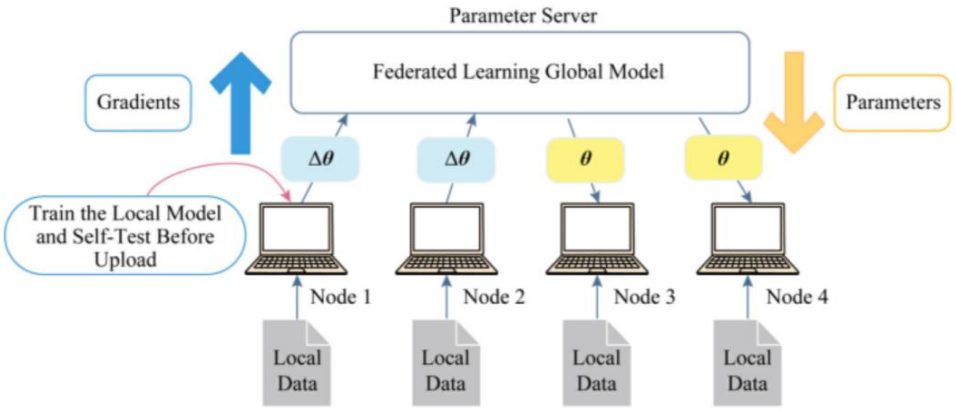
Capstone Project Interim Report

1. Introduction

At the beginning of our research, we focused on the field of credit evaluation of bank users. As we consulted the information, we tried to think about a problem.A person's credit assessment can be derived from many sources, and while it is certainly feasible to utilize a single piece of data to assess a person's credit borrowing and lending situation, it is not comprehensive enough.We certainly hope that the basis used to assess a person's credit rating can come from different sources.However, this raises a key problem that different data sources come from different organizations and need to strictly consider the issue of data privacy among themselves.

So we came up with the federated learning approach.The federated learning approach protects data privacy and solves the problem of heterogeneous data from multiple parties by training the model locally on all parties and sharing only the data updated by the model and not the original data.At the same time, multiple sources of data can allow our system to cover more people, thus improving financial services in general.



Federated learning is mainly categorized into two modes, horizontal federated learning and vertical federated learning, according to the difference in data distribution characteristics.Their core difference lies in the overlap of data features or samples of the participants.Horizontal federation learning is applicable to the scenario where the participants have the same data characteristics but different users.Vertical federation learning is applicable to the scenario where the participants have the same users but different characteristics.

Federated Learning: Horizontal vs. Vertical

Dimension	Horizontal FL (HFL)	Vertical FL (VFL)
Data Overlap	Features overlap, samples differ	Samples overlap, features differ
Communication	Local training, low communication cost	Frequent interactions, high overhead
Privacy	Privacy via parameter aggregation	Requires encrypted alignment & computation
Typical Algorithms	FedAvg, FedProx	SecureBoost, Vertical Logistic Regression

2. Dataset

In **horizontal federated learning**, we leverage the German Credit dataset from the UCI repository, which contains 1,000 loan applicants described by 20 attributes and a binary target variable “kredit.” Approximately 70% of records are labeled as good credit (1) and 30% as bad credit (0). The feature set includes seven strictly numeric variables (e.g., duration in months, amount in Deutsche Marks, age in years, existing_credits, dependents), thirteen categorical or ordinal variables (e.g., checking account status, credit history, loan purpose, savings, employment duration, personal status & sex, housing, property, job), and two binary indicators (telephone ownership, foreign worker status).

In addition to the German Credit dataset mentioned above, **vertical federated learning** tests were also conducted on two other datasets - loan data from LendingClub and MIT credit ranking. For the loan data, the original dataset contains approximately 2.02 million samples, with each sample having 13 attributes. Among them, the label attribute is "loan_status_binary". All attributes are numerical. Due to the excessive sample data, only the first 100,000 samples were intercepted for the subsequent experiments.

For the MIT credit ranking dataset, it consists of 51,336 samples and 86 attribute columns. The "Approved_Flag" attribute can serve as a classification label, with a total of four credit ratings, namely P1 - P4. The "Credit_Score" attribute describes the credit score of the samples and can be used to perform regression tasks. Among the remaining attributes, there are five categorical attributes, and the rest are numerical attributes. Moreover, some attribute columns in this dataset have outliers, which require outlier treatment. The handling method is to discard the attribute columns where the proportion of outliers is greater than 40% (a total of 6), and for other outliers, fill in the average value of the attribute.

All features are standardized using sklearn.preprocessing.StandardScaler, which fits each attribute to zero mean and unit variance. This transformation ensures that every feature lies on the same scale—mean = 0, standard deviation = 1—thereby preventing any single dimension from disproportionately influencing model training and facilitating faster, more stable convergence.

3. Horizontal Federated Learning

To simulate a horizontal federated environment, we first shuffle the fully preprocessed dataset (German Credit dataset) and evenly divide it into three shards of approximately 333 samples each. By preserving the original 70/30 split within each shard, every client receives around 233 good-credit records and 100 bad-credit records. Each shard is then split into a local training set (≈266 samples, 80%) and a local testing set (≈67 samples, 20%) using stratified sampling to maintain class distribution. These three local datasets are distributed to separate clients, which independently train and evaluate their models before any secure aggregation of gradients or parameters.

In horizontal federated learning, we use the Flower framework which is an open-source federated learning framework designed to simplify and accelerate distributed machine learning research and applications, particularly in privacy-sensitive areas like healthcare, finance, and IoT. It supports various machine learning frameworks, including PyTorch, TensorFlow, and scikit-learn, and allows for easy client-server integration. Flower enables secure and scalable federated learning experiments on heterogeneous devices such as smartphones, edge devices, and cloud environments, offering an intuitive and flexible environment for collaborative model training without exchanging raw data. Flower ensures privacy by using secure aggregation protocols, such as FedAvg and FedProx, which allow the server to aggregate model updates from clients without accessing their raw data, thus maintaining data confidentiality throughout the training process.

All experiments utilize a Random Forest classifier configured with 50 trees (n_estimators=50), a max tree depth of 4, a minimum of 30 samples per split, and a minimum of 15 samples per leaf. Additional optimization includes max_features='log2' for feature selection at each split. The class_weight='balanced' parameter adjusts for class imbalance during training, while random_state=42 ensures that results are reproducible across runs. These conservative parameters were chosen to significantly reduce overfitting, achieving an average overfitting gap of only 4.34% compared to the original configuration.

In the standalone setup, each client independently instantiates the shared Random Forest configuration, trains the model on its local training data, and evaluates performance on its local test set. After training, clients extract feature-importance scores from their models to understand which predictors drive decision outcomes in isolation.

In our federated learning implementation, the central server aggregates models using a weighted ensemble strategy through the RandomForestAggregation class. The process involves collecting models from each client, calculating weights based on model accuracy and sample size, and then combining the models using a weighted voting mechanism. This approach ensures privacy by never accessing raw

client data. The ensemble model leverages the contributions of all clients, balancing their impact based on performance and data size, and improves overall prediction accuracy compared to individual client models.

4. Vertical Federated Learning

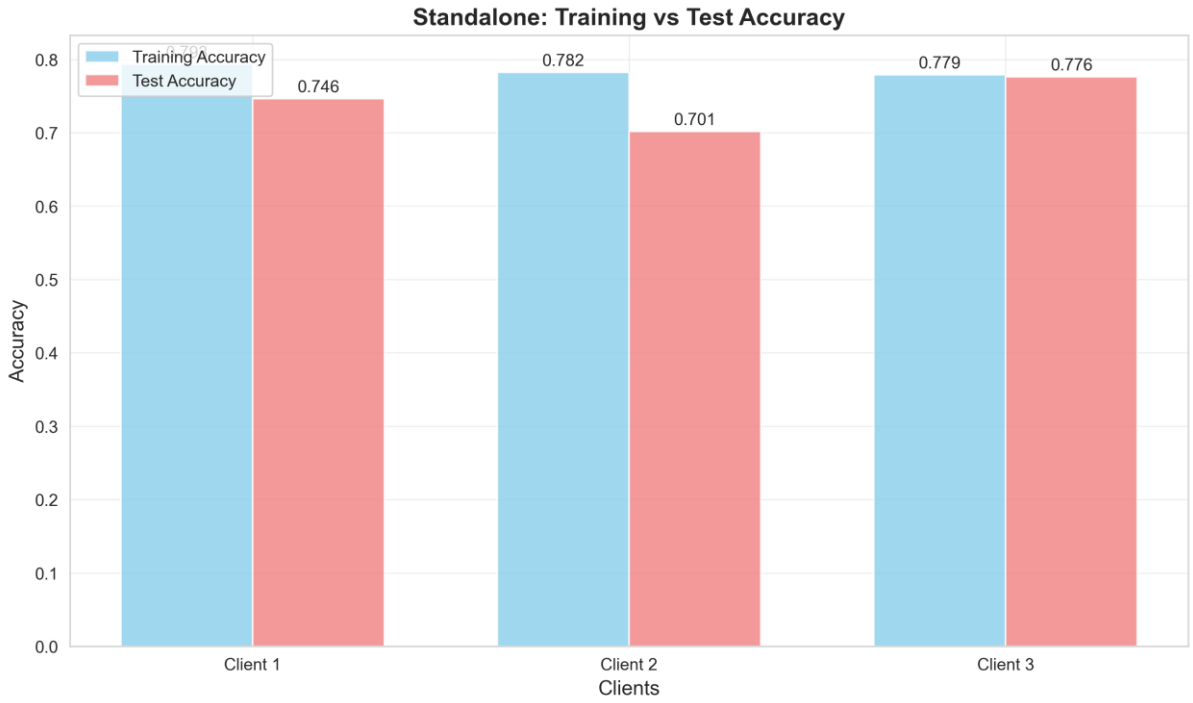
To simulate a vertical federated environment, For the three datasets (German Credit dataset, loan_status_binary, and MIT credit ranking), vertical partitioning was performed to allocate features to three clients in a 1:1:1 ratio. This means each client possesses all samples but only 1/3 of the features for each sample. Subsequently, for each client, the dataset was split into training and testing sets at an 8:2 ratio. The XGBoost model was employed with the following parameters: n_estimators = 50, max_depth = 5, learning_rate = 0.3, max_bin = 10, reg_lambda = 0.1, min_child_weight = 0. The random seed was uniformly set to 42 for reproducibility.

For each dataset, we first conducted single-client training using the training sets held by each client. The XGBoost models trained in this manner were then individually tested on the respective client's test set. This process simulates a scenario where, without the aid of vertical federated learning, each client can only utilize their locally available data with partial features for model training. Subsequently, we employed vertical federated learning to aggregate the training data from all three parties, forming a more comprehensive training set with complete feature attributes. It is important to note that when evaluating the performance of the model trained via vertical federated learning, it is equally necessary to aggregate the test data from the three clients to obtain the model's performance on a complete test set.

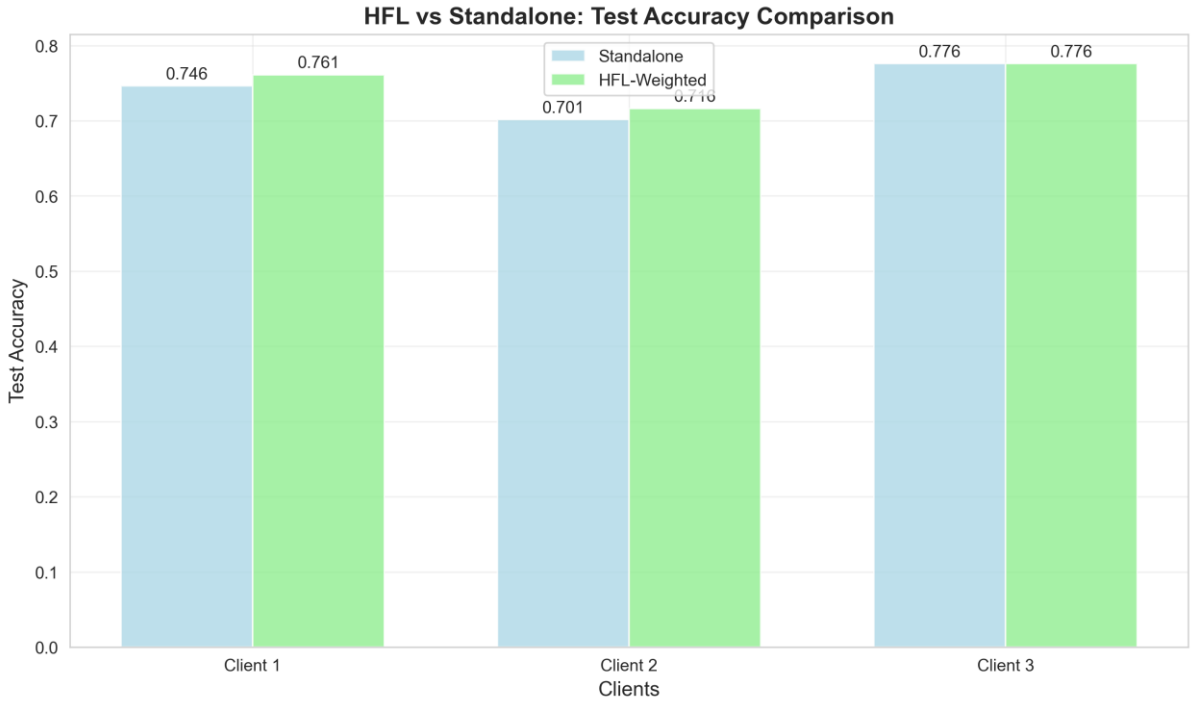
The above experiment uses the SecureBoost algorithm in the SecretFlow framework to perform vertical federated learning. This algorithm prioritizes the protection of label information in vertically partitioned datasets. It uses homomorphic encryption technology to encrypt labels and execute key tree-boosting steps in ciphertext. The result is a distributed boosted tree model composed of PYU objects, where each participant only knows its own split points.

5. Result

The chart below compares the training and test accuracy of the standalone model across three clients, showing that the training accuracy (blue bars) is slightly higher than the test accuracy (red bars) for each client, indicating that there is no overfitting.



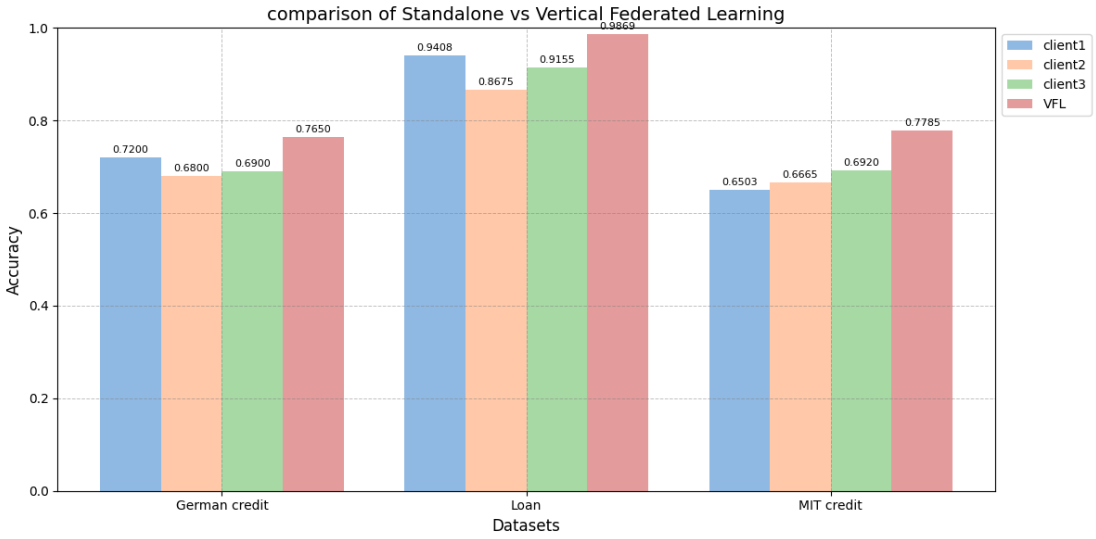
The table below presents a comparison of accuracy on test dataset between Independent Learning and Federated Learning for each client in Horizontal Federated Learning, along with the improvement achieved by Federated Learning:

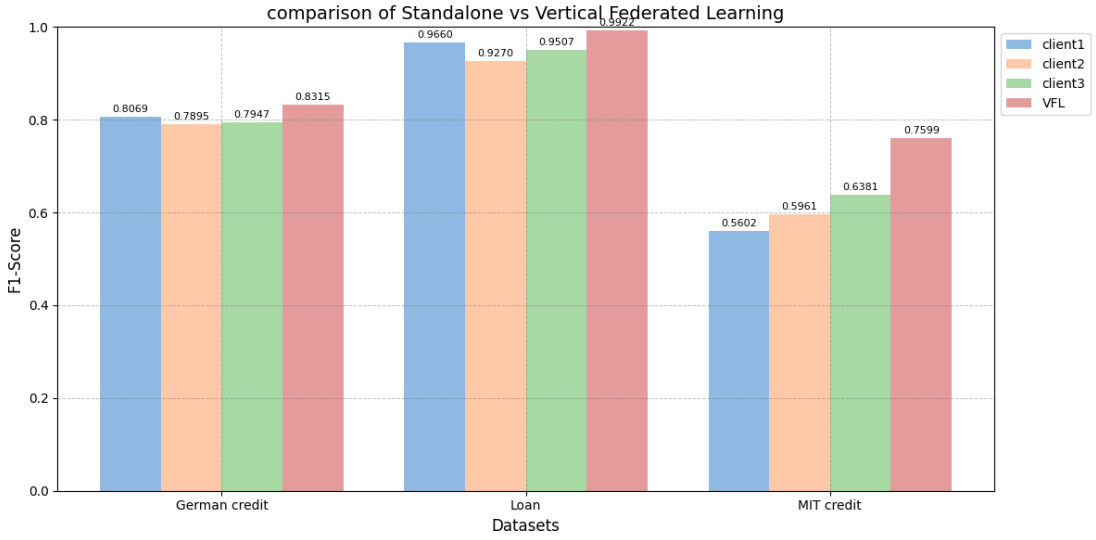


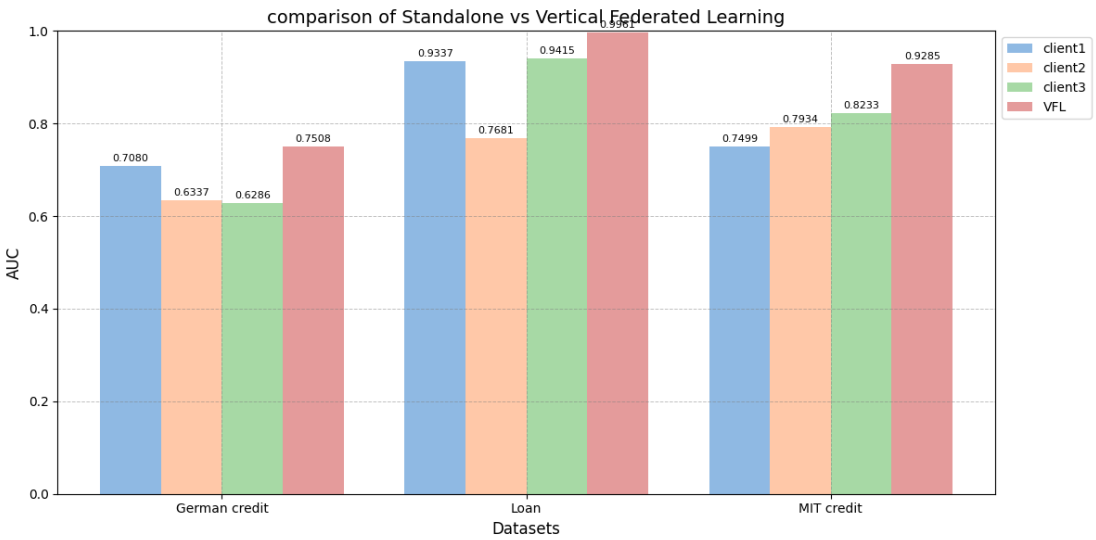
The table below presents a comparison of accuracy between Independent Learning and Vertical Federated Learning for each client in Vertical Federated Learning, along with the improvement achieved by Federated Learning:

Dataset: German credit

	Client1	Client2	Client3	VFL	Ave_Improvement
Accuracy	0.7200	0.6800	0.6900	**0.7650**	+0.0683
F1-Score	0.8069	0.7895	0.7947	**0.8315**	+0.0345
AUC	0.7080	0.6337	0.6286	**0.7508**	+0.0940
Dataset: Loan data from LendingClub					
	Client1	Client2	Client3	VFL	Ave_Improvement
Accuracy	0.9408	0.8675	0.9155	**0.9869**	+0.0790
F1-Score	0.9660	0.9270	0.9507	**0.9922**	+0.0443
AUC	0.9337	0.7681	0.9415	**0.9961**	+0.1150
Dataset: MIT credit ranking					
	Client1	Client2	Client3	VFL	Ave_Improvement
Accuracy	0.6503	0.6665	0.6920	**0.7785**	+0.1089
F1-Score	0.5602	0.5961	0.6381	**0.7599**	+0.1618
AUC	0.7499	0.7934	0.8233	**0.9285**	+0.1396


comparison of Standalone vs Vertical Federated Learning
Accuracy
Datasets
German credit
Loan
MIT credit
client1
client2
client3
VFL
0.7200
0.6800
0.6900
0.7650
0.9408
0.8675
0.9155
0.9869
0.6503
0.6665
0.6920
0.7785


comparison of Standalone vs Vertical Federated Learning
F1-Score
Datasets
German credit
Loan
MIT credit
client1
client2
client3
VFL
0.8069
0.7895
0.7947
0.8315
0.9660
0.9270
0.9507
0.9922
0.5602
0.5961
0.6381
0.7599


comparison of Standalone vs Vertical Federated Learning
AUC
Datasets
German credit
Loan
MIT credit
client1
client2
client3
VFL
0.7080
0.6337
0.6286
0.7508
0.9337
0.7681
0.9415
0.9961
0.7499
0.7934
0.8233
0.9285

6. Discussion

The experiments struck a balance between privacy protection and model performance, achieving data privacy through aggregation voting (horizontal federated learning) and SecureBoost (vertical federated learning), and experimentally validating the feasibility of both solutions. In the horizontal federated learning part, clients do not share the original data during local training, instead transferring only the model updates to ensure data confidentiality. In the vertical federated learning part, label information is protected through homomorphic encryption (e.g., SecureBoost), ensuring that participants only have access to the split point of their own features while maintaining privacy.

However, privacy protection may introduce performance loss: In German credit dataset, the independent client model of vertical federation is unstable due to data heterogeneity (such as Client 2's AUC is only 0.6337), while the AUC after federation aggregation is improved to 0.7508, proving that aggregation effectively improves robustness; In MIT dataset, the performance of a single client is limited due to incomplete features (such as the accuracy of Client 1 in the MIT dataset is 0.6503), but the accuracy jumps to 0.7785 (+12.89%) after feature combination, which emphasizes the importance of feature completeness.

In our current experiments, overfitting control is successful: Horizontal federation uses conservative random forest parameters (max_depth=4, min_samples_leaf=15), and the training-test accuracy difference is only 4.34%, which is significantly lower than the baseline.

However, potential problems still exist: the XGBoost model of vertical federation has slight overfitting on some clients, such as LendingClub's Client 1 training accuracy of 0.9408 vs. federation test accuracy of 0.9869. We may further introduce regularization or early stopping strategy to address this problem.

For the aspect of data heterogeneity, horizontal federated learning assumes that client data distribution is similar, but the data differences among institutions in actual financial scenarios are significant, such as the German credit data set is evenly divided, while the real scenario may be skewed. The feature allocation (1:1:1) of the vertical federated part also does not take into account the difference in feature importance, which may cause key features to be concentrated on a single client.

7. Future Work

In future experiments, we will put some effort into introducing differential privacy (DP): Inject Gaussian noise into the model update of FedAvg to defend against potential attacks, especially for highly sensitive financial data. We will also explore some other aggregation algorithms: FedProx to solve client drift or FedNova to adapt to non-IID data to improve the convergence efficiency under heterogeneous financial data.

We may also design experiments on hybrid federated learning: combine horizontal and vertical federated learning to address more complex scenarios. Hybrid model will hierarchically combine horizontal grouping (by client) and vertical feature partitioning. Validation will be tested on multi-source financial data to quantify gains in metrics like accuracy and F1-score over single-mode federated learning.

Finally, we will use larger and more reliable financial datasets to provide a more robust and accurate evaluation of federated learning models. Such datasets will help evaluate the scalability, feasibility, and effectiveness of privacy-preserving methods under real-world conditions, thereby ensuring the versatility and applicability of the models in the financial area.