# A Marketplace for Data: An Algorithmic Solution

**Anish Agarwal, Munther Dahleh and Tuhin Sarkar** *
LIDS, SDSC, IDSS at Massachusetts Institute of Technology
{anish90, dahleh, tsarkar}@mit.edu

## Abstract

In this work, we aim to create 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 vendors 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 rigorously prove 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 for auctioning a particular class of combinatorial goods based on utilizing Myerson's payment function and the Multiplicative Weights algorithm. These might be of independent interest.

## 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 an 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.

**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 (such as the online ad auctions) the commodity (ad-space) is not a replicable good and buyers usually have a strong prior on the value of the commodity (cf. [23, 37]). In contrast for a data market, *it is infeasible for a firm to make bids on specific datasets as they have no 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 a seller artificially restricts the number of replications, which may be suboptimal for maximizing revenue). This problem only gets exacerbated 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* - a capability online ad markets and prediction markets do not 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 mathematically model a full system design for a data marketplace; we rigorously parametrize the participants of our proposed market - the buyers, sellers and the marketplace itself (Section 2.1) - and the mechanism by which they interact (Section 2.2). *To the best of our knowledge, we are the first to lay out an architecture for a data marketplace that 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, 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) how to incentive buyers to report their internal valuations truthfully; (ii) how to update the price for a collection of correlated datasets such that revenue is maximized over time; (iii) how to divide the generated revenue "fairly" among the training features so they get paid for their marginal contribution; (iv) how to construct 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; (ii) a truthful, zero regret mechanism for auctioning a particular class of combinatorial goods based on utilizing Myerson's payment function (cf. [3]) and the Multiplicative Weights algorithm (cf. [29]). 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 (refer to Section 2 for a mathematical formalization of these dynamics). We refer back to it throughout the paper as we introduce various notations and algorithms to explicitly construct such a marketplace.

*Example from Inventory Optimization*: 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 that best forecast future inventory demand. In such a setting, even though a logistics company clearly knows there is some value in these data streams on sale, it is very unrealistic for such a company to have a prior on what collection of foot-traffic data streams are predictive for its demand forecasting and make separate bids on each of them (this is even without taking into account the 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: (i) a 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; (ii) the mechanism then 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 foot-traffic data streams; (iii) revenue is collected based *only on the increased accuracy* in forecasting inventory demand empirically seen and the supplied bid; (iv) revenue is divided amongst all the retail stores who provided foot-traffic data; (v) the price associated with the foot-traffic data streams is then updated.

What we find exciting is that this example can easily be adapted to a variety of commercial settings (e.g. hedge funds sourcing alternative data to predict certain financial instruments, utility companies sourcing electric vehicle charging data to forecast electricity demand during peak hours etc.). Thus we believe the *dynamic described above can potentially 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, specifically 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 being sold (or service being provided) 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 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 - (i) 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 and 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); (ii) 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 particular 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**. See [36] for a detailed overview. Such markets are a recent phenomenon and have generated a lot of interest, rightly so. 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 $\Delta_W$ i.e. predictions on the chance of each outcome; (ii) the true outcome, $w$, is observed; (iii) the market pays the experts based on $\Delta_W$ and $w$. In such literature, payment functions such as those inspired by Kullback–Leibler divergence are utilized as they incentivize “experts" to be truthful (cf. [19]). Despite similarities, prediction markets remain infeasible for data - “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 (not even taking into account combinatorial issues). Thus a data market must instead provide a real-time mechanism to match training features to prediction tasks based on collective empirical predictive value.

**Information Economics**. There has been an exciting recent line of work that directly tackle 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) there is a feedback effect between data sellers and data buyers and incentives for individuals to provide data when they know the information will be used to adjust prices for goods they consume in the future are modeled [9] (e.g. individuals providing health data knowing that it could potentially affect their insurance prices in the future). These are all extremely important lines of work to pursue, but they focus on different (but complementary) objectives. Specifically, referring 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 collecting through their operations. We focus our work on such a setting, and 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

### 2.1 Market Participants

#### 2.1.1 Sellers

Let there be $M$ sellers, each trying to sell streams of data 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 single univariate time series (i.e. a single feature) over $T$ time steps. 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 $\boldsymbol{X}_S$, $S \subset [M]$. In line with the motivation we provide in Section 1.2 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 have at hand.

#### 2.1.2 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 time series over $T$ time steps that buyer $n$ wants to predict well. To avoid confusion, we clarify what we mean by a "time series" for both $Y_n$ and $X_j$ using the inventory optimization example in Section 1.2 - The historical inventory demand over time for the logistics company is the "time series" $Y_n$. Similarly the time stamped foot-traffic data sold by retailers is referred to as the 'time series" of features, $X_j, j \in [M]$. Hence each scalar in $Y_n$ and $X_j$ has an associated time stamp and the "prediction task" in this example would be to forecast inventory demand from time-lagged foot traffic data. Our aim is to keep the mechanism as agnostic as possible to the particulars of the prediction task (i.e. data sellers having no knowledge of the prediction task) and so we think of time stamps as the most natural way to connect features $X_j$ to the labels $Y_n$.

**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 $1 -$ RMSE [2] (root-mean-squared-error). For classification, an example of $\mathcal{G}_n$ is Accuracy [3]. 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 for prediction quality.** $\mu_n \in \mathbb{R}_+$ parametrizes how 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. $\mu_n$ is what a buyer is willing to pay for a unit increase in $\mathcal{G}$.*

Though a seemingly straightforward definition, we view this 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, bur rather from an increase in prediction accuracy of a quantity of interest.*

*Public Bid Supplied to Market*: Note that $\mu_n$ is a private valuation - it is not necessarily what is provided to the marketplace. Let $b_n \in \mathbb{R}_+$ refer to the actual bid revealed to the marketplace (which may not necessarily be equal to $\mu_n$ if the buyer is strategic).

### 2.1.3 Marketplace

The function of the marketplace is to match buyers and sellers as defined above. We formally parameterize a marketplace through the following set of quantities - For $n \in [N]$:

**Price.** $p_n \in \mathbb{R}_+$ is the price associated with the features on sale when buyer $n$ arrives.

**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; for example each buyer 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. [28, 30, 1]).

**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 $\mathcal{AF}$. In Section 4.2, we provide 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 $\psi_n$, 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 associated with each of the features.

### 2.1.4 Buyer Utility

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

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

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

**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.

## 2.2 Marketplace Dynamics

We can now formally define the per-step dynamic within the marketplace (refer to Figure 1 for a more intuitive graphical overview). Whenever a 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 $\mathcal{G}\Big(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}_M)\Big)$ 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 $\psi_n$, where $\psi_n = \mathcal{PD}(Y_n, \widetilde{\boldsymbol{X}}_M; \mathcal{M}, \mathcal{G})$

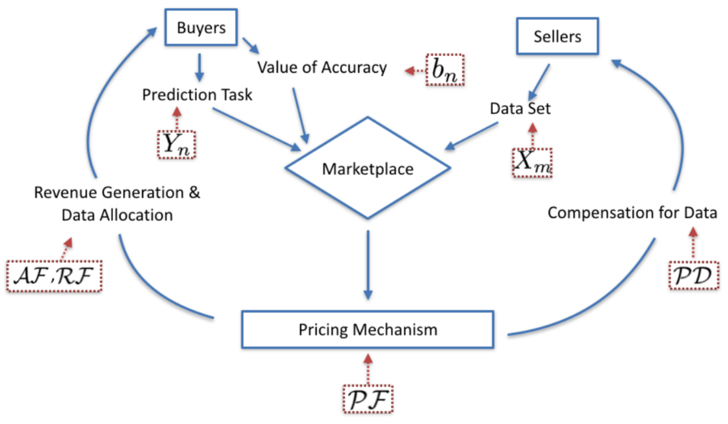

**Figure 1:** Overview of marketplace dynamics.

It is worth noting from Step 3 of the dynamics laid out above, a buyer is "myopic" over a single-stage; it comes into the market once and leaves after being provided the estimate $\hat{Y}_n$, maximizing utility only over that stage. In particular, we do not study the additional complication if the buyer's utility is defined over multiple-stages.

## 3 Desirable Properties of Marketplace

We define the key properties required of such a marketplace for it to be feasible 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. We now formally define these properties.

## 3.1 Truthfulness

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

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

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.

## 3.2 Revenue Maximization

**Property 3.2 (Revenue Maximizing).** *Let $\{(\mu_1, b_1, Y_1), (\mu_2, b_2, Y_2), \dots, (\mu_N, b_N, Y_N)\}$ 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, $\{p_1, p_2, \dots, p_n\}$, such that the "worst-case" average regret, relative to the optimal price $p^*$ in hindsight, goes to $0$,*

$$\lim_{N\to\infty} \frac{1}{N}\Big[\sup_{\{(b_n,Y_n):n\in[N]\}} \Big(\sup_{p^*\in\mathbb{R}_+} \sum_{n=1}^{N} \mathcal{RF}(p^*, b_n, Y_n) - \sum_{n=1}^{N} \mathcal{RF}(p_n, b_n, Y_n)\Big)\Big] = \lim_{N\to\infty} \frac{1}{N}\mathcal{R}(N, M) = 0$$

where $\mathcal{R}(N, M)$ i.e. regret, refers to the expression with the bracket in the middle term. 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, $p^*$ goes to zero as $N$ increases. Note property 3.2 is a robust guarantee - it must hold over the worst case sequence of buyers (i.e. $\mu_n, b_n, Y_n$).

## 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, $\psi_n$):*

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 $\psi_n'$ 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 are the standard axioms of fairness laid out in [34]. 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). Note that if we chose a much simpler notion of fairness such as computing the marginal change in the prediction accuracy with and without every single feature, one at a time, then the correlation between features would lead to the market "undervaluing" each feature. As a toy example of this phenomenon, consider a market with only two identical features on sale. It is easy to see that the simple mechanism above would lead to zero value being allocated to each feature, even though they collectively might have had great value. This is clearly undesirable. That is why Property 3.3 is a necessary notion of fairness as it takes into account the combinatorial nature of the different features, $X_j$.

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

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

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

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

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 $\epsilon$-"robust-to-replication" if $\forall\, n \in [N], \forall\, Y_n$, the following holds on $\mathcal{PD}$:*

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

Property 3.4 is a novel notion of fairness, which can be considered as 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 (refer to 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 where they are both identical to one another. 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 his or 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).*

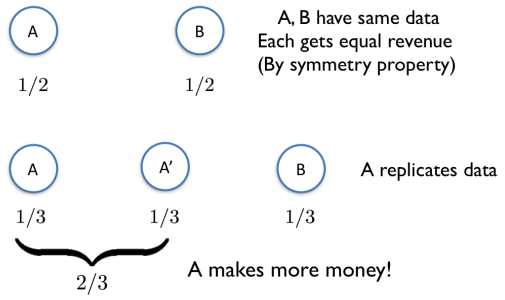

**Figure 2:** Shapley fairness is inadequate for freely replicable goods.

Such a notion of fairness in attribution is especially important in the "computer" age where digital goods can be produced at close to zero marginal cost, and yet users get utility from bundles of digital goods with potentially complex combinatorial interactions between them (e.g. battery cost attribution among smartphone applications, reward allocation among "experts" in a prediction market).

## 3.4 Computational Efficiency

We assume that access to 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 then have 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 2.2 runs in polynomial time in $M$, where $M$ is the number of sellers. In addition, the computation complexity 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 run as close to linear-time in the number of sellers, $M$, as possible and not be dependent 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 $M$. For example, the Shapley Algorithm in Theorem 3.1 runs in $\Theta(2^M)$. Similarly, standard price update algorithms for combinatorial goods satisfying Property 3.2 scale very poorly in $M$ (cf. [15]) - for example, 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 computationally intractable.

## 4 Marketplace Construction

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

### 4.1 Allocation and Revenue Functions

**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 at which buyer $n$ gets allocated the features on sale $\boldsymbol{X}_M$. *In essence, $\mathcal{AF}$ adds noise to/degrades $\boldsymbol{X}_M$ based on the difference between $p_n$ and $b_n$ to supply the features at a quality level the buyers can "afford" (as quantified by $b_n$).* If the bid is greater than the current price (i.e. $b_n > p_n$), then $\boldsymbol{X}_M$ is supplied as is. However if the bid is less than the current price, then the allocation function degrades the set of features being sold in proportion to the difference between $b_n$ and $p_n$. This degradation can take many forms and depends on the structure of $X_j$ itself. Below, we provide some 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 standard 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 $p_n$. While the specific form of the degradation depends on the structure of $X_j$ itself, in Section 5.1 (specifically Assumption 1), we formalize a natural requirement for any such allocation function such that Property 3.1 (i.e. truthfulness) holds.

**Revenue Function.** Observe from Definition 2.1, we parameterize buyer utility through a single parameter $\mu_n$ - how much a buyer values a marginal increase in prediction quality. This crucial modeling choice allows us to use Myerson's celebrated payment function rule (cf. [29]) given below,

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

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

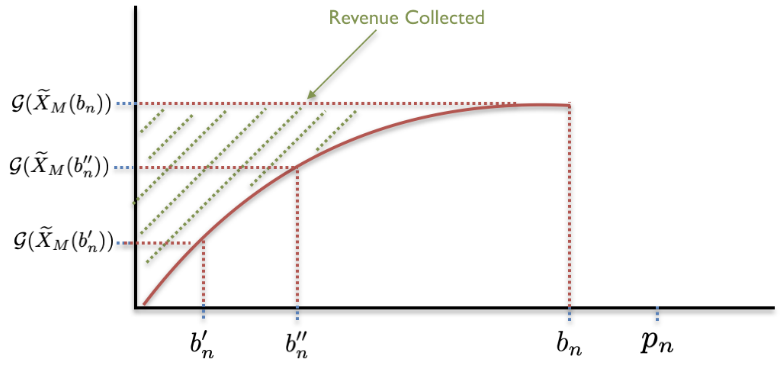

**Figure 3:** Features allocated ($\mathcal{AF}^*$) and revenue collected ($\mathcal{RF}^*$) for a particular price vector $p_n$ and bid $b_n$.

### 4.2 Price Update Function

Recall from Section 3.4 that designing general purpose zero-regret algorithms for combinatorial goods is computationally intractable - if we maintain a separate price for each feature, $X_j$, such algorithms would require exponential running time, and hence Property 3.5 would not be satisfied. This is only exacerbated by the highly non-linear nature of the revenue function $\mathcal{RF}^*$. That is why to achieve a computationally efficient, truthful, zero-regret algorithm, we need to exploit the specific structure of data. We 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 observation we use 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 - the allocation function $\mathcal{AF}(p_n, b_n)$ is then a mechanism to *collectively* adjust the quality of $\boldsymbol{X}_M$ so it is "cheap" enough for buyer $n$ (based on the difference between $p_n$ and $b_n$) as instantiated in Section 4.1.

Thus it suffices to maintain a single price, $p_n \in \mathbb{R}_+$ for all of $\boldsymbol{X}_M$. The market is still tasked with how to pick $p_n$ [4]. 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 and prediction task $(b_n, Y_n)$, 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. However since our particular construction of $\mathcal{AF}^*$ and $\mathcal{RF}^*$ allowed $p_n$ to be a scalar, we can apply off-the-shelf zero-regret algorithms, specifically Multiplicative Weights, to achieve zero-regret as we show in Theorem 5.2.

We now define some quantities needed to construct the price update algorithm. As we make precise in Assumption 4 in Section 5, we assume the bids come from some bounded set, $\mathcal{B} \subset \mathbb{R}_+$. Define $\mathcal{B}_{\max} \in \mathbb{R}$ to be the maximum element of $\mathcal{B}$. Define $\mathcal{B}_{\text{net}}(\epsilon)$ to be a minimal $\epsilon$-net of $\mathcal{B}$ [5]. 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. The price update algorithm is then given in Algorithm 1.

### 4.3 Payment-Division Functions

#### 4.3.1 Shapley Approximation

In our model (as seen in Section 2.2), 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

[4] Recall from Section 2.2 that the market must pick $p_n$ before buyer $n$ arrives. Otherwise no truthfulness guarantees can be made.

[5] We endow $\mathbb{R}$ with the standard Euclidean metric. An $\epsilon$-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 1** PRICE-UPDATE: $\mathcal{PF}^*(b_n, Y_n, \mathcal{B}, \epsilon, \delta)$

```
1: Let $\mathcal{B}_{\text{net}}(\epsilon)$ be an $\epsilon$-net of $\mathcal{B}$
2: for $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$ do
3:     Set $w_1^i = 1$                                   ▷ initialize weights of all experts to 1
4: end for
5: for $n = 1$ to $N$ do
6:     $W_n = \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i$
7:     Let $p_n = c^i$ with probability $w_n^i / W_n$           ▷ note $p_n$ is not a function of $b_n$
8:     for $c^i \in \mathcal{B}_{\text{net}}(\epsilon)$ do
9:         Let $g_n^i = \mathcal{RF}^*(c^i, b_n, Y_n)/\mathcal{B}_{\max}$      ▷ revenue gain if price $c^i$ was used
10:        Set $w_{n+1}^i = w_n^i \cdot (1 + \delta g_n^i)$            ▷ Multiplicative Weights update step
11:    end for
12: end for
13: return $p_n$
```

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. We introduce a randomized version of the Shapley allocation that runs in time $O(M)$, which gives an $\epsilon$-approximation for (2) with high probability (shown in Theorem 5.3). We note some similar sampling based methods, albeit for different applications (cf. [26, 11, 25]).

We first define some quantities needed to construct the revenue division algorithm. 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 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 $\epsilon$-approximation a practitioner desires. We provide guidance on how to pick $K$ in Section 5. The revenue division algorithm is given in Algorithm 2.

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

```
1: for all $m \in [M]$ do
2:     for all $k \in [K]$ do
3:         $\sigma_k \sim \text{Unif}(\sigma_{[M]})$
4:         $G = \mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k < m]}))$
5:         $G^+ = \mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_{[\sigma_k < m \,\cup\, m]}))$
6:         $\hat{\psi}_n^k(m) = [G^+ - G]$
7:     end for
8:     $\hat{\psi}_n(m) = \frac{1}{K} \sum_{k=1}^{K} \hat{\psi}_n^k(m)$
9: end for
10: return $\hat{\psi}_n = [\hat{\psi}_n(m) : m \in [M]]$
```

### 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, 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*

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

**Example 4.4.** *"Inverse" Hellinger distance, a standard metric to define similarity between underlying data distributions, is*

$$1 - \frac{1}{2} \sum_{x \in \mathcal{X}} (\sqrt{p_1(x)} - \sqrt{p_2(x)})^2)^{1/2}, \; p_1 \sim X_1, \; p_2 \sim X_2$$

We now 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, $\mathcal{SM} : \mathbb{R}^{mT} \times \mathbb{R}^{mT} \rightarrow [0, 1]$, 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: $d\mathcal{SM}(X, Y) + d\mathcal{SM}(Y, Z) \geq d\mathcal{SM}(X, Z)$*

Given the observation that we can define a similarity metric for two data vectors (i.e. features), we now have all the necessary notions to define a "robust-to-replication" version of the randomized Shapley approximation algorithm we introduce in Section 4.3.1. The intuition behind the algorithm we design is that it essentially penalizes similar features (relative to the similarity metric, $\mathcal{SM}$) a sufficient amount to disincentive replication. The revenue division algorithm is given in Algorithm 3. See Figure 4 for an illustration of the effect of Algorithm 3 on the original allocation in Example 3.1.

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

```
1: $\hat{\psi}_n(m)$ = SHAPLEY-APPROX($Y_n, \mathcal{M}, \mathcal{G}, K$)
2: $\psi_n(m) = \hat{\psi}_n(m) \exp\Big(-\lambda \sum_{j \in S_m \setminus \{m\}} \mathcal{SM}(X_m, X_j)\Big)$
3: return $\psi_n = [\psi_n(m) : m \in [M]]$
```

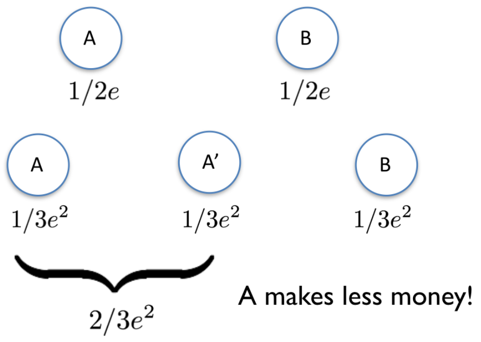

**Figure 4:** A simple example illustrating how SHAPLEY-ROBUST down weights similar data to ensure robustness to replication.

## 5 Main Results

### 5.1 Assumptions.

To give performance guarantees, we state four mild and natural assumptions we need on: (i) $\mathcal{M}$ (prediction algorithm); (ii) $\mathcal{G}$ (prediction gain function); (iii) $\mathcal{AF}^*$ (allocation function) (iv) $b_n$ (bids made). We list them below.

**Assumption 1.** *$\mathcal{M}, \mathcal{G}, \mathcal{AF}^*$ 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 $\widetilde{\boldsymbol{X}}^{(1)}_M, \widetilde{\boldsymbol{X}}^{(2)}_M$ be the outputs of $\mathcal{AF}(p_n, b^{(1)}; \boldsymbol{X}_M), \mathcal{AF}(p_n, b^{(2)}; \boldsymbol{X}_M)$ respctively. Then if $b^{(1)} \leq b^{(2)}$, we have $\mathcal{G}\Big(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}^{(1)}_M)\Big) \leq \mathcal{G}\Big(Y_n, \mathcal{M}(\widetilde{\boldsymbol{X}}^{(2)}_M)\Big)$.*

**Assumption 2.** *$\mathcal{M}, \mathcal{G}$ are 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 $\mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_S)) = \mathcal{G}(Y_n, \mathcal{M}(\boldsymbol{X}_{S^+}))$*

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

**Assumption 4.** *For all steps $n$, let the set of possible bids $b_n$ come from a closed, bounded set $\mathcal{B}$ i.e. $b_n \in \mathcal{B}$, where diameter$(\mathcal{B}) = D$, where $D < \infty$.*

## 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 definition of buyer utility in Definition 2.1 - it lets us parameterize a buyers value for increased accuracy by a scalar, $\mu_n$, which is what allows us to use Myerson's payment function (unfortunately generalization of Myerson's payment function to the setting where $\mu_n$ 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 $\mathcal{RF}^*$ 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 the algorithm hyper-parameters $\epsilon = \frac{1}{\mathcal{L}\sqrt{N}}$, $\delta = \sqrt{\frac{\log(|\mathcal{B}_{net}(\epsilon)|)}{N}}$ we have that for some positive constant $C > 0$, the total average regret is bounded by,*

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

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

Theorem 5.2 proves Algorithm 1 is a zero regret algorithm (where the bound is independent of the number of features sold since the maximum prediction gain is bounded above i.e. $\mathcal{G}(\cdot) \leq 1$). See Appendix B for the proof. Note that a limitation of our mechanism is that $\mathcal{AF}^*$ is fixed and we degrade each feature by the same scaling (since $p_n$ is the same across features). An interesting line of future work would be to study whether we can make the allocation $\mathcal{AF}$ adaptive - so far we fix $\mathcal{AF}^*$ a priori, using some standard notions from the literature but it could potentially be made adaptive to the prediction tasks that arrive over time to further increase the revenue generated.

## 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: $K > \frac{M\log(2/\delta)}{2\epsilon^2}$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output $\hat{\psi_n}$ of Algorithm 2, achieves the following,*

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

Theorem 5.3 is worth noting as it gives an $\epsilon$-approximation for $\psi_{n,\text{shapley}}$, the *unique* vector satisfying Property 3.3, in $O(M)$. In comparison, to compute it exactly would take $\Theta(2^M)$ complexity. 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. See Appendix C for the proof.

**Theorem 5.4.** *Let Assumption 2 hold. For Algorithm 3, pick the following hyperparameters: $K \geq \frac{M\log(2/\delta)}{2(\frac{\epsilon}{3})^2}$, $\lambda = \log(2)$, where $\delta, \epsilon > 0$. Then with probability $1-\delta$, the output, $\psi_n$, of Algorithm 3 is $\epsilon$-"Robust to Replication" i.e. Property 3.4 (Robustness-to-Replication) holds. Additionally Conditions 2-4 of Property 3.3 continue to hold for $\psi_n$ with $\epsilon$-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. See Appendix C for the proof. The key observation is that unlike general digital goods, there is a precise way to compute how similar one data stream is to another (refer to Definition 4.1). A natural question to ask is whether Condition 1 of Property 3.3 and Property 3.4 can hold together. Unfortunately they cannot,

**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 the other features on sale i.e. has unique information.*

In 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 allocation mechanism is the most balanced preserving while being robust to replication? We provide a necessary and sufficient condition for a penalty function to be robust to replication for a given similarity metric, $\mathcal{SM}$, below,

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

See Appendix E for a proof.

### 5.5 Efficiency.

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

## 6 Conclusion

In this paper, we introduce the first mathematical model for a two-sided data market - we define key challenges, construct algorithms, and give theoretical performance guarantees. We highlight two technical contributions: (i) a new notion of "fairness" required for cooperative games with freely replicable goods (and associated algorithms); (ii) a truthful, zero regret mechanism for auctioning a particular class of combinatorial goods based on utilizing Myerson's payment function and the Multiplicative Weights algorithm. There might exist applications of our overall framework and the corresponding theoretical guarantees outside data markets for other types of combinatorial goods, when there is a need to design efficient, truthful, zero-regret payment and pricing mechanisms. We believe the key requirement in such a setting is to find a way to model buyer utility through a scalar parameter (e.g. number of unique views for multimedia ad campaigns, total battery usage for smartphone apps). Our framework might be of independent interest in such use cases. We end with some interesting lines of questioning for future work: (i) Which revenue division mechanism is the most balanced preserving while being robust to replication? (ii) Note that in this work we do not take into account an important attribute of data - a 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 them). So an important question to answer is how does a firm efficiently convey to the market (through their bidding) the externalities they experiences associated with a dataset being replicated too many times and how will the associated mechanism compare with what we describe in this work?

## References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016.

[2] B. Aiello, Y. Ishai, and O. Reingold. Priced oblivious transfer: How to sell digital goods. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 119–135. Springer, 2001.

[3] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[4] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

[5] M. Babaioff, R. Kleinberg, and R. Paes Leme. Optimal mechanisms for selling information. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, EC '12, pages 92–109, New York, NY, USA, 2012. ACM.

[6] Y. Bachrach, E. Markakis, E. Resnick, A. D. Procaccia, J. S. Rosenschein, and A. Saberi. Approximating power indices: theoretical and empirical analysis. *Autonomous Agents and Multi-Agent Systems*, 20(2):105–122, Mar 2010.

[7] E. Balkanski, U. Syed, and S. Vassilvitskii. Statistical cost sharing. In *Advances in Neural Information Processing Systems*, pages 6222–6231, 2017.

[8] S. Banerjee, C. Riquelme, and R. Johari. Pricing in ride-share platforms: A queueing-theoretic approach. 2015.

[9] D. Bergemann, A. Bonatti, and A. Smolin. The design and price of information. *American Economic Review*, 108(1):1–48, January 2018.

[10] B. Caillaud and B. Jullien. Chicken & egg: Competition among intermediation service providers. *RAND journal of Economics*, pages 309–328, 2003.

[11] J. Castro, D. Gómez, and J. Tejada. Polynomial calculation of the shapley value based on sampling. *Comput. Oper. Res.*, 36(5):1726–1730, May 2009.

[12] M. K. Chen and M. Sheldon. Dynamic pricing in a labor market: Surge pricing and flexible work on the uber platform.

[13] R. Cummings, K. Ligett, A. Roth, Z. S. Wu, and J. Ziani. Accuracy for sale: Aggregating data with a variance constraint. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pages 317–324, New York, NY, USA, 2015. ACM.

[14] C. Daskalakis. Multi-item auctions defying intuition?

[15] C. Daskalakis and V. Syrgkanis. Learning in auctions: Regret is hard, envy is easy. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 219–228, Oct 2016.

[16] A. Ghosh and A. Roth. Selling privacy at auction. In *Proceedings of the 12th ACM Conference on Electronic Commerce*, EC '11, pages 199–208, New York, NY, USA, 2011. ACM.

[17] R. Gomes. Optimal auction design in two-sided markets. *The RAND Journal of Economics*, 45(2):248–272, 2014.

[18] A. A. Goshtasby. Similarity and dissimilarity measures. In *Image registration*, pages 7–66. Springer, 2012.

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

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

[21] D. P. Heyman and M. J. Sobel. *Stochastic models in operations research: stochastic optimization*, volume 2. Courier Corporation, 2004.

[22] K. Ligett and A. Roth. Take it or leave it: Running a survey when privacy comes at a cost. In P. W. Goldberg, editor, *Internet and Network Economics*, pages 378–391, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

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

[24] Y. Ma, N. Wang, A. Che, Y. Huang, and J. Xu. The bullwhip effect on product orders and inventory: a perspective of demand forecasting techniques. *International Journal of Production Research*, 51(1):281–302, 2013.

[25] S. Maleki, L. Tran-Thanh, G. Hines, T. Rahwan, and A. Rogers. Bounding the estimation error of sampling-based shapley value approximation with/without stratifying. *CoRR*, abs/1306.4265, 2013.

[26] I. Mann and L. Shapley. Values for large games iv: Evaluating the electoral college exactly. Technical report, RAND Corp Santa Monica CA, 1952.

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

[28] X. Meng, J. K. Bradley, B. Yavuz, E. R. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. B. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar. Mllib: Machine learning in apache spark. *CoRR*, abs/1505.06807, 2015.

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

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, Nov. 2011.

[31] J. G. Riley and W. F. Samuelson. Optimal auctions. *The American Economic Review*, 71(3):381–392, 1981.

[32] J.-C. Rochet and J. Tirole. Platform competition in two-sided markets. *Journal of the european economic association*, 1(4):990–1029, 2003.

[33] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry. Adversarially robust generalization requires more data. *CoRR*, abs/1804.11285, 2018.

[34] L. Shapley. A value for n-person games. Technical report, RAND Corp Santa Monica CA, 1952.

[35] H. R. Varian. Online ad auctions. *American Economic Review*, 99(2):430–34, 2009.

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

[37] W. Zhang, S. Yuan, and J. Wang. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1077–1086, New York, NY, USA, 2014. ACM.

## 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 $\epsilon = \frac{1}{\mathcal{L}\sqrt{N}}$, $\delta = \sqrt{\frac{\log(|\mathcal{B}_{net}(\epsilon)|)}{N}}$ we have that for some positive constant $C > 0$, the total average regret is bounded by,*

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

*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, $\mathcal{G} \in [0, 1]$, we have that 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 $g_n^{\text{alg}}$ be the expected (normalized) gain of Algorithm 1 for buyer $n$. By construction we have that

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

Observe we have the following inductive relationship regarding $W_n$

$$\begin{aligned}
W_{n+1} &= \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_{n+1}^i && (13)\\
&= \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i + \delta w_n^i g_n^i && (14)\\
&= W_n + \delta \sum_{i=1}^{|\mathcal{B}_{\text{net}}(\epsilon)|} w_n^i g_n^i && (15)\\
&= W_n(1 + \delta g_n^{\text{alg}}) && (16)\\
&= W_1 \Pi_{i=1}^n (1 + \delta g_n^{\text{alg}}) && (17)\\
&\overset{(a)}{=} |\mathcal{B}_{\text{net}}(\epsilon)| \cdot \Pi_{i=1}^n (1 + \delta g_n^{\text{alg}}) && (18)
\end{aligned}$$

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

$$\begin{aligned}
\log(W_{N+1}) &= \log(|\mathcal{B}_{\text{net}}(\epsilon)|) + \sum_{i=1}^{N} \log\big(1 + \delta g_n^{\text{alg}}\big)) && (19)\\
&\leq \log(|\mathcal{B}_{\text{net}}(\epsilon)|) + \sum_{i=1}^{N} \delta g_n^{\text{alg}} && (20)
\end{aligned}$$

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

$$\begin{aligned}
\log(W_{N+1}) &\geq \log\big(w_{n+1}^i\big) && (21)\\
&= \sum_{n=1}^{N} \log\big(1 + \delta g_n^i\big)) && (22)\\
&\geq \sum_{n=1}^{N} \delta g_n^i - (\delta g_n^i)^2 && (23)\\
&\overset{(a)}{\geq} \sum_{i=1}^{N} \delta g_n^i - \delta^2 N && (24)
\end{aligned}$$

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

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

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

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

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

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 $g_n^{\text{opt}}$ 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

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

We thus have

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

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 $|\mathcal{B}_{\text{net}}(\epsilon)| \leq \frac{3\mathcal{B}_{\max}}{\epsilon}$, we have that for some positive constant $C > 0$,

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

□

## C Fairness

**Theorem C.1 (Theorem 5.3).** *Let $\psi_{n,\textit{shapley}}$ 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*

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

*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

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

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

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

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

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

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

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

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

$$\begin{aligned}
&\sum_{i \in c_m} \hat{\psi^+}\Big((m,i)\Big) \exp\left(-\lambda \sum_{(j,k) \in S_m^+ \setminus \{(m,i)\}} \mathcal{SM}(X_{(m,i)}, X_{(j,k)})\right) \\
&\overset{(a)}{\leq} \sum_{i \in c_m} \hat{\psi^+}\Big((m,i)\Big) \exp\left(-\lambda \sum_{j \in [c_m] \setminus i} \mathcal{SM}(X_{(m,i)}, X_{(m,j)})\right) \exp\left(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,i)}, X_{(l,1)})\right) \\
&\overset{(b)}{=} \sum_{i \in c_m} \hat{\psi^+}\Big((m,i)\Big) \exp(-\lambda(c_m - 1)) \exp\left(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,i)}, X_{(l,1)})\right) \\
&\overset{(c)}{\leq} c_m\left(\hat{\psi^+}\Big((m,1)\Big) + \frac{1}{3}\epsilon\right) \exp(-\lambda(c_m - 1)) \exp\left(-\lambda \sum_{l \in S_m \setminus \{m\}} \mathcal{SM}(X_{(m,1)}, X_{(l,1)})\right) \\
&\leq c_m\left(\hat{\psi^+}\Big((m,1)\Big) + \frac{1}{3}\epsilon\right) \exp(-\lambda(c_m - 1)) \exp\left(-\lambda \sum_{j \in S_m \setminus \{m\}} \mathcal{SM}(X_m, X_j)\right)
\end{aligned}$$

(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 $c_m\left(\hat{\psi^+}\Big((m,1)\Big) + \frac{1}{3}\epsilon\right) \exp(-\lambda(c_m - 1)) \leq \hat{\psi}(m) + \epsilon \ \forall c_m \in \mathbb{N}$.

We have

$$c_m \exp(-\lambda(c_m - 1))\left(\hat{\psi^+}\Big((m,1)\Big) + \frac{1}{3}\epsilon\right) \overset{(d)}{\leq} c_m \exp(-\lambda(c_m - 1))\Big(\frac{\psi(m)}{c_m} + \frac{2}{3}\epsilon\Big)$$

$$\overset{(e)}{\leq} c_m \exp(-\lambda(c_m - 1))\Big(\psi(m) + \frac{2}{3}\epsilon\Big)$$
$$\overset{(f)}{\leq} c_m \exp(-\lambda(c_m - 1))\Big(\hat{\psi}(m) + \epsilon\Big)$$
$$\overset{(g)}{\leq} \Big(\hat{\psi}(m) + \epsilon\Big)$$

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 $\psi_n$ with $\epsilon$-precision follow easily from Theorem 5.3 and the construction of $\psi_n$. □

**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 $\epsilon, \delta$ 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, $\psi$, 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

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

□

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.