gradient Descent (FedSGD) or Federated Averaging (FedAVG) respectively:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \sum_{i=1}^{n} \frac{|D_i|}{\sum_j |D_j|} \nabla\mathcal{L}_i \quad (1) \qquad \theta^{(t+1)} = \sum_{i=1}^{n} \frac{|D_i|}{\sum_j |D_j|} \theta_i^{(t+1)} \quad (2)$$

In FedSGD (Eq. 1), the server updates the global model by computing a weighted average of the gradients received from all clients, where each client computes $\nabla\mathcal{L}_i$ using a small batch of its local dataset D_i. The weight $\frac{|D_i|}{\sum_j |D_j|}$, ensures that clients with larger datasets contribute more to the global update, whereby $\sum_j |D_j|$ represents the total size of the data of all participants and η is the learning rate. In FedAVG (Eq. 2), the server aggregates θ_i^t of all clients, after each client has trained on its entire dataset for multiple rounds. After aggregation, the new global model $\theta^{(t+1)}$ is sent back to all clients for the next round of training. This iterative process continues until the model converges or a stopping criterion is met.

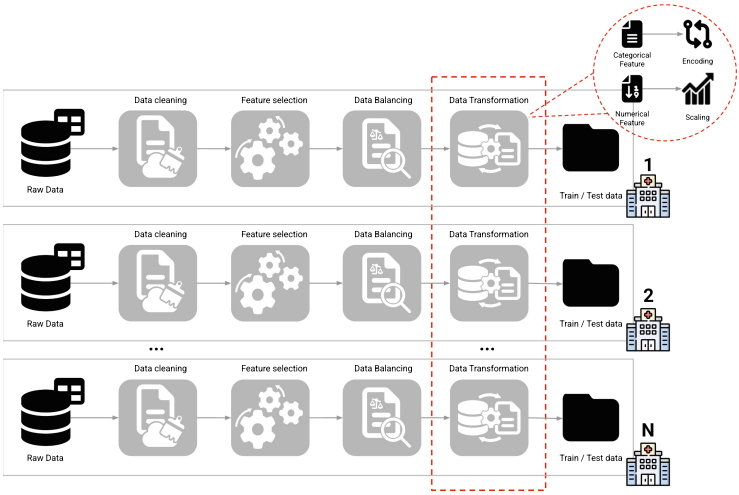

Fig. 1. Example of a federated preprocessing pipeline applying uniform data transformations across all clients.

2.2 Preprocessing of Tabular Data

Preprocessing of training and testing data is crucial for ensuring high data quality and enhancing the learning capability of machine learning models. Figure 1 illustrates a potential preprocessing pipeline for tabular data in federated learning, comprising feature selection, data cleaning, balancing, and transformation. As research on preprocessing in federated tabular settings is still limited, this pipeline represents our conceptual framework to guide future exploration.

The goal of feature selection is to identify and select the most relevant features from the dataset, eliminate irrelevant, redundant, or highly correlated features, and reduce dimensionality if necessary.

Data cleaning aims to remove duplicates and outliers from the dataset and to handle missing values. Duplicates add little value to ML model training, while outliers can negatively affect the model's performance. Missing values often arise in datasets, such as time-series data, due to failures during data collection. If not addressed, these gaps prevent the model from interpreting the affected data points. Common handling methods include imputing likely values manually, replacing missing values by the feature's mean, or removing incomplete data points.

The data balancing phase applies sampling (over-sampling or under-sampling) of data to ensure a more equal distribution of classes to improve model performance and prevent bias due to dominant classes.

In this work, we focus on the data transformation phase as highlighted in Fig. 1. This phase encompasses encoding of categorical features and scaling of numerical features to ensure consistency and to enhance the resulting model's performance.

Scaling is a transformation that maps data from its original space $\mathbb{R}^n$ into a standardized subspace of $\mathbb{R}^n$ to ensure numerical stability and improve model performance. Mathematically, it applies a function $f : \mathbb{R}^n \to \mathbb{R}^n$, where each feature x_i is transformed into $\tilde{x}_i = f(x_i)$ to adjust its range, variance, or distribution while preserving relative structures within the dataset. This transformation facilitates better optimization, by preventing features with larger magnitudes from dominating the learning process.

Encoding methods, such as one-hot encoding and label encoding, map categorical variables to numerical representations, facilitating their processing by ML models. Both methods begin by extracting the unique categorical values of a feature, forming a set $S = \{s_1, s_2, \ldots, s_n\}$, commonly referred to as "bag of words." A dictionary function $f : S \to \mathbb{N}_0$ assigns each value a unique non-negative integer, based on alphabetical order or metrics like term frequency-inverse document frequency [5]. In label encoding, each $s_i \in S$ is replaced by its label $f(s_i)$, which is especially suitable for ordinal features with natural ordering, such as European dress sizes.

If a categorical feature cannot be sorted or relationships between the values are of importance, one-hot encoding should be preferred. For one-hot encoding, a categorical value s_i is represented as a binary vector $v_i \in \{0, 1\}^{|S|}$, where the j-th entry is a 1 if $f(s_i) = j$ and a 0 otherwise.

This ensures that each categorical value is mapped to a unique basis vector in $\mathbb{R}^{|S|}$, preserving categorical distinctions without imposing an artificial ordinal structure. For example, when using this method to encode the feature "illness" with a dictionary $d_{illness} = \{arthritis = 0, diabetes = 1, pneumonia = 2\}$, the resultant vector for a patient with arthritis would be represented as $v = [1\ 0\ 0]$.

3 Threat Model and Privacy Requirements

We consider FML systems involving a small number (n < 20) of independent participants, such as hospitals, that collaborate to address shared challenges while considering organizational and legal constraints. The FML system uses a central server provided by one of the participants or by a third-party orchestrator. We assume that the participants have already completed the data cleaning and balancing tasks locally and reached a consensus on feature selection as well. Now, they want to encode and scale relevant features in a federated manner. To properly encode categorical features, participants must transmit plaintext data, e.g., about patients' medical conditions, to the server. Hospitals are understandably unwilling to share such data, as it could expose details about the prevalence of specific diseases within their patient populations. Such disclosures would compromise privacy and violate trust. Scaling of numerical data, on the other hand, requires the exchange of local statistical properties, like *std* or *mean* of each feature. Features, such as the Charlson Comorbidity Index (CCI) may contain highly sensitive information. The CCI is a weighted metric designed to predict the one-year mortality risk for patients with specific comorbidities. Access to statistical properties of the CCI could allow FML participants to gain insights

into others' datasets. For instance, observing higher *mean* or *std* values than those in their own data could indicate an elevated mortality risk among patients at a specific institution or reveal trends in patients' health.

With regard to trustworthiness, we assume that every participant and the server are Honest-but-Curious (HBC) entities. An HBC participant basically follows the protocol, but attempts to extract as much information as possible from legitimately received messages. In our scenario, it includes categorical feature values in clear text and numerical statistics of local data sets.

Concerning privacy, we aim to minimize the amount of data that have to be exchanged in the FML system with emphasis on these requirements:

1. The server cannot access or learn any specific value of the categorical data.
2. The server cannot access the local statistics about numerical data of individual participants.
3. Each participant gains only knowledge about the categorical values that are also present in its own data set.
4. No participant has access or can learn statistics about numerical data of other participants.

4 Related Work

We are not aware of previous research that explicitly addresses the privacy problem of encoding and scaling in FML systems. However, there are publications that discuss security aspects of feature selection in the field of FML analysis. FML analytics has emerged to enable secure federated data insights without disclosing data [24].

Akavia et al. [2] proposed the first Secure Multiparty Computation (SMPC) based protocol for private feature selection. It combines multiple SMPC methods and leverages Gini impurity for efficient feature scoring. Li et al. [17] propose a privacy-preserving feature selection method using Homomorphic Encryption (HE). It builds sparse ridge regression models on horizontally partitioned data, outsourcing computation to two non-colluding servers and integrating Recursive Feature Elimination for feature pruning.

Eleks et al. [9] enable data consumers to analyze data without exposure and propose strategies for handling missing or unclean data, such as reformatting, feature generation, or adding attributes. Hsu and Huang [14] propose an FML framework using HE to secure training data and model aggregation. An initiator selects participants by sending encrypted queries and suggests pre-processing based on a shared schema. While providing strong data protection, HE incurs high computational costs and scales poorly with complex models.

Privacy vulnerabilities in training have been widely studied. FedSGD-based systems, in particular, are susceptible to data reconstruction under HBC assumptions [11,19,28]. In contrast research on FedAVG when participants have large data silos is limited, especially for tabular data, as most studies focus on image or speech data. Fowl et al. [10] demonstrate the feasibility of inferring local data

even in a FedAVG environment. Vero et al. [23] propose the first attack on tabular data in FML settings. The paper even highlights our privacy implications of sharing data during preprocessing, as the server leverages information about categorical values and numerical statistics to reconstruct the data of participants. Wu et al. [27] also explored tabular data, using label property inference with a Generative Adversarial Network to extract information about target class distributions.

While general privacy-enhancing techniques such as SMPC, HE and Differential Privacy (DP) are widely applied to model aggregation [1,4,18] to tackle these attacks. Recent research includes Das et al. [7], which propose a communication-efficient protocol that combines DP and SMPC for collaborative deep learning in vertical and horizontal FML, significantly reducing overhead while maintaining strong privacy guarantees, and highlighting the feasibility of scalable privacy-preserving training. Further Eichner et al. [8] introduce a federated system architecture combining Trusted Execution Environments, Differential Privacy, and SMPC to secure both client and server-side computations, enabling verifiable privacy guarantees at scale.

However, application of these techniques to preprocessing remains limited. Our work addresses this gap by researching privacy-preserving protocols for encoding and scaling in horizontal FML.

5 Local and Server-Based Data Transformation

This section presents an analysis of local versus server-based data transformation in FML, highlighting the trade-offs and the information exchanges required for the server-based approach. We will examine the strengths and limitations of both methods, before assessing their performance in terms of model quality.

5.1 Local Scaling and Encoding in FML

A numerical feature f can be scaled using methods, such as Min-Max or Z-Score Normalization, reducing values to a standardized range:

$$\tilde{x}_i = \frac{x_i - f_{\min}}{f_{\max} - f_{\min}} \quad (3) \qquad z_i = \frac{x_i - \mu_f}{\sigma_f} \quad (4)$$

Min-Max normalization (Eq. 3) uses the feature's minimum and maximum values $f_{\min}$ and $f_{\max}$ to transform a value x_i. Z-Score normalization (Eq. 4) requires the mean μ_f and the standard deviation σ_f of f for standardization. Due to deviations of the parameters μ_f and σ_f as well as of f_{max} and f_{min} in the local data sets of participants, performing the normalization independently at each site would result in inconsistent scaling of numerical features, making it challenging for the model to recognize similarities in the input data during training.

Categorical feature encoding involves each participant generating a local mapping of unique values in their dataset to numerical identifiers, creating a dictionary for all distinct categorical values. These values are then transformed into

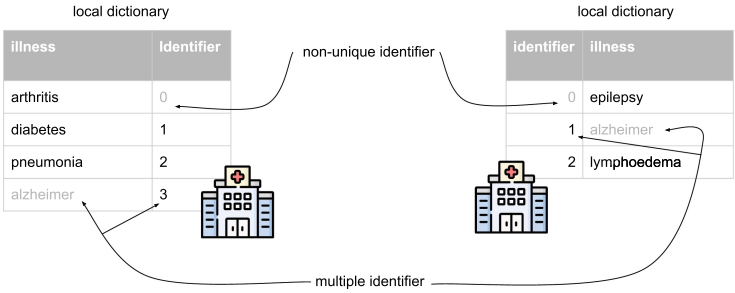

Fig. 2. The dictionary problems visualized on illnesses held by two FML hospitals (Icon source: https://www.flaticon.com/de/kostenlose-icons/krankenhaus)

numerical representations, through label encoding or one-hot encoding. However, isolated local encoding in FML can lead to two types of *dictionary problems*, as illustrated in Fig. 2. When participants independently perform the encoding, the same identifiers are used for different categorical values, thus creating non-unique identifiers. In the example, both participants use a 0 to identify two different illnesses, "arthritis" and "epilepsy", resulting in a collision. Furthermore, identical values are represented by multiple different identifiers. For example, both participants possess the feature value "alzheimer" but encode it as 3 and 1, respectively. Consequently, a system-wide unique identification of the strings is not guaranteed.

5.2 Server-Based Encoding and Scaling in FML

In FML, the central server typically oversees the orchestration of the protocol and the model aggregation, making it a suitable entity to facilitate uniform encoding and scaling of features. However, achieving consistency requires participants to share specific information with the server. For example, for Min-Max normalization, each participant provides its local minimum and maximum values of the numerical features. The server then determines the global minimum and maximum across all datasets and shares these values with the clients.

The uniform encoding of categorical data on the other hand relies on establishing a shared dictionary. In a centralized FML environment, the server aggregates the clear text local dictionaries into a global dictionary, eliminating the dictionary problems. This global dictionary is then shared with the participants. Alternatively, an open-source dictionary can be used to encode categorical features containing words of a language, such as diseases. For example, hospitals collaborating to train a model might have a feature listing treated illnesses. Using a global dictionary of worldwide known diseases for encoding would produce an excessively large one-hot encoding, containing more illnesses than encountered in the local datasets. This leads to sparse representations that reduce model performance and computational efficiency. Moreover, this approach fails when

Table 1. Comparison of Local vs. Server-Based preprocessing using the F1-Score in an FML environment with four participants

	Method			
Dataset	Local Preproc.	Uniform Encoding	Uniform Scaling	Server-Based
Adult	0.60	0.66	0.65	**0.69**
German	0.30	0.77	0.56	**0.79**
Health Heritage	0.63	0.67	0.64	**0.69**
Lawschool	0.69	0.69	0.70	**0.71**
CC Fraud	0.80	0.84	0.81	**0.90**

categorical features include custom terms, as often seen in computer science, due to the lack of predefined dictionaries for such data.

5.3 Performance Assessment

In the following, we investigate the impact of unified federated scaling and encoding on model quality compared to isolated local preprocessing. In our experiments we used the datasets Adult Census [3], German Credit Card [13], Health Heritage[1], Lawschool Admission [25], and the Simulated Credit Card Transactions[2]. These datasets include numerical and categorical features, representing a two-class prediction problem.

We partitioned each dataset into four subsets, each assigned to a different participant in the simulated federation. The data is non-independently and identically distributed (non-IID) among participants to simulate real-world scenarios, with varying categorical values, dataset sizes, and label distributions across participants, thus maximizing divergence. However, this distribution approach is inherently constrained, particularly for datasets with limited variability in their categorical features.

We conducted four experiments by varying the encoding and scaling methods during data preprocessing. In the first, the participants performed isolated local preprocessing and exchanged only input sizes to apply the padding needed to construct the encoding vectors. In the second, the server created a global dictionary for uniform encoding, and the clients scaled their own data locally. The third experiment combined server-based uniform scaling with client-side data encoding. In the fourth, the server orchestrated both uniform encoding and scaling tasks. After each preprocessing, we performed federated model training using FedAVG. Hereby, a small neural network tailored to each dataset, optimizing its architecture to fit the specific features and patterns was used. The results shown in Table 1 demonstrate that uniform scaling and encoding using a server, significantly improve the model's quality. A notable example is the German Credit Card dataset, which showed a 49% increase in the F1-score. The

[1] https://www.kaggle.com/c/hhp.
[2] https://www.kaggle.com/datasets/kartik2112/fraud-detection.

Lawschool dataset, on the other hand, showed minimal benefit from federated preprocessing, likely due to its limited categorical values. Therefore, most unique values are already present locally, thus participants' dictionaries closely align or match with the global dictionary. The best results can be achieved, combining both uniform encoding and scaling.

In addition, the experiments showed that the datasets mostly benefit from uniform encoding, while uniform scaling seems to have a lower impact on these datasets. We attribute this to the influence of categorical values on the model's decision-making.

6 Privacy-Preserving Z-Score Normalization

Z-score normalization (see Eq. 4) is a well-known and widely used data preprocessing technique to scale numerical features in ML training datasets. In FML systems, it especially helps avoid model quality deterioration caused by undetected outliers in participants' private datasets.

As the required μ_f and σ_f differ on each local dataset, we propose a protocol that achieves uniform Z-Score in FML systems. However, in FML system, revealing local statistics raises privacy concerns. To address this, we extend our approach with a privacy preserving method so that no local statistics need to be exchanged. The n participants have to collaboratively compute the global mean μ_{gf} and the global standard deviation σ_{gf} for each numerical feature f that requires normalization:

$$\mu_{gf} = \frac{\sum_{i=1}^{n}(\mu_{if} * |D_i|)}{\sum_{i=1}^{n}|D_i|} \quad \sigma_{gf} = \sqrt{\frac{\sum_{i=1}^{n}\sigma_{if}^2}{\sum_{i=1}^{n}|D_i|}} \tag{5}$$

First, μ_{gf} is computed as the weighted average of local means μ_{if}, where weights are the dataset sizes $|D_i|$. Next, each participant calculates its local variance σ_{if}^2 using μ_{gf}. Afterwards, σ_{gf} is derived by aggregating these σ_{if}^2 values, devided by the total dataset size, and taking the square root.

Knowing both values, each participant can carry out the Z-score normalization of a feature f locally. However, to carry out the required calculations, each participant P_i would normally disclose $|D_i|$ as well as μ_{if} and σ_{if}^2 of each numerical feature. To protect participants' sensitive data, we adopt and apply the Secure Average Computation (SAC) algorithm introduced by Wink and Nochta [26]. Originally designed for decentralized FML, we adapt SAC to the centralized setting with the server acting as a passive message relay.

To compute $\sum_{i=1}^{n}(\mu_{if} * |D_i|)$ for a feature f, each participant P_j generates n positive random numbers, $r_{1j}, r_{2j}, ..., r_{nj}$, one for each participant in the federation and multiplies its local μ_n by$|D_i|$. Afterwards, each participant P_j calculates n weighted parts of its own μ_n accordingly:

$$\mu_{i,j} = \mu_j * \frac{r_{ij}}{\sum_{i=1}^{n} r_{ij}} \tag{6}$$

where P_i is the intended receiver, P_j is the sender, and μ_j is the local mean of f that P_j wants to keep secret. These parts are exchanged between the participants, with each participant keeping its self-calculated $\mu_{i,j}$ value for $i = j$. By summing their own value where $i = j$ with the received values, each participant computes the partial sum $s_i = \sum_{i=1}^{i=n} \mu_{i,j}$, which is then exchanged with the others. Finally, the sum $s = \sum_{i=1}^{i=n} s_i$ is computed by each participant, representing $\sum_{i=1}^{n}(\mu_{if} * |D_i|)$.

By using this algorithm, all three required sums $\sum_{i=1}^{n}(\mu_{if} * |D_i|)$, $\sum_{i=1}^{n} |D_i|$ and $\sum_{i=1}^{n} \sigma_{if}^2$ can be calculated without disclosing any local values.

7 Privacy-Preserving Feature Encoding Methods

In the following, we present eight approaches to encode categorical data that aim to minimize or even eliminate the need for exchanging sensitive data. We categorize them as Round-Robin Identifier Allocation Protocols, Secure Multiparty Computation Protocols, and Alternative Encoding Protocols.

7.1 Round-Robin Identifier Allocation Protocols

All three protocols in this group utilize a round-robin (RR) approach—proposed in this work—for assigning globally unique identifiers to categorical values across the FML system. As shown in Fig. 3, this approach organizes all FML participants into a logical ring, enabling message exchange in a coordinated RR fashion. The process starts at participant P_1, who creates a dictionary d_f for a given categorical feature f. P_1 then transmits the highest identifier value assigned in the dictionary $max_1(d_f)$ to the next participant P_2 in the ring. P_2 generates its own dictionary by incrementing the just received value to create the lowest identifier $min_2(d_f) = max_1(d_f) + 1$. This procedure continues sequentially through all participants until the last participant, P_n, sends $max_n(d_f)$ to P_1, which then broadcasts $max_n(d_f)$ as the highest identifier value of f to all other participants. This guarantees that all participants are aware of the maximum encoding length, enabling, for example, a consistent one-hot encoding of f at all FML nodes. To ensure scalability and efficiency, each participant's message includes the highest identifiers from all their categorical features, rather than covering just one feature at a time.

Unique Identifier One-Hot Encoding (UHE) requires the participants to generate a unique one-hot encoding for all categorical features locally, after the RR protocol established unique identifiers and the global maximum $max_n(d_f)$ for each categorical feature. These encodings have a length corresponding to the $max_n(d_f)$ for a given feature f, replacing the original features.

This process enhances upon isolated local one-hot encoding by ensuring FML-wide unique identifiers for categorical values, while minimizing information disclosure in contrast to server-based encoding. This approach does not address the

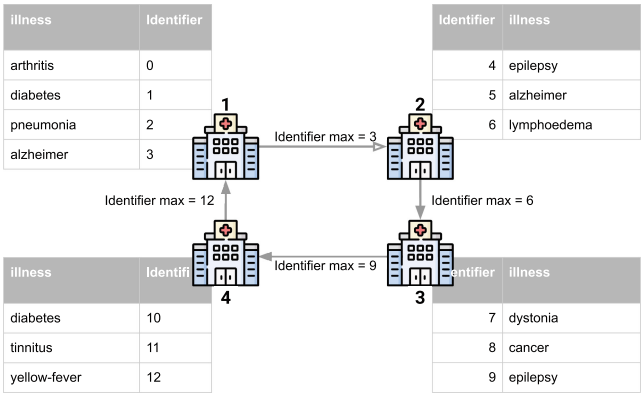

Fig. 3. Our Round-Robin protocol to assign unique identifiers to categorical features.

challenge of multiple identifiers for the same categorical value. Instead, we investigate whether simply establishing unique identifiers is sufficient for the model to learn and generalize across different encodings of the same underlying value.

Private Set Intersection (PSI) is a cryptographic protocol that allows users, to determine the intersection of their datasets (i.e., common elements) without revealing the rest of their data [6]. We apply PSI to resolve cases where multiple identifiers have been assigned to the same categorical value in participants' dictionaries. This collaborative process ensures a consistent encoding of identical values of a feature f while maintaining privacy.

Our protocol follows these steps: After assigning unique identifiers using the RR protocol, all n participants perform PSI with all other $n-1$ participants to identify their categorical value intersections $\bigcap_n = concat(\bigcap_1, \bigcap_2, ..., \bigcap_{n_1})$, then standardize encoding by adopting the lowest identifier value among intersections. For example, if participant P_x assigns $alzheimer = 3$ and P_y sets $alzheimer = 0$, P_x updates to 0, ensuring consistent representation of this disease across the federation.

We focus on the asymmetric PSI methods Asymmetrical Labeled PSI (APSI) and Practical Over-Threshold Multi-Party PSI (OT-MP-PSI), chosen for their adaptability to varying dictionary sizes, code availability, scientific recognition, and suitability for centralized and decentralized FML settings.

Asymmetrical Labeled Private Set Intersection (APSI) introduced by Cong et al. [6], uses a one-to-one communication pattern, where one participant acts as a client and the other as a server. Note that only the client learns the intersection of the two sets during a single protocol execution.

In our FML setting each participant runs the APSI protocol twice with every other participant: once as a client and once as a server, ensuring that all parties learn all potential intersections. To deploy APSI in FML participant must offer

separate service instances for each other participant, because requests cannot be processed simultaneously due to the encryption. For a federation with n participants, this design requires $n \times (n-1)$ protocol executions to compute pairwise intersections across all participants.

Given the central FML server's role in managing communication in the federation, our protocol ensures that all APSI interactions are routed through the server. Since PSI requests are encrypted, the server cannot access the exchanged information, preserving participant privacy. One key advantage of APSI is its ability to exchange labels during the intersection computation. Thus, participants automatically receive the corresponding labels from intersecting values, eliminating the need for sending additional label requests.

Practical Over-Threshold Multi-Party PSI (OT-MP-PSI) by Mahdavi et al. [20] enables n entities to compute the intersection of their datasets, revealing an element only if it appears in at least $t \leq n$ datasets. It employs Shamir's Secret Sharing, encoding each element into a polynomial with a secret fixed at 0. An intersection is detected if any t shares reconstruct to 0.

Chosen participants are assigned as a keyholder (generating polynomials) or a reconstructor (verifying intersections). The keyholder engage in polynomial share generation with the participants, which then put these shares into padded bins. The reconstructor iterates over these bins, reconstructing secrets from different participants' shares. If a reconstructed value is 0, the intersection is reported.

To adapt this approach in our use case, we set $t = 2$, detecting all pairwise intersections. As the method is designed for numerical sets, we convert each categorical value x into an integer by first encoding it as bytes and then casting to integer: $x_t = int(bytes(x))$. At the start, the central server assigns the first and last participants in the RR ring structure as keyholder and reconstructor, respectively.

The reconstructor learns only the bin positions of intersections, not their contents. Upon detection, it informs participants of intersecting positions. For instance, P_1 learns that its element at position $(2, 3)$ in the padded bins intersects with P_2's element at position $(1, 4)$. Using these positions, participants exchange their identifiers, but not their values.

To minimize data transfer between participants, only participants with lower indices in the RR ring provide identifiers, as they assigned lower identifiers. Specifically, P_1 never requests identifiers, while P_n queries $P_1, \ldots, P_{n-1}$ as needed.

7.2 Secure Multiparty Computation Protocols

The following methods were designed to achieve encoding within the federation with minimal effort and without relying on complex encryption techniques. Both approaches leverage the SAC protocol previously employed in Sect. 6, enabling participants to collaboratively compute specific metrics without disclosing local values.

Bytes-Norm (BN) employs the encoding scheme for categorical features introduced in Sect. 7.1, where each categorical value x of a feature f is cast to an integer via its byte representation, formally $x_t = int(bytes(x))$. We treat the resulting numerical values x_t as standard numerical features, enabling the application of privacy-preserving Z-Score normalization using the μ_g and σ_g of the transformed feature f_t. The Z-Score normalization eliminates the extreme dimension differences of the x_t values. Unlike one-hot encoding, this approach treats categorical values as continuous numerical data, yielding a distinct encoding scheme. However, similar to label encoding, it does not preserve semantic relationships between categorical values when no natural order is present.

Mod-Norm (MN) extends the typecasting approach $x_t = \text{int}(\text{bytes}(x))$ by incorporating a binning approach based on a modulo operation, enabling the mapping of categorical values into a smaller numerical range. Each participant first transforms every categorical value x of the categorical feature f_i to x_t. Participants then compute the number of unique values $unq_i = |\text{unq}(f_i)|$ locally and securely aggregate the global sum $s_i = \sum_{j=1}^{n} unq_{ij}$ with the other n participants using SAC. The result s_i serves as an upper bound estimate of the distinct categorical values (duplicates or rather intersections are not considered between participants). Finally, each x_t is then encoded as $H_i = x_t \bmod (2s_i)$, with the factor 2 included to reduce the likelihood of collisions.

7.3 Alternative Encoding Strategies

This section explores encoding strategies from related work, proposed as alternatives to generic methods like one-hot or label encoding. Originating from natural language processing tasks or standard (i.e., non-federated) training settings, we evaluate their feasibility to encode tabular data in a FML setting.

FastText (FT) is a pre-trained word embedding model designed to map words into high-dimensional vector spaces [15]. Formally, given a bag of words S and a predefined embedding dimension d, FT assigns each entry $x \in S$ a dense vector representation $\phi(x)$. These embeddings are typically trained on extensive corpora to capture semantic relationships effectively. For example, synonyms or related terms like "illness" and "alzheimer" are positioned close to each other in the feature space, reflecting their contextual similarities. While this property is valuable for text representation, our work focuses on a different application: We utilize FT mainly to encode categorical features within a dataset.

A key advantage of FT is its ability to generate embeddings for out-of-vocabulary words using subword information. This property ensures that any categorical value $x \notin S$ can still be mapped to a unique vector $\phi(x)$ by leveraging character-level n-grams.

In a FML setting, this guarantees that all participants can independently compute identical vector representations for the same categorical feature value without exchanging raw data. Specifically, given a categorical feature f with a

dictionary of unique values D_f, we apply FT to obtain a fixed-length vector representation for each entry $\forall x \in D_f, v_x = \phi(x), v_x \in \mathbb{R}^{300}$ where we utilize the pre-trained English word embedding model of FT with a vector dimension of $d = 300$. This enables a consistent and privacy-preserving encoding mechanism for categorical features across all federated learning participants.

Target Encoding (TE) replaces categorical values with the mean of their corresponding target variable, with the target class typically set to 1 in binary classification problems [21]. A categorical value x of a feature f is transformed such that:

$$x_t = \frac{frequency(x) * p(f = x|c = 1) + \alpha * \mu_c}{frequency(x) + \alpha} \tag{7}$$

where frequency(x) represents the frequency of x in the feature, $p(f = x \mid c = 1)$ is the probability of x occurring given the target class $c = 1$, μ_c is the mean of the target label c, and α is a smoothing parameter to mitigate overfitting or poor encoding for small datasets. TE is prone to overfitting and target leakage, where information from the target variable unintentionally influences predictions, causing overly optimistic results and misleading model performance assessments.

For federated TE, we propose two strategies: local TE without communication, ensuring privacy but risking inconsistent encodings, and sharing probability values to unify encodings, compromising privacy, as probabilities and clear text values are revealed. To prioritize privacy, we adopt the first approach, allowing participants to encode values independently without data exchange.

Rolling Hash (RH) proposed by Nagy et al. [22] serves as a benchmark to compare our proposed approaches against an established technique for federated encoding. While originally applied to NLP tasks, this method is one the only work we are aware of that addresses federated encoding, making it a valuable point of reference. Adapting the protocol for categorical features, each categorical value is lowercased and hashed using $H = \sum_{i=1}^{|s|} s_i * P^i \bmod M$ where $|s|$ is the string length, s_i is the position of the i-th character in the alphabet ($a = 1, b = 2, \ldots, z = 26$), $P = 31$ is a prime exceeding the alphabet size, and $M = 5000$ defines the vector length. The resulting H determines the position in the M-length vector set to 1 for a word's occurrence. Repeating this for each categorical value produces a 5000-length one-hot vector encoding. Following the original work, which represents an entire text features in a single 5000-length vector, we similarly encode all categorical features in a shared 5000-length vector. This reduces sparsity compared to the standard one-hot encoding per feature, leveraging the assumption that this dimensionality suffices for multiple categorical variables, as it does for text data.

8 Experimental Evaluation

To identify the most suitable secure encoding protocol, we evaluate them based on four dimensions: impact on model quality, scalability in federated settings

Table 2. Dataset specifications, including amount of categorical and numerical features as well as categorical features encoded under different approach

Dataset	Features	Categorical	Numerical	One-Hot	FT	RH	Data Points
Adult	13	8	5	105	2404	5013	45222
German	20	13	7	63	3908	5020	1000
Health Heritage	17	6	11	110	1811	5011	218415
Lawschool	7	5	2	39	1502	5006	96584
CC Fraud	7	4	4	1206	904	5004	137775

(message exchange and runtime), changes in input vector dimensionality, and information leakage to participants or the central server.

For this evaluation, we utilize the same datasets described in Sect. 5.3. In Table 2 we collected an overview of the datasets, incl. the number of numerical and categorical features, size of the full dataset, and input dimensions using different encodings. The number of features describes the state after the datasets were feature engineered, i.e., by adding or removing different features.

The proposed protocols were implemented using Protocol Buffers for message formatting and ZeroMQ for message transmission between clients and the server. Preprocessing and federated training were simulated on a single machine with four NVIDIA TITAN Xp GPUs (12 GB VRAM), four Intel(R) Xeon(R) CPU E5-2660 v4 @ 2.00GHz, and 64 GB RAM. Unless stated otherwise, the simulated federation included one server and four participants. This baseline was chosen because the SAC protocol requires at least three participants and aligns with the available resources, allowing each participant to use a dedicated GPU and CPU.

8.1 Impact on Model Quality

We first assess each method's impact on model quality by applying the selected encoding for preprocessing, followed by federated training. We track test F1-scores across epochs and stop training once performance stabilizes. The F1-scores across the federation were calculated by summing the local confusion matrices derived from each participant's test data. However, in real federations, such matrices should never be exchanged, as they could reveal sensitive information, including participant labels and data quantities. This approach was used solely for evaluation in the context of this work.

Both PSI methods produce the same results, therefore we collectively evaluated them as "PSI". We integrated the authors' PSI implementations into our preprocessing service using a C-to-Python wrapper. As shown in Table 3, PSI outperformed all other methods by preserving both the input data and the underlying model. It delivers results identical to unsecured server-based encoding with a global dictionary while also ensuring data privacy.

Table 3. F1-Scores of federated model training with four participants using different approaches for encoding, including isolated local preprocessing, in a non-IID setting. The best results are highlighted in bold for each dataset

	Server-Based	Privacy Preserving Methods							
Dataset	Benchmark	PSI	Local	UHE	BN	MN	FT	RH	TE
Adult	0.69	**0.69**	0.60	0.50	0.0	0.30	0.62	0.65	**0.69**
German	0.79	**0.79**	0.30	0.55	0.0	0.53	0.75	0.31	0.55
Health Heritage	0.69	0.69	0.63	0.63	0.54	0.59	0.67	**0.70**	0.65
Lawschool	0.71	**0.71**	0.69	0.46	0.53	0.56	0.66	0.69	0.67
CC Fraud	0.90	**0.90**	0.80	0.64	0.0	0.83	0.89	0.88	0.84

The Bytes algorithm performed worst, consistently underperforming local preprocessing and yielding F1-Scores of 0 for three datasets. MN and UHE showed moderate but unstable performance, often degrading model quality. UHE only outperformed local preprocessing once, highlighting that establishing unique identifiers alone is insufficient.

FT, RH, and TE emerged as the best-performing alternatives to PSI. Although TE matched federated preprocessing in one instance, it struggled on the German dataset. Similarly, RH beats local preprocessing by only 1% on the German dataset. FT demonstrated stable performance across all five datasets, outperforming local preprocessing on those with numerous categorical values. Apart from FT and PSI, the other approaches resulted in significant loss of model quality. We attribute this to the fact that they introduce a lot of sparsity or use numerical representation for features that can't be sorted.

8.2 Scaling Analysis

We assess federated scalability by evaluating message complexity among n participants using O-notation. For OT-MP-PSI we estimate the worst-case scenario, where participants in the ring must request intersections from all preceding participants. PSI achieves the highest model quality but incurs the highest communication overhead. APSI, in particular, exhibits rapid growth in communication demands as n increases, while BN, MN, and UHE scale moderately. FT, RH, and TE are the most efficient, requiring no inter-participant communication.

Empirical runtime evaluations on the Adult dataset confirm these trends. We measure the time from the server's preprocessing request to completion (Table 4). Since experiments were conducted locally, network-related delays in real-world federations were not considered. Execution times were measured as n increased from 4 to 10. BN, MN, and Local Encoding exhibit the fastest runtimes (< 100 ms), with Local Preprocessing unaffected by federation size. RH, TE, and UHE take longer, with TE being four times faster than RH.

Among research-based methods, FT is the slowest (20 s) due to embedding model loading. PSI-based methods have the highest latency. APSI is more effi-

Table 4. Runtimes (in msec) of encoding approaches using the Adult dataset

Participants	Local	MPSI	APSI	UHE	BN	MN	FT	RH	TE
4	≈41	63673	28280	264	89	52	≈20799	≈853	≈183
6	≈41	105210	64819	377	96	52	≈20799	≈853	≈183
8	≈41	119530	140913	520	99	62	≈20799	≈853	≈183
10	≈41	148776	227524	636	111	67	≈20799	≈853	≈183

cient than OT-MP-PSI for small federations (4âĂŞ6 participants) but less so beyond eight. These results indicate a clear trade-off between model quality and preprocessing time.

8.3 Input Dimension

We evaluate the input dimension by measuring the size of the original data point vector $|V|$ after encoding the categorical features $|C|$ and adding the numerical features $|N|$. For instance, in one-hot encoding, the original features are removed from the vector, and the encoding is added to the data point. The same applies to FT and RH, where a new encoding replaces the original features.

TE, MN, and BN achieve the smallest input dimensions by converting categorical features into numerical values. Unique Identifier One-Hot and PSI methods result in input dimensions comparable to traditional one-hot encoding, offering more efficient dimensions compared to isolated local encoding by avoiding padding. FT and RH substantially increase input dimensions, leading to higher hardware costs. FT scales linearly with the embedding size (300 in these experiments) and the number of categorical values, whereas RH generates fixed-size sparse vectors of 5000 in our setup.

8.4 Privacy Analysis

All introduced algorithms meet the established privacy requirements, defined in Sect. 3. While neither the server nor participants gain knowledge of others' local datasets, this does not eliminate the need for exchanging or disclosing certain information. PSI-based approaches reveal intersections $\bigcap_{\mathrm{f}}$ of a feature f between participants, thereby they only gain information about values present in their local datasets. Therefore, all parties have to consent to share potential intersections.

Additionally, BN and the protocols utilizing the RR, disclose an approximation of the sum of all unique values $\approx |unq(f)|$ in feature f. However, no intersections between participants are considered, and meaningful inferences are complex to draw for categorical features without limiting the value range. Furthermore, the second participant in the ring during RR can infer the size of the first participant's dictionary. Adding padding or noise can obscure this but increases one-hot encoding dimensionality, thus introducing sparsity.

Table 5. Comparison of the privacy-preserving categorical feature encodings

Method	Federated Scaling	Input Dimension	Δ F1-Score	Shared Infos
Local	$O(1)$	$\|N\|$ + One-Hot	15.4%	$\|V\|$
APSI	$O(4n - 2n - 1)$	$\|N\|$ + One-Hot	0%	$\approx \|unq(f)\| + \bigcap_{\mathrm{f}}$
OT-MP-PSI	$O(8n - 7 + \sum_{i=1}^{n} n - i)$	$\|N\|$ + One-Hot	0%	$\approx \|unq(f)\| + \bigcap_{\mathrm{f}}$
UHE	$O(2n - 1)$	$\|N\|$ + One-Hot	18.2%	$\approx \|unq(f)\|$
BN	$O(2n - 2)$	$\|V\|$	52.6%	Z-Score statistics
MN	$O(4n - 4)$	$\|V\|$	19.4%	$\approx \|unq(f)\|$
FT	$O(1)$	$\|N\|$ + Embed	3.8%	None
RH	$O(1)$	$\|V\|$ + Bucket	11.0%	None
TE	$O(1)$	$\|V\|$	7.6%	None

Z-Score normalization does not disclose local values. However, it reveals aggregated metrics, such as the sums of weighted local means, the total number of data points in the federation, and the sum of variances. Furthermore, both the server and participants gain access to μ_g and σ_g for each normalized feature. The same applies to MN, which uses the Z-Score to normalize the casted categorical values. Nonetheless, the exposure of these values offers limited actionable information, as the large numerical range of encoded strings makes it challenging to derive meaningful insights.

The FT, RH, and TE protocols meet all of our established privacy goals and exceed them in several areas. By eliminating the need to share data among participants, they prevent disclosing sensitive information and introduce no additional communication overhead. As a result, these protocols provide a higher level of privacy than all other approaches evaluated, achieving a standard comparable to the robust data confidentiality of locally isolated label encoding.

8.5 Discussion

Categorical feature preprocessing in FML must balance privacy, scalability, and model quality. Among the evaluated methods, PSI and FT offer the best trade-offs, while others compromise model quality, provide limited privacy benefits, or perform similarly to local preprocessing. Table 5 summarizes these results.

PSI ensures consistent encoding and model quality preservation, especially on datasets like the German dataset where other methods underperform. However, it poses privacy risks by revealing intersections and leaking metadata via the RR protocol. PSI also incurs high runtimes and scalability costs, though it integrates easily without hardware changes or model adjustments. We recommend PSI when model quality is critical, scalability is manageable, and a certain trust between participants and the server can be ensured. It is also preferable when FT's hardware demands are impractical. Between OT-MP-PSI and APSI, the latter is better for federations with up to 10 participants due to its decentralized

design, which improves resilience by enabling seamless continuation despite node failures or late additions.

FT offers a privacy-first approach by encoding data locally without participant communication, ensuring scalability with constant computational costs. However, it demands substantial resources, including a 7 GB pre-trained model, high memory usage, and increased neural network input sizes due to word embedding size. Additionally, it degrades model quality by about 3.8%. FT is ideal when privacy and scalability are priorities, particularly in untrusted environments where HBC assumptions don't hold and extra hardware is available.

Limitations: Privacy-related limitations of secure Z-Score normalization in FML arise from the exposure of global statistics, such as μ_g and σ_g. While these values might be accessible, inferring exact local contributions becomes increasingly difficult as the number of participants grows, reducing the risk of individual data leakage. However, global statistics remain essential to ensure consistent preprocessing across all participants.

As an alternative to SAC, we considered DP and HE for Secure Z-Score normalization, but their limitations make them unsuitable for this problem. DP protects individual values by introducing noise and could be viable in two-party settings where SMC is infeasible. However, it reduces precision while still requiring the exchange of global scaling parameters for consistent preprocessing. HE, on the other hand, imposes significant computational overhead, particularly in multi-party settings. Operations like standard deviation require square roots, which HE does not natively support and must approximate using complex polynomial functions, further increasing computational costs. Since both methods ultimately necessitate sharing global statistics, they offer no advantage over SMC, which we selected for its efficiency and practicality.

Additionally, we only cover Z-Score in its paper due to its resistance to outliers. If an Outlier Detection is integrated into a federated preprocessing pipeline, other alternatives, such as Min-Max scaling or Robust Scaling should be adapted and investigated in FML as well. Incorporating these techniques while preserving privacy poses challenges, requiring methods, such as Garbled Circuits, which demand further refinement. A comprehensive evaluation and implementation of various scaling methods within a FL setting warrant a dedicated research study.

In conclusion, none of the investigated approaches fully preserved model quality while ensuring complete privacy. Striking a balance between privacy and performance often necessitates some degree of data exchange, highlighting the ongoing challenge and the need for further novel methodologies to achieve both objectives effectively.

9 Conclusion

This paper examined the impact of encoding and scaling in horizontal FML, showing that inconsistent local preprocessing significantly degrades model performance. Primary contributors to poor model quality are the so-called "dictionary problems", resulting in duplicate or mismatched identifiers for the same

feature values. However, achieving uniform scaling and encoding requires sharing sensitive clear text categorical values and statistical properties, posing privacy risks. To address these concerns, we proposed a privacy-preserving Z-Score normalization method using Secure Multiparty Computation, ensuring that only aggregated global statistics are shared. For categorical encoding, we introduced eight privacy-preserving methods, each offering different trade-offs between privacy, scaling, input dimensionality and model quality. Experiments on five datasets demonstrated that a Round-Robin protocol combined with PSI matched the performance of unsecured uniform preprocessing but incurred high computational costs and requires trust between participants. The most private methods Rolling Hash and fastText avoid direct data exchange, though only fastText maintained competitive model quality. However, it also increased model complexity and hardware demands due to its use of word embeddings. Since our scaling and encoding methods are ML framework-agnostic, they are compatible with TensorFlow, PyTorch, and Scikit-learn. Leveraging our gRPC and ZMQ setup, the preprocessing prototype can be integrated into Flower pipelines prior to training, enabling secure and scalable preprocessing in practice. We recommend PSI when high model quality is essential, provided trust among participants and scalable resources are available. When privacy and scaling are the primary concerns, FT offers a secure alternative, though with efficiency trade-offs and local hardware requirements. To support future research and adoption, our source code and implementation details will be publicly released upon publication.

Acknowledgments. This work was co-funded by the European Union in the Interreg Upper Rhine program (project aura.ai).

References

1. Adnan, M., Kalra, S., Cresswell, J.C., Taylor, G.W., Tizhoosh, H.R.: Federated learning and differential privacy for medical image analysis. Sci. Rep. **12**(1), 1953 (2022)
2. Akavia, A., Galili, B., Shaul, H., Weiss, M., Yakhini, Z.: Privacy preserving feature selection for sparse linear regression. Cryptology ePrint Archive (2023)
3. Becker, B., Kohavi, R.: Adult. UCI Machine Learning Repository (1996). https://doi.org/10.24432/C5XW20
4. Byrd, D., Polychroniadou, A.: Differentially private secure multi-party computation for federated learning in financial applications (2020). https://doi.org/10.48550/ARXIV.2010.05867
5. Ceri, S., Bozzon, A., Brambilla, M., Valle, E.D., Fraternali, P.: Web Information Retrieval. Data-Centric Systems and Applications. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39314-3
6. Cong, K., et al.: Labeled psi from homomorphic encryption with reduced computation and communication. Cryptology ePrint Archive, Paper 2021/1116 (2021). https://eprint.iacr.org/2021/1116

7. Das, S., Ray Chowdhury, S., Chandran, N., Gupta, D., Lokam, S., Sharma, R.: Communication efficient secure and private multi-party deep learning. Proc. Priv. Enhancing Technol. **2025**, 169–183 (2025)
8. Eichner, H., et al.: Confidential federated computations. arXiv preprint arXiv:2404.10764 (2024)
9. Eleks, M., Rebstadt, J., Kortum, H., Thomas, O.: Privacy aware processing (2023)
10. Fowl, L., Geiping, J., Czaja, W., Goldblum, M., Goldstein, T.: Robbing the fed: Directly obtaining private data in federated learning with modified models. arXiv preprint arXiv:2110.13057 (2021)
11. Geiping, J., Bauermeister, H., Dröge, H., Moeller, M.: Inverting gradients - how easy is it to break privacy in federated learning? CoRR abs/2003.14053 (2020). https://arxiv.org/abs/2003.14053
12. Hard, A., et al.: Federated learning for mobile keyboard prediction. CoRR abs/1811.03604 (2018)
13. Hofmann, H.: Statlog (German Credit Data). UCI Machine Learning Repository (1994). https://doi.org/10.24432/C5NC77
14. Hsu, R.H., Huang, T.Y.: Private data preprocessing for privacy-preserving federated learning. In: 2022 IEEE 5th International Conference on Knowledge Innovation and Invention (ICKII), pp. 173–178. IEEE (2022)
15. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
16. Kalra, S., Wen, J., Cresswell, J.C., Volkovs, M., Tizhoosh, H.R.: ProxyFL: decentralized federated learning through proxy model sharing. CoRR abs/2111.11343 (2021). https://arxiv.org/abs/2111.11343
17. Li, X., Dowsley, R., Cock, M.D.: Privacy-preserving feature selection with secure multiparty computation (2021). https://arxiv.org/abs/2102.03517
18. Liu, Z., Chen, S., Ye, J., Fan, J., Li, H., Li, X.: DHSA: efficient doubly homomorphic secure aggregation for cross-silo federated learning (2022)
19. Lyu, L., Chen, C.: A novel attribute reconstruction attack in federated learning (2021). https://doi.org/10.48550/ARXIV.2108.06910
20. Mahdavi, R.A., et al.: Practical over-threshold multi-party private set intersection. In: Proceedings of the 36th Annual Computer Security Applications Conference, pp. 772–783. ACSAC '20, Association for Computing Machinery (2020). https://doi.org/10.1145/3427228.3427267
21. Micci-Barreca, D.: A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. ACM SIGKDD Explor. Newsl **3**(1), 27–32 (2001)
22. Nagy, B., et al.: Kiss: privacy-preserving federated learning and its application to natural language processing. Knowl.-Based Syst. **268**, 110475 (2023). https://www.sciencedirect.com/science/article/pii/S0950705123002253
23. Vero, M., Balunović, M., Dimitrov, D.I., Vechev, M.: TabLeak: tabular data leakage in federated learning (2023)
24. Wang, Z., Ji, H., Zhu, Y., Wang, D., Han, Z.: A survey on federated analytics: taxonomy, enabling techniques, applications and open issues (2025). https://arxiv.org/abs/2404.12666
25. Wightman, F.L.: LSAC national longitudinal bar passage study (2017)
26. Wink, T., Nochta, Z.: An approach for peer-to-peer federated learning. In: Proceedings of 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W) (2021)

27. Wu, H., Zhao, Z., Chen, L.Y., Moorsel, A.V.: Federated learning for tabular data: exploring potential risk to privacy. In: 2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE), pp. 193–204. IEEE Computer Society, Los Alamitos, CA, USA (2022). https://doi.ieeecomputersociety.org/10.1109/ISSRE55969.2022.00028
28. Zhu, J., Blaschko, M.B.: R-GAP: recursive gradient attack on privacy. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=RSU17UoKfJF