Fed-Tree Lab

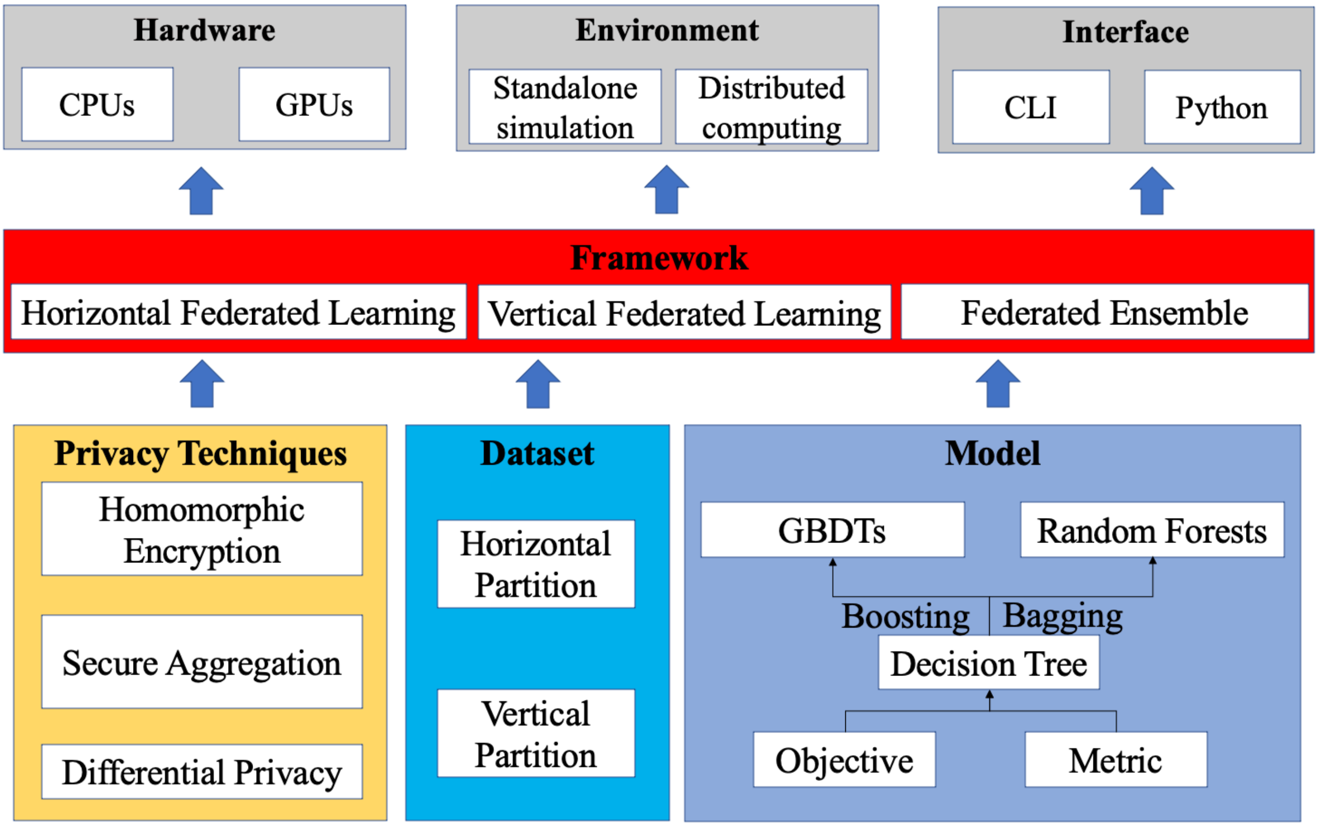
52275903003 胡梓锐 华东师范大学

1. 简介

FedTree is a federated learning system for tree-based models. It is designed to be highly efficient, effective, and secure. It has the following features currently.

- Federated training of gradient boosting decision trees.
- Parallel computing on multi-core CPUs and GPUs.
- Supporting homomorphic encryption, secure aggregation and differential privacy.
- Supporting classification and regression.

  The overall architecture of FedTree is shown below.



2. 数据集介绍

数据集使用的是Kaggle上的Credit Card Fraud Detection数据集，下载地址如下：https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

该数据集包含 2013 年 9 月欧洲持卡人使用信用卡进行的交易。 其中 284,807 笔交易中有 492 笔欺诈。数据集高度不平衡，正类(欺诈)占所有交易的 0.172%。

3. 实验准备

3.1 环境配置

本项目运行Fed-Tree的单机模式，在Ubuntu环境下进行实验，需要准备Cmake（已准备好），GMP和NTL

```
# 安装GMP
sudo apt-get install libgmp-dev
# ------------------------------------
# 安装NTL
wget https://libntl.org/ntl-11.5.1.tar.gz
tar -xvf ntl-11.5.1.tar.gz
cd ntl-11.5.1/src
./configure SHARED=on
make
make check
sudo make install
```

3.2 部署FedTree

```
git clone git@github.com:Tamiflu233/FedTree.git
cd FedTree
git submodule init
git submodule update
# under the directory of FedTree
mkdir build && cd build
cmake .. -DDISTRIBUTED=OFF
make -j
git clone https://github.com/FedTree/ECNU-FedTree-Challenge
cd ECNU-FedTree-Challenge
```

3.3 模拟联邦学习配置

We need to download the fraud detection dataset [here](https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud) (https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud) and put `creditcard.csv` under `data` directory. Then, we can run `python train_test_split.py` to split the dataset into training dataset and test dataset.

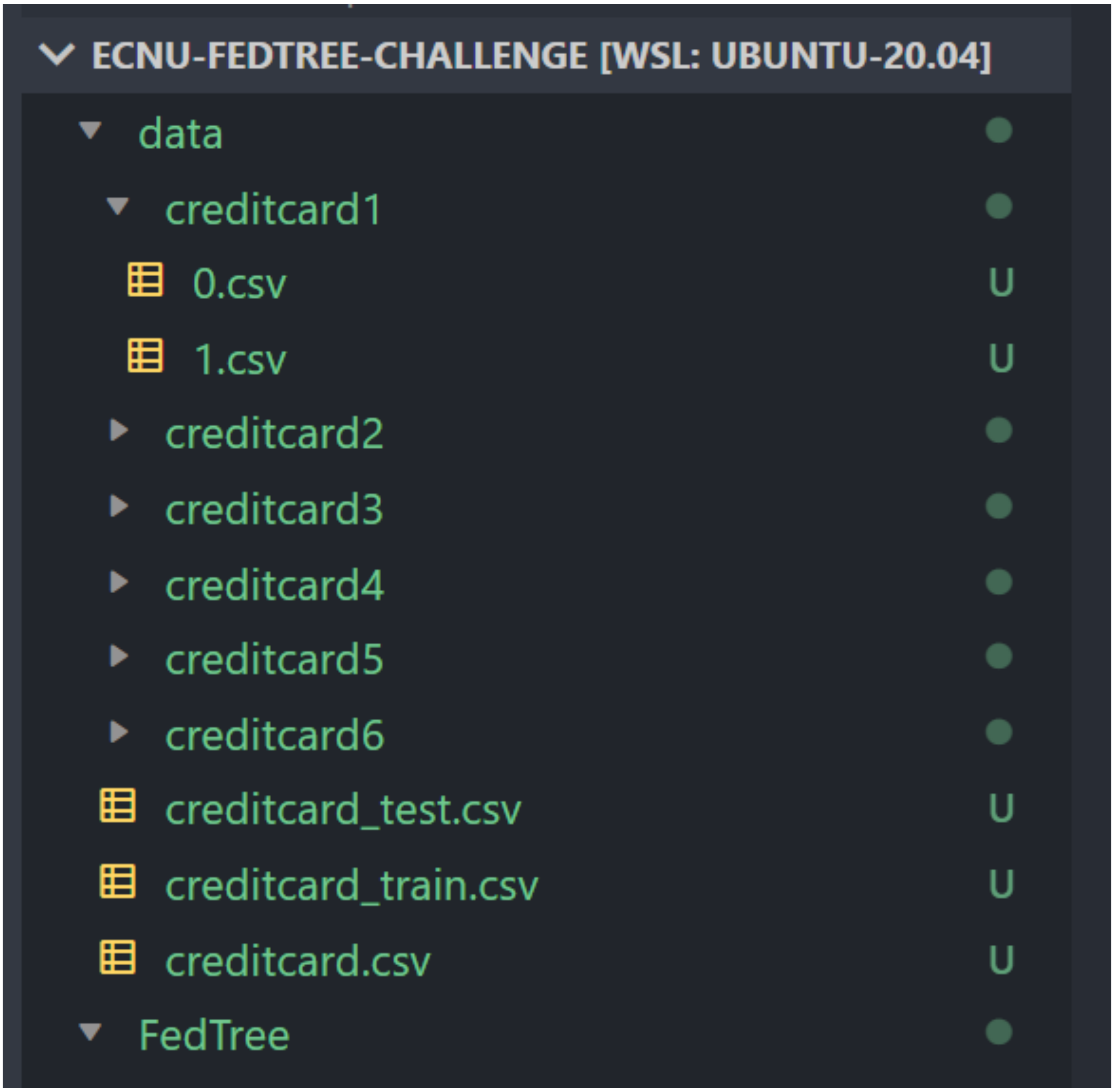
Then, we can create the partitions of the dataset to simulate the federated setting with the help of `partitions/partition.py`, which has the following parameters

Parameter	Description
`n_parties`	Number of parties, default = `2`.
`partition`	The partition way. Options: `homo`, `noniid-labeldir`, `noniid-#label1` (or 2, 3, ..., which means the fixed number of labels each party owns), `iid-diff-quantity`. Default = `homo`
`init_seed`	The initial seed, default = `0`.
`datadir`	The path of the dataset, default = `./data/creditcard_train.csv`.
`outputdir`	The path of the output directory, default = `./data/partitioned_creditcard/`.
`beta`	The concentration parameter of the Dirichlet distribution for heterogeneous partition, default = `0.5`.

我们需要尝试以下六种配置：

- n_parties = 2, partition = `homo`
- n_parties = 2, partition = `noniid-labeldir`, beta = 0.5
- n_parties = 10, partition = `noniid-labeldir`, beta = 0.5
- n_parties = 10, partition = `noniid-labeldir`, beta = 0.1
- n_parties = 10, partition = `noniid-#label1`
- n_parties = 10, partition = `iid-diff-quantity`, beta = 0.5

数据集切分结果如下：



3.4 运行模型

使用如下配置进行模型的训练与配置：

```
data=../data/creditcard1/0.csv,../data/creditcard1/1.csv # tra
test_data=../data/creditcard_test.csv # test data location
n_parties=2 # number of party
num_class=2 # number of class
mode=horizontal # federal mode
objective=binary:logistic # objective function
data_format=csv
privacy_tech=none
model_path=fedtree.model
max_num_bin=16
learning_rate=0.1
max_depth=6
n_trees=10
```

执行如下命令开始训练和测试：

```
./build/bin/FedTree-train example.conf
```

初始测试结果如下表所示：

分区方式	具体参数	AUC
parition1.sh	n_parties = 2, partition = homo	0.853675
parition2.sh	n_parties = 2, partition分区 = noniid-labeldir, beta = 0.5	0.934496
parition3.sh	n_parties = 10, partition = noniid- labeldir, beta = 0.5	0.929194
parition4.sh	n_parties = 10, partition = noniid- labeldir, beta = 0.1	0.95182
parition5.sh	n_parties = 10，partition = noniid-#label1	0.143383
parition6.sh	n_parties = 10, partition = iid-diff- quantity, beta = 0.5	0.825801

平均AUC = 0.7718525

以partion1.sh运行结果为例，其运行结果如下图所示：

```
2022-12-29 14:37:07,103 INFO FLtrainer.cpp:466 : averaged AUC = 0.840699
2022-12-29 14:37:07,103 INFO FLtrainer.cpp:468 : Training round 8 end
2022-12-29 14:37:07,103 INFO FLtrainer.cpp:202 : Training round 9 start
2022-12-29 14:37:07,367 INFO FLtrainer.cpp:466 : averaged AUC = 0.843439
2022-12-29 14:37:07,367 INFO FLtrainer.cpp:468 : Training round 9 end
2022-12-29 14:37:07,367 INFO FLtrainer.cpp:472 : training time = 2.51266s
2022-12-29 14:37:07,367 INFO FLtrainer.cpp:481 : end of training
2022-12-29 14:37:07,367 INFO fedtree_train.cpp:223 : end horizontal training
2022-12-29 14:37:07,396 INFO gbdt.cpp:104 : AUC = 0.853675
```

4. 调参

首先对 GBDT 的深度，学习率，决策树个数，max_num_bin等参数进行了多次调优，最终实践出的最优解如下所示：其中树深对整体决策调优具有较为明显的影响

分区方式	Maximum depth of tree	Number of bins	Learning rate	number of trees	AUC
parition1.sh	24	10	0.1	10	0.978388
parition2.sh	20	10	0.05	10	0.97791
parition3.sh	6	20	0.1	11	0.956748
parition4.sh	10	16	0.1	11	0.951723
parition5.sh	9	20	0.1	8	0.811788
parition6.sh	6	18	0.01	10	0.861585

平均AUC =0.9230236，增长了119%，通过贪心搜索调优，最终得到了较好的结果，也是对联邦学习的一次有益尝试。