A Marketplace for Data: An Algorithmic Solution

ANISH AGARWAL, Massachusetts Institute of Technology
MUNTHER DAHLEH, Massachusetts Institute of Technology
TUHIN SARKAR, Massachusetts Institute of Technology

In this work, we aim to design a data marketplace; a robust real-time matching mechanism to efficiently buy and sell training data for Machine Learning tasks. While the monetization of data and pre-trained models is an essential focus of industry today, there does not exist a market mechanism to price training data and match buyers to sellers while still addressing the associated (computational and other) complexity. The challenge in creating such a market stems from the very nature of data as an asset: (i) it is freely replicable; (ii) its value is inherently combinatorial due to correlation with signal in other data; (iii) prediction tasks and the value of accuracy vary widely; (iv) usefulness of training data is difficult to verify a priori without first applying it to a prediction task. As our main contributions we: (i) propose a mathematical model for a two-sided data market and formally define the key associated challenges; (ii) construct algorithms for such a market to function and analyze how they meet the challenges defined. We highlight two technical contributions: (i) a new notion of "fairness" required for cooperative games with freely replicable goods; (ii) a truthful, zero regret mechanism to auction a class of combinatorial goods based on utilizing Myerson's payment function and the Multiplicative Weights algorithm. These might be of independent interest.

CCS Concepts: • **Theory of computation → Algorithmic game theory and mechanism design**.

Additional Key Words and Phrases: Data Marketplaces, Value of Data, Shapley Value, Online Combinatorial Auctions

ACM Reference Format:
Anish Agarwal, Munther Dahleh, and Tuhin Sarkar. 2019. A Marketplace for Data: An Algorithmic Solution. In *ACM EC '19: ACM Conference on Economics and Computation (EC '19), June 24–28, 2019, Phoenix, AZ, USA.* ACM, New York, NY, USA, 27 pages. https://doi.org/10.1145/3328526.3329589

EC '19, June 24–28, 2019, Phoenix, AZ, USA

ACM ISBN 978-1-4503-6792-9/19/06…$15.00
https://doi.org/10.1145/3328526.3329589

1 INTRODUCTION

A Data Marketplace - Why Now? Machine Learning (ML) is starting to take the place in industry that "Information Technology" had in the late 1990s: businesses of all sizes and in all sectors, are recognizing the necessity to develop predictive capabilities for continued profitability. To be effective, ML algorithms rely on high-quality training data – however, obtaining relevant training data can be very difficult for firms to do themselves, especially those early in their path towards incorporating ML into their operations. This problem is only further exacerbated, as businesses increasingly need to solve these prediction problems in real-time (e.g. a ride-share company setting prices, retailers/restaurants sending targeted coupons to clear inventory), which means data gets "stale" quickly. Therefore, we aim to design a data marketplace – a real-time market structure for the buying and selling of training data for ML.

What makes Data a Unique Asset? (i) Data can be replicated at zero marginal cost – in general, modeling digital goods (i.e., freely replicated goods) as assets is a relatively new problem (cf. [2]). (ii) Its value to a firm is inherently combinatorial i.e., the value of a particular dataset to a firm depends on what other (potentially correlated) datasets are available - hence, it is not obvious how to set prices for a collection of datasets with correlated signals. (iii) Prediction tasks and the value of an increase in prediction accuracy vary widely between different firms - for example, a 10% increase in prediction accuracy has very different value for a hedge fund maximizing profit compared to a logistics company trying to decrease inventory costs. (iv) The authenticity and usefulness of data is difficult to verify a priori without first applying it to a prediction task - continuing the example from above, a particular dataset of say satellite images may be very predictive for a specific financial instrument but may have little use in forecasting demand for a logistics company (and this is infeasible to check beforehand).

Why Current Online Markets Do Not Suffice? Arguably, the most relevant real-time markets to compare against are: (i) online ad auctions (cf. [35]); (ii) prediction markets (cf. [36]). Traditionally, in these markets (e.g. online ad auctions) the commodity (e.g. ad-space) is not a replicable good and buyers have a strong prior (e.g. historical click-through-rate) on the value of the good sold (cf. [23, 37]). In contrast for a data market, *it is infeasible for a firm to make bids on specific datasets as it is unlikely they have a prior on its usefulness.* Secondly, it is infeasible to run something akin to a second price auction (and variants thereof) since data is freely replicable (unless a seller artificially restricts the number of replications, which may be suboptimal for maximizing revenue). This problem only gets significantly more complicated due to the combinatorial nature of data. *Thus any market which matches prediction tasks and training features on sale, needs to do so based on which datasets collectively are, empirically the most predictive and "cheap" enough for a buyer.* This is a capability online ad markets and prediction markets do not currently have. See Section 1.3 for a more thorough comparison with online ad and prediction markets.

1.1 Overview of Contributions

Mathematical Model of Two-Sided Data Market. Formal Definition of Key Challenges. As the main contribution of this paper, we propose a mathematical model of a system design for a data marketplace; we rigorously parametrize the participants of our proposed market - the buyers, the sellers and the marketplace itself (Sections 2.1, 2.2, 2.3) - and the mechanism by which they interact (Section 2.4). *This is a new formulation, which lays out a possible architecture for a data marketplace, and takes into account some of the key properties that make data unique*; it is freely replicable, it is combinatorial (i.e., features have overlapping information), buyers having no prior on usefulness of individual datasets on sale and the prediction tasks of buyers vary widely. In Section 3, we study the key challenges for such a marketplace to robustly function in real-time,

which include: (i) incentivizing buyers to report their internal valuations truthfully; (ii) updating the price for a collection of correlated datasets such that revenue is maximized over time; (iii) dividing the generated revenue "fairly" among the training features so sellers get paid for their marginal contribution; (iv) constructing algorithms that achieve all of the above and are efficiently computable (e.g. run in polynomial time in the parameters of the marketplace)?

Algorithmic Solution. Theoretical Guarantees. In Section 4, we construct algorithms for the various functions the marketplace must carry out: (i) allocate training features to and collect revenue from buyers; (ii) update the price at which the features are sold; (iii) distribute revenue amongst the data sellers. In Section 5, we prove these particular constructions do indeed satisfy the desirable marketplace properties laid out in Section 3. We highlight two technical contributions: (i) Property 3.4, a novel notion of "fairness" required for cooperative games with freely replicable goods, which generalizes the standard notion of Shapley fairness; (ii) a truthful, zero regret mechanism for auctioning a particular class of combinatorial goods based on utilizing Myerson's payment function (cf. [29]) and the Multiplicative Weights algorithm (cf. [3]). These might be of independent interest.

1.2 Motivating Example from Inventory Optimization

We begin with an example from inventory optimization to help build intuition for our proposed architecture for a data marketplace (see Section 2 for a mathematical formalization of these dynamics). We refer back to this example throughout the paper as we introduce various notations and algorithmic constructions.

Inventory Optimization Example: Imagine data sellers are retail stores selling anonymized minute-by-minute foot-traffic data streams into a marketplace and data buyers are logistics companies who want features (i.e. various time series) that best forecast future inventory demand. In such a setting, even though a logistics company clearly knows there is predictive value in these data streams on sale, it is reasonable to assume that the company does not have a good prior on what collection of foot-traffic data streams are most predictive for demand forecasting, and within their budget. Thus, practically speaking, such a logistics company cannot make accuracy, individual bids for each data stream (this is even without accounting for the significant additional complication arising from the overlap in signal i.e., the correlation that invariably will exist between the foot-traffic data streams of the various retail stores).

Instead what a logistics company does realistically have access to is a well-defined cost model for not predicting demand well (cf. [21, 24]) - e.g., "10% over/under-capacity costs $10,000 per week". Hence it can make a bid into a data market of what a marginal increase in forecasting accuracy of inventory demand is worth to it - e.g. "willing to pay $1000 for a percentage increase in demand forecasting accuracy from the previous week".

In such a setting, the marketplace we design performs the following steps:

(1) The logistics company supplies a prediction task (i.e., a time series of historical inventory demand) and a bid signifying what a marginal increase in accuracy is worth to it
(2) The market supplies the logistics company with foot-traffic data streams that are "cheap" enough as a function of the bid made and the current price of the data streams
(3) A ML model is fit using the foot-traffic data streams sold and the historical inventory demand
(4) Revenue is collected based *only on the increased accuracy in forecasting inventory demand* [1]
(5) Revenue is divided amongst all the retail stores who provided foot-traffic data
(6) The price associated with the foot-traffic data streams is then updated

[1]Model evaluation could potentially be done on an out-of-sample test set or based on actual prediction performance on future unseen demand

What we find especially exciting about this example is that it can easily be adapted to a variety of commercial settings. Examples include: (i) hedge funds sourcing alternative data to predict certain financial instruments; (ii) utility companies sourcing electric vehicle charging data to forecast electricity demand during peak hours; (iii) retailers sourcing online social media data to predict customer churn.

Thus we believe the *dynamic described above can be a natural, scalable way for businesses to source data for ML tasks, without knowing a priori what combination of data sources will be useful.*

1.3 Literature Review

Auction design and Online Matching. In this work, we are specifically concerned with online auction design in a two–sided market. There is a rich body of literature on optimal auction design theory initiated by [29], [31]. We highlight some representative papers. In [32] and [10], platform design and the function of general intermediary service providers for such markets is studied; in [17], advertising auctions are studied; in the context of ride–sharing such as those in Uber and Lyft, the efficiency of matching in [12] and optimal pricing in [8] are studied. An extensive survey on online matching, in the context of ad allocation, can be found in [27]. These paper generally focus on the tradeoff between inducing participation and extracting rent from both sides. Intrinsic to such models is the assumption that the value of the goods or service being sold is known partially or in expectation. This is the key issue in applying these platform designs for a data marketplace; as stated earlier, it is unrealistic for a buyer to know the value of the various data streams being sold a priori (recall the inventory example in Section 1.2 in which a logistic company cannot realistically make accurate bids on separate data streams or bundles of data streams). Secondly these prior works do no take into account the freely replicable, combinatorial nature of a good such as data.

Online Ad Auctions. See [35] for a detailed overview. There are two key issues with online ad markets that make it infeasible for data. Firstly, ad-space is not a replicable good i.e., for any particular user on an online platform, at any instant in time, only a single ad can be shown in an ad-space. Thus an *online ad market does not need to do any "price discovery"* - it simply allocates the ad-space to the highest bidder; to ensure truthfulness, the highest bidder pays the second highest bid i.e., the celebrated second price auction (and variants thereof). In contrast, for a freely replicable good such as data, a second price auction does not suffice (unless a seller artificially restricts a dataset to be replicated a fixed number of times, which may be suboptimal for maximizing revenue). Secondly, buyers of online ad-space have a strong prior on the value of a particular ad-space - for example, a pharmaceutical company has access to historical click-through rates (CTR) for when a user searches for the word "cancer ". So it is possible for firms to make separate bids for different ad-spaces based on access to past performance information such as CTR (cf. [23, 37]). In contrast, since prediction tasks vary so greatly, past success of a specific training feature on sale has little meaning for a firm trying to source training data for its highly specific ML task; again, making it is infeasible for a firm to make bids on specific datasets as they have no prior on its usefulness.

Prediction Markets. Such markets are a recent phenomenon and have generated a lot of interest, rightly so. See [36] for a detailed overview. Typically in such markets, there is a discrete random variable, W, that a firm wants to accurately predict. The market executes as follows: (i) "experts" sell probability distributions Δ_W i.e., predictions on the chance of each outcome; (ii) the true outcome, w, is observed; (iii) the market pays the "experts" based on Δ_W and w. In such literature, payment functions based on the Kullback–Leibler divergence are commonly utilized, as they incentivize "experts" to be truthful (cf. [19]). Despite similarities, prediction markets remain infeasible for data as "experts" have to explicitly choose which tasks to make specific predictions for. In contrast, it is not known a priori whether a particular dataset has any importance for a prediction task;

in the inventory optimization example in Section 1.2, retail stores selling foot-traffic data cannot realistically know which logistics company's demand forecast their data stream will be predictive for (again, this is only exacerbated when taking into account the overlap of information between features). A data market must instead provide a real-time mechanism to match training features to prediction tasks based on the increase in predictive value from the allocated features.

Information Economics. There has been an exciting recent line of work that directly tackles data as an economic good which we believe to be complimentary to our work. We divide them into three major buckets and highlight some representative papers: (i) data sellers have detailed knowledge of the specific prediction task and incentives to exert effort to collect high-quality data (e.g. reduce variance) are modeled [5, 13]; (ii) data sellers have different valuations for privacy and mechanisms that tradeoff privacy loss vs. revenue gain are modeled [16, 22]; (iii) studying the profitability of data intermediaries who supply consumer data to firms that want to sell the very same customers more targeted goods [9]. These are all extremely important lines of work to pursue, but they focus on different (but complementary) objectives.

Referring back to the inventory optimization example in Section 1.2), we model the sellers (retail stores) as simply trying to maximize revenue by selling foot-traffic data they already collect. Hence we assume they have *(i) no ability to fundamentally increase the quality of their data stream; (ii) no knowledge of the prediction task; (iii) no concerns for privacy.* In many practical commercial settings, these assumptions do suffice as the data is sufficiently anonymized, and these sellers are trying to monetize data they are already implicitly collecting through their operations. We focus our work on such a setting, where firms are trying to buy features to feed their ML models, and believe our formulation to be the most relevant for it. It would be interesting future work to find ways of incorporating privacy, feedback and the cost of data acquisition into our model.

2 THE MODEL - PARTICIPANTS AND DYNAMICS

2.1 Sellers

Let there be M sellers, each supplying data streams in this marketplace. We formally parameterize a seller through the following single quantity:

Feature. $X_j \in \mathbb{R}^T,\ j \in [M]$ is a vector of length T.

For simplicity, we associate with each seller a single feature and thus restrict X_j to be in $\mathbb{R}^T$. Our model is naturally extended to the case where sellers are selling multiple streams of data by considering each stream as another "seller" in the marketplace. We refer to the matrix denoting any subset of features as $\boldsymbol{X}_S$, $S \subset [M]$. Recall from the motivations we provide in Sections 1.2 and 1.3 for our model, we assume data sellers do not have the ability to change the quality of the data stream (e.g. reducing variance) they supply into the market nor any concerns for privacy (we assume data is sufficiently anonymized as is common in many commercial settings). Additionally sellers have no knowledge of the prediction tasks their data will be used for and simply aim to maximize revenue from the datasets that they already have on hand.

2.2 Buyers

Let there be N buyers in the market, each trying to purchase the best collection of datasets they can afford in this marketplace for a particular prediction task. We formally parameterize a buyer through the following set of quantities, for $n \in [N]$:

Prediction Task. $Y_n \in \mathbb{R}^T$ is a vector of T labels that Buyer n wants to predict well [2].

[2]To reduce notational overload, we abstract away the partition of Y_n into training and test data.

We provide a clarifying example of Y_n and X_j, using the inventory optimization example in Section 1.2. There, the historical inventory demand for the logistics company is Y_n and each historical foot-traffic data stream sold by retailers is X_j. The "prediction task" is then to forecast inventory demand, Y_n, from time-lagged foot traffic data, X_j for $j \in [M]$.

Prediction Gain Function. $\mathcal{G}_n : \mathbb{R}^{2T} \to [0, 1]$, the prediction gain function, takes as inputs the prediction task Y_n and an estimate $\hat{Y}_n$, and outputs the quality of the prediction.

For regression, an example of $\mathcal{G}_n$ is $1 - \text{RMSE}$ [3] (root-mean-squared-error). For classification, an example of $\mathcal{G}_n$ is Accuracy [4]. In short, a larger value for $\mathcal{G}_n$ implies better prediction accuracy. To simplify the exposition (and without any loss of generality of the model), we assume that all buyers use the same gain function i.e., $\mathcal{G} = \mathcal{G}_n$ for all n.

Value of Accuracy. $\mu_n \in \mathbb{R}_+$ is how much Buyer n values a *marginal* increase in accuracy.

As an illustration, recall the inventory optimization example in Section 1.2 where a logistics company makes a bid of the form, "willing to pay \$1000 for a percentage increase in demand forecasting accuracy from the previous week". We then have the following definition for how a buyer values an increase in accuracy,

Definition 2.1. *Let $\mathcal{G}$ be the prediction gain function. We define the value Buyer n gets from estimate $\hat{Y}_n$ as:*

$$\mu_n \cdot \mathcal{G}(Y_n, \hat{Y}_n)$$

i.e., μ_n is what a buyer is willing to pay for a unit increase in $\mathcal{G}$.

Remark 2.1. *Though a seemingly natural definition, we view it as one of the key modeling decisions we make in our design of a data marketplace. In particular, a buyer's valuation for data does not come from specific datasets, but rather from an increase in prediction accuracy of a quantity of interest.*

Remark 2.2. *A potential source of confusion is we require μ_n to be linear while many ML error metrics are non-linear. For example, in balanced, binary classification problems, randomly guessing labels has expected accuracy of 50%, which has zero value (and not $\mu_n/2$). However, such non-linearities can easily be captured in the gain function, $\mathcal{G}$. For the balanced, binary classification problem, $\mathcal{G}$ can easily be normalized such that 50%-accuracy has value 0 and 100%-accuracy has value 1 (specifically, $G = \frac{max(0, \widehat{accuracy})}{0.5}$). μ_n can thus be thought of as a buyer-specific scaling of how much they value an increase in accuracy. Indeed, linear utility models are standard in information economics (c.f. [9]).*

Remark 2.3. *Recall that to reduce notational overload, we let Y_n refer to both test and train data. Specifically, $Y_n = (Y_n^{train}, Y_n^{test})$. The ML-algorithm accesses Y_n^{train} and the Gain function, $\mathcal{G}$ accesses Y_n^{test} (i.e. $\hat{Y}_n$), as is standard in ML workflows.*

Public Bid Supplied to Market. $b_n \in \mathbb{R}_+$ is the public bid supplied to the marketplace.

Note that μ_n is a private valuation. If Buyer n is strategic, μ_n is not necessarily what is revealed to the marketplace. Thus we define b_n, which refers to the actual bid supplied to the marketplace (not necessarily equal to μ_n).

---

[3] $\text{RMSE} = \frac{1}{Y_{\max} - Y_{\min}} \sqrt{\sum_{i=1}^{T} (\hat{Y}_i - Y_i)^2 / T}$, where: (i) $\hat{Y}_i$ is the predicted value for $i \in [T]$ produced by the machine learning algorithm, $\mathcal{M}$, (ii) $Y_{\max}$, $Y_{\min}$ are the max and min of Y_n respectively.

[4] $\text{Accuracy} = \frac{1}{T} \sum_{i=1}^{T} \mathbb{1}(\hat{Y}_i = Y_i)$, with $\hat{Y}_i$ defined similarly to that above

2.3 Marketplace

The function of the marketplace is to match buyers and sellers as defined above. As we make precise in Section 2.4, we model the M sellers as fixed and the N buyers as coming one at a time. We formally parameterize a marketplace through the following set of quantities, for $n \in [N]$:

Price. $p_n \in \mathbb{R}_+$ is the price the marketplace sets for the features on sale when Buyer n arrives.

As we make precise in Property 3.2, we measure the quality of the prices $(p_1, \ldots, p_N)$ set by the marketplace for each buyer by comparing against the optimal fixed price in hindsight (i.e., standard definition of regret). However, it is well-known that standard price update algorithms for combinatorial goods, which satisfy Property 3.2, scale very poorly in M (cf. [15]). Specifically, if we maintain separate prices for every data stream (i.e., if $p_n \in \mathbb{R}_+^M$) it is easily seen that regret-minimizing algorithms such as Multiplicative Weights (cf. [3]) or Upper Confidence Bandits (cf. [4]), will have exponential running time or exponentially loose guarantees (in M) respectively. In fact from [15], we know regret minimizing algorithms for even very simple non-additive buyer valuations are provably computationally intractable.

Thus to achieve a zero-regret price update algorithm, without making additional restrictive assumptions, we restrict p_n to be a scalar rather than a M-dimensional vector. This is justified due to Definition 2.1, where we model a buyer's "value for accuracy" (and the associated public bid) through the scalar, μ_n (and the scalar b_n respectively). This allows the marketplace to control the quality of the predictions based on the difference between p_n and b_n (see Section 4.1 for details).

Machine Learning/Prediction Algorithm. $\mathcal{M} : \mathbb{R}^{MT} \to \mathbb{R}^T$, the learning algorithm utilized by the marketplace, takes as input the features on sale $\boldsymbol{X}_M$, and produces an estimate $\hat{Y}_n$ of Buyer n's prediction problem Y_n.

$\mathcal{M}$ does not necessarily have to be supplied by the marketplace and is a simplifying assumption. Instead buyers could provide their own learning algorithm that they intend to use, or point towards one of the many excellent standard open-source libraries widely used such as SparkML, Tensorflow and Scikit-Learn (cf. [1, 28, 30]) [5].

Allocation Function. $\mathcal{AF} : (p_n, b_n; \boldsymbol{X}_M) \to \widetilde{\boldsymbol{X}}_M, \widetilde{\boldsymbol{X}}_M \in \mathbb{R}^M$, takes as input the current price p_n and the bid b_n received, to decide the quality at which Buyer n gets allocated the features on sale $\boldsymbol{X}_M$ (e.g. by adding noise or subsampling the features).

In Section 4.1, we provide explicit instantiations of $\mathcal{AF}$ and detailed reasoning for why we choose this particular class of allocation functions.

Revenue Function. $\mathcal{RF} : (p_n, b_n, Y_n; \mathcal{M}, \mathcal{G}, \boldsymbol{X}_M) \to r_n,\ r_n \in \mathbb{R}_+$, the revenue function, takes as input the current price p_n, in addition to the bid and the prediction task provided by the buyer (b_n and Y_n respectively), to decide how much revenue r_n to extract from the buyer.

Payment Division Function. $\mathcal{PD} : (Y_n, \widetilde{\boldsymbol{X}}_M; \mathcal{M}, \mathcal{G}) \to \psi_n,\ \psi_n \in [0, 1]^M$, the payment-division function, takes as input the prediction task Y_n along with the features that were allocated $\widetilde{\boldsymbol{X}}_M$, to compute ψ_n, a vector denoting the marginal value of each allocated feature for the prediction task.

Price Update Function. $\mathcal{PF} : (p_n, b_n, Y_n; \mathcal{M}, \mathcal{G}, \boldsymbol{X}_M) \to p_{n+1},\ p_{n+1} \in \mathbb{R}_+$, the price-update function, takes as input the current price p_n, in addition to the bid and the prediction task provided by the buyer (b_n and Y_n respectively) to update the price for Buyer $n + 1$.

[5] Indeed a key trend in many business use cases, is that the ML algorithms used are simply lifted from standard open-source libraries. Thus the accuracy of predictions is primarily a function of the quality of the data fed to these ML algorithms.

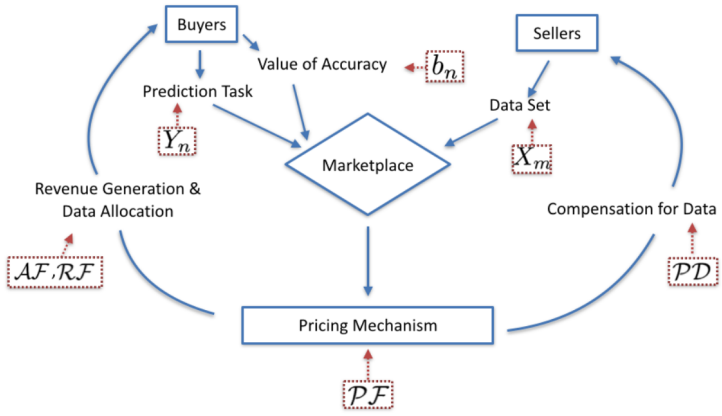

Fig. 1. Overview of marketplace dynamics.

2.3.1 Buyer Utility. We can now precisely define the utility function, $\mathcal{U} : \mathbb{R}_+ \times \mathbb{R}^T \to \mathbb{R}$, each buyer is trying to maximize,

Definition 2.2. *The utility Buyer n receives by bidding b_n for prediction task Y_n is given by*

$$\mathcal{U}(b_n, Y_n) := \mu_n \cdot \mathcal{G}(Y_n, \hat{Y}_n) - \mathcal{RF}(p_n, b_n, Y_n) \tag{1}$$

where $\hat{Y}_n = \mathcal{M}(Y_n, \widetilde{\boldsymbol{X}}_M)$ and $\widetilde{\boldsymbol{X}}_M = \mathcal{AF}(p_n, b_n; \boldsymbol{X}_M)$.

In words, the first term on the right hand side (r.h.s) of (1) is the value derived from a gain in prediction accuracy (as in Definition 2.1). Note this is a function of the quality of the features that were allocated based on the bid b_n. The second term on the r.h.s of (1) is the amount the buyer pays, r_n. Buyer utility as in Definition 2.2 is simply the difference between these two terms.

2.4 Marketplace Dynamics

We can now formally define the per-step dynamic within the marketplace (see Figure 1 for a graphical overview). Note this is a formalization of the steps laid out in the inventory optimization example in Section 1.2. When Buyer n arrives, the following steps occur in sequence (we assume p_0, b_0, Y_0 are initialized randomly):

For $n \in [N]$:

(1) Market sets price p_n, where $p_n = \mathcal{PF}(p_{n-1}, b_{n-1}, Y_{n-1})$
(2) Buyer n arrives with prediction task Y_n
(3) Buyer n bids b_n where $b_n = \arg\max_{z \in \mathbb{R}_+} \mathcal{U}(z, Y_n)$
(4) Market allocates features $\widetilde{\boldsymbol{X}}_M$ to Buyer n , where $\widetilde{\boldsymbol{X}}_M = \mathcal{AF}(p_n, b_n; \boldsymbol{X}_M)$
(5) Buyer n achieves $\mathcal{G}\left(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_M)\right)$ gain in prediction accuracy
(6) Market extracts revenue, r_n, from Buyer n, where $r_n = \mathcal{RF}(p_n, b_n, Y_n; \mathcal{M}, \mathcal{G})$
(7) Market divides r_n amongst allocated features using ψ_n, where $\psi_n = \mathcal{PD}(Y_n, \widetilde{\boldsymbol{X}}_M; \mathcal{M}, \mathcal{G})$

Remark 2.4. *A particularly important (albeit implicit) benefit of the above proposed architecture is that the buyer's do not ever access the underlying features. Rather they only receive predictions*

through the ML model trained on the allocated features. This circumvents a known, difficult problem in designing data markets where sellers are reluctant to release potentially valuable data streams as they do not have control over who subsequently accesses it (since data streams are freely replicable).

Remark 2.5. *In our proposed architecture, the price for each buyer is set centrally by the marketplace rather than by the sellers individually. A sellers simply supplies data streams to the marketplace and is assigned revenue based on the marginal contribution the data stream provides to the prediction task. Thus from the perspective of price setting, our model can equivalently be thought of as a single seller supplying multiple data streams to the market and adjusting p_n to maximize overall revenue.*

Remark 2.6. *We note from the dynamics laid out above (specifically Step 3), a buyer is "myopic" over a single-stage i.e., Buyer n comes into the market once and leaves after being provided the estimate $\hat{Y}_n$. Thus Buyer n maximizing utility only over Step n. In particular, we do not study the additional complication if the buyer's utility is defined over multiple-stages.*

Remark 2.7. *Our proposed architecture does not take into account an important attribute of data; a firm's utility for a particular dataset may be heavily dependent on what other firms get access to it (e.g. a hedge fund might pay a premium to have a particularly predictive dataset only go to it). By modeling buyer's coming to the market one at a time, we do not study the externalities associated with a dataset being replicated multiple times.*

3 DESIRABLE PROPERTIES OF MARKETPLACE

We define key properties for such a marketplace to robustly function in a large-scale, real-time setting, where buyers are arriving in quick succession and need to be matched with a large number of data sellers within minutes, if not quicker. Intuitively we require the following properties: (i) buyers are truthful in their bids; (ii) overall revenue is maximized; (iii) revenue is fairly divided amongst sellers; (iv) marketplace runs efficiently. In Sections 3.1-3.4, we formally define these properties.

3.1 Truthfulness

Property 3.1 (Truthful). *A marketplace is "truthful" if for all Y_n,*

$$\mu_n = \arg\max_{z \in \mathbb{R}_+} \mathcal{U}(z, Y_n)$$

where $\mathcal{U}(z, Y_n)$ is defined as in Definition 2.2.

Property 3.1 requires that the allocation function, $\mathcal{AF}$, and the revenue function, $\mathcal{RF}$, incentivize buyers to bid their true valuation for an increase in prediction accuracy. Note that we assume buyers do not alter their prediction task, Y_n.

3.2 Revenue Maximization

Property 3.2 (Revenue Maximizing). *Let $\{(\mu_1, b_1, Y_1), (\mu_2, b_2, Y_2), \ldots, (\mu_N, b_N, Y_N)\}$ be a sequence of buyers entering the market. A marketplace is "revenue maximizing" if the price-update function, $\mathcal{PF}(\cdot)$, produces a sequence of prices, $\{p_1, p_2, \ldots, p_n\}$, such that the "worst-case" average regret, relative to the optimal price p^* in hindsight, goes to 0, i.e.,*

$$\lim_{N\to\infty} \frac{1}{N}\left[\sup_{\{(b_n, Y_n): n \in [N]\}} \left(\sup_{p^* \in \mathbb{R}_+} \sum_{n=1}^{N} \mathcal{RF}(p^*, b_n, Y_n) - \sum_{n=1}^{N} \mathcal{RF}(p_n, b_n, Y_n) \right) \right].$$

As is convention, we term the expression with the square bracket as regret and denote it, $\mathcal{R}(N, M)$. Property 3.2 is the standard worst-case regret guarantee (cf. [20]). It necessitates the price-update

function, $\mathcal{PF}$, produce a sequence of prices p_n such that the average difference with the unknown optimal price in hindsight, p^* goes to zero as N increases. Note Property 3.2 must hold over the worst case sequence of buyers i.e, no distributional assumptions on μ_n, b_n, Y_n are made.

3.3 Revenue Division

In the following section, we abuse notation and let $S \subset [M]$ refer to both the index of the training features on sale and to the actual features, $\boldsymbol{X}_S$ themselves.

3.3.1 *Shapley Fairness.*

Property 3.3 (Shapley Fair). *A marketplace is "Shapley-fair" if* $\forall\ n \in [N], \forall\ Y_n$, *the following holds on* $\mathcal{PD}$ *(and its output,* ψ_n*):*

(1) ***Balance**:* $\sum_{m=1}^{M} \psi_n(m) = 1$
(2) ***Symmetry**:* $\forall\ m, m' \in [M], \forall S \subset [M] \setminus \{m, m'\}$, *if* $\mathcal{PD}(S \cup m, Y_n) = \mathcal{PD}(S \cup m', Y_n)$, *then* $\psi_n(m) = \psi_n(m')$
(3) ***Zero Element**:* $\forall\ m \in [M], \forall S \subset [M]$, *if* $\mathcal{PD}(S \cup m, Y_n) = \mathcal{PD}(S, Y_n)$, *then* $\psi_n(m) = 0$
(4) ***Additivity**: Let the output of* $\mathcal{PD}([M], Y_n^{(1)}), \mathcal{PD}([M], Y_n^{(2)})$ *be* $\psi_n^{(1)}, \psi_n^{(2)}$ *respectively. Let* ψ_n' *be the output of* $\mathcal{PD}([M], Y_n^{(1)} + Y_n^{(2)})$. *Then* $\psi_n' = \psi_n^{(1)} + \psi_n^{(2)}$.

The conditions of Property 3.3, first laid out in [34], are considered the standard axioms of fairness. We choose them as 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).

Remark 3.1. *A naive definition of the marginal value of feature m would be a "leave-one-out" policy, i.e.,* $\psi_n(m) = \mathcal{G}(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_{[M]})) - \mathcal{G}(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_{[M]\setminus m}))$. *As the following toy example shows, the correlation between features would lead to the market "undervaluing" each feature. Consider the simple case where there are two sellers each selling identical features. It is easy to see the "leave-one-out" policy above would lead to zero value being allocated to each feature, even though they collectively might have great predictive value. This is clearly undesirable. That is why Property 3.3 is a necessary notion of fairness as it takes into account the overlap of information that will invariably occur between the different features,* X_j.

We then have the following celebrated theorem from [34],

Theorem 3.1 (Shapley Allocation). *Let* $\psi_{shapley} \in [0,1]^{[M]}$ *be the output of the following algorithm,*

$$\psi_{shapley}(m) = \sum_{T \subset [M]\setminus\{m\}} \frac{|T|!(M - |T| - 1)!}{M!} \left(\mathcal{G}\left(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_{T\cup m})\right) - \mathcal{G}\left(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_{T})\right) \right) \tag{2}$$

Then $\psi_{shapley}$ *is the unique allocation that satisfies all conditions of Property 3.3*

Intuitively, this algorithm is computing the average marginal value of feature m over all subsets $T \subset [M] \setminus \{m\}$. It is easily seen that the running time of this algorithm is $\Theta(2^M)$, which makes it infeasible at scale if implemented as is. But it still serves as a useful standard to compare against.

3.3.2 *Robustness to Replication.*

Property 3.4 (Robustness to replication). *For all* $m \in [M]$, *let* m_i^+ *refer to the* i^{th} *replicated copy of m i.e.,* $X_{m,i}^+ = X_m$. *Let* $[M]^+ = \cup_m (m \cup_i m_i^+)$ *refer to the set of original and replicated features. Let* $\psi_n^+ = \mathcal{PD}([M]^+, Y_n)$. *Then a marketplace is* ϵ*-"robust-to-replication" if* $\forall\ n \in [N], \forall\ Y_n$, *the following holds on* $\mathcal{PD}$*:*

$$\psi_n^+(m) + \sum_i \psi_n^+(m_i^+) \leq \psi_n(m) + \epsilon.$$

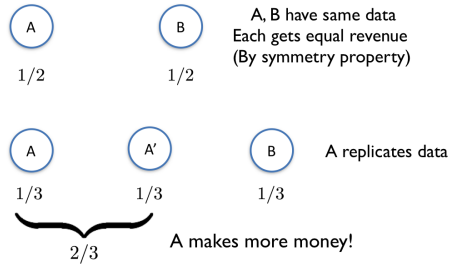

Fig. 2. Shapley fairness is inadequate for freely replicable goods.

Property 3.4 is a novel notion of fairness, which can be considered a necessary additional requirement to the Shapley notions of fairness for freely replicable goods. We use Example 3.1 below to elucidate how adverse replication of data can lead to grossly undesirable revenue divisions (see Figure 2 for a graphical illustration). Note that implicit in the definition of Property 3.4 is that the "strategy-space" of the data sellers is the number of times they replicate their data.

Example 3.1. *Consider a simple setting where the marketplace consists of only two sellers, A and B, each selling one feature which are both identical. By Property 3.3, the Shapley value of A and B are equal, i.e.,* $\psi(A) = \frac{1}{2}, \psi(B) = \frac{1}{2}$. *However if seller A replicated her feature once and sold it again in the marketplace, it is easy to see that the new Shapley allocation will be* $\psi(A) = \frac{2}{3}, \psi(B) = \frac{1}{3}$. *Hence it is not robust to replication since the aggregate payment remains the same (no change in accuracy).*

Such a notion of fairness in cooperative games is especially important in modern day applications where: (i) digital goods are prevalent and can be produced at close to zero marginal cost; (ii) users get utility from bundles of digital goods with potentially complex combinatorial interactions between them. Two examples of such a setting are battery cost attribution among smartphone applications and reward allocation among "experts" in a prediction market.

3.4 Computational Efficiency

We assume the Machine Learning algorithm, $\mathcal{M}$, and the Gain function, $\mathcal{G}$, each require computation running time of $O(M)$, i.e., computation complexity scales at most linearly with the number of features/sellers, M. We define the following computational efficiency requirement of the market,

Property 3.5 (Efficient). *A marketplace is "efficient" if for each Step n, the marketplace as laid out in Section 2.4 runs in polynomial time in M, where M is the number of sellers. In addition, the computational complexity for each step of the marketplace cannot grow with N.*

Such a marketplace is feasible only if it functions in real-time. Thus, it is pertinent that the computational resources required for any Buyer n to interface with the market are low i.e., ideally with run-time close to linear in M, the number of sellers, and not growing based on the number of buyers seen thus far. Due to the combinatorial nature of data, this is a non–trivial requirement as such combinatorial interactions normally lead to an exponential dependence in M; recall from earlier sections, the Shapley Algorithm in Theorem 3.1 runs in $\Theta(2^M)$ and a naive implementation of Multiplicative Weights Algorithm for combinatorial goods runs in $\Theta(\exp(M))$.

4 MARKETPLACE CONSTRUCTION

We now explicitly construct instances of $\mathcal{AF}, \mathcal{RF}, \mathcal{PF}$ and $\mathcal{PD}$ and argue in Section 5 that the properties laid out in Section 3 hold for these particular constructions.

Remark 4.1. *In line with Remark 2.5, we can think of $\mathcal{AF}, \mathcal{RF}$ as instances of how to design a robust bidding, data allocation and revenue generation scheme from the buyer's perspective with the features sold held fixed (see Property 3.1). Analogously, $\mathcal{PD}$ is a function for fair revenue division from the seller's perspective for a fixed amount of generated revenue (see Properties 3.3 and 3.4). And $\mathcal{PF}$ is a function to centrally adjust the price of the features sold dynamically over time from the marketplace's perspective (see Property 3.2).*

4.1 Allocation and Revenue Functions (Buyer's Perspective)

Allocation Function. Recall the allocation function, $\mathcal{AF}$, takes as input the current price p_n and the bid b_n received, to decide the quality of the features $\boldsymbol{X}_M$, used for Buyer n's prediction task.

Recall from Definition 2.1 that a buyer's utility comes solely from the quality of the estimate $\hat{Y}_n$ received, rather than the particular datasets allocated. Thus the key structure we exploit in designing $\mathcal{AF}$ is that from the buyer's perspective, *instead of considering each feature X_j as a separate good (which leads to computational intractability), it is greatly simplifying to think of $\boldsymbol{X}_M$ as the total amount of "information" on sale.* $\mathcal{AF}$ can thus be thought of as a function to *collectively* adjust the quality of all of $\boldsymbol{X}_M$ based on the difference between p_n and b_n

Specifically, we choose $\mathcal{AF}$ to be a function that adds noise to/degrades $\boldsymbol{X}_M$ proportional to the difference between p_n and b_n. This degradation can take many forms and depends on the structure of X_j itself. Below, we provide examples of commonly used allocation functions for some typical X_j encountered in ML.

Example 4.1. *Consider $X_j \in \mathbb{R}^T$ i.e. sequence of real numbers. Then an allocation function (i.e. perturbation function), $\mathcal{AF}_1^*(p_n, b_n; X_j)$, commonly used (cf. [13, 16]) would be for t in $[T]$,*

$$\tilde{X}_j(t) = X_j(t) + \max(0, p_n - b_n) \cdot \mathcal{N}(0, \sigma^2)$$

where $\mathcal{N}(0, \sigma^2)$ is a univariate Gaussian.

Example 4.2. *Consider $X_j \in \{0, 1\}^T$ i.e. sequence of bits. Then an allocation function (i.e. masking function), $\mathcal{AF}_2^*(p_n, b_n; X_j)$, commonly used (cf. [33]) would be for t in $[T]$,*

$$\tilde{X}_j(t) = B(t; \theta) \cdot X_j(t)$$

where $B(t; \theta)$ is an independent Bernoulli random variable with parameter $\theta = \min(\frac{b_n}{p_n}, 1)$.

In both examples if $b_n \geq p_n$, then the buyer is given $\boldsymbol{X}_M$ as is without degradation. However if $b_n < p_n$, then X_j is degraded in proportion to the difference between b_n and p_n.

Remark 4.2. *Through Assumption 1 (see Section 5.1), we formalize a natural and necessary property required of any such allocation function so that Property 3.1 (truthfulness) holds. Specifically, for a fixed price p_n, increasing the bid b_n cannot lead to a decrease in prediction quality. The space of possible allocations functions that meet this criteria is clearly quite large. We leave it as future work to study what is the optimal $\mathcal{AF}$ from the space of feasible allocation functions to maximize revenue.*

Remark 4.3. *A celebrated result from [29] is that for single-parameter buyers, a single take-it-or-leave-it price for all data is optimal, i.e. if the bid is above the singe posted price then allocate all the data without any noise and if the bid is less than the price, allocate no data. However, maybe surprisingly, in our setting this result does not apply. This is due to an important subtlety in our formalism - while μ_n (how much a buyer values a marginal increase in accuracy), is a scalar, a buyer is also parametrized by Y_n, the prediction task.*

This leads to the following simple counter-example - imagine buyers are only of two types: (i) Type I with prediction task Y_1 and valuation μ_1; (ii) Type II with prediction task Y_2 and valuation μ_2. Further,

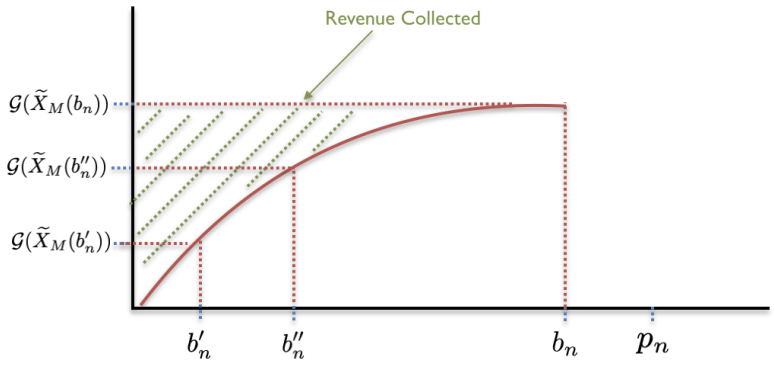

Fig. 3. Features allocated ($\mathcal{AF}^*$) and revenue collected ($\mathcal{RF}^*$) for a particular price vector p_n and bid b_n.

let there be only two types of features on sale, X_1 and X_2. Assume X_1 is perfectly predictive of prediction task Y_1 and has zero predictive value for Y_2. Analogously, assume X_2 is perfectly predictive of Y_2 and has no predictive value for Y_1. Then it is easy to see that the optimal pricing mechanism is to set the price of X_1 to be μ_1 and X_2 to be μ_2. Thus a single posted price is not optimal [6].

More generally, if different datasets have varying amounts of predictive power for different buyer types, it is not even clear that a take-it-or-leave-it price per feature sold is optimal.

Revenue Function. Recall from Definition 2.1, we parameterize buyer utility through the parameter μ_n, i.e., how much a buyer values a marginal increase in prediction quality. This crucial modeling choice allows us to use Myerson's payment function rule (cf. [29]) given below,

$$\mathcal{RF}^*(p_n, b_n, Y_n) = b_n \cdot \mathcal{G}\Big(Y_n, \mathcal{M}\Big(\mathcal{AF}^*(b_n, p_n)\Big)\Big) - \int_0^{b_n} \mathcal{G}\Big(Y_n, \mathcal{M}\Big(\mathcal{AF}^*(z, p_n)\Big)\Big) dz. \tag{3}$$

In Theorem 5.1, we show that $\mathcal{RF}^*$ *ensures Buyer n is truthful* (as defined in Property 3.1). Refer to Figure 3 for a graphical view of $\mathcal{AF}^*$ and $\mathcal{RF}^*$.

4.2 Price Update Function (Marketplace Perspective)

The market is tasked with how to pick p_n for $n \in [N]$. Recall from Section 2.4, the market must decide p_n before Buyer n arrives (otherwise, it is easily seen that truthfulness cannot be gauranteed).

We now provide some intuition of how increasing/decreasing p_n affects the amount of revenue collected, and the implicit tradeoff that lies therein. Observe from the construction of $\mathcal{RF}^*$ in (3) that for a fixed bid, b_n, and prediction task Y_n, it is easily seen that if p_n is picked to be too large, then the positive term in $\mathcal{RF}^*$ is small (as the degradation of the signal in $\boldsymbol{X}_M$ is very high), leading to lower than optimal revenue generation. Similarly, if p_n is picked to be too small, it is easily seen that the negative term in $\mathcal{RF}^*$ is large, which again leads to an undesired loss in revenue.

In Algorithm 1, we apply the Multiplicative Weights method to pick p_n in an online fashion and to balance the tradeoff described above. To construct Algorithm 1 more precisely, we need to define some additional quantities. As we make precise in Assumption 4 in Section 5, we assume bids come from some bounded set, $\mathcal{B} \subset \mathbb{R}_+$. We define $\mathcal{B}_{\max} \in \mathbb{R}$ to be the maximum element of $\mathcal{B}$

[6]When it comes to truthfulness (see Property 3.1), the fact that we can parametrize a buyerâĂŹs valuation through a scalar, μ_n, does indeed mean MyersonâĂŹs payment function (see (3)) is truthful as long as the data allocation function is monotonic.

ALGORITHM 1: PRICE-UPDATE: $\mathcal{PF}^*(b_n, Y_n, \mathcal{B}, \epsilon, \delta)$

Input: $b_n, Y_n, \mathcal{B}, \epsilon, \delta$
Output: p_n
Let $\mathcal{B}_{\text{net}}(\epsilon)$ be an ϵ-net of $\mathcal{B}$;
for $c^i \in \mathcal{B}_{net}(\epsilon)$ **do**
    Set $w_1^i = 1$; `// initialize weights of all experts to 1`
end
for $n = 1$ *to* N **do**
    $W_n = \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i$;
    Let $p_n = c^i$ with probability w_n^i / W_n ; `// note` p_n `is not a function of` b_n
    for $c^i \in \mathcal{B}_{net}(\epsilon)$ **do**
        Let $g_n^i = \mathcal{RF}^*(c^i, b_n, Y_n)/\mathcal{B}_{\max}$; `// revenue gain if price` c^i `was used`
        Set $w_{n+1}^i = w_n^i \cdot (1 + \delta g_n^i)$; `// Multiplicative Weights update step`
    end
end

and $\mathcal{B}_{\text{net}}(\epsilon)$ to be a minimal ϵ-net of $\mathcal{B}$ [7]. Intuitively, the elements of $\mathcal{B}_{\text{net}}(\epsilon)$ serve as our "experts" (i.e. the different prices we experiment with) in the Multiplicative Weights algorithm.

In Theorem 5.2, we show that this algorithm does indeed achieve zero-regert with respect to the optimal $p^* \in \mathbb{R}_+$ in hindsight.

4.3 Payment-Division Functions (Seller's Perspective)

4.3.1 Shapley Approximation. In our model (see Section 2.4), a buyer only makes an aggregate payment to the market based on the increase in accuracy experienced (see $\mathcal{RF}^*$ in (3)). It is thus up to the market to design a mechanism to fairly (as defined in Property 3.3) allocate the revenue among the sellers to incentivize their participation. Following the seminal work in [34], there have been a substantial number of applications (cf. [6, 7]) leveraging the ideas in [34] to fairly allocate cost/reward among strategic entities cooperating towards a common goal. Since the Shapley algorithm stated in (2) is the unique method to satisfy Property 3.3, but unfortunately runs in time $\Theta(2^M)$, the best one can do is to approximate (2) as closely as possible.

In Algorithm 2, we uniformly sample from the space of permutations over $[M]$ to construct an approximation of the Shapley value in (2). To construct Algorithm 2 more precisely, we need to define some additional quantities. Let $\sigma_{[M]}$ refer to the set of all permutations over $[M]$. For any permutation $\sigma \in \sigma_{[M]}$, let $[\sigma < m]$ refer to the set of features in $[M]$ that came before m.

The key observation in showing that Algorithm 2 is effective, is that instead of enumerating over all permutations in $\sigma_{[M]}$ as in the Shapley allocation, it suffices to sample $\sigma_k \in \sigma_{[M]}$ uniformly at random with replacement, K times, where K depends on the ϵ-approximation a practitioner desires. We provide guidance on how to pick K in Section 5.4. We note some similar sampling based methods, albeit for different applications (cf. [11, 25, 26]).

In Theorem 5.3, we show that Algorithm 2 gives an ϵ-approximation for (2) with high probability while running in time $O(M^2)$.

4.3.2 Robustness to Replication. Recall from Section 3.3.2 that for freely replicable goods such as data, the standard Shapley notion of fairness does not suffice (see Example 3.1 for how it can lead to undesirable revenue allocations). Though this issue may seem difficult to overcome in general,

[7] We endow $\mathbb{R}$ with the standard Euclidean metric. An ϵ-net of a set $\mathcal{B}$ is a set $K \subset \mathcal{B}$ such that for every point $x \in \mathcal{B}$, there is a point $x_0 \in K$ such that $|x - x_0| \leq \epsilon$.

ALGORITHM 2: SHAPLEY-APPROX: $\mathcal{PD}_A^*(Y_n, \widetilde{X}_M, K)$

Input: $Y_n, \widetilde{X}_M, K$
Output: $\hat{\psi}_n = [\hat{\psi}_n(m) : m \in [M]]$
Let $\mathcal{B}_{\text{net}}(\epsilon)$ be an ϵ-net of $\mathcal{B}$;
for $m \in [M]$ **do**
  for $k \in [K]$ **do**
    $\sigma_k \sim \text{Unif}(\sigma_{[M]})$;
    $G = \mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k < m]}))$;
    $G^+ = \mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k < m \cup m]}))$;
    $\hat{\psi}_n^k(m) = [G^+ - G]$
  end
  $\hat{\psi}_n(m) = \frac{1}{K} \sum_{k=1}^K \hat{\psi}_n^k(m)$
end

we again exploit the particular structure of data as a path forward. Specifically, we note that there are *standard methods to define the "similarity" between two vectors of data.* A complete treatment of similarity measures has been done in [18]. We provide two examples:

Example 4.3. *Cosine similarity, a standard metric used in text mining and information retrieval, is:*

$$\frac{|\langle X_1, X_2 \rangle|}{||X_1||_2 ||X_2||_2}, \; X_1, X_2 \in \mathbb{R}^T$$

Example 4.4. *"Inverse" Hellinger distance, a standard metric to define similarity between underlying data distributions, is:* $1 - \frac{1}{2} \sum_{x \in \mathcal{X}} (\sqrt{p_1(x)} - \sqrt{p_2(x)})^{1/2}, \; p_1 \sim X_1, \; p_2 \sim X_2$.

We introduce some natural properties any such similarity metric must satisfy for our purposes,

Definition 4.1 (**Adapted from [18]**). *A similarity metric is a function,* $\mathcal{SM} : \mathbb{R}^T \times \mathbb{R}^T \to [0, 1]$, *that satisfies: (i) Limited Range:* $0 \leq \mathcal{SM}(\cdot, \cdot) \leq 1$*; (ii) Reflexivity:* $\mathcal{SM}(X, Y) = 1$ *if and only if* $X = Y$*; (iii) Symmetry:* $\mathcal{SM}(X, Y) = \mathcal{SM}(Y, X)$*; (iv) Define* $d\mathcal{SM}(X, Y) = 1 - \mathcal{SM}(X, Y)$*, then Triangle Inequality:* $d\mathcal{SM}(X, Y) + d\mathcal{SM}(Y, Z) \geq d\mathcal{SM}(X, Z)$

In Algorithm 3, we construct a "robust-to-replication" version of the randomized Shapley approximation algorithm by utilizing Definition 4.1 above.

Intuitively, the algorithm penalizes similar features (relative to the similarity metric, $\mathcal{SM}$) to disincentivize replication. We provide guidance on how to pick the hyper-parameter λ in Section 5.

In Theorem 5.4, we show Algorithm 3 is ϵ-"Robust to Replication" i.e. Property 3.4 (Robustness-to-Replication) holds. See the example below for an illustration of the effect of Algorithm 3 on undesired replication.

Example 4.5. *Recall Example 3.1 where there are two sellers, A and B, each selling an identical feature. In that example, if Seller A replicated her feature, her Shapley allocation increased from* $\frac{1}{2}$ *to* $\frac{2}{3}$*. If we instead apply Algorithm 3 (with* $\lambda = 1$*), then it is easy to see that her Shapley allocation decreases from* $\frac{1}{2e}$ *to* $\frac{2}{3e^2}$*, ensuring Property 3.4 holds. See Figure 4 for an illustration.*

5 MAIN RESULTS

5.1 Assumptions.

To give performance guarantees, we state four mild and natural assumptions we need on: (i) $\mathcal{AF}^*$ (allocation function); (ii) $\mathcal{M}$ (ML algorithm); (iii) $\mathcal{RF}^*$ (revenue function); (iv) b_n (bids made).

ALGORITHM 3: SHAPLEY-ROBUST: $\mathcal{PD}_B^*(Y_n, \widetilde{X}_M, K, \mathcal{SM}, \lambda)$

Input: $Y_n, \widetilde{X}_M, K, \mathcal{SM}, \lambda$
Output: $\psi_n = [\psi_n(m) : m \in [M]]$
$\hat{\psi}_n(m)$ = SHAPLEY-APPROX$(Y_n, \mathcal{M}, \mathcal{G}, K)$;
$\psi_n(m) = \hat{\psi}_n(m) \exp(-\lambda \sum_{j \in [M] \setminus \{m\}} \mathcal{SM}(X_m, X_j))$;

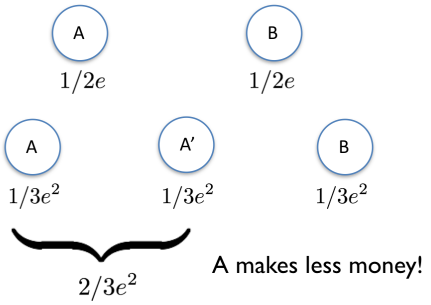

Fig. 4. A simple example illustrating how SHAPLEY-ROBUST down weights similar data to ensure robustness to replication.

Assumption 1 ($\mathcal{AF}^*$ is Monotonic). *$\mathcal{M}, \mathcal{AF}^*$ are such that an increase in the difference between p_n and b_n leads to a decrease in $\mathcal{G}$ i.e. an increase in "noise" cannot lead to an increase in prediction accuracy. Specifically, for any Y_n, p_n, let $\widetilde{X}_M^{(1)}, \widetilde{X}_M^{(2)}$ be the outputs of $\mathcal{AF}(p_n, b^{(1)}; X_M), \mathcal{AF}(p_n, b^{(2)}; X_M)$ respctively. Then if $b^{(1)} \leq b^{(2)}$, we have $\mathcal{G}\left(Y_n, \mathcal{M}(\widetilde{X}_M^{(1)})\right) \leq \mathcal{G}\left(Y_n, \mathcal{M}(\widetilde{X}_M^{(2)})\right)$.*

Assumption 2 ($\mathcal{M}$ is Invariant to Replicated Data). *$\mathcal{M}$ is such that replicated features do not cause a change in prediction accuracy. Specifically, $\forall\ S \subset [M],\ \forall\ Y_n,\ \forall\ m \in S$, let m_i^+ refer to the i^{th} replicated copy of m (i.e. $X_{m,i}^+ = X_m$). Let $S^+ = \cup_m (m \cup_i m_i^+)$ refer to the set of original and replicated features. Then $\mathcal{G}(Y_n, \mathcal{M}(X_S)) = \mathcal{G}(Y_n, \mathcal{M}(X_{S^+}))$*

Assumption 3 ($\mathcal{RF}^*$ is Lipschitz). *The revenue function $\mathcal{RF}^*$ is $\mathcal{L}$-Lipschitz with respect to price. Specifically, for any $Y_n, b_n, p^{(1)}, p^{(2)}$, we have $|\mathcal{RF}^*(p^{(1)}, b_n, Y_n) - \mathcal{RF}^*(p^{(2)}, b_n, Y_n)| \leq \mathcal{L}|p^{(1)} - p^{(2)}|$.*

Assumption 4 (Bounded Bids). *The set of possible bids b_n for $n \in [N]$ come from a closed, bounded set $\mathcal{B}$. Specifically, $b_n \in \mathcal{B}$, where diameter$(\mathcal{B}) = D$, where $D < \infty$.*

Remark 5.1. *We provide some justification for Assumptions 1 and 2 above, which impose requirements of the ML algorithm and the accuracy metric (i.e. $\mathcal{M}$ and $\mathcal{G}$). These assumptions require that: (i) as more noise is added to data, the less gain in prediction accuracy; (ii) replicated features do not have an effect on the accuracy. Essentially all ML algorithms and the accuracy metrics function like this. Thus these assumptions reflects standard, weak statistical assumptions. Assumptions 3 and 4 are self-explanatory.*

5.2 Truthfulness.

Theorem 5.1. *For $\mathcal{AF}^*$, Property 3.1 (Truthfulness) can be achieved if and only if Assumption 1 holds. In which case, $\mathcal{RF}^*$ guarantees truthfulness.*

Theorem 5.1 is an application of Myerson's payment function (cf. [29]) which ensures $b_n = \mu_n$. See Appendix A for the proof.

Again, the key is the modeling choice made to define buyer utility as in Definition 2.1. It lets us parameterize a buyers value for increased accuracy by a scalar, μ_n, which allows us to exploit Myerson's payment function (unfortunately generalization of Myerson's payment function to the setting where μ_n is a vector are severely limited cf. [14]).

5.3 Revenue Maximization.

THEOREM 5.2. *Let Assumptions 1, 3 and 4 hold. Let $p_{n:n\in[N]}$ be the output of Algorithm 1. Let $\mathcal{L}$ be the Lipschitz constant of $\mathcal{RF}^*$ (defined as in Assumption 3). Let $\mathcal{B}_{\max} \in \mathbb{R}$ be the maximum element of $\mathcal{B}$ (where $\mathcal{B}$ is defined as in Assumption 4). Then by choosing the algorithm hyper-parameters $\epsilon = (\mathcal{L}\sqrt{N})^{-1}$, $\delta = \sqrt{\log(|\mathcal{B}_{net}(\epsilon)|)/N}$, the total average regret is bounded by,*

$$\frac{1}{N}\mathbb{E}[\mathcal{R}(N)] \leq C\mathcal{B}_{\max}\sqrt{\frac{\log(\mathcal{B}_{\max}\mathcal{L}\sqrt{N})}{N}} = O(\sqrt{\frac{\log(N)}{N}}),$$

for some positive constant $C > 0$. Here, the expectation is taken over the randomness in Algorithm 1. Hence, Property 3.2 (Revenue Maxmization) holds.

Theorem 5.2 proves Algorithm 1 is a zero regret algorithm. We note the bound is independent of M, the number of features sold. See Appendix B for the proof.

As we note in Remark 4.2, a limitation of the $\mathcal{AF}^*$ we design is that it is fixed, i.e., we degrade each feature by the same scaling. We leave it as future work to design an adaptive $\mathcal{AF}$; instead of fixing $\mathcal{AF}^*$ a priori (as we currently do using standard noising procedures), can we make the noising procedure adaptive to the prediction tasks to further increase the revenue generated (potentially by adding distributional assumptions to the prediction tasks)?

5.4 Fairness in Revenue Division.

THEOREM 5.3. *Let $\psi_{n,shapley}$ be the unique vector satisfying Property 3.3 (Shapley Fairness) as given in* (2)*. For Algorithm 2, pick the following hyperparameter: $K > (M\log(2/\delta))/(2\epsilon^2)$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output $\hat{\psi}_n$ of Algorithm 2, achieves the following,*

$$||\psi_{n,shapley} - \hat{\psi}_n||_\infty < \epsilon.$$

Theorem 5.3 gives an ϵ-approximation for $\psi_{n,\text{shapley}}$, the *unique* vector satisfying Property 3.3, in $O(M)$. Recall, computing it exactly would take $\Theta(2^M)$ running time. See Appendix C for the proof.

To the best of our knowledge, the direct application of random sampling to compute feature importances for ML algorithms along with finite sample guarantees is novel. We believe this random sampling method could be used as a model-agnostic tool (not dependent on the particulars of the prediction model used) to assess feature importance - a prevalent question for data scientists seeking interpretability from their prediction models.

THEOREM 5.4. *Let Assumption 2 hold. For Algorithm 3, pick the following hyperparameters: $K \geq (M\log(2/\delta))/(2(\epsilon/3)^2)$, $\lambda = \log(2)$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output, ψ_n, of Algorithm 3 is ϵ-"Robust to Replication" i.e. Property 3.4 (Robustness-to-Replication) holds. Additionally Conditions 2-4 of Property 3.3 continue to hold for ψ_n with ϵ-precision.*

Theorem 5.4 states Algorithm 3 protects against adversarial replication of data, while maintaining the conditions of the standard Shapley fairness other than balance. Again, the key observation, which makes Algorithm 3 possible is that we can precisely compute similarity between data streams (see Definition 4.1). See Appendix C for the proof.

A natural question is whether Property 3.4 and Condition 1 of Property 3.3 and can hold together. Unfortunately, as we see from Proposition 5.1, they cannot (see Appendix C for the proof),

Proposition 5.1. *If the identities of sellers in the marketplace is anonymized, the balance condition in Property 3.3 and Property 3.4 cannot simultaneously hold.*

Note however, Algorithm 3, down-weights features in a "local" fashion i.e. highly correlated features are individually down-weighted, while uncorrelated features are not. *Hence, Algorithm 3 incentivizes sellers to provide data that is: (i) predictive for a wide variety of tasks; (ii) uncorrelated with other features on sale i.e. has unique information.*

In Step 2 of Algorithm 3, we exponentially penalize (i.e. down weight) each feature, for a given similarity metric, $\mathcal{SM}$. An open question for future work is which revenue division mechanism is the most balanced preserving while being robust to replication? As an important first step, we provide a necessary and sufficient condition for any penalty function [8] to be robust to replication for a given similarity metric, $\mathcal{SM}$ (see Appendix E for the proof),

Proposition 5.2. *Let Assumption 2 hold. Then for a given similarity metric $\mathcal{SM}$, a penalty function f is "robust-to-replication" if and only if it satisfies the following relation for any $c \in \mathbb{Z}_+, x \in \mathbb{R}_+$,*

$$(c+1)f(x+c) \leq f(x)$$

5.5 Efficiency.

Corollary 5.1. *$\mathcal{AF}^*, \mathcal{RF}^*, \mathcal{PF}^*$ run in $O(M)$. $\mathcal{PD}_a^*, \mathcal{PD}_b^*$ run in $O(M^2)$. Property 3.5 holds.*

See Appendix D for the proof. $\mathcal{AF}^*, \mathcal{RF}^*, \mathcal{PF}^*$ running in $O(M)$ is desirable as they need to be re-computed in real-time for every buyer. However, the revenue division algorithms (which run in $O(M^2)$) can conceivably run offline as we assume the sellers to be fixed.

6 CONCLUSION

Modeling contributions. Our main contribution is a mathematical model for a two-sided data market (Section 2). We hope our proposed architecture can serve as a foundation to operationalize real-time data marketplaces, which have applicability in a wide variety of important commercial settings (Section 1.2). To further this goal, we define key challenges (Section 3), construct algorithms to meet these challenges (Section 4) and theoretically analyze their performance (Section 5).

To make the problem tractable, we make some key modeling choices. Two of the most pertinent ones include: (i) Buyer n's utility comes solely from the quality of the estimate $\hat{Y}_n$ received, rather than the particular datasets allocated (Definition 2.1); (ii) the marketplace is allowed to centrally set prices for all features for each buyer, rather than sellers individually setting prices for each feature (Remark 2.5).

Technical contributions. We highlight two technical contributions, which might be of independent interest. First, a new notion of "fairness" required for cooperative games with freely replicable goods (and associated algorithms). As stated earlier (Section 3.3.2), such a notion of fairness is especially important in modern applications where users get utility/cost from bundles of digital goods with potentially complex combinatorial interactions (e.g. battery cost attribution for smartphone applications, reward allocation in prediction markets). Second, a truthful, zero regret mechanism for auctioning a particular class of combinatorial goods, which utilizes Myerson's payment function and the Multiplicative Weights algorithm. Specifically, if one can find a way of modeling buyer utility/cost through a scalar parameter (e.g. number of unique views for multimedia ad campaigns, total battery usage for smartphone apps), then the framework described can potentially be applied.

[8] We define a general penalty function to be of the form $\hat{\psi}_n(m) f(\cdot)$, instead of $\hat{\psi}_n(m) \exp(-\lambda \sum_{j \in [M] \setminus \{m\}} \mathcal{SM}(X_m, X_j))$ as in Step 2 of Algorithm 3.

Future Work. We reiterate some interesting lines of questioning for future work: (i) how to take into account the externalities of replication experienced by buyers (Remark 2.7); (ii) how to design an adaptive allocation function that further increases revenue generated (Remark 4.2); (iii) which "robust-to-replication" revenue division mechanism is the most balanced preserving (Section 5.4)?

ACKNOWLEDGMENTS

During this work, the authors were supported in part by a MIT Institute for Data, Systems and Society (IDSS) WorldQuant and Thompson Reuters Fellowship.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-scale Machine Learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI'16)*. USENIX Association, Berkeley, CA, USA, 265–283. http://dl.acm.org/citation.cfm?id=3026877.3026899

[2] Bill Aiello, Yuval Ishai, and Omer Reingold. 2001. Priced Oblivious Transfer: How to Sell Digital Goods. In *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 119–135.

[3] Sanjeev Arora, Elad Hazan, and Satyen Kale. 2012. The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. *Theory of Computing* 8, 6 (2012), 121–164. https://doi.org/10.4086/toc.2012.v008a006

[4] Peter Auer. 2003. Using Confidence Bounds for Exploitation-exploration Trade-offs. *Journal of Machine Learning Research* 3 (March 2003), 397–422. http://dl.acm.org/citation.cfm?id=944919.944941

[5] Moshe Babaioff, Robert Kleinberg, and Renato Paes Leme. 2012. Optimal Mechanisms for Selling Information. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC '12)*. ACM, New York, NY, USA, 92–109.

[6] Yoram Bachrach, Evangelos Markakis, Ezra Resnick, Ariel D. Procaccia, Jeffrey S. Rosenschein, and Amin Saberi. 2010. Approximating power indices: theoretical and empirical analysis. *Autonomous Agents and Multi-Agent Systems* 20, 2 (01 Mar 2010), 105–122.

[7] Eric Balkanski, Umar Syed, and Sergei Vassilvitskii. 2017. Statistical Cost Sharing. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 6221–6230. http://papers.nips.cc/paper/7202-statistical-cost-sharing.pdf

[8] Siddhartha Banerjee, Ramesh Johari, and Carlos Riquelme. 2015. Pricing in Ride-Sharing Platforms: A Queueing-Theoretic Approach. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation (EC '15)*. ACM, New York, NY, USA, 639–639. https://doi.org/10.1145/2764468.2764527

[9] Dirk Bergemann, Alessandro Bonatti, and Alex Smolin. 2018. The Design and Price of Information. *American Economic Review* 108, 1 (January 2018), 1–48. https://doi.org/10.1257/aer.20161079

[10] Bernard Caillaud and Bruno Jullien. 2003. Chicken & Egg: Competition Among Intermediation Service Providers. *The RAND Journal of Economics* 34, 2 (2003), 309–328. http://www.jstor.org/stable/1593720

[11] Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial Calculation of the Shapley Value Based on Sampling. *Computer and Operations Research* 36, 5 (May 2009), 1726–1730. https://doi.org/10.1016/j.cor.2008.04.004

[12] M. Keith Chen. 2016. Dynamic Pricing in a Labor Market: Surge Pricing and Flexible Work on the Uber Platform. In *Proceedings of the 2016 ACM Conference on Economics and Computation (EC '16)*. ACM, New York, NY, USA, 455–455. https://doi.org/10.1145/2940716.2940798

[13] Rachel Cummings, Katrina Ligett, Aaron Roth, Zhiwei Steven Wu, and Juba Ziani. 2015. Accuracy for Sale: Aggregating Data with a Variance Constraint. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS '15)*. ACM, New York, NY, USA, 317–324. https://doi.org/10.1145/2688073.2688106

[14] Constantinos Daskalakis. 2015. Multi-item Auctions Defying Intuition? *SIGecom Exch.* 14, 1 (Nov. 2015), 41–75. https://doi.org/10.1145/2845926.2845928

[15] C. Daskalakis and V. Syrgkanis. 2016. Learning in Auctions: Regret is Hard, Envy is Easy. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. 219–228. https://doi.org/10.1109/FOCS.2016.31

[16] Arpita Ghosh and Aaron Roth. 2011. Selling Privacy at Auction. In *Proceedings of the 12th ACM Conference on Electronic Commerce (EC '11)*. ACM, New York, NY, USA, 199–208. https://doi.org/10.1145/1993574.1993605

[17] Renato Gomes. 2014. Optimal Auction Design in Two-Sided Markets. *The RAND Journal of Economics* 45, 2 (2014), 248–272.

[18] A Ardeshir Goshtasby. 2012. Similarity and Dissimilarity Measures. In *Image Registration*. Springer, 7–66.

[19] Robin Hanson. 2012. Logarithmic markets coring rules for modular combinatorial information aggregation. *The Journal of Prediction Markets* 1, 1 (2012), 3–15.

[20] Elad Hazan et al. 2016. Introduction to online convex optimization. *Foundations and Trends® in Optimization* 2, 3-4 (2016), 157–325.

[21] Daniel P Heyman and Matthew J Sobel. 2004. *Stochastic Models in Operations Research, Volume II. Stochastic Optimization.* Vol. 2. Courier Corporation.

[22] Katrina Ligett and Aaron Roth. 2012. Take It or Leave It: Running a Survey When Privacy Comes at a Cost. In *Internet and Network Economics*, Paul W. Goldberg (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 378–391.

[23] De Liu and Jianqing Chen. 2006. Designing online auctions with past performance information. *Decision Support Systems* 42, 3 (2006), 1307–1320.

[24] Yungao Ma, Nengmin Wang, Ada Che, Yufei Huang, and Jinpeng Xu. 2013. The Bullwhip Effect on Product Orders and Inventory: A Perspective of Demand Forecasting Techniques. *International Journal of Production Research* 51, 1 (2013), 281–302.

[25] Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. 2013. Bounding the Estimation Error of Sampling-based Shapley Value Approximation With/Without Stratifying. *CoRR* abs/1306.4265 (2013).

[26] I Mann and LS Shapley. 1952. *Values for large games IV: Evaluating the electoral college exactly.* Technical Report. RAND Corp Santa Monica CA.

[27] Aranyak Mehta et al. 2013. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science* 8, 4 (2013), 265–368.

[28] Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, Doris Xin, Reynold Xin, Michael J. Franklin, Reza Zadeh, Matei Zaharia, and Ameet Talwalkar. 2016. MLlib: Machine Learning in Apache Spark. *Journal of Machine Learning Research* 17, 34 (2016), 1–7.

[29] Roger B Myerson. 1981. Optimal auction design. *Mathematics of operations research* 6, 1 (1981), 58–73.

[30] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2825–2830. http://dl.acm.org/citation.cfm?id=1953048.2078195

[31] John G Riley and William F Samuelson. 1981. Optimal Auctions. *The American Economic Review* 71, 3 (1981), 381–392.

[32] Jean-Charles Rochet and Jean Tirole. 2003. Platform Competition in Two-Sided Markets. *Journal of the European Economic Association* 1, 4 (2003), 990–1029.

[33] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. 2018. Adversarially Robust Generalization Requires More Data. In *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 5019–5031. http://papers.nips.cc/paper/7749-adversarially-robust-generalization-requires-more-data.pdf

[34] LS Shapley. 1952. *A VALUE FOR N-PERSON GAMES.* Technical Report. RAND Corp Santa Monica CA.

[35] Hal R Varian. 2009. Online Ad Auctions. *American Economic Review* 99, 2 (2009), 430–34.

[36] Justin Wolfers and Eric Zitzewitz. 2004. Prediction markets. *Journal of economic perspectives* 18, 2 (2004), 107–126.

[37] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal Real-time Bidding for Display Advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. ACM, New York, NY, USA, 1077–1086. https://doi.org/10.1145/2623330.2623633

A TRUTHFULNESS

Theorem A.1 (**Theorem 5.3**). *For $\mathcal{AF}^*$, Property 3.1 (Truthfulness) can be achieved if and only if Assumption 1 holds. In which case, $\mathcal{RF}^*$ guarantees truthfulness.*

Proof. This is a classic result from [29]. We provide the arguments here for completeness and for consistency with the properties and notation we introduce in our work. We begin with the backward direction. By Assumption 1 the following then holds $\forall\ b'_n \geq b_n$

$$\mathcal{G}(Y_n, \mathcal{M}(\mathcal{AF}^*(b'_n, p_n))) \geq \mathcal{G}(Y_n, \mathcal{M}(\mathcal{AF}^*(b_n, p_n))) \tag{4}$$

To simplify notation, let $h(z;\ \mathcal{G}, p_n, Y_n, \mathcal{M}) = \mathcal{G}(Y_n, \mathcal{M}(\mathcal{AF}^*(z, p_n)))$. In words, $h(z)$ is the gain in prediction accuracy as a function of the bid, z, for a fixed $\mathcal{G}, Y_n, \mathcal{M}, p_n$.

By definition of (1), it suffices to show that if $b_n \neq \mu_n$, the following holds

$$\mu_n \cdot h(\mu_n) - \mu_n \cdot h(\mu_n) + \int_0^{\mu_n} h(z)dz \geq \mu_n \cdot h(b_n) - b_n \cdot h(b_n) + \int_0^{b_n} h(z)dz \tag{5}$$

This is equivalent to showing that

$$\int_0^{\mu_n} h(z)dz \geq \int_0^{b_n} h(z)dz - (b_n - \mu_n) \cdot h(b_n) \tag{6}$$

Case 1: $b_n > \mu_n$. In this case, (6) is equivalent to

$$(b_n - \mu_n) \cdot h(b_n) \geq \int_{\mu_n}^{b_n} h(z)dz \tag{7}$$

This is immediately true due to monotonicity of $h(z)$ which comes from (4). Case 2: $b_n < \mu_n$. In this case, (6) is equivalent to

$$\int_{b_n}^{\mu_n} h(z)dz \geq (\mu_n - b_n) \cdot h(b_n) \tag{8}$$

Again, this is immediately true due to monotonicity of $h(z)$.

Now we prove the opposite direction, i.e. if we have a truthful payment mechanism, which we denote as $\mathcal{RF}'$, an increased allocation of features cannot decrease accuracy. Our definition of a truthful payment function implies the following two inequalities $\forall\ b > a$

$$a \cdot h(a) - \mathcal{RF}'(\cdot, a, \cdot) \geq a \cdot h(b) - \mathcal{RF}'(\cdot, b, \cdot) \tag{9}$$

$$b \cdot h(b) - \mathcal{RF}'(\cdot, b, \cdot) \geq b \cdot h(a) - \mathcal{RF}'(\cdot, a, \cdot) \tag{10}$$

These two inequalities imply

$$a \cdot h(a) + b \cdot h(b) \geq a \cdot h(b) + b \cdot h(a) \implies h(b)(b - a) \geq h(a)(b - a) \tag{11}$$

Since by construction $b - a > 0$, we can divide both sides of the inequality by $b - a$ to get

$$h(b) \geq h(a) \iff \mathcal{G}_n(Y_n, \mathcal{M}(\mathcal{AF}^*(b, p_n))) \geq \mathcal{G}_n(Y_n, \mathcal{M}(\mathcal{AF}^*(a, p_n))) \tag{12}$$

Since the allocation function $\mathcal{AF}^*(z, p_n)$ is increasing in z, this completes the proof. □

B PRICE UPDATE - PROOF OF THEOREM 5.2

THEOREM B.1 (**THEOREM 5.2**). *Let Assumptions 1, 3 and 4 hold. Let $p_{n:n\in[N]}$ be the output of Algorithm 1. Let $\mathcal{L}$ be the Lipschitz constant of $\mathcal{RF}^*$ with respect to price (where $\mathcal{L}$ is defined as in Assumption 3). Let $\mathcal{B}_{\max} \in \mathbb{R}$ be the maximum element of $\mathcal{B}$ (where $\mathcal{B}$ is defined as in Assumption 4). Then by choosing algorithm hyper-parameters $\epsilon = \frac{1}{\mathcal{L}\sqrt{N}}$, $\delta = \sqrt{\frac{\log(|\mathcal{B}_{net}(\epsilon)|)}{N}}$ for some positive constant $C > 0$, the total average regret is bounded by,*

$$\frac{1}{N}\mathbb{E}[\mathcal{R}(N)] \leq C\mathcal{B}_{\max}\sqrt{\frac{\log(\mathcal{B}_{\max}\mathcal{L}\sqrt{N})}{N}} = O(\sqrt{\frac{\log(N)}{N}}).$$

where the expectation is taken over the randomness in Algorithm 1. Hence, Property 3.2 (Revenue Maxmization) holds.

PROOF. Our proof here is an adaptation of the classic result from [3]. We provide the arguments here for completeness and for consistency with the properties and notation we introduce in our work. It is easily seen by Assumption 1 that the revenue function $\mathcal{RF}^*$ is non-negative. Now since by construction the gain function, $\mathcal{G} \in [0, 1]$, the range of $\mathcal{RF}^*$ is in $[0, \mathcal{B}_{\max}]$. This directly implies that for all i and n, $g_n^i \in [0, 1]$ (recall g_n^i is the (normalized) revenue gain if we played price i for every buyer n).

We first prove a regret bound for the best fixed price in hindsight within $\mathcal{B}_{\text{net}}(\epsilon)$. Let g_n^{alg} be the expected (normalized) gain of Algorithm 1 for buyer n. By construction,

$$g_n^{\text{alg}} = \frac{\sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i g_n^i}{W_n}$$

Observe we have the following inductive relationship regarding W_n

$$W_{n+1} = \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_{n+1}^i \tag{13}$$

$$= \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i + \delta w_n^i g_n^i \tag{14}$$

$$= W_n + \delta \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i g_n^i \tag{15}$$

$$= W_n(1 + \delta g_n^{\text{alg}}) \tag{16}$$

$$= W_1\Pi_{i=1}^n(1 + \delta g_n^{\text{alg}}) \tag{17}$$

$$\overset{(a)}{=} |\mathcal{B}_{\text{net}}(\epsilon)| \cdot \Pi_{i=1}^n(1 + \delta g_n^{\text{alg}}) \tag{18}$$

where (a) follows since W_1 was initialized to be $|\mathcal{B}_{\text{net}}(\epsilon)|$.

Taking logs and utilizing the inequality $\log(1 + x) \leq x$ for $x \geq 0$, we have

$$\log(W_{N+1}) = \log(|\mathcal{B}_{\text{net}}(\epsilon)|) + \sum_{i=1}^{N} \log(1 + \delta g_n^{\text{alg}})) \tag{19}$$

$$\leq \log(|\mathcal{B}_{\text{net}}(\epsilon)|) + \sum_{i=1}^{N} \delta g_n^{\text{alg}} \tag{20}$$

Now using that $\log(1+x) \geq x - x^2$ for $x \geq 0$, we have for all prices $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$,

$$\log(W_{N+1}) \geq \log(w^i_{n+1}) \tag{21}$$

$$= \sum_{n=1}^{N} \log(1 + \delta g^i_n)) \tag{22}$$

$$\geq \sum_{n=1}^{N} \delta g^i_n - (\delta g^i_n)^2 \tag{23}$$

$$\overset{(a)}{\geq} \sum_{i=1}^{N} \delta g^i_n - \delta^2 N \tag{24}$$

where (a) follows since $g^i_n \in [0,1]$.

Thus for all prices $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$

$$\sum_{n=1}^{N} \delta g^{\text{alg}}_n \geq \sum_{n=1}^{N} \delta g^i_n - \log(|\mathcal{B}_{\text{net}}(\epsilon)|) - \delta^2 N$$

Dividing by δN and picking $\delta = \sqrt{\frac{\log(|\mathcal{B}_{\text{net}}(\epsilon)|)}{N}}$, we have for all prices $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$

$$\frac{1}{N}\sum_{n=1}^{N} g^{\text{alg}}_n \geq \frac{1}{N}\sum_{i=1}^{N} g^i_n - 2\sqrt{\frac{\log(|\mathcal{B}_{\text{net}}(\epsilon)|)}{N}}$$

So far we have a bound on how well Algorithm 1 performs against prices in $\mathcal{B}_{\text{net}}(\epsilon)$. We now extend it to all of $\mathcal{B}$. Let g^{opt}_n be the (normalized) revenue gain from buyer n if we had played the optimal price, p^* (as defined in Property 3.2). Note that by Assumption 4, we have $p^* \in \mathcal{B}$. Then by the construction of $|\mathcal{B}_{\text{net}}(\epsilon)|$, there exists $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$ such that $|c^i - p^*| \leq \epsilon$. Then by Assumption 3, we have that

$$|g^{\text{opt}}_n - g^i_n| = \frac{1}{\mathcal{B}_{\max}} |\mathcal{RF}^*(p^*, b_n, Y_n) - \mathcal{RF}^*(c^i, b_n, Y_n)| \leq \frac{\mathcal{L}\epsilon}{\mathcal{B}_{\max}}$$

We thus have

$$\frac{1}{N}\sum_{n=1}^{N} g^{\text{alg}}_n \geq \frac{1}{N}\sum_{i=1}^{N} g^{\text{opt}}_n - 2\sqrt{\frac{\log(|\mathcal{B}_{\text{net}}(\epsilon)|)}{N}} - \frac{\mathcal{L}\epsilon}{\mathcal{B}_{\max}}$$

Multiplying throughout by $\mathcal{B}_{\max}$, we get

$$\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}[\mathcal{RF}^*(p_n, b_n, Y_n)] \geq \frac{1}{N}\mathcal{RF}^*(p^*, b_n, Y_n) - 2\mathcal{B}_{\max}\sqrt{\frac{\log(|\mathcal{B}_{\text{net}}(\epsilon)|)}{N}} - \mathcal{L}\epsilon$$

Now setting $\epsilon = \frac{1}{\mathcal{L}\sqrt{N}}$ and noting that $|\mathcal{B}_{\text{net}}(\epsilon)| \leq \frac{3\mathcal{B}_{\max}}{\epsilon}$, for some positive constant $C > 0$, we have

$$\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}[\mathcal{RF}^*(p_n, b_n, Y_n)] \geq \frac{1}{N}\mathcal{RF}^*(p^*, b_n, Y_n) - C\mathcal{B}_{\max}\sqrt{\frac{\log(\mathcal{B}_{\max}\mathcal{L}\sqrt{N})}{N}}$$

□

C FAIRNESS

THEOREM C.1 (**THEOREM 5.3**). *Let $\psi_{n,shapley}$ be the unique vector satisfying Property 3.3 as given in* (2). *For Algorithm 2, pick the following hyperparameter: $K > \frac{M\log(2/\delta)}{2\epsilon^2}$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output $\hat{\psi}_n$ of Algorithm 2, achieves the following*

$$\|\psi_{n,shapley} - \hat{\psi}_n\|_\infty < \epsilon \tag{25}$$

PROOF. It is easily seen that $\psi_{n,\text{shapley}}$ can be formulated as the following expectation

$$\psi_{n,\text{shapley}}(m) = \mathbb{E}_{\sigma\sim\text{Unif}(\sigma_{S_n})}[\mathcal{G}_n(Y_n, \mathcal{M}_n(\boldsymbol{X}_{[\sigma<m\,\cup\, m]})) - \mathcal{G}_n(Y_n, \mathcal{M}_n(\boldsymbol{X}_{[\sigma<m]})] \tag{26}$$

The random variable $\hat{\psi}_n^k(m)$ is distributed in the following manner:

$$\mathbb{P}\Big(\hat{\psi}_n^k(m) = \mathcal{G}_n(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k<m\,\cup\, m]})) - \mathcal{G}_n(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k<m]})); \sigma \in \sigma_{S_n}\Big) = \frac{1}{S_n!} \tag{27}$$

We then have

$$\mathbb{E}[\hat{\psi}_n(m)] = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}[\hat{\psi}_n^k(m)] = \psi_{n,\text{shapley}} \tag{28}$$

Since $\hat{\psi}_n(m)$ has bounded support between 0 and 1, and the $\hat{\psi}_n^k(m)$ are i.i.d, we can apply Hoeffding's inequality to get the following bound

$$\mathbb{P}\Big(|\psi_{n,\text{shapley}} - \hat{\psi}_n(m)| > \epsilon\Big) < 2\exp(\frac{-2\epsilon^2}{K}) \tag{29}$$

By applying a Union bound over all $m \in S_n \le M$, we have

$$\mathbb{P}\Big(\|\psi_{n,\text{shapley}} - \hat{\psi}_n\|_\infty > \epsilon\Big) < 2M\exp(\frac{-2\epsilon^2}{K}) \tag{30}$$

Setting $\delta = 2M\exp(\frac{-2\epsilon^2}{K})$ and solving for K completes the proof. □

THEOREM C.2 (**THEOREM 5.4**). *Let Assumption 2 hold. For Algorithm 3, pick the following hyperparameters: $K \ge \frac{M\log(2/\delta)}{2(\frac{\epsilon}{3})^2}$, $\lambda = \log(2)$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output, ψ_n, of Algorithm 3 is ϵ-"Robust to Replication" i.e. Property 3.4 (Robustness-to-Replication) holds. Additionally Conditions 2-4 of Property 3.3 continue to hold for ψ_n with ϵ-precision.*

PROOF. To reduce notational overhead, we drop the dependence on n of all variables for the remainder of the proof. Let $S = \{X_1, X_2, \ldots, X_K\}$ refer to the original set of allocated features without replication. Let $S^+ = \{X_{(1,1)}, X_{(1,2)}, \ldots, X_{(1,c_1)}, X_{(2,1)}, \ldots, X_{(K,c_K)}\}$ (with $c_i \in \mathbb{N}$), be an appended version of S with replicated versions of the original features, i.e. $X_{(m,i)}$ is the $(i-1)$-th replicated copy of feature X_m.

Let $\hat{\psi}, \hat{\psi^+}$ be the respective outputs of Step 1 of Algorithm 3 for S, S^+ respectively. The total revenue allocation to seller m in the original and replicated setting is given by the following:

$$\psi(m) = \hat{\psi}(m)\exp(-\lambda \sum_{j\in S_m\setminus\{m\}} \mathcal{SM}(X_m, X_j)) \tag{31}$$

$$\psi^+(m) = \sum_{i\in c_m}\hat{\psi}^+\Big((m,i)\Big)\exp(-\lambda \sum_{(j,k)\in S_m^+\setminus\{(m,i)\}} \mathcal{SM}(X_{(m,i)}, X_{(j,k)})) \tag{32}$$

For Property 3.4 to hold, it suffices to show that $\psi^+(m) \leq \psi(m) + \epsilon$. We have that

$$\sum_{i \in c_m} \hat{\psi}^+\Big((m,i)\Big) \exp(-\lambda \sum_{(j,k) \in S_m^+ \setminus \{(m,i)\}} \mathcal{SM}(X_{(m,i)}, X_{(j,k)}))$$

$$\overset{(a)}{\leq} \sum_{i \in c_m} \hat{\psi}^+\Big((m,i)\Big) \exp(-\lambda \sum_{j \in [c_m] \setminus i} \mathcal{SM}(X_{(m,i)}, X_{(m,j)})) \exp(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,i)}, X_{(l,1)}))$$

$$\overset{(b)}{=} \sum_{i \in c_m} \hat{\psi}^+\Big((m,i)\Big) \exp(-\lambda(c_m - 1)) \exp(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,i)}, X_{(l,1)}))$$

$$\overset{(c)}{\leq} c_m \Big(\hat{\psi}^+\Big((m,1)\Big) + \frac{1}{3}\epsilon\Big) \exp(-\lambda(c_m - 1)) \exp(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,1)}, X_{(l,1)}))$$

$$\leq c_m \Big(\hat{\psi}^+\Big((m,1)\Big) + \frac{1}{3}\epsilon\Big) \exp(-\lambda(c_m - 1)) \exp(-\lambda \sum_{j \in S_m \setminus \{m\}} \mathcal{SM}(X_m, X_j))$$

(a) follows since $\lambda, \mathcal{SM}(\cdot) \geq 0$; (b) follows by condition (i) of Definition 4.1; (c) follows from Theorem 5.3;

Hence it suffices to show that $c_m \Big(\hat{\psi}^+\Big((m,1)\Big) + \frac{1}{3}\epsilon\Big) \exp(-\lambda(c_m - 1)) \leq \hat{\psi}(m) + \epsilon \ \forall c_m \in \mathbb{N}$. We have

$$\begin{aligned} c_m \exp(-\lambda(c_m - 1)) \Big(\hat{\psi}^+\Big((m,1)\Big) + \frac{1}{3}\epsilon\Big) &\overset{(d)}{\leq} c_m \exp(-\lambda(c_m - 1)) \Big(\frac{\psi(m)}{c_m} + \frac{2}{3}\epsilon\Big) \\ &\overset{(e)}{\leq} c_m \exp(-\lambda(c_m - 1)) \Big(\psi(m) + \frac{2}{3}\epsilon\Big) \\ &\overset{(f)}{\leq} c_m \exp(-\lambda(c_m - 1)) \Big(\hat{\psi}(m) + \epsilon\Big) \\ &\overset{(g)}{\leq} \Big(\hat{\psi}(m) + \epsilon\Big) \end{aligned}$$

where (d) and (f) follow from Theorem 5.3; (e) follows since $c_m \in \mathbb{N}$; (g) follows since $c_m \exp(-\lambda(c_m - 1)) \leq 1 \ \forall c_m \in \mathbb{N}$ by picking $\lambda = \log(2)$.

The fact that Conditions 2-4 of Property 3.3 continue to hold for follow ψ_n with ϵ-precision follow easily from Theorem 5.3 and the construction of ψ_n. □

Proposition C.1 (**Proposition 5.1**). *If the identities of sellers in the marketplace is anonymized, the balance condition in Property 3.3 and Property 3.4 cannot simultaneously hold.*

Proof. We show this through an extremely simple counter-example consisting of three scenarios.

In the first scenario, the marketplace consists of exactly two sellers, A, B, each selling identical features i.e. $X_A = X_B$. By Condition 1 and 2 of Property 3.3, both sellers must receive an equal allocation i.e. $\psi_1(A) = \psi_1(B) = \frac{1}{2}$ for any prediction task.

Now consider a second scenario, where the marketplace against consists of the same two sellers, A and B, but this time seller A replicates his or her feature once and sells it again in the marketplace as A'. Since by assumption the identity of sellers is anonymized, to achieve the "balance" condition in Property 3.3, we require $\psi_2(A) = \psi_2(B) = \psi_2(A') = \frac{1}{3}$. Thus the total allocation to seller A is $\psi_2(A) + \psi_2(A') = \frac{2}{3} > \frac{1}{2} = \psi_1(A)$ i.e. Property 3.4 does not hold.

Finally consider a third scenario, where the marketplace consists of three sellers A, B and C, each selling identical features i.e. $X_A = X_B = X_C$. It is easily seen that to achieve "balance", we require $\psi_3(A) = \psi_3(B) = \psi_3(C) = \frac{1}{3}$.

Since the marketplace cannot differentiate between A' and C, we either have balance or Property 3.4 i.e. "robustness to replication". □

D EFFICIENCY

Corollary D.1 (**Corollary 5.1**). *$\mathcal{AF}^*, \mathcal{RF}^*, \mathcal{PF}^*$ run in $O(M)$. $\mathcal{PD}_a^*, \mathcal{PD}_b^*$ run in $O(M^2)$ time. Hence, Property 3.5 holds.*

Proof. This is immediately seen by studying the four functions: (i) $\mathcal{AF}^*$ simply tunes the quality of each feature X_j for $j \in [M]$, which is a linear time operation in M; (ii) $\mathcal{RF}^*$ again runs in linear time as we require a constant number of calls to $\mathcal{G}$ and $\mathcal{M}$; (iii) $\mathcal{PF}^*$ runs in linear time as we call $\mathcal{G}$ and $\mathcal{M}$ once for every price in $\mathcal{B}_{\text{net}}(\epsilon)$; (iv) $\mathcal{PD}_a^*$ has a running time of $\frac{M^2 \log(2/\delta)}{2\epsilon^2}$ for any level of precision and confidence given by ϵ, δ respectively i.e. we require $\frac{M \log(2/\delta)}{2\epsilon^2}$ calls to $\mathcal{G}$ and $\mathcal{M}$ to compute the Shapley Allocation for each feature X_j for $j \in [M]$. The additional step in $\mathcal{PD}_b^*$ i.e. Step 2, is also a linear time operation in M (note that the pairwise similarities between X_i, X_j for any $i, j \in [M]$ can be precomputed). □

E OPTIMAL BALANCE-PRESERVING, ROBUST-TO-REPLICATION PENALTY FUNCTIONS

In this section we provide a necessary and sufficient condition for "robustness-to-replication" any penalty function $f : \mathbb{R}_+ \to \mathbb{R}_+$ must satisfy, where f takes as argument the cumulative similarity of a feature with all other features. In Algorithm 3, we provide a specific example of such a penalty function given by exponential down-weighting. We have the following result holds

Proposition E.1 (**Proposition 5.2**). *Let Assumption 2 hold. Then for a given similarity metric $\mathcal{SM}$, a penalty function f is "robust-to-replication" if and only if it satisfies the following relation*

$$(c+1)f(x+c) \leq f(x)$$

where $c \in \mathbb{Z}_+, x \in \mathbb{R}_+$.

Proof. Consider the case where a certain data seller with feature X_i has original cumulative similarity x, and makes c additional copies of its own data. The following relation is both necessary and sufficient to ensure robustness,

$$\hat{\psi}_i(c+1)f(x+c) \leq \psi_i f(x)$$

We first show sufficiency. By Assumption 2, the new Shapley value (including the replicated features) for a single feature X_i denoted by $\hat{\psi}$, is no larger than the original Shapley value, ψ, for the same feature. Then it immediately follows that $(c+1)f(x+c) \leq f(x)$.

We now show that it is also necessary. We study how much the Shapley allocation changes when only one player duplicates data. The Shapley allocation for feature X_i is defined as

$$\psi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} (v(S \cup \{i\}) - v(S))$$

A key observation to computing the new Shapley value is that $v(S \cup \{i\}) - v(S) \geq 0$ if i appears before all its copies. Define M to be the number of original sellers (without copying) and c are the

additional copies. By a counting argument one can show that

$$\begin{aligned}
\hat{\psi}_i(v) &= \sum_{i=0}^{M-1} \frac{1}{(M+c)!} \binom{M-i+c-1}{M-i-1} [v(S \cup \{i\}) - v(S)] \\
&= \sum_{i=0}^{M-1} \frac{M!}{(M+c)!} \binom{M-i+c-1}{M-i-1} \frac{1}{M!} [v(S \cup \{i\}) - v(S)] \\
&= \sum_{i=0}^{M-1} \frac{M!}{(M+c)!} \binom{M-i+c-1}{c} \frac{1}{M!} [v(S \cup \{i\}) - v(S)] \\
&\leq \frac{M}{M+c} \sum_{i=0}^{M-1} \frac{1}{M!} [v(S \cup \{i\}) - v(S)] \\
&= \frac{M}{M+c} \psi_i(v)
\end{aligned}$$

Observe this inequality turns into an equality when all the original sellers have exactly the same data. We observe that for a large number of unique sellers then copying does not change the Shapley allocation too much $\simeq -c/M$. In fact, this bound tells us that when there are a large number of sellers, replicating a single data set a fixed number of times does not change the Shapley allocation too much, *i.e.*, $\hat{\psi}_i \approx \hat{\psi}_i$ (with the approximation being tight in the limit as M tends to infinity). Therefore, we necessarily need to ensure that

$$(c+1) f(x+c) \leq f(x)$$

□

Remark E.1. *If we make the* extremely loose *relaxation of letting $c \in \mathbb{R}_+$ instead of $\mathbb{Z}_+$, then the exponential weighting in Algorithm 3 is minimal in the sense that it ensures robustness with least penalty in allocation. Observe that the penalty function (assuming differentiability) should also satisfy*

$$\begin{aligned}
\frac{f(c+x) - f(x)}{c} &\leq -f(x+c) \\
\lim_{c \to 0^+} \frac{f(c+x) - f(x)}{c} &\leq -f(x) \\
f'(x) &\leq -f(x)
\end{aligned}$$

By Gronwall's Inequality we can see that $f(x) \leq Ce^{-Kx}$ for suitable $C, K \geq 0$. This suggests that the exponential class of penalty ensure robustness with the "least" penalty, and are minimal in that sense.