Collaborative Machine Learning Markets with Data-Replication-Robust Payments

Olga Ohrimenko **Shruti Tople** **Sebastian Tschiatschek**

Microsoft Research
{oohrim,t-shtopl,setschia}@microsoft.com

Abstract

We study the problem of collaborative machine learning markets where multiple parties can achieve improved performance on their machine learning tasks by combining their training data. We discuss desired properties for these machine learning markets in terms of fair revenue distribution and potential threats, including data replication. We then instantiate a collaborative market for cases where parties share a common machine learning task and where parties' tasks are different. Our marketplace incentivizes parties to submit high quality training and true validation data using a novel payment-division function that is robust-to-replication and customized output models that perform well only on requested machine learning tasks. In experiments, we validate the assumptions underlying our theoretical analysis and show that these are approximately satisfied for commonly used machine learning models.

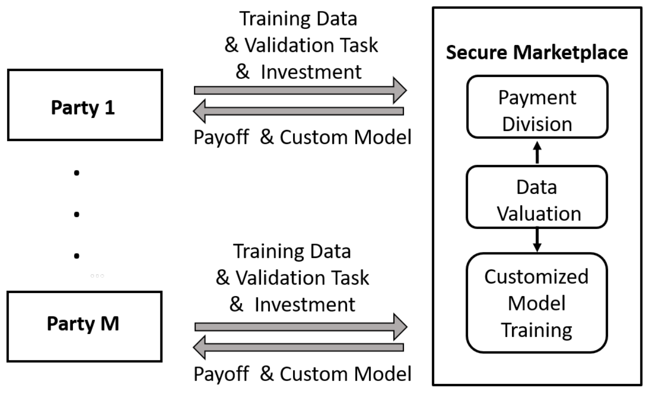


Figure 1: Collaborative marketplace setup.

1 Introduction

One of the main obstacles for training well-performing machine learning models is the limited availability of sufficiently diverse labeled training data. However, the data needed to train good models often exists but is not easy to leverage as it is distributed and owned by multiple parties. For instance, in the medical domain, important data about patients that could be used for learning diagnostic support systems for cancer might be in possession of different hospitals, each of which holding different data, (e.g., from a specific geographical region with different demographics). Typically, by pooling the available data, the hospitals could train better machine learning models for their application than they could using only their own data. As all hospitals would benefit from a better machine learning model obtained through data sharing, there is a need for collaborative machine learning.

Naturally, this type of collaboration raises questions in terms of how to incentivize parties to participate in such a collaborative machine learning effort. Unfortunately, in many applications there is a clear and natural incentive for parties to provide quality data or share their data to begin with. Financially rewarding parties to incentivize participation seems to be natural in such cases but has to be done with great care. For example, a fixed price per data point could motivate parties to gather large amounts of low quality or fake data if there is no mechanism to control data quality. Another reason that may dis-incentivize parties from sharing data could stem from privacy and integrity concerns regarding the use of party's data once it is shared. For example, once a party's data is released it can be easily reused or sold.

To overcome these challenges and enable collaborative machine learning in the outlined setting, there is a need for a secure machine learning marketplace for joint model training that guarantees fair incentives for participation and ensures secure data handling. In this work, we propose a cloud-based collaborative machine learning platform accessible to parties for submitting data and machine learning tasks. Submitted training data represents the data that a party is willing to contribute/sell, while submitting the validation data can be seen as a specification of the machine learning model a party is willing to buy. After a *trade* in this market, a participating party obtains a model trained on the data available to the market and customized for its task. Crucially, there is no sharing of data and parties are only provided this customized model (or query interface). As a result, only information relevant to the validation task is released through the model, limiting the possibility of copying and reusing the data for other tasks. Such markets allow multiple parties to jointly train machine learning models based on the training data provided by all of the parties and achieve improved performance on their own tasks. Parties pay *to* the market for the improvement on their validation tasks and get paid *by* the market for the contribution of their training data to the tasks of others. The market can support a single validation task scenario, for example, where hospitals bring together their data to train a single model for detecting cancer. Furthermore, it also supports scenarios where one's data can contribute to multiple tasks. An overview of our envisioned marketplace is shown in Figure 1.

Our market is enabled by three main components: 1. *Data valuation.* Each model in the market is trained on the data that is best suited for the specified validation task. Similar to recent and concurrent work (Jia et al. 2019; Ghorbani and Zou 2019) we can use Shapley values (Shapley 1953), a solution concept from game theory discussed later, to match data to tasks. 2. *Customized model training.* We ensure that the model that a party receives is only suited for its own task but not the other tasks available on the market. As a result each party is incentivized to provide its true validation task. 3. *Payment division.* Each party receives a reward proportional to how useful its data is for training other models. We again use Shapley values, however, in this case to determine fair payoffs.

One of the key challenges in designing such data marketplaces comes from the very nature of data, i.e., free replication. This means that the reward of a party submitting copies of the same dataset should not be more than the party submitting it once. Indeed the authors of (Agarwal, Dahleh, and Sarkar 2019) also point out that, if used naively for machine learning, Shapley value is not robust to replication (albeit in a different market setup than ours). We design a market that is robust-to-replication. The key idea behind our approach is to allow a party to contribute data only if it also submits a validation task for which it requires a trained model. Submitting a task, in turn, requires a participation fee. Hence, for every replica, a party has to pay a participation fee that depends on its validation task. As a result, the fee it pays for the improvement on its task balances the payoff it gets from the use of its training data.

Finally, we ensure that our cloud-based market is secure and can be trusted by the parties. That is, the market itself cannot resell, copy or release the data or the trained models. Such a market can be instantiated using secure hardware, such as Intel SGX (Hoekstra et al. 2013), which allows parties to submit their data encrypted and ensure that it is only decrypted and processed in secure enclaves and data is always encrypted in memory. It also provides attestation capabilities that parties can use to verify the integrity of the market (e.g., that the data was used only to train specific models and was not copied in plaintext outside of an enclave). See (Ohrimenko et al. 2016) for details.

We summarize our contribution as follows:

1. *Marketplace Definition*: We introduce a collaborative marketplace to sell data and buy machine learning models for learning single and multiple validation tasks.
2. *Payment Division*: We propose a novel and robust-to-replication payment division function.
3. *Customized model training*: We propose to release customized models to each party and not data in order to prevent malicious parties from reselling the data while incentivizing them to submit honest validation tasks (which in our market can be seen as an indirect bid on the data).
4. *Evaluation*: We empirically evaluate the properties of our marketplace. Our experimental results confirm that our marketplace generates fair payoffs, customized models and is robust to replication.

2 Background and Notation

Machine Learning Models and Evaluation In supervised machine learning, we often consider training of models for regression or classification tasks, i.e., learning of a prediction function $\mathcal{M}\colon \Omega \to \mathbb{R}^k$ or $\mathcal{M}\colon \Omega \to [K]$, respectively, where Ω is some input space, $k \in \mathbb{N}$, and $[K] = \{1, \ldots, K\}$. We introduce concepts for classification problems only but the same concepts apply to regression problems. The model $\mathcal{M}$ is a part of some hypothesis space $\mathcal{H}$, e.g., the set of one layer neural-networks with a fixed number of hidden neurons and sigmoid activations. These functions are learned from labeled training data $\mathcal{X}$, i.e., collections/sets of tuples of the form (ω, z), where $z \in [K]$ for classification. Learning in that context commonly refers to minimizing a sample-wise loss function $l\colon \Omega \times [K] \to \mathbb{R}$, e.g., the cross-entropy loss, overloading notation: $\mathcal{M}(\mathcal{X}) = \arg\min_{\mathcal{M}' \in \mathcal{H}} \frac{1}{|\mathcal{X}|} \sum_{(\omega,z)\in\mathcal{X}} l(\mathcal{M}'(\omega), z)$. The goal of the learning process is to identify functions $\mathcal{M}$ that perform well on unseen data, i.e., test data, and, for instance, achieve good classification accuracy. As a proxy, for estimating the performance on test data, we use validation data $\mathcal{V}$. The average classification performance on the validation data is $\mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X})) = \frac{1}{|\mathcal{V}|} \sum_{(\omega,z)\in\mathcal{V}} \mathbf{1}_{\mathcal{M}(\omega)=z}$, where $\mathbf{1}$ is the indicator function. Clearly, the performance measure is application dependent and can, for instance, also be the RMSE, ranking accuracy, etc. We refer to $\mathcal{G}$ as (performance) gain function.

Properties of $\mathcal{G}$ In the paper and for simplicity, we assume an idealized gain function $\mathcal{G}$ and dependencies of the model on the training data. For training datasets $\mathcal{X}, \mathcal{X}'$ we assume that:

(i) *replicated data does not change performance*: $\forall \mathcal{X}, \mathcal{X}'\colon \mathcal{M}(\mathcal{X}) = \mathcal{M}(\mathcal{X} \oplus \mathcal{X})$;

(ii) *monotonicty*: $\mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X}))) \leq \mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X} \oplus \mathcal{X}')))$;

(iii) *supermodularity*: $\mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X} \cup \{x\})) - \mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X})) \leq \mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X}' \cup \{x\})) - \mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X}'))$ for $\mathcal{X} \subseteq \mathcal{X}'$ and $x \notin \mathcal{X}'$;

(iv) *boundedness*: $\mathcal{G}(\mathcal{V}, \mathcal{M}(\mathcal{X})) \leq 1$.

Here, (i) captures that duplication of training data does not change the learned model. This is, for instance, true for 1-NN classifiers; (ii) characterizes that additional data either improves or does not change the performance; (iii) captures the collaborative market setting where each party provides complementary data that contributes towards a ML task even in the presence of other parties' data.

Shapley Values for Fair Payoffs Consider a (machine learning) task which M parties $\mathbf{M} = \{1, \dots, M\}$ aim to solve with a joint effort. To quantify the value of the contribution of each party towards solving the task, we consider a characteristic function $v \colon 2^{\mathbf{M}} \to \mathbb{R}$. For every set $S \subseteq \mathbf{M}$ of parties, $v(S)$ quantifies how well the parties in S can solve the task, e.g., $v(S)$ could be the prediction accuracy of the best model the parties in S can train by combining their training data, i.e., $v(S) = \mathcal{G}(\mathcal{V}, \cup_{i \in S} \mathcal{X}_i)$, where $\mathcal{X}_i$ is the ith party's training data. If parties are to be compensated for helping to solve a machine learning task, it is natural to ask what a *fair* payoff for each parties' effort is. To this end, we consider Shapley values (Shapley 1953), i.e., unique payoffs, studied in game theory for collaborative games, that satisfy certain natural fairness properties discussed later. The Shapley value for characteristic function v and party $i \in \mathbf{M}$ is

$$\psi(v,i) = \sum_{S \subseteq \mathbf{M} \setminus \{i\}} \frac{|S|!(M - |S| - 1)!}{M!} \big(v(S \cup \{i\}) - v(S)\big), \quad (1)$$

i.e., $\psi(v,i)$ quantifies the average marginal contribution of party i wrt all possible subsets of parties. If $v(\mathbf{M}) \neq 1$, Shapley values can be normalized using $1/v(\mathbf{M})$. When clear from the context which characteristic function is used, we use $\psi(i)$. For a fixed characteristic function v, Shapley values $\psi(i)$ are the unique payoffs satisfying properties of *efficiency, symmetry, linearity and null player*. These properties ensure that parties are paid equally for equal contributions and all gains are distributed among the parties.

We will use Shapley values with different characteristic functions in our marketplace: u for deciding which data to use for training customized models (§5) vs. v and w for computing the payoffs for parties in §4.1. This is necessary as Shapley value itself is not robust to replication.

3 A Collaborative Marketplace

In our marketplace, we consider M parties $P_1, \dots, P_M$ which aim to collaborate towards training machine learning models for their tasks. Each party simultaneously takes the role of a seller and a buyer. The ith party has training data $\mathcal{X}_i$ and validation data $\mathcal{V}_i$. The performance of a machine learning model $\mathcal{M}(\mathcal{X})$ trained on some training data $\mathcal{X}$ is evaluated using a performance/gain function $\mathcal{G}(\mathcal{V}_i, \mathcal{M}(\mathcal{X})) \in [0, 1]$.

The goal of the market is to provide party i with a customized model trained on the subset of (or potentially all) datasets of other parties' that best fits its task (based on $\mathcal{V}_i$). At the same time the market uses $\mathcal{X}_i$ to train models of parties where this dataset fits the corresponding task (based on validation data of other parties).

Definition 1 (Marketplace). *A* marketplace *is a tuple* $(\mathcal{P}, PD)$*, where* $\mathcal{P} = (P_1, \dots, P_M)$ *is the list of parties engaging with the market place and PD is the payment division function.*

We consider the following interaction with the data marketplace:

1. Parties $P_1, \dots, P_M$ arrive.
2. The marketplace collects all training data sets $\mathcal{X}_1, \dots, \mathcal{X}_M$ and validation tasks $\mathcal{V}_1, \dots, \mathcal{V}_M$.
3. Every party pays the market a participation fee A_i that is determined proportional to unit increase in personal performance gain $A_i = 1 - \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\mathcal{X}_i))$, i.e., valuation for increasing performance to 100% on their validation data. The market can also set a fixed value for a unit increase $c > 0$ in performance, such that $A_i = c \cdot (1 - \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\mathcal{X}_i)))$.
4. The marketplace trains a machine learning model $\mathcal{M}^i$ for every party i where $\mathcal{M}^i = \mathcal{M}(\mathcal{V}_i, \oplus_{j \in S_i} \mathcal{X}_j)$ and $S_i \subseteq \mathbf{M}$.
5. The model $\mathcal{M}^i$ is shared with party i. Let $a_i = \mathcal{G}(\mathcal{V}_i, \mathcal{M}^i) - \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\mathcal{X}_i))$. Party i receives payoff t_i which depends on the increase in performance on its validation data $\mathcal{V}_i$ (i.e., they receive $A_i - a_i$) and how much its data helps in improving performance of models for other validation tasks, b_i. Hence, in total party i gains (or loses) $(A_i - a_i) + b_i - A_i = b_i - a_i$.

3.1 Desired Properties

In the following we enumerate desired properties for a machine learning marketplace such that participating parties receive *fair* payoffs for their engagement and benefit from participation.

Revenue division and payment Our list of properties is inspired by the "standard axioms of fairness" since they are the de facto method to assess the marginal value of goods (i.e., features in our setting) in a cooperative game (i.e., prediction task in our setting). They include: *Balance:* $\sum b_i = \sum a_i$. *Symmetry:* if two parties i and j enter the market with same training and validation sets then the their payoff, $t_i = t_j$. *Zero element buyer:* if there is a party whose performance does not increase, it should at least get its participation fee back, $t_i = A_i$.

Incentives Party i decides on whether to enter the collaborative market or not and what $\mathcal{V}_i$ and $\mathcal{X}_i$ to contribute. As a result, the marketplace should incentivize the parties to join the market with good training data and honest validation data:

Joining the market: The market should incentivize new parties to join. Our setting incentivizes this as follows. A new party brings a new task to the market, hence, existing parties' data may be useful for this task, increasing their payoff b_i. At the same time, the new party is also bringing new training data which can increase performance of tasks submitted in the existing market, resulting in an increased payoff. Hence a party joining the market can benefit from increased utility.

Validation data: A dishonest party can try to manipulate training and validation data (including their relationship) in order to gain more than it would with its true training and validation datasets. For example, $\mathcal{V}_i$ can be seen as an implicit bid that party i places on the model $\mathcal{M}^i$ it will obtain. There can be a case that $\mathcal{G}(\mathcal{V}', \mathcal{M}^i) - \mathcal{G}(\mathcal{V}', \mathcal{M}(\mathcal{X}_i)) > \mathcal{G}(\mathcal{V}_i, \mathcal{M}^i) - \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\mathcal{X}_i))$.

If it is the case, $\mathcal{M}^i$ has higher utility than what is determined by $\mathcal{V}_i$. To this end, the marketplace needs to ensure that the model $\mathcal{M}^i$ that is returned to the party does not allow for existence of $\mathcal{V}'$ in the current marketplace, incentivizing the party to provide the best $\mathcal{V}_i$ to get the best utility model. We enforce this incentive by training models that are customized for a specific task, i.e., maximizing their accuracy towards a specific task while minimizing their accuracy on all other tasks in the market.

Training data: The market needs to ensure that the payment b_i that party i receives for the use of its data to train other models incentivizes it to provide its best $\mathcal{X}_i$. We note that our market does not have an explicit way for parties to bid or price training data.

Robustness to replication Parties may not behave honestly and may replicate their data and create new replica parties to join the market on their behalf. The market should be robust to replication. That is, a party that replicates its training data should not earn more than it would in the original market. This is a crucial property for any data market since data as compared to physical goods is easily replicable. This problem was already highlighted for a different marketplace setup in (Agarwal, Dahleh, and Sarkar 2019).

Privacy and Integrity Since data can be easily leaked and manipulated compared to physical goods, the market should provide assurance to its participants about the correct handling of their data and model training. At the very least it should ensure that information about the training data $\mathcal{X}_i$ of party i is revealed only to other parties through models for validation tasks where ith training data is useful. The information about $\mathcal{V}_i$ has to be kept secret as well. As we mention in the introduction, these properties can be provided if the market infrastructure is instantiated using Trusted Execution Environments (or enclaves): parties submit their data encrypted and allow only attested (verified) code to decrypt, compute on it, and finally encrypt the models under the encryption keys of the parties who can access these models (i.e., the party who specified the validation data for this model).

4 Market Instantiations

We provide market instantiations for both a single validation task among all the parties and multiple validation tasks.

4.1 Single Validation Task

Let us describe the marketplace for a single validation task $\mathcal{V}$ that all M parties agree on. We slightly abuse the notation by letting $\mathcal{M}_S = \mathcal{M}(\oplus_{k \in S} \mathcal{X}_k)$ be the model trained on data from parties in S for task $\mathcal{V}$. The parties agree on the same marginal payment per increase of the model performance they gain (for example, per percentage increase): if $\mathcal{G}(\mathcal{V}, \mathcal{M}_{\mathbf{M}}) = 1$ then party i pays the amount proportional to $A_i = 1 - \mathcal{G}(\mathcal{V}, \mathcal{M}_i)$. Hence, the largest amount that can be distributed among market participants is $\sum_{i \in \mathbf{M}} A_i$. Recall that, when entering the market the party submits $\mathcal{X}_i$ and fee A_i. After the market completes, i obtains $\mathcal{M}_{\mathbf{M}}$ and payout $b_i \geq 0$.

In §5 we explain how to train $\mathcal{M}_{\mathbf{M}}$ from parties' datasets and here describe how we instantiate the market, compute the payoffs using Shapley and show that the market is robust to replication.

Characteristic Function Our characteristic function captures the value of data to parties in a set S for the task $\mathcal{V}$ as the value of the model trained on the data as well as value the model brings to every party. As a result the characteristic function for this market is defined as

$$v(S) = \underbrace{\mathcal{G}(\mathcal{V}; \mathcal{M}_S)}_{\text{value of the model}} + \sum_{j \in S} \underbrace{\left[\mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \mathcal{G}(\mathcal{V}; \mathcal{M}_j)\right]}_{\text{model value for party } j}$$

This function could be seen as the value of the model trained on the datasets of all parties in S plus marginal gains for each party. Note that for a single party the value of the data is expressed as the value of the model trained on its own training dataset.

Payment Division The total amount, $\mathbf{a}$, that is distributed among the participants depends on the individual gains obtained from the final model. Let $a_i = \mathcal{G}(\mathcal{V}, \mathcal{M}_{\mathbf{M}}) - \mathcal{G}(\mathcal{V}, \mathcal{M}_i)$. Then, $\mathbf{a} = \sum_{i \in \mathbf{M}} a_i$. We use (normalized) Shapley values for characteristic function v to determine the distribution of $\mathbf{a}$ for each party i: $b_i = \mathbf{a} \times \psi(v, i)$. (To simplify the notation we use $\psi(i)$ to denote $\psi(v, i)$.) In total, party i obtains $t_i = (A_i - a_i) + b_i$ where $(A_i - a_i)$ is the return of the original investment if the final model performance is not 1. Hence, party i gains/loses the following amount by participating in the market: $(A_i - a_i) + b_i - A_i = \psi(i)\mathbf{a} - a_i$. Note that i gets $\psi(i)$ portion from each a_j and it pays $1 - \psi(i)$ of a_i to the market.

Properties Our single-task market has the following properties: *Balance:* $\sum(b_i - a_i) = \sum \mathbf{a} \times \psi(i) - \sum a_i = \mathbf{a} \times 1 - \mathbf{a} = 0$. *Symmetry:* This follows from using Shapley value to calculate payout b_i and a_i would be the same for the parties with same data. *Zero element buyer:* If i does not benefit from other parties, that is $\mathcal{G}(\mathcal{V}, \mathcal{M}_i) = \mathcal{G}(\mathcal{V}, \mathcal{M}_{\mathbf{M}})$, then A_i is returned to i since $a_i = 0$.

Robustness-to-replication Recall that a market is robust if a party does not gain a higher payoff in the market with the replicas compared to its gain the original market.

Let us consider the total payoff of i when it replicates itself. Let $\mathbf{a}$ and $\mathbf{a}^R$ be the values of the original and replicated markets. By definition of market value, $\mathbf{a}^R = \mathbf{a} + a_i$. Let t denote i's payoff in the original market and t^R be its total payoff when it replicates itself. Similarly, let $\psi()$ and $\psi^R()$ denote Shapley values before and after replication. $t^R = 2 \times \psi^R(i)(\mathbf{a}^R - a_i) - 2 \times (1 - \psi^R(i))a_i = 2 \times (\psi^R(i)\mathbf{a}^R - a_i)$. In the market where each party is equivalent, that is $\forall k, a_k = a$, replication does not help. Using the symmetry property of Shapley value we can show that $t^R = 0$.

Let us consider the case when the contributions of the parties are different. Recall that: $t = \psi(i)\mathbf{a} - a_i$ and $t^R = 2 \times (\psi^R(i)(\mathbf{a} + a_i) - a_i)$. Replication is useful to party i only if $t < t^R$. If this is the case, then: $\psi(i)\mathbf{a} + a_i < 2\psi^R(i)(\mathbf{a} + a_i)$ should hold.

Condition 1. *Market in §4.1 is robust to replication if* $\psi^R(i) \leq \frac{\psi(i)\mathbf{a}+a_i}{2(\mathbf{a}+a_i)}$.

We show that our single-task market instantiation is robust to replication as long as the gain function $\mathcal{G}$ has the properties outlined in §2. Our proof proceeds as follows. We first derive the payoff that i needs to make in the replicated market in order to break even. We then determine the marginal contribution of party i for new coalitions that are created due to the addition of its replica i' (see Appendix Lemma 1). We then devise the relationship between the Shapley values of party i before and after replication. Finally we show that the Shapley value of i in the new replicated market satisfies the condition of robustness to replication, yielding the following theorem (proof details are provided in the appendix):

Theorem 1. *Single market instantiation in §4.1 is robust to replication as per Condition 1.*

4.2 Multiple Validation tasks

The market setup is similar to the single task market where there is a task from each party, $\mathcal{V}_i$. Similarly when entering the market the party submits $\mathcal{X}_i$ and fee A_i. After the market completes, i obtains $\mathcal{M}^i_{\mathbf{M}}$ and payout $b_i \geq 0$. In §5 we explain how to train an individual $\mathcal{M}^i_{\mathbf{M}}$ from parties' datasets and use the rest of the section to describe how we instantiate the market and compute the payoffs. The *characteristic function* w for the multi-task market is defined as:

$$w(S) = \underbrace{\sum_{i \in \mathbf{M}} \mathcal{G}(\mathcal{V}_i; \mathcal{M}^i_S)}_{\text{value of all models of set } \mathbf{M}} + \sum_{i \in S} \big[\underbrace{\mathcal{G}(\mathcal{V}_i; \mathcal{M}^i_S) - \mathcal{G}(\mathcal{V}_i; \mathcal{M}^i_i)}_{\text{value for party } i} \big]$$

We observe that this is a natural extension from a single validation task: the goal of the market is to achieve better performance on all tasks of the market as well as individually for each party. As we will see it also helps with lowering the impact of replicated parties on the value of the market: an addition of a replicated party will affect only the second part of the value function while a party with new data has a chance to contribute to both parts of the characteristic function.

Payment division and properties The payment division is identical to the one for a single task market in §4.1 except that w is used as the characteristic function when computing the Shapley value.

Since the payment function is the same as in single market, the condition on robustness is also the same and marginal contribution of i to coalitions that are created due to its replication is limited in the same manner as in the single-task market (see Appendix Lemma 3).

Since multi-task market payout is defined in the same manner as the one for single task, the following properties also hold: balance, zero element and symmetry. The main difference is the computation of the characteristic function w.

5 Training over Multi-Party Data

In this section, we describe (1) how to align the training data provided by all parties for training models for different tasks, i.e., select the relevant training data; and (2) how to train a model for party i such that it performs well only on its corresponding task and not other tasks in the market.

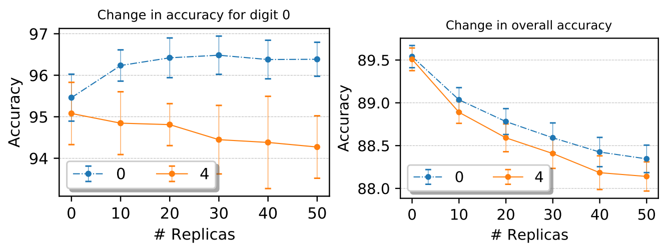
Training Data Selection With multi-party data we need to select the right training data for training each parties' model. To this end, we also make use of Shapley values. Note that this step happens prior to the computation of the payoffs which again involves the computation of Shapley values but for a different characteristic function. For training data selection for party i we follow the following two steps: (1) Compute the Shapley values $\psi(u_i, i)$ for $u_i(S) = \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\cup_{j \in S} \mathcal{X}_j))$. (2) Determine the relevant data for the task of party i as $\mathcal{D}_i = \{j \in \mathbf{M} \mid \psi(u_i, j) \geq \tau\}$, where τ is a threshold that can be used to control the amount of used training data. Note that the above steps can be trivially applied on a sample-level instead of a dataset-level. Computing sample-level Shapley value would typically result in better model performance at increased computational cost for deciding on the training data to use. In the case of a single common validation set, $\mathcal{D}_i = \mathcal{D}_j, \forall i \neq j$.

The characteristic functions used for computing payoffs in §4.2 are computed using only relevant training data. In particular, when computing Shapley values for determining each parties' payoff, we use the shorthand $\mathcal{M}^i_S$ for $\mathcal{M}_{S \cap \mathcal{D}_i}$. Superscript is omitted for single validation task.

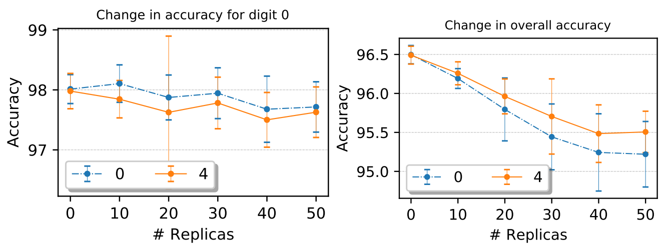
Customized Model Training To minimize usefulness of the model provided to party i after interacting with the market for other parties' validation tasks, we adopt the following strategy for training the model for party i: Party i receives a model trained on the data of parties $\mathcal{D}_i$. Additionally the model that party i receives is optimized for maximizing the loss wrt all $\mathcal{V}_k$ for $k \neq i$, i.e.

$$\begin{aligned} \min_{\theta} \quad & \sum_{j \in \mathbf{M} \setminus \{i\}} \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\oplus_{k \in \mathbf{M}} \mathcal{X}_k)) \qquad (2) \\ \text{s.t.} \quad & \mathcal{G}(\mathcal{V}_i, \mathcal{M}(\oplus_{k \in \mathbf{M}} \mathcal{X}_k)) \geq \mathcal{G}^* - \epsilon, \end{aligned}$$

where $\mathcal{G}^* = \mathcal{G}(\mathcal{V}_i, \mathcal{M}^i_{\mathbf{M}})$ is the best performance achievable on the validation data of party i by standard model training, $\epsilon \geq 0$ is some constant, and θ are parameters of the models, e.g., weights of a neural network. To avoid overfitting to the validation data by using it during model training (we still need to resort to the validation data for pricing) and have a straightforwardly approachable optimization problem, we consider the proxy $\max_\theta \ \mathcal{G}(\cup_{k \in \mathcal{D}_i} \mathcal{X}_k, \mathcal{M}^i_{\mathbf{M}}) - \lambda \sum_{j \neq i} \mathcal{G}(\mathcal{V}_j, \mathcal{M}^i_{\mathbf{M}})$, where $\lambda > 0$ is a hyperparameter. This approach is evaluated in our experiments.



(a) Logistic regression



(b) DNN

Figure 2: Accuracy for MNIST dataset trained using logistic regression (two left plots) and a DNN (two right plots) with replication of digit 0. Replication of other digits showed similar results.

6 Experiments

We implemented a prototype of our marketplace and evaluated it on the task of classifying handwritten digits from the MNIST dataset (LeCun et al. 1998). Here we validate assumptions made in the paper and evaluate the effectiveness of our marketplace setup. The goals of our evaluation are: (i) Understand the effect of replicated training data on the accuracy on validation data; (ii) Measure the (importance) value of training data towards a single validation task; (iii) Calculate the payoffs with our payment division function for training data with and w/o replication and compare them to those of the naive payment division; (iv) Evaluate the effectiveness of an output model customized for a given validation task.

6.1 Effect of Data Replication on ML models

We aim to understand how data replication affects model accuracy and validate the assumption we made in §2. To this end, we consider a collaboration among 10 parties for the validation task of 10K images with all 10 digits. We measure the accuracy on this validation task on models that are trained using the data from all parties. We train a logistic regression model and a single hidden layer (500 neurons) DNN model for 100 epochs with a learning rate of 10^{-2}, Adam optimizer and a value of 10^{-4} for $\ell 2$ regularization.

In the initial setup, each party contributes 1000 images corresponding to a single digit, which are randomly sampled from the training data for this digit. To understand the effect of replication, we compare the accuracy of the model trained in the initial setup with the following replication configurations. The party with digit 0 or 4, respectively, replicates and creates new parties with the same 1000 images as their training data. We vary the number of replicas from 0 to 50. Hence, there will be $50{,}000$ samples corresponding to the digit 0 in the combined training dataset for 50 replicas. All observations are similar for the replication of other digits. In addition to the overall accuracy of the validation task, we looked at the accuracy of the digits 0 and 4 present in the validation dataset.

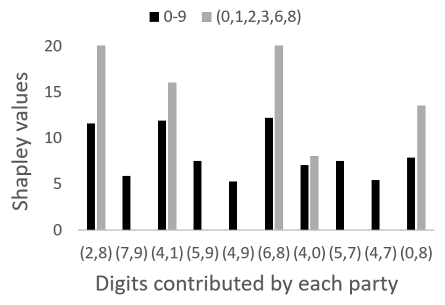
Results. Figure 2 shows results for a logistic regression and a DNN model averaged over 50 runs (random draws of training and validation data, and of the initial network weights). The dotted and the solid lines denote the accuracy over the number of replicas. For logistic regression, the accuracy of classifying digit 0 increases initially with replication of digit 0 and then remains constant. This confirms that replication does not necessarily benefit the accuracy of the replicated digit on a model that is trained on sufficient samples. On the other hand, the accuracy of classifying the digit 0 decreases slightly with replication of 4. This result shows that replication may hurt the accuracy of other validation tasks in the market. Finally, we observe that the accuracy of the overall validation task of 10K images reduces only slightly with the replication of both digits. The results for DNNs show similar trends. Thus, in summary, the assumptions needed for our theoretical results hold approximately in practice.

6.2 Customized Model Training

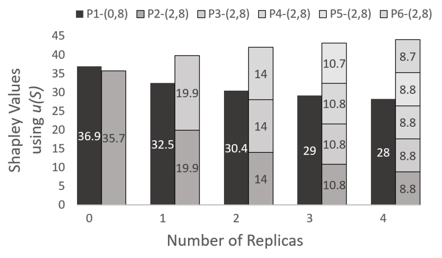
An important component of our marketplace is customized model training, cf. §5, which consists of two steps: data selection and model training. We empirically investigate both steps here.

Training data selection. To understand the usefulness of training data with respect to a validation task, we consider a setting where each party holds 2000 training samples for two randomly selected labels. As validation tasks, we consider classification of all ten digits and classifying digits $(0, 1, 2, 3, 6, 8)$. Figure 3a shows the Shapley values using characteristic function u for both validation tasks. For all the digits in the validation data (label *0-9* in the figure), we observe that the parties with unique labels such as 1, 2 and 6 have higher Shapley values indicating higher utility of their training data. For the validation task on digits (0,1,2,3,6,8) we observe that the parties which do not contribute any of these digits have zero Shapley values. This confirms that Shapley values are well suited for selecting training data for task-specific model training.

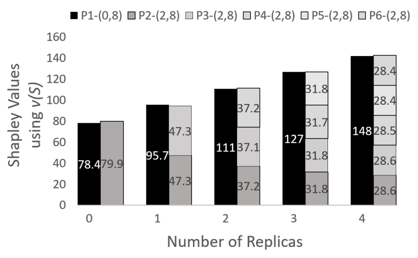
Model training. To evaluate the customized model training we considered a setting with ten parties in which each party holds data for two different digits s.t. party j has data for labels $(j, (j+1) \mod 10)$ for $j \in \{0, 1, \ldots, 9\}$. For party j the training data consists of all training samples from the MNIST dataset with either of its two labels. The same holds for its validation data with respect to the validation data from MNIST. In Figure 4 we see results for applying the customized model training approach when training a lo-



(a)



(b)



(c)

Figure 3: (a) Shapley values for parties contributing training data for validation tasks with all ten digits and only digits $(0, 1, 2, 3, 6, 8)$. Figures (b) and (c): Shapley values on validation dataset of digits (0,2,8) with replicating parties for characteristic function u (accuracy) in (b) and our proposed characteristic function v (§4.1) in (c).

gistic regression model for the validation task of each party. We observe that although the combined training points consist of all the digits in MNIST, the model trained with our approach is useful only for the specific validation task and has mediocre performance on (other) validation tasks with overlapping labels. That is, it is not useful to classify digits from the remaining labels outside the specific validation task.

6.3 Payoffs in Single Task Marketplace

We now empirically validate that our proposed characteristic function v in §4.1 for computing payoffs is indeed robust to replication. We also compare it to the case where Shapley values are computed for characteristic function u (model accuracy), cf. §5, which is not robust to replication. Here, we consider two parties: Honest party P_1 with digits (0,8), and replicating party P_2 with digits (2,8). P_2 creates replicas P_3 to P_6 with the same dataset. The single validation task is classification of digits 0,2,8. The parties combine their data and train a logistic regression model.

Figure 3 shows Shapley values of our results as number of replicas increases. We observe that the combined Shapley value of the replicating parties increases with the number of replicas while that of the honest party P_1 decreases for u (Figure 3b) when using Shapley values with accuracy as the characteristic function. In contrast, Figure 3c shows that when Shapley values are instantiated with our new characteristic function, Shapley values of P_1 and the combined values for the replicating parties are equal. This verifies that replication does not benefit the replicating party when using our single-task collaborative market instantiation.

7 Related Work

Multiple lines of work are related to our proposed collaborative machine learning markets framework. We organized it into work on using *Shapley values for valuating data* for model training and *machine learning marketplaces*. Also general work on game theory and mechanism design is related to our work but not discussed here in detail, cf. (Nisan et al. 2007) for an overview.

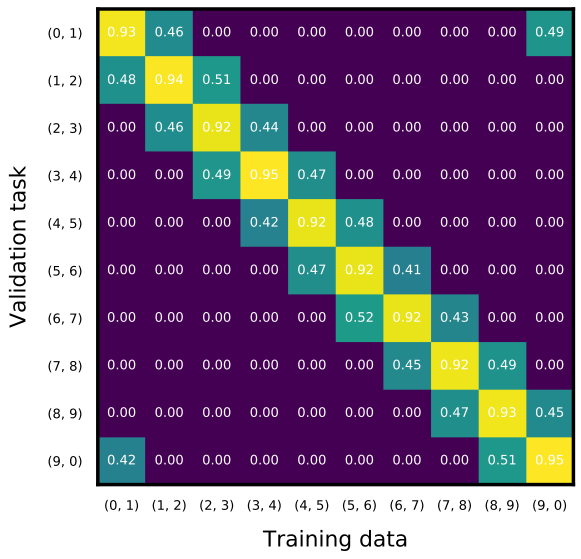


Figure 4: Classification accuracy of the customized models for different validation tasks.

Shapley values for valuating data. Prior work has proposed the use of Shapley values for characteristic function u to valuate data in machine learning settings. Most of these works focus on the problem of efficiently approximating Shapley values. Closest to ours, Agarwal, Dahleh, and Sarkar (2019) uses Shapley values to compute payoffs in non-collaborative marketplace. Before that, Datta et al. (2016) proposed the use of Shapley values to quantify feature importance for classification problems, and proposed sampling based approximations of Shapley values. Ghorbani et al. (2019) show that Shapley values are good metric to quantify the usefulness of data for machine learning tasks. They show that valuating data for model training based on Shapley value is better than using leave-one-out cross-validation, and propose sampling and gradient based approaches for approximating Shapley values. Also Jia et al. (2019) propose several sampling based approaches and the use of influence functions (Koh and Liang 2017) to reduce the computational cost of computing Shapley values.

Machine learning marketplaces. The closest to our paper is work by Agarwal et al. (2019) who study an algorithmic approach for data marketplaces. They propose a general marketplace setup which makes use of Shapley values for computing payoffs. The marketplace sets prices for data in an online fashion and matches sellers to buyers. They highlight the problem of data replication in combination with Shapley values and propose to overcome it heuristically by downweighing Shapley values according to data similarity. Critically, their approach also reduces payoffs for honest, i.e., non-replicating, parties. In contrast, we achieve robustness-to-replication naturally through the multi-party setting in combination with novel characteristic functions. Furthermore, in our marketplace no data is given to party, avoiding further reselling of the data while in (Agarwal, Dahleh, and Sarkar 2019) this is possible. Other related work tries to price machine learning models instead of data directly (Chen, Koutris, and Kumar 2018). In that work, a broker forms the interface between data sellers and model buyers. Their main focus is to ensure affordability of models for all buyers by adjusting the model's performance through noise-injection and prevention of arbitrage. Abernethy and Frongillo (2011) study incentive mechanisms in the setting where solving a task (e.g., ML prediction) is done in a collaborative decentralized manner with parties contributing towards solving the task with their expertise (e.g., improvements to a classifier), as opposed to contributing data. Marketplaces that treat data similar to classical goods are available at (Big Data Exchange ; QLik Data Market).

8 Future Work

We consider the following extensions of our work: *Performance:* Improve the time to select training data using optimizations for evaluation of Shapley values such as sampling or influence functions (Koh and Liang 2017; Jia et al. 2019; Ghorbani and Zou 2019). *Data Privacy:* Use differential privacy techniques to strengthen privacy of training data in the customized models, as a result mitigating model inversion or membership inference attacks on the output model (Shokri et al. 2017; Fredrikson, Jha, and Ristenpart 2015). *Iterative Market:* Consider marketplace properties when the parties interact with the market over multiple rounds. Iterative markets would allow parties to learn additional information about the market such as the type of validation tasks and training data available, which might give advantage to adversarial parties.

References

Abernethy, J. D., and Frongillo, R. M. 2011. A collaborative mechanism for crowdsourcing prediction problems. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, 2600–2608.

Agarwal, A.; Dahleh, M.; and Sarkar, T. 2019. A marketplace for data: An algorithmic solution. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, 701–726. ACM.

Big Data Exchange. Big Data Exchange. htt://www.bigdataexchange.com.

Chen, L.; Koutris, P.; and Kumar, A. 2018. Model-based pricing for machine learning in a data marketplace. *arXiv preprint arXiv:1805.11450*.

Datta, A.; Sen, S.; and Zick, Y. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, 598–617. IEEE.

Fredrikson, M.; Jha, S.; and Ristenpart, T. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 1322–1333. ACM.

Ghorbani, A., and Zou, J. 2019. Data Shapley: Equitable valuation of data for machine learning. *arXiv preprint arXiv:1904.02868*.

Hoekstra, M.; Lal, R.; Pappachan, P.; Rozas, C.; Phegade, V.; and del Cuvillo, J. 2013. Using innovative instructions to create trustworthy software solutions. In *Workshop on Hardware and Architectural Support for Security and Privacy (HASP)*.

Jia, R.; Dao, D.; Wang, B.; Hubis, F. A.; Hynes, N.; Gurel, N. M.; Li, B.; Zhang, C.; Song, D.; and Spanos, C. 2019. Towards efficient data valuation based on the Shapley value. *arXiv preprint arXiv:1902.10275*.

Koh, P. W., and Liang, P. 2017. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 1885–1894.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Nisan, N.; Roughgarden, T.; Tardos, E.; and Vazirani, V. V. 2007. *Algorithmic game theory*. Cambridge university press.

Ohrimenko, O.; Schuster, F.; Fournet, C.; Mehta, A.; Nowozin, S.; Vaswani, K.; and Costa, M. 2016. Oblivious multi-party machine learning on trusted processors. In *USENIX Security Symposium*.

QLik Data Market. QLik Data Market. http://www.qlik.com/us/products/qlik-data-market.

Shapley, L. S. 1953. A value for n-person games. *Contributions to the Theory of Games* 2(28):307–317.

Shokri, R.; Stronati, M.; Song, C.; and Shmatikov, V. 2017. Membership inference attacks against machine learning models. In *Security and Privacy (SP), 2017 IEEE Symposium on*, 3–18. IEEE.

A Proof details

We show that our single-task market instantiation is robust to replication. Recall that a market is robust if a party does not gain a higher payoff in the market with the replicas compared to its gain the original market. Let M be the number of parties in the market before replication. Let i be the party that replicates itself. Our proof proceeds as follows. We first derive the payoff that i needs to make in the replicated market in order to break even (Condition 1) and estimate what is the marginal contribution of party i for new coalitions that are created due to the addition of its replica i' (Lemma 1). We then devise the relationship between the Shapley values of party i before and after replication. (Lemma 2). Finally, we show that, under certain conditions on the gain function $\mathcal{G}$, ψ^R of the new market satisfies the condition on robustness to replication (Theorem 1).

Our main argument depends on the following lemmas where we capture the influence of the replicated party on the computation of the Shapley value $\psi^R(i)$.

Lemma 1. *Let i and i' be replicas and let $i' \in S$ then $v(S \cup \{i\}) - v(S) \leq a_i$ for any set S.*

Proof. Let us expand $v(S \cup \{i\})$ using the property of the gain function that states that replicated does not change its value (i.e., $\mathcal{G}(\mathcal{V}; \mathcal{M}_{S\cup\{i\}}) = \mathcal{G}(\mathcal{V}; \mathcal{M}_S)$):

$$\begin{aligned} v(S \cup \{i\}) = \quad & \mathcal{G}(\mathcal{V}; \mathcal{M}_{S\cup\{i\}}) + \sum_{j\in S\cup\{i\}} \left[\mathcal{G}(\mathcal{V}; \mathcal{M}_{S\cup\{i\}}) - \mathcal{G}(\mathcal{V}; \mathcal{M}_j)\right] = \\ & \mathcal{G}(\mathcal{V}; \mathcal{M}_S) + \sum_{j\in S\cup\{i\}} \left[\mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \mathcal{G}(\mathcal{V}; \mathcal{M}_j)\right] \end{aligned}$$

Then

$$\begin{aligned} v(S \cup \{i\}) \quad - \quad & v(S) \\ = \quad & \mathcal{G}(\mathcal{V}; \mathcal{M}_S) + \sum_{j\in S\cup\{i\}} \left[\mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \mathcal{G}(\mathcal{V}; \mathcal{M}_j)\right] \\ & -\mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \sum_{j\in S} \left[\mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \mathcal{G}(\mathcal{V}; \mathcal{M}_j)\right] \\ = \quad & \mathcal{G}(\mathcal{V}; \mathcal{M}_S) - \mathcal{G}(\mathcal{V}; \mathcal{M}_i) \leq a_i \end{aligned}$$

□

Lemma 2. *Let ϕ and ϕ^R be the Shapley values of the original market and the market where party i replicates itself. Then, if $\mathcal{G}$ used in the characteristic function v is monotonic and supermodular,*

$$\psi^R(i) \leq \frac{\psi(i)v(\mathbf{M})}{2(v(\mathbf{M}) + a_i)} + \frac{a_i}{2(v(\mathbf{M}) + a_i)} = \frac{\psi(i)v(\mathbf{M}) + a_i}{2(v(\mathbf{M}) + a_i)}$$

Proof. Let us express $\psi^R(i)$ in terms of ψ. Compare the coalition subsets S between the two values. All the new subsets in ψ^R will contain the replica of i. As shown in Lemma 1, for such subsets $v(S \cup i) - v(S) = a_i$. The other subsets are the same as in ψ except that their value towards the overall Shapley value decreases. Let $\psi^R(i) = \psi^R_{\textsf{old}}(i) + \psi^R_{\textsf{new}}(i)$ where $\psi^R_{\textsf{old}}$ is the value brought from old (coalition) subsets and $\psi^R_{\textsf{new}}$ from the new ones.

Computing $\psi^R_{\textsf{old}}(i)$: $\psi^R_{\textsf{old}}(i) \leq \frac{\psi(i)v(\mathbf{M})}{2(v(\mathbf{M})+a_i)}$ The (normalized) Shapley value of the subsets that existed before replication for both markets is:

$$\psi(i) = \frac{1}{v(\mathbf{M})} \sum_{S\subseteq \mathbf{M}\setminus\{i\}} \frac{|S|!(M - |S| - 1)!}{M!} \big(v(S\cup\{i\}) - v(S)\big)$$

$$\psi^R_{\textsf{old}}(i) = \frac{1}{v(\mathbf{M}^R)} \sum_{S\subseteq \mathbf{M}\setminus\{i\}} \frac{|S|!(M - |S|)!}{(M + 1)!} \big(v(S\cup\{i\}) - v(S)\big)$$

where $v(\mathbf{M}^R) = v(\mathbf{M}) + a_i$. We need to show that

$$\psi^R_{\textsf{old}}(i) = \frac{1}{v(\mathbf{M}) + a_i} \sum_{S\subseteq \mathbf{M}\setminus\{i\}} \frac{|S|!(M - |S|)!}{(M + 1)!} \big(v(S\cup\{i\}) - v(S)\big) \leq \frac{\psi(i)v(\mathbf{M})}{2(v(\mathbf{M}) + a_i)} \tag{3}$$

Simplifying the inequality and substituting $\psi(v)$:

$$\begin{aligned}
\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|)!}{(M+1)!}\big(v(S\cup\{i\})-v(S)\big) &\leq \frac{\psi(i)v(\mathbf{M})}{2}\\
\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|)!}{(M+1)!}\big(v(S\cup\{i\})-v(S)\big) &\leq \frac{1}{v(\mathbf{M})}\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|-1)!}{M!}\big(v(S\cup\{i\})-v(S)\big)\frac{v(\mathbf{M})}{2}\\
\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|)!}{(M+1)!}\big(v(S\cup\{i\})-v(S)\big) &\leq \frac{1}{2}\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|-1)!}{M!}\big(v(S\cup\{i\})-v(S)\big)\\
\sum_{S\subseteq\mathbf{M}\setminus\{i\}} \frac{|S|!(M-|S|)!}{(M+1)}\big(v(S\cup\{i\})-v(S)\big) &\leq \frac{1}{2}\sum_{S\subseteq\mathbf{M}\setminus\{i\}} |S|!(M-|S|-1)!\big(v(S\cup\{i\})-v(S)\big)\\
\sum_{S\subseteq\mathbf{M}\setminus\{i\}} (M-2|S|-1)\big(v(S\cup\{i\})-v(S)\big) &\leq 0
\end{aligned}$$

Note that coefficients $M-2|S|-1$ are symmetric for sets $|R| < \lfloor M/2 \rfloor$ and $|Q| = M-|R|-1$: $M-2|R|-1 = -(M-2|Q|-1)$. Hence, as long as $v(R\cup\{i\})-v(R) \leq v(Q\cup\{i\})-v(Q)$, the inequality is satisfied.

Recall that the characteristic function is defined for the single-task market is defined as $v(S\cup\{i\}) = \mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) + \sum_{j\in S\cup\{i\}}\left[\mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_j)\right]$. Then $v(S\cup\{i\})-v(S)$ is:

$$\begin{aligned}
\mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) + \sum_{j\in S\cup\{i\}}\left[\mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_j)\right] - \mathcal{G}(\mathcal{V};\mathcal{M}_S) - \sum_{j\in S}\left[\mathcal{G}(\mathcal{V};\mathcal{M}_S) - \mathcal{G}(\mathcal{V};\mathcal{M}_j)\right] &=\\
(|S|+1)[\mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_S)] + \left[\mathcal{G}(\mathcal{V};\mathcal{M}_{S\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_i)\right]&
\end{aligned}$$

Hence, $v(R\cup\{i\})-v(R) \leq v(Q\cup\{i\})-v(Q)$ can be written as follows, using the fact that $\mathcal{G}$ is monotonic:

$$(|R|+1)[\mathcal{G}(\mathcal{V};\mathcal{M}_{R\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_R)] \leq (|Q|+1)[\mathcal{G}(\mathcal{V};\mathcal{M}_{Q\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_Q)]$$
$$(|R|+1)[\mathcal{G}(\mathcal{V};\mathcal{M}_{R\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_R)] \leq (M-|R|)[\mathcal{G}(\mathcal{V};\mathcal{M}_{Q\cup\{i\}}) - \mathcal{G}(\mathcal{V};\mathcal{M}_Q)]$$

The inequality holds, as long $\mathcal{G}$ is supermodular since $|R|+1 \leq M-|R|$ for $|R| < M/2$.

Computing $\psi^R_{\text{new}}(i)$: Let us now show that $\psi^R_{\text{new}}(i) = \frac{a_i}{2v(\mathbf{M}^R)}$, where we denote the number of coalitions of size s without party i in the market with M parties using, $n_{s,M}$:

$$n_{s,M} = \binom{M-1}{s}$$

Each new coalition in the new marketplace contains i'. Hence, using Lemma 1, marginal contribution of i in these new coalitions will be limited to a_i. The number of such new subsets is

$$n_{M,M+1} + \sum_{s=1}^{M-1} n_{s,M+1} - n_{s,M} = 1 + \sum_{s=1}^{M-1} n_{s,M+1} - n_{s,M}$$

Overall these subsets bring the following value to $\psi^R(i)$:

$$\begin{aligned}
\psi^R_{\text{new}} = a_i\frac{1}{M+1}\frac{1}{v(\mathbf{M}^R)}\left(1+\sum_{s=1}^{M-1}\frac{n_{s,M+1}-n_{s,M}}{n_{s,M+1}}\right) &=\\
a_i\frac{1}{M+1}\frac{1}{v(\mathbf{M}^R)}\left(1+\frac{M-1}{2}\right) = a_i\frac{1}{M+1}\frac{1}{v(\mathbf{M}^R)}\frac{M+1}{2} &= \frac{a_i}{2v(\mathbf{M}^R)} \qquad (4)
\end{aligned}$$

(see Claim 1 for the expansion of the subset combinations).

Then substituting derivations 3 and 4 into $\psi^R(i)$:

$$\psi^R(i) \leq \frac{\psi(i)v(\mathbf{M})}{2(v(\mathbf{M})+a_i)} + \frac{a_i}{2(v(\mathbf{M})+a_i)} = \frac{\psi(i)v(\mathbf{M})+a_i}{2(v(\mathbf{M})+a_i)}$$

□

Corollary 1. *If $a_i = 0$, then $\psi^R(i) \leq \frac{\psi(i)v(\mathbf{M})}{2v(\mathbf{M})} = \frac{\psi(i)}{2}$.*

Theorem 1. *Single market instantiation in §4.1 is robust to replication as per Condition 1.*

Proof. Let $\mathbf{a} = a_i + a_j$ where a_j refers to the sum of a_k for all other parties in the market except i and i'. Let us prove the contrary that the market is not robust to replication and hence Condition 1 is false. Substituting ψ^R from Lemma 2 in Condition 1 and rearranging we obtain

$$\psi(i)\frac{a_i + a_j}{2(a_i + a_j + a_i)} + \frac{a_i}{2(a_i + a_j + a_i)} < \psi(i)\frac{v(\mathbf{M})}{2(v(\mathbf{M}) + a_i)} + \frac{a_i}{2(v(\mathbf{M}) + a_i)}$$

$$\frac{\psi(i)(a_i + a_j) + a_i}{(a_i + a_j + a_i)} < \frac{\psi(i)v(\mathbf{M}) + a_i}{v(\mathbf{M}) + a_i} \tag{5}$$

Recall that $v(\mathbf{M}) = \mathcal{G}(\mathcal{V}; \mathcal{M}_{\mathbf{M}}) + \sum_{l \in \mathbf{M}} a_l = \mathcal{G}(\mathcal{V}; \mathcal{M}_{\mathbf{M}}) + a_i + a_j$. Note that $a_l < \mathcal{G}(\mathcal{V}; \mathcal{M}_{\mathbf{M}}) \leq 1, \forall l$ since $a_l = \mathcal{G}(\mathcal{V}; \mathcal{M}_{\mathbf{M}}) - \mathcal{G}(\mathcal{V}; \mathcal{M}_i)$. Let $k = \arg\max a_l$.

Replacing $v(\mathbf{M})$ in Equation 5:

$$\frac{\psi(i)(a_i + a_j) + a_i}{(a_i + a_j + a_i)} < \frac{\psi(i)(a_k + a_i + a_j) + a_i}{a_k + (a_i + a_j + a_i)}$$

$$\frac{a_k + a_i + a_j + a_i}{a_i + a_j + a_i} < \frac{\psi(i)(a_k + a_i + a_j) + a_i}{\psi(i)(a_i + a_j) + a_i}$$

Let $A = a_k + a_i + a_j$, $B = a_i + a_j$, $x = 1/\psi(i) \geq 1$ Then

$$\frac{A + a_i}{B + a_i} < \frac{\psi(i)A + a_i}{\psi(i)B + a_i}$$

$$\frac{A + a_i}{B + a_i} < \frac{A + xa_i}{B + xa_i}$$

Let $p = \frac{A + a_i}{B + a_i}$, then $(A + a_i) = p(B + a_i)$. Since $A > B$, $a_i \geq 0$, then $p \geq 1$. We need to show that

$$p < \frac{A + xa_i}{B + xa_i} = \frac{p(B + a_i) - a_i + xa_i}{B + xa_i}$$

$$pB + pxa_i < pB + pa_i - a_i + xa_i$$

$$pxa_i < pa_i - a_i + xa_i$$

$$px < p + x - 1$$

However, $px \geq p + x - 1$ for $p, x \geq 1$ and we arrive to contradiction. □

Lemma 3. *Let i and i' be replicas and let $i' \in S$ then $w(S \cup \{i\}) - w(S) \leq a_i$ for any set S.*

Proof. Consider the value $w(S \cup \{i\}) - w(S)$ for replicated party i.

$$w(S \cup i) = \sum_{j \in M} \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) + \sum_{j \in S,i} [\mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) - \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_j)]$$

$$w(S \cup i, i') = \sum_{j \in M} \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i,i')}) + \sum_{j \in S,i,i'} [\mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i,i')}) - \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_j)] =$$

$$\sum_{j \in M} \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) + \sum_{j \in S,i,i'} [\mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) - \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_j)]$$

Then

$$w(S \cup i, i') - w(S \cup i) =$$

$$\sum_{j \in M} \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) + \sum_{j \in S,i,i'} [\mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) - \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_j)]$$

$$- \sum_{j \in M} \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) - \sum_{j \in S,i} [\mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_{(S,i)}) - \mathcal{G}(\mathcal{V}_j; \mathcal{M}^j_j)] =$$

$$[\mathcal{G}(\mathcal{V}_i; \mathcal{M}^j_{(S,i)}) - \mathcal{G}(\mathcal{V}_i; \mathcal{M}^j_i)] = a_i$$

□

Claim 1. $\sum_{s=1}^{M-1} \frac{n_{s,M+1}-n_{s,M}}{n_{s,M+1}} = \frac{1}{M}\frac{(M-1)M}{2} = \frac{M-1}{2}$.

Proof.

$$\frac{(n_{s,M+1}-n_{s,M})}{n_{s,M+1}} = \frac{\binom{M}{s}-\binom{M-1}{s}}{\binom{M}{s}} = 1-\frac{\binom{M-1}{s}}{\binom{M}{s}} = 1-\frac{(M-1)!s!(M-s)!}{s!(M-s-1)!M!} = 1-\frac{M-s}{M} = \frac{s}{M}$$

and

$$\sum_{s=1}^{M-1}\frac{\binom{M}{s}-\binom{M-1}{s})}{\binom{M}{s}}) = \sum_{s=1}^{M-1}\frac{s}{M} = \frac{1}{M}\frac{(M-1)M}{2} = \frac{M-1}{2}$$

□