

Data Science HW4

111062684 林鼎勳

1. 程式說明

a) select_users

首先先判斷要選擇的人數有沒有超過或剛好為全部的人數，若是，則把所有的 users 回傳；若否，則用 random 隨機挑選需要的人數後回傳。

```
def select_users(self, round, num_users):  
    ## TODO  
    ...  
  
    if num_users >= len(self.users):  
        print(f"iteration: {round}, selected users: ALL")  
        return self.users  
    else:  
        selected = random.sample(self.users, num_users)  
        self.logging.info(f"Selected users: {selected}")  
        print(f"iteration: {round}, selected users: {selected}")  
        return selected
```

b) aggregate_parameters

先將 server model 的參數設為 0，接著開始叫出所有 user 的 model 進行加權的計算，直到最後一個人的時候再除以全部的總和，以獲得最終加權平均的結果。

```
def aggregate_parameters(self):  
    ## TODO  
    ...  
  
    total_samples = 0  
    for param in self.model.parameters():  
        param.data = torch.zeros_like(param.data)  
  
    for idx, user in enumerate(self.selected_users):  
        last = idx + 1 == self.num_users  
        total_samples += user.train_samples  
        for server_param, user_param in zip(self.model.parameters(), user.get_parameters()):  
            server_param.data += user_param.data * user.train_samples  
            if last:  
                server_param.data /= total_samples
```

c) set_parameters

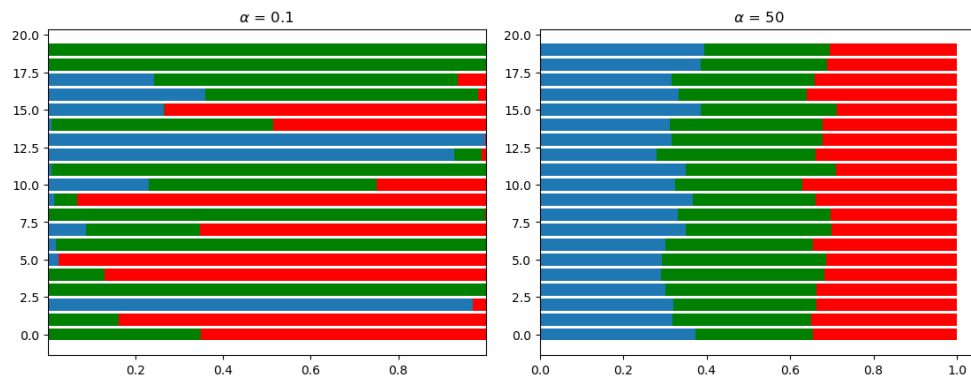
用 for 迴圈依序呼叫 user 和 server 的 parameters 後把後者的參數複製給前者。

```
def set_parameters(self, model, beta=1):  
    ## TODO  
    ...  
  
    for old_param, new_param in zip(self.model.parameters(), model.parameters()):  
        old_param.data = new_param.data.clone()
```

2. 問題探討

a) Data distribution

首先， α 在 Dirichlet distribution 中是控制分布狀態的參數，越大代表會越平均，反之則代表不平均。附圖是參考 NumPy 官網繪製的範例：



此處是以三筆資料為例。由此可明顯看出不同的 α 會造成不同程度的資料分布狀況。

也因為資料分布的差異，所以可以預期 $\alpha = 50$ 的結果會優於 $\alpha = 0.1$ 的結果，因為 $\alpha = 0.1$ 的 model 學習到內容太特定了，不像 $\alpha = 50$ 的 model 可以學到比較全面的資訊，如以下截圖所示：

- $\alpha = 0.1$

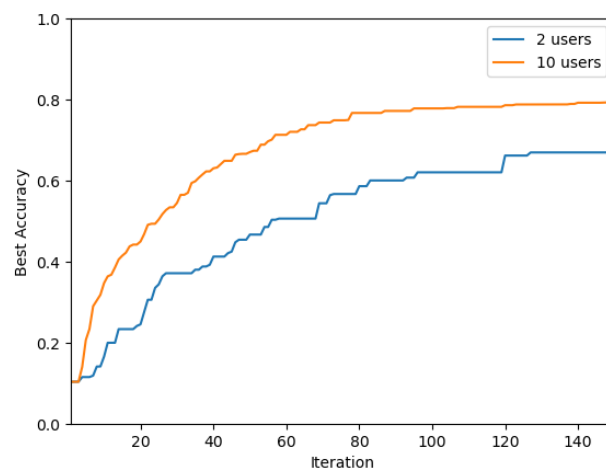
```
-----Round number: 149 -----
2023-05-13 11:45:49,794:INFO: Average Global Accuracy = 0.3757, Loss = 1.73.
2023-05-13 11:45:49,795:INFO: Best Global Accuracy = 0.3896, Loss = 1.60, Iter = 135.
```

- $\alpha = 50$

```
-----Round number: 149 -----
2023-05-16 12:17:52,424:INFO: Average Global Accuracy = 0.7832, Loss = 0.83.
2023-05-16 12:17:52,425:INFO: Best Global Accuracy = 0.7934, Loss = 0.80, Iter = 148.
```

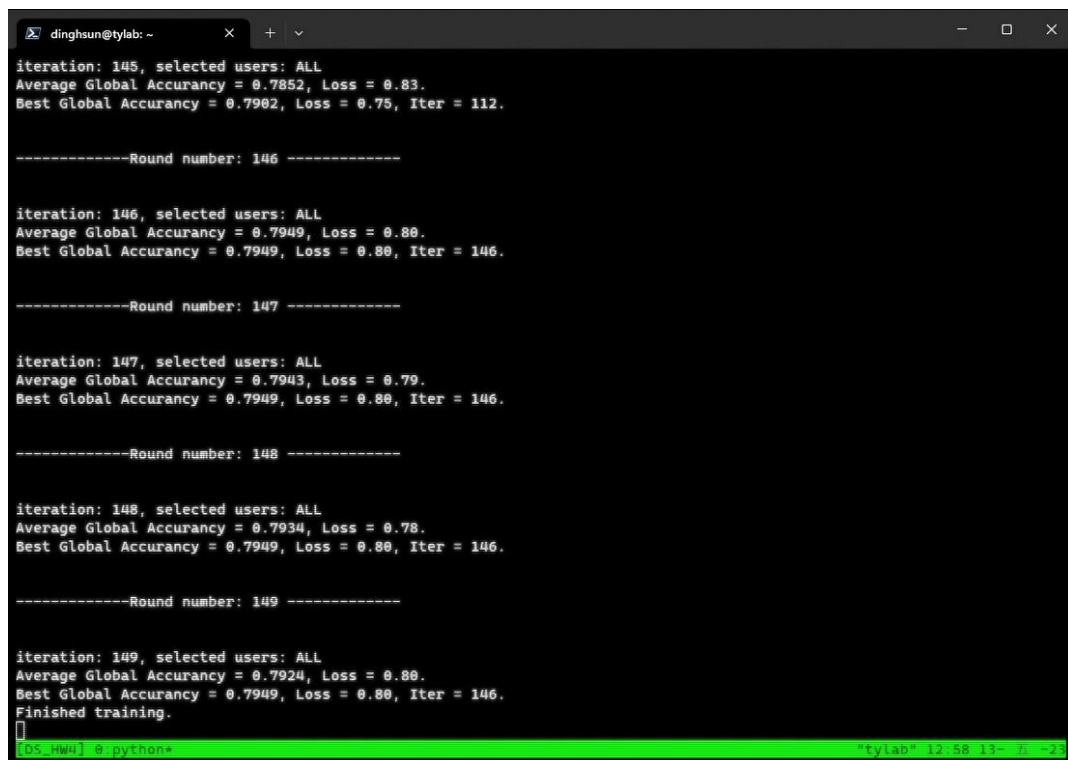
b) Number of users in a round

每一輪參與的人數越多，server 獲得的資料也會更齊全，因此可以預期參與人數較多的結果會較好，且收斂速度較快。附圖橫軸為執行次數，縱軸為截至目前最好的正確率：



從上圖可以看出 10 users 的學習速度較快（收斂速度較快）且準確率也較高。

3. 輸出截圖



```
dinghsun@tylab: ~  
iteration: 145, selected users: ALL  
Average Global Accuracy = 0.7852, Loss = 0.83.  
Best Global Accuracy = 0.7902, Loss = 0.75, Iter = 112.  
  
-----Round number: 146 -----  
  
iteration: 146, selected users: ALL  
Average Global Accuracy = 0.7949, Loss = 0.80.  
Best Global Accuracy = 0.7949, Loss = 0.80, Iter = 146.  
  
-----Round number: 147 -----  
  
iteration: 147, selected users: ALL  
Average Global Accuracy = 0.7943, Loss = 0.79.  
Best Global Accuracy = 0.7949, Loss = 0.80, Iter = 146.  
  
-----Round number: 148 -----  
  
iteration: 148, selected users: ALL  
Average Global Accuracy = 0.7934, Loss = 0.78.  
Best Global Accuracy = 0.7949, Loss = 0.80, Iter = 146.  
  
-----Round number: 149 -----  
  
iteration: 149, selected users: ALL  
Average Global Accuracy = 0.7924, Loss = 0.80.  
Best Global Accuracy = 0.7949, Loss = 0.80, Iter = 146.  
Finished training.  
[DS_HW4] @:python* "tylab" 12:58 13- 5 -23
```

4. 學到甚麼

透過此次作業，讓我概略了解 Horizontal Federated Learning 的內容以及實際上的執行方式。也藉由助教提出的問題了解資料的分布情形和使用者的多寡是如何影響中心 server 訓練的速度以及準確度。

5. 參考資料

a) NumPy 官網對於 dirichlet 函數的說明以及範例：

<https://numpy.org/doc/stable/reference/random/generated/numpy.random.dirichlet.html>