serverbase.py

```python
def select_users(self, round, num_users):

    return np.random.choice(self.users, num_users) #Use numpy module random choice function to choice num_users of users

    ## TODO
    '''
    Randomly select {num_users} users from all users
    Args:
        round: current round
        num_users: number of users to select
    Return:
        List of selected clients objects

    Hints:
        1. Default 10 users to select, you can modify the args {--num_users} to change this hyper-parameter
        2. Note that {num_users} can not be larger than total users (i.e., num_users <= len(self.user))
    '''
```

使用 numpy module 內建的 random choice function 去挑選 users

```python
def aggregate_parameters(self):
    total_train_samples = 0
    new_parameters = []
    for i in self.selected_users[0].get_parameters():
        new_parameters.append(0)
    for users in self.selected_users:
        for i, parameter in enumerate(users.get_parameters()):
            new_parameters[i] += parameter * users.train_samples
        total_train_samples += users.train_samples
    for parameter in new_parameters:
        parameter /= total_train_samples
    for parameter, new_parameter in zip(self.model.parameters(), new_parameters):
        parameter.data = new_parameter.data.clone()
    ## TODO
    '''
    Weighted sum all the selected users' model parameters by number of samples

    Args: None
    Return: None

    Hints:
        1. Use self.selected_users, user.train_samples.
        2. Replace the global model (self.model) with the aggregated model.
    '''
```

將 selected 的 user model 的 parameter 做加權平均並更新 global model 的 parameter，程式中共有四個 block(for 迴圈)，第一個 block 是做初始化，第二個 block 做 user model 的加權(權重是 user 的 train_samples)，第三個 block 對家權過的每個 parameter 做平均，第四個 block 更新 global model 的 parameter

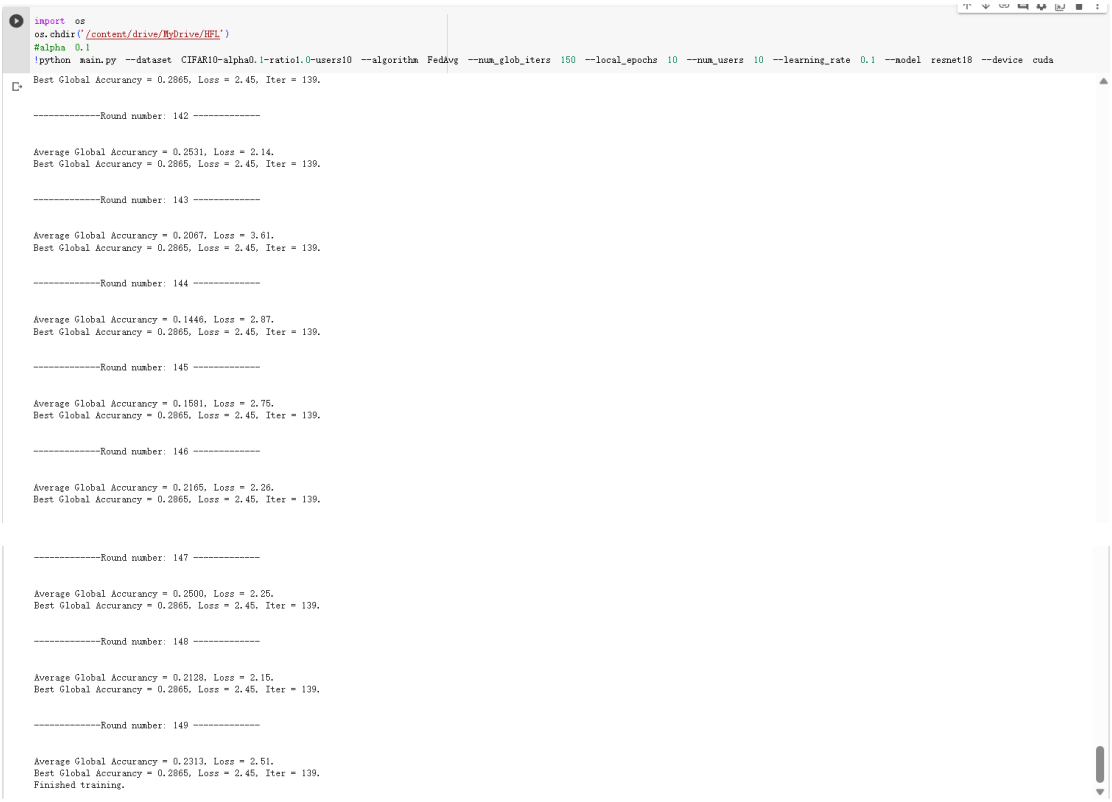
userbase.py

```python
def set_parameters(self, model, beta=1):
    new_parameter = []
    for global_parameter, local_parameter in zip(model.parameters(), self.get_parameters()):
        new_parameter.append(beta * global_parameter + (1 - beta) * local_parameter)
    self.update_parameters(new_parameter)
    ## TODO
    '''
    Replace the user's local model with the global model
    Args:
        model: the global model parameters
        beta: moving average model,
            i.e., user's model parameters = beta * global model parameters + (1 - beta) * user's model parameters
    Return:
        None

    Hint:
        1. You can use self.model (the user's model), model (global model parameters).
    '''
```

利用 global model 的 parameter 與 user model 的 parameter 以特定比例(beta，1 - beta)更新 user model 的 parameter

Data Distribution:

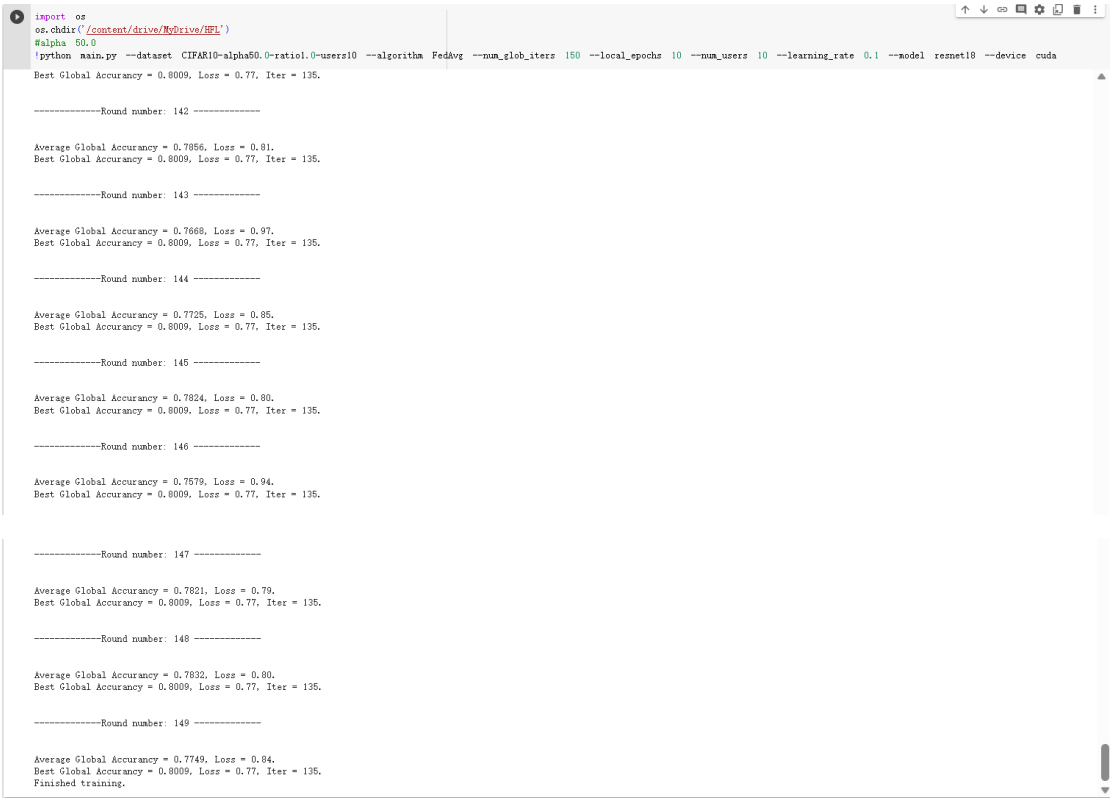
當 alpha 值為 0.1 時：
用戶樣本數量相對較少，對於模型聚合的貢獻較小。
擁有更多樣本的用戶（alpha 值較高）在模型聚合中的影響更大，其模型參數对於全局模型的更新具有更大的權重。
參與訓練的用戶之見的數據分布可能存在較大差異，由於樣本數量不均衡，部分用戶的數據特徵可能無法很好地被捕捉到，可能導致全局模型的準確性下降。



```
import os
os.chdir('/content/drive/MyDrive/HFL')
#alpha 0.1
!python main.py --dataset CIFAR10-alpha0.1-ratio1.0-users10 --algorithm FedAvg --num_glob_iters 150 --local_epochs 10 --num_users 10 --learning_rate 0.1 --model resnet18 --device cuda
```

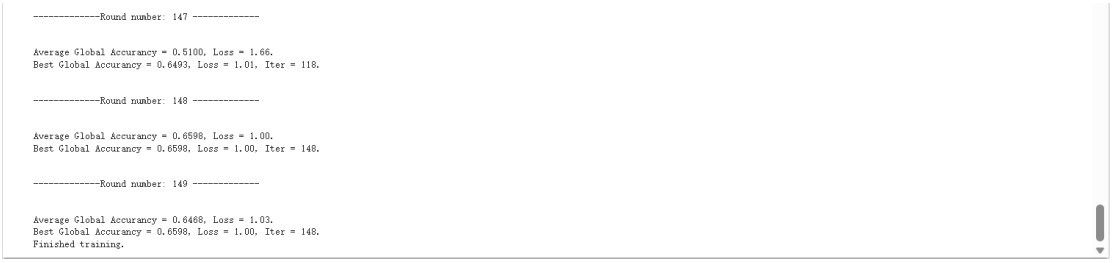
```
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 142 ------------

Average Global Accuracy = 0.2531, Loss = 2.14.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 143 ------------

Average Global Accuracy = 0.2067, Loss = 3.61.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 144 ------------

Average Global Accuracy = 0.1446, Loss = 2.87.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 145 ------------

Average Global Accuracy = 0.1581, Loss = 2.75.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 146 ------------

Average Global Accuracy = 0.2165, Loss = 2.26.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 147 ------------

Average Global Accuracy = 0.2500, Loss = 2.25.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 148 ------------

Average Global Accuracy = 0.2128, Loss = 2.15.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.

------------Round number: 149 ------------

Average Global Accuracy = 0.2313, Loss = 2.51.
Best Global Accuracy = 0.2865, Loss = 2.45, Iter = 139.
Finished training.
```

當 alpha 值為 50.0 時：
用戶樣本數量相對較多，對於模型聚合的貢獻較大。
用戶的數據更具代表性，對於全局模型的更新具有更大的影響力。
數據分布更均衡，不同用戶之間的數據特徵可能更能夠相互補充和提升，有助於提高全局模型的準確性。



Number of users in a round:

當 num_users 值為 2 時：
用戶多樣性較低，較少數量的用戶參與訓練可能導致數據分佈不平衡，無法充分利用數據的多樣性，從而影響準確性。
較少用戶同時參與訓練雖然減少並行計算效率，降低模型訓練的速度，但由於通信開銷降低導致通信效率的上升，導致訓練速度可能升高。



當 num_users 值為 10 時：

用戶多樣性較高，較多數量的用戶參與訓練可以提供更多樣化的數據樣本，有助於全局模型學習更廣泛的特徵和模式，從而提高準確性。

較多用戶同時參與訓練可以提高並行計算效率，加快模型訓練的速度，但由於通信開銷上升導致通信效率的下降，導致訓練速度可能下降。

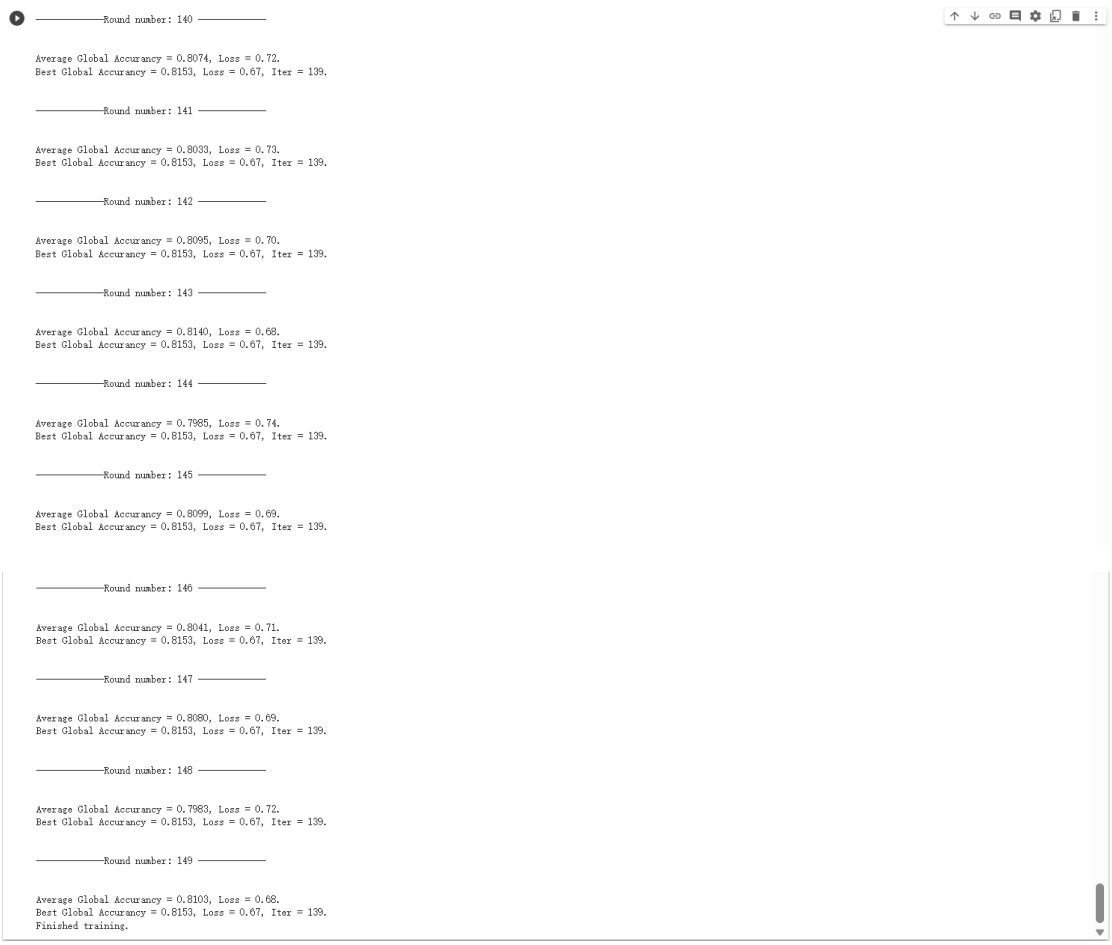
```
[ ] import os
    os.chdir('/content/drive/MyDrive/HFL')
    !python main.py --dataset CIFAR10-alpha100.0-ratio1.0-users10 --algorithm FedAvg --num_glob_iters 150 --local_epochs 10 --num_users 10 --learning_rate 0.1 --model resnet18 --device cuda
```

```
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.


-------------Round number: 142 -------------

Average Global Accurancy = 0.8095, Loss = 0.70.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 143 -------------

Average Global Accurancy = 0.8140, Loss = 0.68.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 144 -------------

Average Global Accurancy = 0.7985, Loss = 0.74.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 145 -------------

Average Global Accurancy = 0.8099, Loss = 0.69.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 146 -------------

Average Global Accurancy = 0.8041, Loss = 0.71.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.


-------------Round number: 147 -------------

Average Global Accurancy = 0.8080, Loss = 0.69.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 148 -------------

Average Global Accurancy = 0.7983, Loss = 0.72.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.

-------------Round number: 149 -------------

Average Global Accurancy = 0.8103, Loss = 0.68.
Best Global Accurancy = 0.8153, Loss = 0.67, Iter = 139.
Finished training.
```

最終 acc 的前十筆及後十筆輸出 (輸出筆數過多)

```
import os
os.chdir('/content/drive/MyDrive/HFL')
!python main.py --dataset CIFAR10-alpha100.0-ratio1.0-users10 --algorithm FedAvg --num_glob_iters 150 --local_epochs 10 --num_users 10 --learning_rate 0.1 --model resnet18 --device cuda
```

```
==================================================
Summary of normal model training process:
Dataset: CIFAR10-alpha100.0-ratio1.0-users10
model: resnet18
Device: cuda
Number of global rounds: 150
Number of local rounds: 10
Clents' learning rate: 0.1
==================================================

        [ Start training iteration 0 ]

Users in total: 10
Number of users per round / total users: 10 / 10
Finished creating FedAvg server.

-------------Round number: 0 -------------

Average Global Accurancy = 0.1068, Loss = 2.30.
Best Global Accurancy = 0.1068, Loss = 2.30, Iter = 0.

-------------Round number: 1 -------------

Average Global Accurancy = 0.1039, Loss = 19565.48.
Best Global Accurancy = 0.1068, Loss = 2.30, Iter = 0.

-------------Round number: 2 -------------

Average Global Accurancy = 0.1367, Loss = 5839.31.
Best Global Accurancy = 0.1367, Loss = 5839.31, Iter = 2.

-------------Round number: 3 -------------

Average Global Accurancy = 0.2087, Loss = 8.96.
Best Global Accurancy = 0.2087, Loss = 8.96, Iter = 3.

-------------Round number: 4 -------------

Average Global Accurancy = 0.2338, Loss = 2.02.
Best Global Accurancy = 0.2338, Loss = 2.02, Iter = 4.

-------------Round number: 5 -------------

Average Global Accurancy = 0.2600, Loss = 2.11.
Best Global Accurancy = 0.2600, Loss = 2.11, Iter = 5.

-------------Round number: 6 -------------

Average Global Accurancy = 0.3297, Loss = 1.81.
Best Global Accurancy = 0.3297, Loss = 1.81, Iter = 6.

-------------Round number: 7 -------------

Average Global Accurancy = 0.3545, Loss = 1.74.
Best Global Accurancy = 0.3545, Loss = 1.74, Iter = 7.

-------------Round number: 8 -------------

Average Global Accurancy = 0.3468, Loss = 1.84.
Best Global Accurancy = 0.3545, Loss = 1.74, Iter = 7.

-------------Round number: 9 -------------

Average Global Accurancy = 0.4054, Loss = 1.60.
Best Global Accurancy = 0.4054, Loss = 1.60, Iter = 9.

-------------Round number: 10 -------------

Average Global Accurancy = 0.4345, Loss = 1.54.
Best Global Accurancy = 0.4345, Loss = 1.54, Iter = 10.
```



這次作業我學習到聯邦平均算法的實作方法，透過實作 selected_users(), aggregate_parameters(), set_parameters()這三個 function，認識此算法中 Server 端是如何與 client 端互動，透過隨機挑選固定數量用戶上傳模型修改全局模型的參數，再由用戶下載全局模型參數，來避免資料外洩的問題。