水平联邦：样本的联合

参与方业务相同，数据不同

# 水平联邦框架介绍

主要解决水平联邦建模问题。

大方向有两个：NN建模、树模型建模（XGBoost）

SF水平联邦特性：

1. 用户友好

接口和明文几乎一致，已有的代码不用过多改变。

在API层面屏蔽不同backend的差异

1. 功能完善、

建模能力和明文差不多

支持多种backend、多种安全等级聚合方法

支持一定的扩展性

1. 性能出色

计算交给了优秀引擎

通信优化（稀疏化、压缩）

流水线优化（并行度提高）

# FLModel实践讲解

## API

Python api：

class FLModel:

…

统一使用这个接口

参数解释：

server：有一些逻辑会在server做。不是聚合器的server

device\_list: 参与方

model：模型定义

aggregator：可以用sf定义好的

backend：后端引擎

FLModel的dispatcher会根据给定参数拉起合适的计算worker

用户需要给定合适参数

二次开发的策略需要注册

## 模型定义

把普通的tensorflow.keras模型挪进来就行了

区别就是需要wrap，为了序列化能够穿透到每个worker，给定一个函数比给定对象更方便

明文model可以直接在flmodel使用

## Aggregator

导入：(现在还有MPCAggregator, TEEAggregator, HEAggregator)

from secretflow.security.aggregation import

(PlainAggregator, SecureAggregator, SPUAggregator)

使用只需要简单的实例化

## Strategy

联邦训练的strategy（联邦学习和分布式机器学习的主要区别）主要有以下两类：

builtin strategy (fedavg, fedprox, fedSTC, fedSCR, SCAFFOLD, FEDDYN)

custom strategy

builtin的strategy支持pytorch和tensorflow

自定义strategy需要定义这4个：

local train strategy

server aggregate strategy

compressor strategy

register strategy(@register\_stragety(strategy\_name=”…”, backend=”…”))

## 水平FLModel实战

参考相应ipynb

# SFXgboost实践

sf的xgboost和明文xgboost基本对齐。

明文xgboost：

import xgboost as xgb

# read data

dtrain=xgb.DMatrix(“demo/data/agaricus.txt.train”)

dtest=xgb.DMatrix(“demo/data/agaricus.txt.test”)

# specify map

param={‘max\_depth’:2, ‘eta’:1, ‘objective’:’binary:logistic’}

num\_round=2

bst=xgb.train(param, dtrain, num\_round)

sf的xgboost：

bst=SFXgboost(server=self.davy, clients={self.alice, slef.bob})

params={

‘max\_depth’:4,

‘eta’:1.0,

‘objective’:’binary:logistic’,

‘label\_key’:’label’ # 标记hdataframe中的label列名

}

bst.train(self.hdf, self.hdf, params=params, num\_boost\_round=4)

注意事项：

目前水平XGBoost只支持HDataFrame类型的数据

模型超参数：和明文几乎一致

FineTune: bst.train()里面传入模型路径就可以了

Predict没有实现

剩下内容（详细代码）见ipynb