

A Further evaluation of the empirical performance of PI-FL

A.1 Ablation study of clustering performance with Incentive in PI-FL

Table 3: Test accuracy of ablation study with Incentive (I) and without incentive (NI)

	10:90		30:70		linear		random	
	c0	c1	c0	c1	c0	c1	c0	c1
PI-FL (I)	58.62%(0)	67.4%(1)	51.12%(0)	64.06%(1)	66.2%(0)	57.02%(1)	64.54%(0)	56.86%(1)
PI-FL (NI)	49.92%(1)	52.8%(1)	48.9%(0)	44.44%(1)	50.82%(1)	45.92%(1)	57.42%(1)	46.76%(1)

We perform an ablation study with the incentive component of PI-FL on the Synthetic CIFAR10 dataset for 200 rounds with $N = 100$ clients, batch size 128, and learning rate $\eta = 0.01$. When the incentive is disabled clients do not consider maximizing their incentive while sending preference bids. Instead, clients send preference bids with random cluster choices to the scheduler as in FedAvg [2].

Table 3 shows the test accuracies of the cluster-level models. PI-FL(I) indicates that incentives are enabled and PI-FL(NI) shows the accuracies when incentives are disabled. In general PI-FL(I) outperforms PI-FL(NI) in terms of test accuracy for all partitions. The important point to note here is that the incentive mechanism in PI-FL directly motivates clients to join clusters in which they can have the most contribution. This results in accurate clustering based on client data distributions and good-quality personalized models. This is indicated by the performance of PI-FL(NI), i.e without incentive, cluster-level models are unable to dominate a single distribution and only perform well for a single distribution for all partitions except 30:70. Compared to this, in PI-FL(I) each cluster-level model dominates and performs well for their distribution.

A.2 Advantages of including client preferences in pFL.

A.2.1 Challenges requiring client autonomy

Prior personalized FL works generate personalized models from the server’s perspective. We argue that the server may not have complete information to produce good-quality models due to a variety of challenges [10, 52]. These challenges are as follows: **Confidentiality**: Some clients may have sensitive data that they do not want to share with others for privacy or security reasons. For example, a company may have confidential customer data that they do not want to share with a third-party vendor, **Competitive Advantage**: In some industries, companies may want to keep their data private to maintain a competitive advantage. For example, a company may not want to share its sales data with competitors. **Data Governance**: Some organizations may have strict data governance policies that prohibit the sharing of certain types of data. For example, a healthcare organization may not be able to share patient data without proper consent. **Resource limitations**: Clients with large datasets may not have the resources to share all their data for training. In these cases, they may choose to share a random sample of their data to keep the training process manageable. **Data Anonymization**: Sometimes, clients may not want to share the raw data, but instead, they may share a subset of the data which has been anonymized to protect the privacy of the individuals. **Compliance with privacy laws**: In order to comply with privacy laws (GDPR [58], HIPA [59]) some clients might only share anonymized data while keeping Personally Identifiable Information (PII) private. Prior clustering-based pFL works also lack support to accurately include new clients into the clusters whose data qualities are unknown. By including client preferences, PI-FL performs accurate clustering and generation of appealing personalized models for new clients.

We use the Synthetic datasets to test PI-FL where the aggregator is unaware of the client’s dataset distribution and goals.

A.2.2 Clustering performance with Synthetic CIFAR10 dataset

Cluster-based pFL methods cluster all heterogeneous clients within different clusters so each cluster has homogeneous clients with similar data distribution in it. So first we test the client-preference driven clustering design of PI-FL with Synthetic CIFAR10 Data to show the performance of cluster-level models. We use the same configurations described in the main paper for the CIFAR10 dataset evaluation. We perform training for 500 rounds with both PI-FL and FedSoft. Table 4 shows the cluster-level model test accuracies for PI-FL with Synthetic CIFAR10 data.

Table 4: Test accuracy for PI-FL on Synthetic CIFAR10 Dataset

	10:90		30:70		linear		random	
	c0	c1	c0	c1	c0	c1	c0	c1
θ_0	62.78%	2.56%	53.28%	34.32%	61.66%	12.08%	66.9%	19.48%
θ_1	1.5%	70.96%	30.38%	61.94%	30.94%	59.44%	19.12%	58.42%

Table 5: Test accuracy for PI-FL and FedSoft on Synthetic CIFAR10 Dataset

	10:90		30:70		linear		random	
	c0	c1	c0	c1	c0	c1	c0	c1
PI-FL	62.68%(0)	70.96%(1)	53.28%(0)	61.94%(1)	61.66%(0)	59.44%(1)	66.90%(0)	58.42%(1)
FedSoft	32.50%(0)	38.62%(1)	20.28%(0)	23.58%(0)	34.42%(1)	49.62%(1)	21.62%(1)	33.12%(1)

PI-FL accurately differentiates between clients of different distributions. This is visible by the accuracy difference of each cluster-level model on different distributions. For example on the 10:90 partition c1 model has a 70.96% accuracy on the θ_1 distribution and has 2.56% accuracy on θ_0 which indicates that cluster-level model c1 trains with clients that have the majority of their training data from θ_1 . Similarly, c0 trains with clients that have their majority of training data from θ_0 and has an accuracy of 62.68%.

Table 5 shows the cluster-level model accuracy comparison of PI-FL and FedSoft. The number inside the parenthesis along with accuracy shows the distribution for which the cluster-level model performs best. For example, for the linear partition, PI-FL c0 cluster-level model has an accuracy of 61.66% for distribution θ_0 and 59.44% accuracy for distribution θ_1 . For the linear partition with FedSoft, both c0 and c1 perform best on only one distribution θ_1 with accuracy 34.42% and 49.62%. PI-FL outperforms FedSoft in terms of accuracy for each partition and is able to distinguish between different distributions accurately.

Table 6: Test accuracy for FedSoft on Synthetic CIFAR10 Dataset

	10:90		30:70		linear		random	
	c0	c1	c0	c1	c0	c1	c0	c1
θ_0	32.5%	13.6%	20.28%	23.58%	8.48%	2.82%	16.18%	0.28%
θ_1	11.76%	38.62%	0.18%	0.08%	34.42%	49.62%	21.62%	33.12%

Table 6 represents the result of FedSoft [25] tested with the Synthetic CIFAR10 dataset. The experimental setup is explained in more detail in the main paper. We can observe that except for the 10:90 partition, FedSoft is unable to cluster clients under individual tiers in a way that each tier-level model could dominate one distribution. Both tier-level models perform well on only one distribution and the other is ignored leading to low test accuracy of both tier-level models for the ignored distribution.

A.2.3 Clustering performance with Synthetic EMNIST dataset

Synthetic EMNIST dataset. This image dataset has images of dimension 28 x 28 and 52 output classes where 26 classes are lower case letters and 26 classes are upper case letters. We test the dataset on different partitions of 10:90, 30:70, linear, and random created in the same way as the Synthetic CIFAR10 data. The only difference is that D_A contains 26 lowercase letters and D_B has 26 uppercase letters.

Since Ditto is not a clustering-based algorithm, we only compare the test accuracy of personalized models on the EMNIST dataset for PI-FL and Ditto in Figure 5. PI-FL outperforms Ditto significantly, especially for highly heterogeneous data partitions such as 10:90 and linear. The reason for this performance improvement is that, unlike Ditto, PI-FL provides autonomy to clients to personalize according to their own goals. With Ditto, the goal of each client which consists of their private data is hidden from the aggregator server which affects the quality of personalized models.

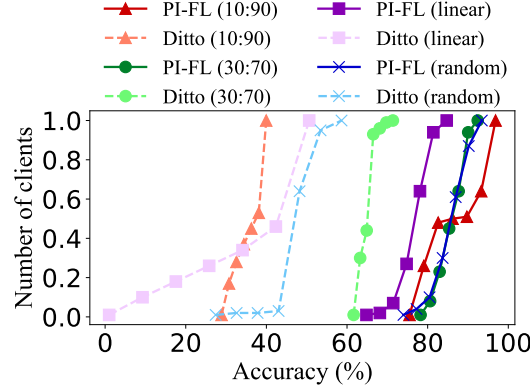


Figure 5: CDF of clients' personalized model test accuracy for PI-FL and Ditto on EMNIST Dataset

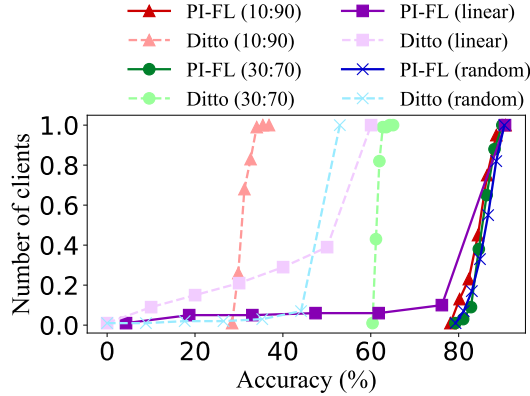


Figure 6: CDF of clients' personalized model test accuracy for PI-FL and Ditto on 4 and more distributions with EMNIST Dataset

617 A.2.4 Multiple distribution Results

618 PI-FL outperforms other FL personalization algorithms in heterogeneous cases when the clients and
619 dataset are divided between 2 distributions (DA & DB). To analyze the reliability of PI-FL it is also
620 evaluated in highly heterogeneous conditions with 4 different distributions (DA, DB, DC, and DD)
621 on the EMNIST dataset. EMNIST dataset has a total of 56 classes, thus each of the 4 distributions
622 gets 25% of total available classes. DA has the first 13 classes, DB has the next 13, and so on. Figure
623 6 shows the CDF of test accuracy for all personalized models at clients. PI-FL outperforms Ditto for
624 all partition types. For the linear partition less than 10% clients have lower than 80% accuracy and
625 we attribute this as an outlier due to the partition type where dividing the data linearly some clients
626 get very few data samples. This trend can also be seen with Ditto for the linear partition.

627 A.2.5 Insights from the analysis with Synthetic datasets

628 We use the synthetic datasets to present a scenario of dynamic data at client or induction of new
629 clients in pFL whose data distributions are unknown. This presents new challenges of accurate
630 clustering and the generation of personalized models for less familiar clients. We observe through
631 empirical evaluation with Synthetic datasets that this can lead to 100% opt-outs from these clients if
632 we use conventional server-driven pFL methods. However, we observe that by the induction of client
633 preferences in clustering, PI-FL is able to do accurate clustering and generate appealing models for
634 new clients and changing data distribution at clients.

Table 7: PMA and opt-outs with the 4 classes per client EMNIST dataset

pFL Method	FedAvg	Ditto	FedProx	FedALA	PerfFedAvg	FedFomo	PI-FL
Personalized Accuracy	82.8±4.2	90.78±2.1	61.23±2.86	62.17±3.9	59.99±1.87	86.57±1.59	98.59±1.2
Optouts	-	0	0.64	0.31	0.68	0	0
Average PMA	-	24.48±4.2	0.57±5.14	1.5±4.74	-0.6±1.5	25.9±3.6	37.93±3.43

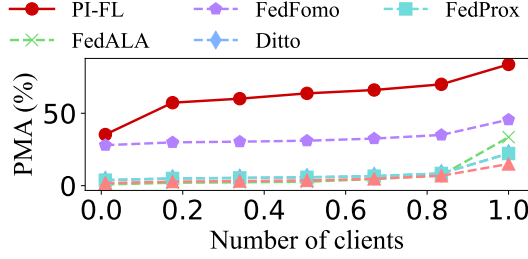


Figure 7: CDF of clients' PMA for PI-FL and pFL algorithms on CIFAR10 Dataset

635 A.3 Opt-out results for 4 classes per client with EMNIST dataset

636 Table 7 shows the complete opt-out, test accuracy, and PMA results for the test conducted with the
 637 EMNIST dataset with 4 classes per client. Some pFL algorithms such as FedFomo and Ditto have
 638 similar performance compared to PI-FL in terms of opt-outs. However, PI-FL outperforms all of
 639 them in terms of PMA and test accuracy.

640 A.4 Opt-out results for 2 classes per client with CIFAR10 dataset

641 We also test highly heterogeneous data conditions where each client has only 2 classes per client. We
 642 use the learning rate = 0.01, batch size = 128, and global epochs = 150 with the CIFAR10 dataset
 643 and the same CNN model mentioned in the main paper. Figure 7 shows the PMA for different pFL
 644 algorithms. PI-FL outperforms all other pFL algorithms by approximately 50% and FedFomo by
 645 30%. This goes to show that PI-FL performs even better in highly heterogeneous conditions where
 646 personalization is used to tackle data heterogeneity.

647 B Shapley value approximation for client contribution

648 Here we present the Shapley Value approximation derivation we use for calculating the client
 649 contributions.

Algorithm 3 Estimated Shapley value of any client in an FL

Input: Test data $(x_i, y_i), i = 1, \dots, n_{\text{test}}$, clients' local model parameters and aggregation weights, W_m, λ_m .
 server's aggregated model parameter $W_M = \sum_{m=1}^M \lambda_m W_m$.

- 1: Calculate $\gamma_M \triangleq n_{\text{test}}^{-1} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M)$
- 2: **for** $k = 1, \dots, M$ **do**
- 3: Calculate SHAP($i \rightarrow [M]$) using

$$-\left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M)\right)^T \lambda_i W_i \quad (16)$$

for unnormalized aggregation, or

$$-\left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M)\right)^T \lambda_i (W_i - W_M) \quad (17)$$

for normalized aggregation.

4: **end for**

Output: Obtains all clients' Shapley values

650 **Notation:** Let $[M]$ denote the set $\{1, \dots, M\}$, and $A - B$ the set of elements in A but not in B . In
 651 this section, $[M]$ denotes all the agents that *participate* in the coalition. We will consider those not
 652 participating in the near future.

653 We aim to look for a reasonable way to quantify the amount of each client's contribution in a round.
 654 Suppose at any particular round, the server obtains an aggregated model with parameter

$$W_{[M]} \triangleq \sum_{m \in [M]} \lambda_m W_m, \quad (18)$$

655 where λ_m is the weight (usually n_m/n where n_m and n are sample sizes of client m and all clients,
 656 respectively), and W_m is the locally updated model of client m .

657 The prediction loss of the model with parameter W , denoted by $\mathcal{L}(W)$, is approximated by

$$\mathcal{L}(W) \approx \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \ell(x_i, y_i; W), \quad (19)$$

658 where (x_i, y_i) , $i = 1, \dots, n_{\text{test}}$, is a set of test data. At round t , we define the value function of a set
 659 of agents C based on how much their contributed model, denoted by W_C , has decreased the loss of
 660 the earlier model, denoted by W_{t-1} , namely

$$v_t(C) \triangleq \mathcal{L}(W_{t-1}) - \mathcal{L}(W_C), \quad (20)$$

661 so that the larger the better. When there is no ambiguity, we simply write v_t as v . It is worth noting
 662 that v is a function of the set while \mathcal{L} is a function of the parameter. Once C is realized, W_C will
 663 become W_t for the next round.

664 Recall that the original Shapley value [60] of agent m given a set of agents A and a value function v
 665 is defined by

$$\sum_{S \in A - \{m\}} \frac{|S|!(|A| - 1 - |S|)!}{|A|!} (v(\{S \cup \{m\}\}) - v(S)), \quad (21)$$

666 whose sum over all agents is equal to $v(A) - v(\emptyset)$. Here, \emptyset represents the *baseline* coalition scenario,
 667 from which the contribution of each agent is quantified. To highlight the dependency on baseline, we
 668 use B to denote the baseline and rewrite (21) as

$$\text{SHAP}(m \rightarrow A \mid B) \quad (22)$$

$$\triangleq \sum_{S \in A - \{m\}} \frac{|S|!(|A| - 1 - |S|)!}{|A|!} (v(\{S \cup \{m\}\} \mid B) - v(S \mid B)), \quad (23)$$

669 where $v(S \mid B)$ means the value of S conditional on the baseline B . In our scenario, B means the set
 670 of agents that are already in coalition and thus

$$v(S \mid B) \triangleq v(S \cup B). \quad (24)$$

671 Let us consider the baseline as $B \triangleq [M] - \{i, j\}$. The corresponding baseline model will be

$$\text{unnormalized version: } W_{[M] - \{i, j\}} \triangleq \sum_{m \in [M] - \{i, j\}} \lambda_m W_m, \quad (25)$$

$$\text{normalized version: } W_{[M] - \{i, j\}}^* \triangleq \frac{1}{\sum_{m \in [M] - \{i, j\}} \lambda_m} \sum_{m \in [M] - \{i, j\}} \lambda_m W_m. \quad (26)$$

672 We consider an unnormalized version for brevity. The additional value by introducing i, j is

$$v(\{i, j\} \mid [M] - \{i, j\}) - v(\emptyset \mid [M] - \{i, j\}) \quad (27)$$

$$\text{use(24)} = v([M]) - v([M] - \{i, j\}) \quad (28)$$

$$\text{use(20) and recall (18) and (25)} = \mathcal{L}(W_{[M]-\{i,j\}}) - \mathcal{L}(W_M) \quad (29)$$

$$= \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \{\ell(x_i, y_i; W_M) - \ell(x_i, y_i; W_{[M]-\{i,j\}})\} \quad (30)$$

$$\approx \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M)^T (W_{[M]-\{i,j\}} - W_M) \quad (31)$$

$$= - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T (\lambda_i W_i + \lambda_j W_j). \quad (32)$$

673 Next, we calculate how much agent i should be attributed to the above gain that is achieved by
 674 i, j jointly. To that end, we calculate the Shapley value of agent i conditional on that agents in
 675 $[M] - \{i, j\}$ already participate, namely

$$\text{SHAP}(i \rightarrow \{i, j\} \mid [M] - \{i, j\}) \quad (33)$$

$$\begin{aligned} \text{recall (23)} &= \sum_{S \in \{j\}} \frac{|S|!(1-|S|)!}{2!} \left(v\left(S \cup \{i\} \cup ([M] - \{i, j\})\right) \right. \\ &\quad \left. - v\left(S \cup ([M] - \{i, j\})\right) \right) \end{aligned} \quad (34)$$

$$= \frac{1}{2} \left(v([M]) - v([M] - \{j\}) + v([M] - \{i\}) - v([M] - \{i, j\}) \right) \quad (35)$$

$$= \frac{1}{2} \left(-\mathcal{L}(W_M) + \mathcal{L}(W_{M,-i}) - \mathcal{L}(W_{M,-j}) + \mathcal{L}(W_{M,-ij}) \right) \quad (36)$$

$$= \frac{1}{2} \left(-\mathcal{L}(W_M) + \mathcal{L}(W_{M,-i}) + \mathcal{L}(W_M) - \mathcal{L}(W_{M,-j}) + \mathcal{L}(W_{M,-ij}) - \mathcal{L}(W_M) \right) \quad (37)$$

$$\text{use (29)–(32) and alike} \approx -\frac{1}{2} \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T (\lambda_i W_i - \lambda_j W_j + \lambda_i W_i + \lambda_j W_j)$$

$$= - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T \lambda_i W_i \quad (38)$$

676 which, interestingly, does not depend on j . As such, we use this to calculate the Shapley value of
 677 client i , denoted by

$$\text{SHAP}(i \rightarrow [M]) \triangleq - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T \lambda_i W_i. \quad (39)$$

678 From Equalities (32) and (38), we can verify that

$$v(\{i, j\} \mid [M] - \{i, j\}) - v(\emptyset \mid [M] - \{i, j\}) = \text{SHAP}(i \rightarrow [M]) + \text{SHAP}(j \rightarrow [M])$$

679 **Remark B.1 (Intuitions).** Intuitively, our derived Shapley value of client i in (38) can be regarded
 680 as the model's marginal reduction of the test loss by introducing client i . To see that, consider the
 681 following approximation based on first-order Taylor expansion:

$$\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \ell(x_i, y_i; W_M - \Delta) - \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \ell(x_i, y_i; W_M) \quad (40)$$

$$\approx - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T \Delta W, \quad (41)$$

682 which becomes the term in (38) when $\Delta W \triangleq \lambda_i W_i$. The above quantity approximates the amount
 683 of client i 's contribution to decreasing the test loss of the server's aggregated model, the larger the
 684 better.

685 *Remark B.2* (Normalized counterpart). Suppose we use the normalized version introduced in (26)
 686 when considering the baseline without clients i, j . Thus,

$$W_{[M]-\{i,j\}}^* = \frac{W_M - \lambda_i W_i - \lambda_j W_j}{\sum_{m \in [M]-\{i,j\}} \lambda_m} \quad (42)$$

$$= W_M + \frac{(\lambda_i + \lambda_j)W_M - \lambda_i W_i - \lambda_j W_j}{\sum_{m \in [M]-\{i,j\}} \lambda_m} \quad (43)$$

$$= W_M - \frac{\lambda_i(W_i - W_M) + \lambda_j(W_j - W_M)}{1 - (\lambda_i + \lambda_j)}. \quad (44)$$

687 Similarly, we have

$$W_{[M]-\{i\}}^* = W_M - \frac{\lambda_i(W_i - W_M)}{1 - \lambda_i}. \quad (45)$$

688 Bringing the above formula into (37), we have

$$\text{SHAP}(i \rightarrow \{i, j\} \mid [M] - \{i, j\}) \quad (46)$$

$$= \frac{1}{2} \left(-\mathcal{L}(W_M) + \mathcal{L}(W_{M,-i}) + \mathcal{L}(W_M) - \mathcal{L}(W_{M,-j}) + \mathcal{L}(W_{M,-ij}) - \mathcal{L}(W_M) \right) \quad (47)$$

$$\approx - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T \Delta W^* \text{ where} \quad (48)$$

$$2\Delta W^* \triangleq \frac{\lambda_i(W_i - W_M)}{1 - \lambda_i} - \frac{\lambda_j(W_j - W_M)}{1 - \lambda_j} + \frac{\lambda_i(W_i - W_M) + \lambda_j(W_j - W_M)}{1 - (\lambda_i + \lambda_j)}. \quad (49)$$

$$\approx 2\lambda_i(W_i - W_M) \quad (50)$$

689 assuming small λ_i and λ_j . Therefore, under normalization we have

$$\text{SHAP}(i \rightarrow \{i, j\} \mid [M] - \{i, j\}) \approx - \left(\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \nabla_W \ell(x_i, y_i; W_M) \right)^T \lambda_i(W_i - W_M). \quad (51)$$

690 The intuition is the same as Remark B.1 except that the server model with client i satisfies

$$\text{unnormalized version : } W_{[M]-\{i\}} = W_M - \lambda_i W_i. \quad (52)$$

$$\text{normalized version : } W_{[M]-\{i\}} \approx W_M - \lambda_i(W_i - W_M). \quad (53)$$

691 C Definitions for PI-FL

692 In this section, we provide definitions for Pi-FL.

693 **Definition 1:** A provider payoff $\mathbf{r}_{ik} \in \mathbb{R}$ is individually rational if each provider can
 694 obtain a benefit no less than that by acting alone, i.e., $r_{ik} \geq r_i \mid \forall i \in N, \forall k \in K$

Definition 2: On the basis of definition 1, we extend this definition to group rationality. Providers will join tiers with greater similarity indexes compared to other tiers if their incentive is directly correlated with similarity and performance, i.e.,

$$\delta(i, k_1) \leq \delta(i, k_2)$$

, then

$$r_{ik_1} \geq r_{ik_2} \mid \forall i \in N, \forall (k_1, k_2) \in K$$

695 where $i \in N$ and N represents all clients, $k \in K$, K represents all tiers. r_{ik}
 696 represents the reward for client i in tier k .

697 **Definition 3:** As the tier-level model converges, the personalized model generated
 698 from tier-level models will have a similar performance compared to the tier-level
 699 model performance on the local dataset of client $i \mid \forall i \in N$ as it is a weighted
 700 mixture of the tier-level models

701 We have made the code available as part of the supplementary material.