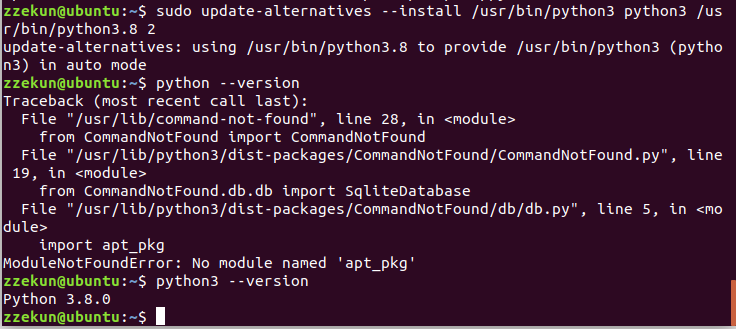
Secretflow与Rosetta

这个Secretflow是基于它所定义的AI framework和device flow 之上的，这一点有点像tensorflow就是新的框架语法有点多，可能源代码迁移的耗费的精力比较大。

但是对于环境配置这一块儿，如果是使用虚拟机的环境，对于这个本来的Secretflow的环境很离谱，18.04虚拟机镜像自带的python版本是3.6.9，需要python 3.8，得自己下python压缩包，重新编译然后重新设置几个python版版本的优先级。（虽然Rosetta也有点儿难配，Rosetta难配的点在于python3.6.9的版本与tensorflow 1.14.0版本的兼容）（可以参考

[https://blog.csdn.net/wojuzi/article/details/107847835#:~:text=Ubuntu18.04x64](https://blog.csdn.net/wojuzi/article/details/107847835#:~:text=Ubuntu18.04x64%E6%9B%B4%E6%96%B0Python3.7%EF%BC%8C%E5%B9%B6%E6%94%B9%E5%8F%98python3%E6%8C%87%E5%90%91%EF%BC%8C%E5%92%8C%E9%81%87%E5%88%B0%E7%9A%84%E5%9D%91%201%20%E6%B3%95%E4%B8%80%E3%80%81%EF%BC%88%E6%B2%A1%E8%AF%95%E8%BF%87%EF%BC%89%E5%BB%BA%E7%AB%8B%E8%BD%AF%E8%BF%9E%E6%8E%A5%E3%80%80%E6%9B%B4%E6%96%B0Python%203%E4%BB%A5%E6%8C%87%E5%90%91Python,3.7%202%20%E6%B3%95%E4%BA%8C%E3%80%81%EF%BC%88%E6%8E%A8%E8%8D%90%EF%BC%89%E5%B0%86Python%203.6%E5%92%8CPython%203.7%E6%B7%BB%E5%8A%A0%E5%88%B0%E6%9B%B4%E6%96%B0%E6%9B%BF%E4%BB%A3%E9%A1%B9)

<https://www.codenong.com/2a623dae4fb19107b44e/>



或者是在虚拟机之中下载一个conda然后虚拟环境设置）

<https://linux.cn/article-14871-1.html（在Ubuntu22.04>上安装docker，这个Linux中文论坛的博客挺好）（应该本机上安装运行docker也是可以的）

Facebook CrypTen安全多方计算（MPC）框架，其底层依赖于深度学习框架PyTorch。

华控清交基于多方安全计算技术，实现了高性能通用的安全计算框架 PrivPy 平台，没开源是个平台。（这是啥？？）

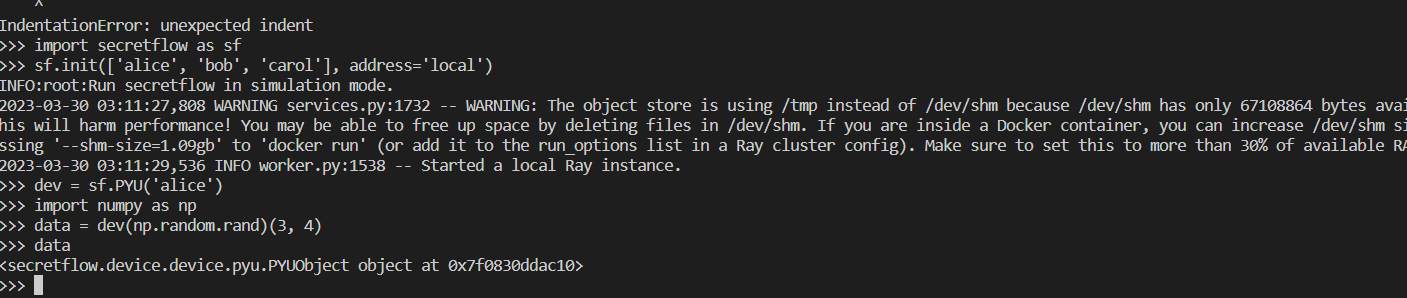
PySyft就是一个python库，支持tenserflow和pytorch，github上8700多star，google的反而不太行一直不更新了，FaceBook那个是基于pytorch的（好像可以考虑），隐语表现不错（完成基本部署），TF-Encrypted看着好像也还行

目前隐语还没有测试NVIDIA的GPU支持模块，这个本机上是配置好的炼丹环境，但是不知道虚拟机能不能也支持。

<https://blog.csdn.net/Taylor_0522/article/details/117655579>（这一套流程就可以在本机上的vscode之中连接上docker的环境了，docker就有点类似于独立的虚拟环境，配置一下config里面的socket的虚拟机IP和端口）

在Secretflow的框架之中，能使用secretinit直接模拟多个用户，Rosetta的话可以通过那种多个命令行执行的模式。

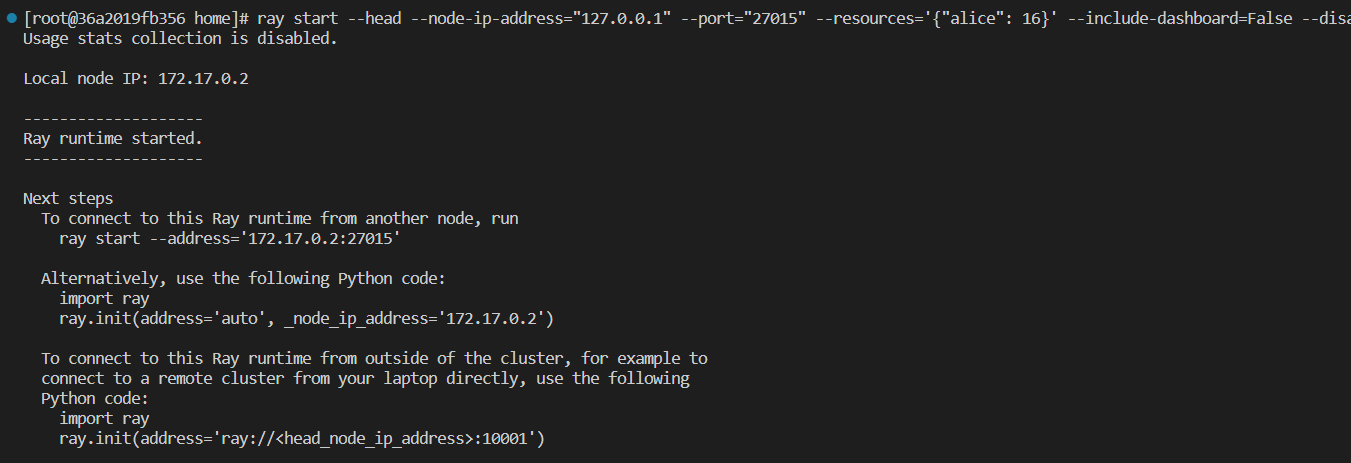
复现最简单的tutorial成功，但是遇到了一个warning和/dev/shm以及/tmp有关，需要扩增/dev/shm的size



使用修改后的命令即可：

sudo docker run -it --shm-size=2.0gb secretflow/secretflow-anolis8

sf.init(['alice', 'bob', 'carol'], address='local')可以使用init方法直接在本机上进行模拟。Secretflow框架还给出了一个单节点的cluster的模拟方式。



上图为单节点的ray cluster模拟方式。应该是目前只能够完成对于单头节点以及单层的集群架构方式（目前应该是不能做成DNS层次化结构），因为子节点应该是没有办法使用head的命令行选项的。

provide `cluster\_config` when running with production mode，在运行在production mode时需要对集群的config进行设置，但是在模拟的时候就不需要了。Secretflow专门设置了为了部署而使用的production mode。

Docker可以进行部署仿真，host网络和bridge桥接网络部署。可以在一台机器上使用多个docker容器部署仿真集群。

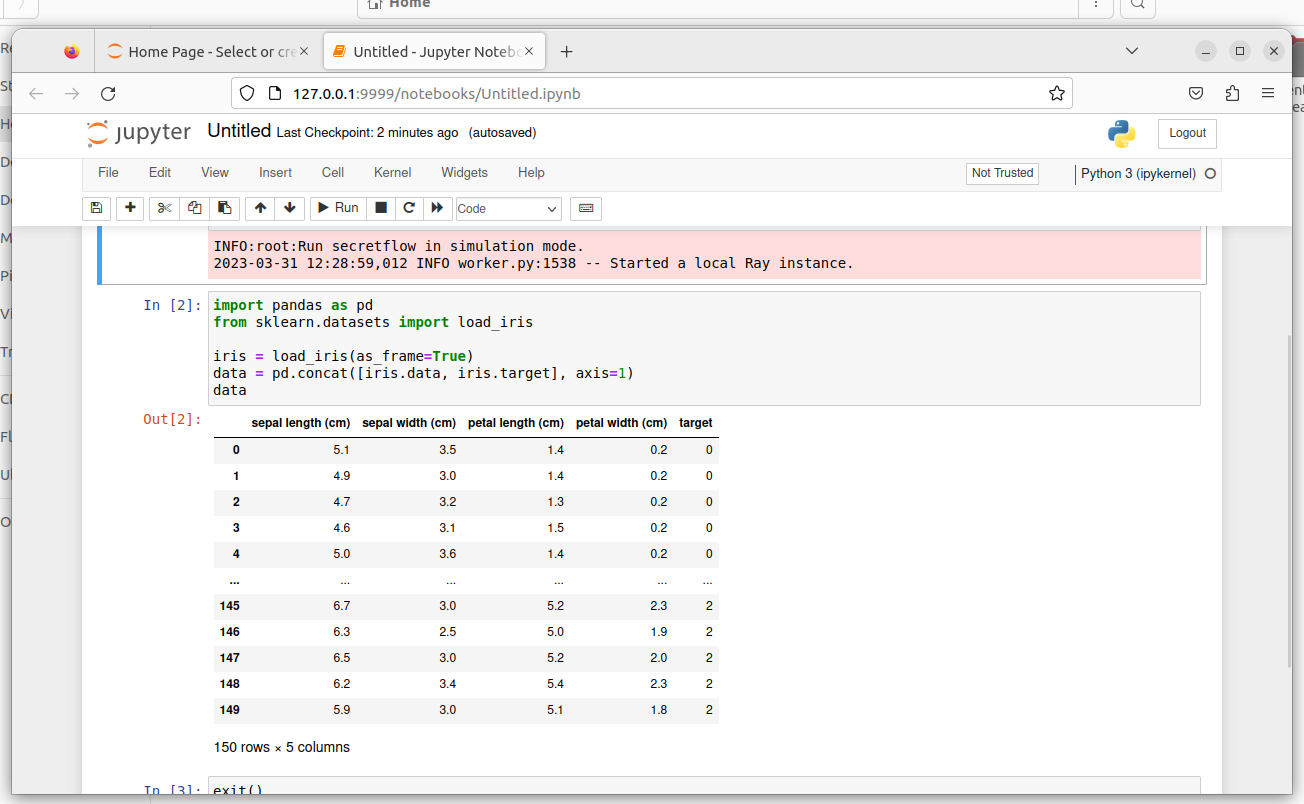
bRPC基础介绍：RPC是远程过程调用（Remote Procedure Call）。RPC 的主要功能目标是让构建分布式计算（应用）更容易，在提供强大的远程调用能力时不损失本地调用的语义简洁性。为实现该目标，RPC 框架需提供一种透明调用机制，让使用者不必显式的区分本地调用和远程调用。

框架组件：

* 预处理（提供了多种工具方法和类，可以将原始特征转换为更适合后续流程（比如训练）的数据形式）
* PSI（SecretFlow SPU目前支持ECDH-PSI, KKRT16-PSI和BC22-PCG-PSI）
* MPC机器学习（通过多方安全计算技术和同态加密技术，SecretFlow提供了MPC建模能力。）（线性回归模型，决策树，特征工程皮尔逊相关系数统计学方面的内容）
* 联邦学习（在联邦学习中，多个拥有本地数据的设备可以在没有交互原始数据的情况下训练机器学习模型。除了样本ID和特征空间都不相同的情况无法满足），也能训练神经网络啥的

数据预处理模块：目前DataFrame兼容了部分pandas接口，使用体感和pandas一致。在计算过程中，原始数据始终保持在数据拥有方本地，不会出域。为了保护数据隐私，DataFrame不允许对原始数据进行查看。

数据预处理部分之中，构建了两种partition方式，一种是水平的partition，另一种是垂直的partition，类比于pandas，可以使用类似于hdf或者vdf的语法就可以实现。

https://blog.csdn.net/weixin\_40008349/article/details/81135847

可以完成的操作是在Ubuntu虚拟机之上能够运行jupyter notebook使用sf框架的数据预处理部分，这一部分接入了一部分pandas的接口，使用起来还行。

看了眼隐私集合求交集的部分，支持两方协议与三方协议的实现，实现了带差分隐私的隐私求交协议，协议内容具体的真没看懂..

SecretFlow使用MPC协议和同态加密实现了多个可证安全的建模算法。（线性回归模型，决策树模型和特征工程）

支持的MPC协议有：

ABY3:一个多数诚实的3PC协议。SPU实现了半诚实模型。

Semi2k-SPDZ :一个半诚实NPC协议。和SPDZ协议类似，但是需要一个可信第三方来产生离线的随机数。目前，在协议中默认使用可信第一方。因此，目前主要用于调试。

Cheetah :一个快速半诚实的2pc模型。因为协议不需要可信第三方，因此会需要一些计算开销。

线性回归：SS-SGD与HESS-SGD，一个是基于秘密共享协议的，另一个是基于使用同态加密计算下降梯度。SSRegression() 是secretflow根据regression自己集成的API函数，使用秘密共享协议实现了针对垂直划分数据集的线性回归和二分类回归训练，求解方式为批量随机梯度下降。该API函数能够完成线性回归与逻辑回归。

一些样例代码可见：

<https://www.secretflow.org.cn/docs/secretflow/zh_CN/components/mpc_ml/linear_model.html>

其中包含了不同模型训练的样例代码，模型参数能够被保存为密态的。

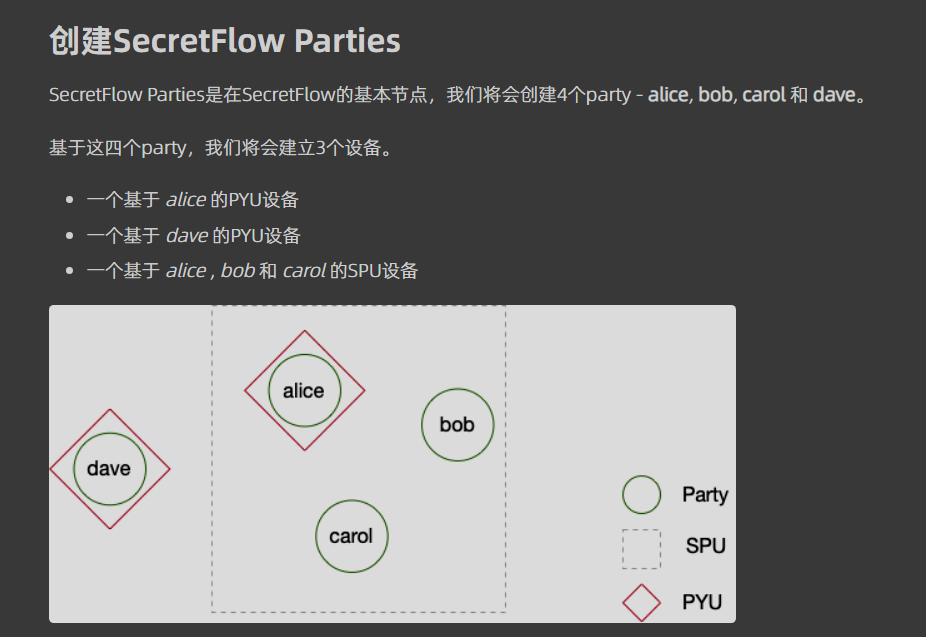
SecretFlow使用多方安全计算的秘密分享技术实现了可证安全的梯度下降决策树模型 Xgb() ，目前支持线性回归问题训练和二分类问题训练。

SecretFlow 的 SSPValue 模块可以用于探查模型的P-Value，计算过程使用秘密分享协议保护。（特征工程部分的内容）

在SPU部分的官方文档讲解之中有百万富翁问题的实例，这一部分讲解了明文与密文的各种运算的基本过程。

Secretflow可以部分API使用的backend是pytorch，可以使用一些包装器来包装一下原来的API函数，或者直接继承替换原来的pytorch的API函数。

补充一些具体的算法流程：PYU单元可以本地执行明文计算，PYU的object可以与SPU单元object进行转换。SPU设备在SecretFlow中负责执行MPC计算，个人感觉SPU设备是比较核心的设备单元，



我们选择在本机上进行测试，可以创建PYU和SPU设备，PYU主要是用于本地明文计算用的。

sf.init(['alice', 'bob', 'carol', 'dave'], address='local')（本机模拟）

下一步：

aby3\_config = sf.utils.testing.cluster\_def(parties=['alice', 'bob', 'carol'])（基于ABY3协议的SPU单元）

下一步：

可以将一个Python Object从本机传到SPU：

bank\_account = [{'id': 12345, 'deposit': 1000.25}, {'id': 12345, 'deposit': 100000.25}]

bank\_account\_spu = sf.to(alice, bank\_account).to(spu\_device)

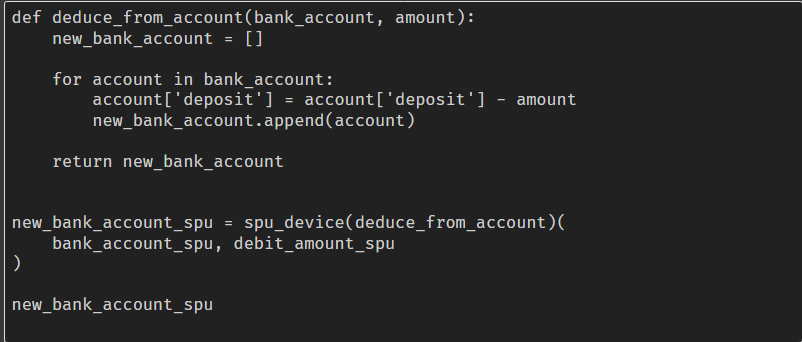
直接运用to(spu\_device)这个函数即可，如果转化为了SPU设备对象之后在通信的过程之中就是密文传输了：

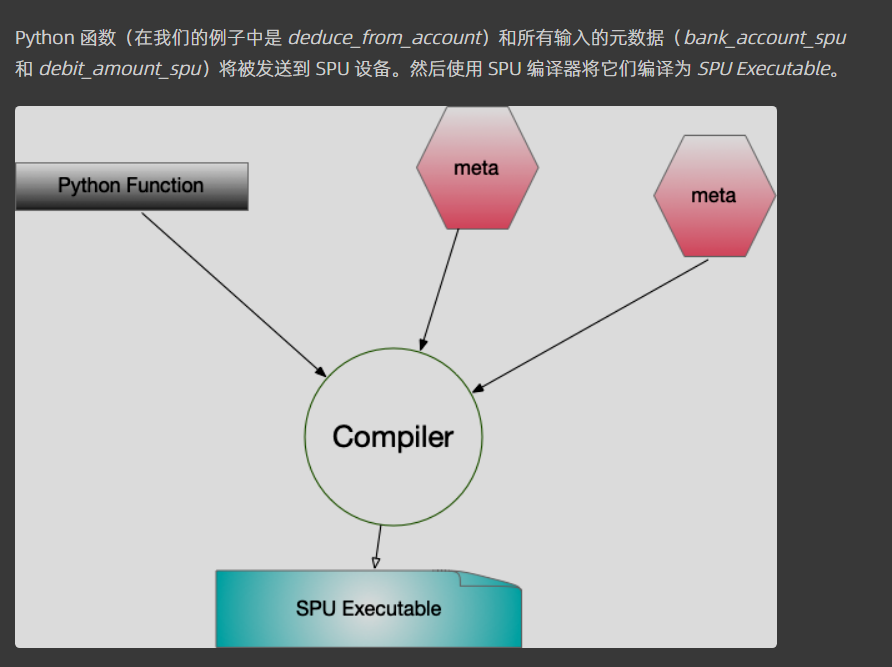


可以看到meta保留了数据的结构，但是数值被进行了替换，ABY3协议会为三方分出三个分片，每个分片为密文表示。sf.reveal 检查原始值，只能在SPU Object位于本机（其实也就相当于是本地数据）的时候检查 meta 和 shares。否则，只能使用 sf.reveal获取plaintext。

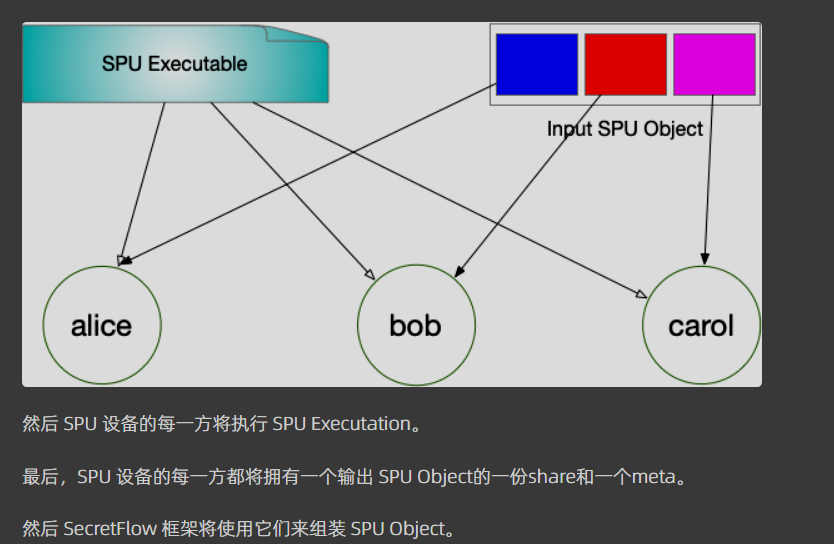
可以使用SPU设备进行一部分的运算：

例如银行账户减去向外贷款的份额：





然后就是要执行的函数与数据内容都会被发送到应该是每一个SPU设备，然后每一个设备对其进行编译获得SPU Executable，然后也获得一份儿SPU Object，然后本机运算获得一份结果。



一些框架内容：PYU（明文单元）,silo（企业之间隔绝的结构？？）,TLS密钥共享交互证书（一种协议），Ray集群（由一个主节点和零或若干个从节点组成），

SPU（Secure Processing Unit）, SPU设备在SecretFlow中负责执行MPC计算，



可以用于解决姚氏百万富翁问题，这里应该是默认转换为SPU Object了然后其实也就是对于密文来进行计算的。类似于如下代码：

new\_bank\_account\_spu = spu\_device(deduce\_from\_account)(

bank\_account\_spu, debit\_amount\_spu

)

百万富翁问题运行事例之前要使用iptables先把所有的端口打开，或者设置自己想要打开的端口。具体可以参考：

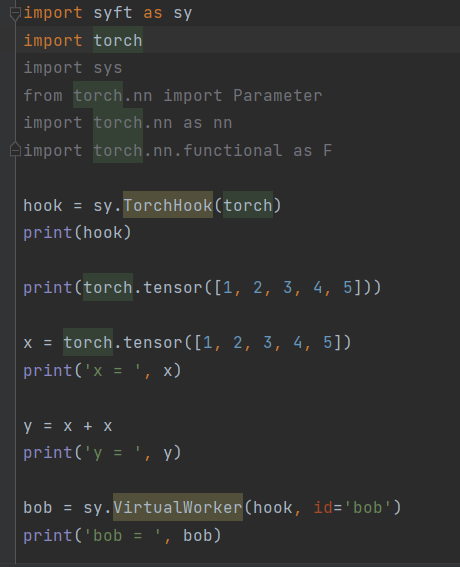
https://blog.csdn.net/wangzhaoyoung/article/details/108678355#:~:text=iptables%20-P%20INPUT%20ACCEPT%20iptables%20-P%20FORWARD%20ACCEPT,-I%20INPUT%20-p%20tcp%20--dport%208080%20-j%20ACCEPT%EF%BC%89%EF%BC%8C%E4%BD%86%E6%98%AF%E5%A6%82%E6%9E%9C%E6%9C%8D%E5%8A%A1%E5%99%A8%E9%87%8D%E5%90%AF%EF%BC%8C%E4%B8%8D%E4%BC%9A%E4%BF%9D%E5%AD%98%EF%BC%8C%E5%A6%82%E6%9E%9C%E6%83%B3%E6%8C%81%E7%BB%AD%E5%8C%96%E8%A7%84%E5%88%99%EF%BC%9A%E4%BD%BF%E7%94%A8

对于这个问题我们进一步加深对SPU单元的理解，其实这里只是拿了两个主机模拟的两方SPU单元，但是输入的数据不一定非要是SPU单元内的主机进行输入，外部PYU主机输入也是可以的，另外就是不同的SPU单元不同的ray cluster也是可以进行通信的。（确实就是运行起来通信+运算有点慢）

PySyft

感觉PySyft对联邦学习的支持力度更大。Github官网上给出的示例是建立一个服务器，然后数据拥有方向服务器发送数据，数据分析方来进行分析。（但是在官方的课程讲解之中包含了MPC，DP等各个方面）。

Pysyft也是使用的是pytorch，PySyft只是提供了联邦学习和安全计算相关的一些技术，但是要实际结合真实场景使用，还 要用到PyGrid形成点对点的访问。而HaGrid就是可以用一行命令快速部署PyGrid。



老版本的这个看起来还行，只需要对torch进行一步hook然后通过设置virtual worker就可以来进行模拟了，新版本目前的一切文档都没有提供完整，目前是无法尝试的，Openminded还有一个python第三方库是SyMPC这个库他们正在和PySyft联合集成，如果要使用PySyft的话，推荐使用老版本的，新版本的遇到问题忒多了，还没啥博客讲，应该是还没开发完）

<https://www.cnblogs.com/mlblog27/p/14258662.html（这篇博客部署的是PySyft0.2.4>版本的）

新版本的PySyft就不在继续使用hook函数了，使用的是sy.tensor，就是sy集成好的tensor，但是官网上显示0.2-0.5的已经荒废了，目前新版本还处于开发阶段，很多0.7.0的doc还没有，还是纯英文的理论视频教程……