The Value of Data

by

Dalton James Jones

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2018

Author .... Signature redacted ....
Department of Electrical Engineering and Computer Science
November 30, 2017

Certified by.................... Signature redacted ....
Munther A. Dahleh
William A. Coolidge Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .... Signature redacted ....
Leslie Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

The Value of Data

by

Dalton James Jones

Submitted to the Department of Electrical Engineering and Computer Science
on November 30, 2017, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

Data and information are integral to the modern economic system. Advances in technology have allowed companies to both collect and utilize vast amounts of data. At times this data can be very private and collected surreptitiously. Smartphones and other devices that keep us in constant contact with the internet provide companies like Google and Facebook with a wealth of information to sell. Despite all this, there currently does not exist a systematic way to value data. In the absence of such valuations, gross economic inefficiencies are inevitable. In this thesis, we seek to model ways in which data can be bought, sold, and used fairly in an economic environment. We also develop a theory to value data in different settings. Our models and results are applied to a variety of different domains to demonstrate their efficacy. Results from game theory and mathematical programming allow us to provide fair and efficient allocations of data. This research shows that there exists an efficient and fair method with which to determine the value of information and data and to trade it fairly.

Thesis Supervisor: Munther A. Dahleh
Title: William A. Coolidge Professor of Electrical Engineering and Computer Science

Acknowledgments

First of all, I would like to thank Prof. Munther Dahleh. Without his encouragement and steadfast support, this research would never have been possible. He has been a wonderful source of inspiration and ideas. Invariably, upon meeting with Dr. Dahleh, I have had a renewed feeling of inspiration. I can't say enough about how much I appreciate his unique perspective. He has truly made me feel at home here at MIT. Aside from Dr. Dahleh, I would like to thank Dr. Behrouz Touri, without whom I would never have had the opportunities I have today.

I would like to thank my friends for making grad school the best it could be; Shreya, who has been a great mentor and friend through some of the frustrations of last year; Gavin, who has been a good and supportive friend when it mattered most; Diego and Jason, for being two of greatest office mates I could ask for; Tuhin and Ian for showing the new members of our groups the ropes in both research and life in Boston; Flora, whose constant friendliness and positivity invariably brightens our little corner of LIDS; Anish, without whom I wouldn't have had half the fun or results in the last year; And Scott and Tim, whose lively discussions on everything mathematical and historical kept me going in the darkest times, this wouldn't have been possible without your support.

Finally, I would like to thank my family for their guidance and affection through this whole process. Without their unconditional love and support I would not have had the amazing opportunity to be here. I would especially like to thank my father, who's been my best friend and stood by me through think and thin. You have given me the strength and skills to accomplish anything.

Contents

1 Introduction **9**

1.1 Examples 11

1.2 Previous Work 14

1.3 Contributions of This Thesis 17

1.3.1 Outline 20

2 Preliminaries **21**

2.1 Machine Learning 21

2.1.1 Supervised Learning 22

2.2 Algorithmic Game Theory 27

2.2.1 Mechanism Design 27

2.2.2 Combinatorial Auctions 29

2.3 Valuing Data 30

2.3.1 Individual Data Valuation 32

3 Data Matchings **35**

3.1 Data Matching Mechanisms 35

3.1.1 Separable Data 39

3.1.2 ϵ Separability 41

3.1.3 Simulations 44

3.1.4 Profit Maximization and Optimal Replication 46

4 Integer Programming Methods **49**

4.1 Integer Programming and Incentive Compatibility 49

4.1.1 Representational Complexity 50

4.1.2 Iterative Auctions 52

5 Price Driven Mechanisms **59**

5.1 Greedy Truthful Mechanisms 59

5.2 Competitive Pricing 65

6 Conclusion **71**

6.1 Directions for Future Work 72

Chapter 1

Introduction

In November 1856 at the height of the Crimean war, an unknown nurse arrived in southern Ukraine. What Florence Nightingale found at the British army hospital was unbelievable. In war, tragedies are unavoidable, but what Nightingale observed wasn't battle killing soldiers, it was medicine. Poor hospital sanitation was leading to the bulk of deaths among soldiers. Nightingale used statistics to persuade military leaders to improve sanitation, resulting in a drop in the death rate from 42% to 2%. This dramatic improvement was the product of the acquisition and interpretation of data. The ability to extrapolate from experience and learn from data propelled a little known hominid from its enclave in western Africa to the dominant species on the planet in the evolutionary blink of an eye. For millennia, the knowledge of which rocks made the best arrowheads to which roots were safe to eat to how to use the stars to navigate was indispensable in the expansion and success of our species. This knowledge came from data. Without input, the human learning machine would be lost. Data is the lifeblood of humanity's success.

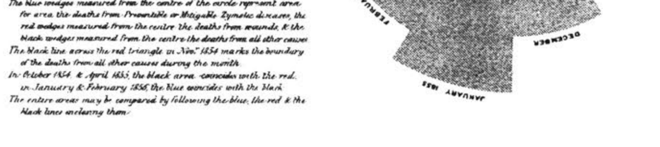


Figure 1-1: Florence Nightingale's Statistics.

If information and data were valuable in the 19th century, it is nothing compared to their importance in the modern world. From finance to insurance to your washing machine, data gives systems the ability to make better informed decisions (including the optimal composition of a portfolio or your preferred tumble dry setting.) The ability to process information has grown hand in hand with increases in computing power. No longer do statisticians have to rely on painstakingly hand-collected data. Modern technology gives us a repository of information larger than we even know what to do with. It is partially for this that machine learning has gone from a relatively obscure branch of statistics and computer science to the hottest academic topic taught in schools today.

While machine learning has made great strides, several issues remain. Modern machine learning is highly domain specific. For example, algorithms that work quite well with data collected from one distribution might not work well for data collected from another. Algorithms that are quite successful at using sparse medical data might fail miserably when used to analyze high dimensional financial data. Different application domains require different tasks. Google uses website data to solve an eigenvalue problem to produce search results. Such an approach would not work if one wanted to predict the price of Google's stock in a month's time. In this way, machine learning consists of a menagerie of different algorithms cobbled together, most being specialized to particular problems in particular settings. Many of these algorithms don't even have provable performance guarantees. Clearly these methods are highly dependent data structure and on the required task. Hence, the value of data to one algorithm that can use it, might be very different to one that cannot.

Furthermore, sources of information are becoming more aware and more reticent to release data. Consider the recent hacks of Equifax which resulted in a leak of sensitive private information on more than 100 million Americans. This data was originally collected by Equifax without the knowledge or consent of these individuals, highlighting the importance of modern privacy. It remains to be seen whether future legislation will address these issues, however, the ability to control where your data goes is something more and more people are becoming concerned about. As such, some data is destined to become a scarce resource in the future as individuals recognize and correct their breaches of privacy. Thus, machine learning algorithms should be designed to acquire useful data while at the same time mitigating privacy concerns.

One solution to these problems would be to monetize data transmission. If google would like to use my search history to give me personalized advertisements, they must pay me something. While we understand much more about data collection and manipulation than we used to, the economics of information transfer is practically unstudied. There is currently no systematic way to assign value to data. Despite this, individuals are beginning to realize the value of their own data. It has become common practice, for example, to hide one's location and browsing online when shopping for airline tickets as different online retailers will increase their prices if they see you've previously shopped for particular tickets. The value of data and the value of privacy are two sides of the same coin, each resulting from a cycle of learning and resulting outcomes.

This thesis seeks to find a solution to the problem of valuing information. We develop a structure through which data can be bought and sold fairly and describe under what conditions optimal learning or profit can occur. We show how the economics of data and information differ from other types of assets and develop specialized analysis techniques to both value information and model market dynamics. Before this however, it would be nice to develop some intuition about this problem. Consider the following examples.

1.1 Examples

Example 1

During the financial crisis in 2008, many factors contributed to a sudden systemic economic failure. One component was uncertainty surrounding certain types of complex financial instruments. Assets like securitized mortgages played a part in stoking and maintaining the panic that led to the downturn. Let's place ourselves in the shoes of a financial firm in 2006, before the crisis hit, who is considering either buying or selling such securitized mortgages. In order to make a decision on whether or not to buy or sell such assets, we as the firm must make a decision about what these assets are worth. To do so requires a minor digression on the structure of these investments.

A mortgage-backed security, made infamous in the aftermath of the great recession, is simply a mixture of different pieces of debt. In general, securitization is a technique to transform illiquid asset which generate receivables into securities that may be bought and sold in the open market. The process of securitization requires that first the originator of the assets create a portfolio of unwanted assets. Next, this pool of assets is sold to an intermediate financial firm who finances this acquisition by issuing tradable, interest-bearing securities. Investors who buy such securities receive interest from the returns on the asset. In the specific case of mortgage backed securities, banks package and sell unwanted mortgage debt to financial firms who package this debt in the form of securities to sell to investors. Banks still collect on the loans and pass on profits, minus a small fee, to the financial firm who in turn yields most of that profit to the investors. The way that firms package this debt however is to split up individual pieces of debt into tiny pieces, and then recollect these pieces based upon their riskiness. For example, if you were to go out and buy one of these securities, it would contain small pieces of hundreds if not thousands of mortgages, This means that when your neighbors, or equally a household across the country, pays their mortgage this month and your security stipulates that you own 1% of their debt, you will receive 1% of their interest (minus some processing fee.) Thus, securitization is a means of diversifying the risk of an asset.

Now let's get back to the firm trying to value such securities. It is clear that their value is directly associated with the probability that each of the mortgages get paid each month. However, since securitization may pack a huge number of assets into one security, one would need to know the default rate on the same large number of mortgages spread across the US (or even the world.) Hence, data on mortgage payments would be central in determining its value. The banks issuing individual loans still collect on them after securitization and would have access to data such as payment history and other personal information. They could sell this information to financial firms who would in turn be able to form an accurate assessment of the value of these securities, allowing them to make optimal decisions in the future.

Mathematically speaking, let us consider a security to be a vector $s \in \mathbb{R}^n$ such that s_i represents the percentage of each piece of n debts. Then, to simplify matters somewhat, say that each of the debts d_i are i.i.d. Bernoulli random variables paying 1 with probability p_i and 0 with probability $1 - p_i$. Hence, given some sample of the debts, the profit of the security is defined as $\sum\limits_i s_i d_i$. For this reason, it would be helpful to know the values p_i in order to calculate the expected value of the investment. Thus, our goal as the financial firm is to obtain data to learn the expected value of the security by learning the p_i's. In order to do this we have access to several pieces of information:

1. We know that each piece of debt d_i we hold in the security has some associated features (e.g. type of employment, location etc.) represented by the vector v_i
2. For each piece of debt d_i we hold a prior belief on the possible distribution of p_i given by $\mathcal{P}_i(p_i)$.
3. Banks who lend to households have access to v_i and the payment history $h_i \in \mathbb{R}^T$ which is simply a time series with entries $\{0, 1\}$.
4. Each bank can sell the pair (h_i, v_i).

With these conditions established, it becomes clear that our goal is to design a function $f : \mathbb{R}^n \to \mathbb{R}$ with the property that $f(v_i) \approx p_i$. With this, we can accurately calculate both the distribution and the expectation of the returns of the security s. Then we would be able to determine if the security is under or overpriced, and what its risk is to then behave accordingly. All we need in order to learn the function f is access to data (h_i, v_i) which we may purchase from banks.

The question then becomes, how much should we purchase this information for. To answer this question, first we need to define our utility as a financial firm. In other words, we need a quantitative description of how much we value accuracy in predicting the values of the p_i. Suppose this function takes the form $U_i(f, s) = \sum\limits_i s_i g_i(f(v_i), p_i)$ where g_i is some monotone decreasing function in $|f(v_i) - p_i|$ with maximum where $f(v_i) = p_i$. Then, suppose we have an algorithm $\mathcal{A}$ that takes as in the prior distribution $\mathcal{P}_i$ and the data (h_i, v_i) and returns a new distribution $\hat{\mathcal{P}}_i$ such that as $T \to \infty$, $\mathcal{A}(\mathcal{P}_i, (h_i, v_i))$ approaches the dirac delta function $\hat{\mathcal{P}}_i(p) \to \delta(p_i - p)$. In other words, with infinite information, the algorithm allows

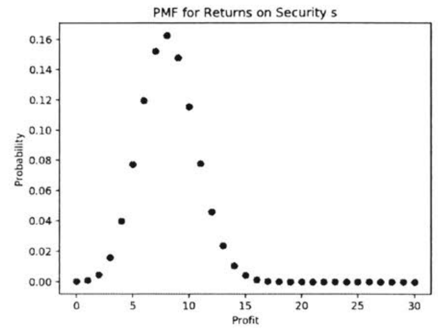


Figure 1-2: The goal of the firm is to estimate the PMF of the security s, seen above.

us to exactly learn the value of p_i. Then one may suppose that the value of the information, and hence the amount we pay for (v_i, h_i) to us is exactly the marginal change in U_i given this data is processed by $\mathcal{A}$. Sadly, this doesn't work, since in terms of our utility we are equally well off simply not buying any data and there is no incentive to learn.

One further complication is the addition of other financial firms competing with us. It may be the case that the more other firms learn there we lose utility commensurately. We want to find a fair, systematic way to obtain information from banks, or vendors of data, who have the ability to sell to multiple firms. Such a system does not exist to date. One of the goals of this thesis is to find a mechanism that, given some sources of data and firms who want to use that data, assigns an allocation of data in a fair way.

This example highlights the multifaceted nature of this problem. In order to assign value to data, one must consider the preferences of those selling and buying it, the preferences and information held by other sources and buyers of information, the combinatorial value of subsets of data and the method with which data is learned. Such a problem might seem intractable, however, it consists of only three major pieces. First, we need to understand the economics designing a market to share data. Second a thorough knowledge of the domain from which the information originated is integral to data valuation. Finally, in order to calculate the value of data, an algorithmic mechanism must factor in domain specific knowledge and economic preferences to produce an allocation.

Example 2

Rare diseases like ALS are particularly difficult to research for several reasons. First of all, there is minimal data. If a drug company wanted to develop a new therapy to manage or cure such a disease, the sample size they could work with is incredibly small. It is also difficult to justify the large cost of this development in order to sell a drug to such a small population, despite the possible improvement in many people's lives. The act of sharing data is risky to patients, whose information could end up in the hands of insurance companies that might decide to charge more to these individuals. Furthermore, in the realm of medicine, sharing data may be as simple as obtaining genetic information to something as complicated and risky as participating in a drug trial. These methods of data collection could cost people their lives, something not necessarily taken into account by statisticians and scientists collecting information.

Imagine several drug companies are competing to develop a drug to treat a rare disease. They each have different goals they would like to achieve with their treatment and can evaluate and tune their methodology through data collection from patients. These patients in turn could be incentivized to share their information in some way, but must be compensated for possible risks to their well being. In this case, we may design a system similar to the one discussed above in which the marketplace provides a matching between patients and drug manufacturers upon the stipulation that expected harm patients is commensurately compensated. This highlights the importance of the feedback between sharing data and future costs, and how such feedback must be accounted for in valuing data. The value of data and information is not limited to training algorithms. In fact, data gives agents the ability to assess the current state a game in order to choose their action.

Example 3

Consider the following game. There exists firms $f_1, ..., f_n$, each with functions $\mathcal{F}_i$ that map data to utility. Additionally, there exist m sources of data, $S_1, ..., S_m$, such that each S_i is a collection of datasets $\cup_j d_{i,j}$. Firms f_i may choose to buy data from each source meaning they sample some dataset from these collections. Suppose at each step they can sample or not sample from each source. Multiple sampling from the same source is not allowed. Each firm has past information about the sources of information encoded in the function $\mathcal{G}_i(\kappa)$ that gives the expected value, in terms of the utility given by $\mathcal{F}_i$, of buying data from a subset of sources $\kappa \subseteq \{S_1, ..., S_m\}$. Sampling from S_i comes at a price p_i, where $p = \{p_i\}$ is the vector of all such prices. Given p and $\mathcal{G}_i$, each firm calculates which sources to sample from by optimizing: $\kappa_i(p, \mathcal{G}_i) = \underset{\kappa \subseteq \{S_1, ..., S_m\}}{\text{argmax}} \; \mathcal{G}_i(\kappa) - \sum_{j \in \kappa} p_j$. Now, the if we imagine the subsets themselves operating as autonomous agents attempting to collect as much profit as possible, each source of data S_j will perform the optimization (where c_j is the cost of producing the data set.) $\max_{p_j \in \mathbb{R}, p_j \geq c_j} \sum_{i | j \in \kappa_j(p_j, p_{-j}, \mathcal{G}_i)} p_j$

A Nash equilibrium in this game would consist of a set of prices p such that no one source S_1 has an incentive to change their price. It is unclear, without knowing the structure of the functions $\mathcal{G}_i$ and by extension $\kappa_i(p, \mathcal{G}_i)$, whether or not such equilibria exist, whether or not they are unique and finally whether they can be calculated. It stands to reason however, if such an equilibrium p can be found, that this represents one interpretation of the value of data. Note that this could be generalized such that the function $\mathcal{F}_i$ maps data to an estimate of some state θ_i, which in turn dictates the action a_i and subsequent utility u_i of firm f_i. In this case, all of this information would give f_i the ability to update $\mathcal{G}_i$ over time as data is collected.

1.2 Previous Work

Scoring Rules

Buying, selling and valuing information and data has been studied in several contexts in the past. In the 1950's, weather stations discovered that they needed a way to incentivize forecasters to provide accurate predictions [1]. This problem is not limited meteorology [2]. In fact, a large number of problems in statistics attempt to estimate the distribution of a given random variable. Probabilistic forecasting is central in both finance and economics. Development of Markov chain Monte Carlo methods [3] dramatically increased the value of such methods. In order to estimate such distributions however, one must first collect and aggregate information from sources of information and belief. For example, a weather station might collect information from a variety of different forecasters, aggregate it, and publish a forecast based upon the result of this procedure. The question then becomes how to get good information. In mathematical terms, suppose there is a risk-neutral agent who holds a belief, in this case represented by a probability distribution $\mathcal{P} = \{p_i\}$ over a set of r states, such that $\sum_{i=1}^{r} p_i = 1$. If one were attempting to use this agent's information along with others to assess the true distribution of a variable, one could offer a reward to this agent for reporting a distribution $\mathcal{R}$ which we would hope equals $\mathcal{P}$. To encourage both truthful reporting and incentivize good forecasting, the reward c for the agent's reported distribution $\{r_i\} = \mathcal{R}$, would have the form $c = \sum_i p_i s_i(\mathcal{P}) \geq 0$ where the function s_i, called a scoring rule, is constructed such that c is maximized when $\mathcal{R} = \mathcal{P}$ and $\mathcal{P} = \hat{\mathcal{P}}$ where $\hat{\mathcal{P}}$ is the true distribution of the variable in question. Hence, this value c attempts to ascribe a value of the information or belief provided, depending on both its truthfulness and accuracy. Brier [1] introduced the first such scoring rule as $s_i = a_i + b\left(2r_i - \sum_k r_k^2\right)$ which was later refined by Good [4] to the logarithmic rule $s_i = a_i + b\log(r_i)$. While these scoring rules give a good way to describe the value of certain types of data, they run into the problem that, when aggregating data in most cases the output distribution is equal to one of the inputs [5], called the dictator, meaning that the information provided by other agents was not a posteriori useful. Modern results in scoring rules [6], [7], [8] demonstrate how the convexity of such rules can guarantee truthfulness in a wide variety of situations and additionally that different estimable values can be properly predicted using the theory of elicitible functionals. While interesting and useful from the perspective of designing a fair market, these results don't consider the economic aspects of data.

Active Learning

The problem of sampling informative, and possibly costly, data is the guiding principle behind active learning. Problems here can be categorized as either stream based or pool-based depending on the manner in which new data can be obtained. In the stream based case, the learner is presented with one piece of data to label or not to label at each time in the process. The active learning algorithm then provides a suggestion, based upon observed properties of the data, whether or not it is worth labeling. The algorithm employed must hence take in the observed properties of data and output an estimate of how valuable it will be to the learning task. These problem and algorithms have good performance both empirically and theoretically [9], [10], [11]. Pool-based learning [12] takes the approach that most instances of unlabeled data are known to the learner at each time who then must pick the best data to label given past information. Specifically, in the pool-based case, the learner is presented with a set of labeled data, here denoted as $D_L = \{(x_1, y_1, ..., (x_n, y_n)\}$ and unlabeled data $D_u = \{x_{1u}, ..., x_{mu}\}$, where y_i is the associated label to the point x_i. We assume the total set $D = D_L \cup D_u$ is sampled independently and identically from the same distribution. The learner then uses the labeled points to train an algorithm f and faces the objective of learning points in D_u in order to improve the algorithm's performance. These problems generally feature a small labeled set and a much larger unlabeled set from which to choose points to label. With a small budget to query the unlabeled points, the challenge is to extract as much information from the unknown set as possible while staying within budget constraints.

Results in pool-based active learning focus mainly on measures that indicate which unknown point to learn. Such measures can be considered to be functions that estimate the value of previously unlearned data. Lewis and Gale [13] developed uncertainty sampling, labeling the data point that maximizes uncertainty while Tong and Koller [14] applied this notion to SVM's by sampling points based upon their distance to the boundary of the classifier. Uncertainty sampling is in fact a broad term for a variety of different metrics used to guide active learning.

Another direction of research in this area is representative sampling. Here researchers attempt to find a measure of how representative a certain unknown data type is of the distribution in question thereby avoiding outliers. In the case of a linear classifier, there may be a point that is far away from the bulk of the rest of the data but happens to be close to the decision boundary, in which case, learning the type of this point is not very useful to classify the rest of the data accurately. Nguyen and Smeulders [15] and Donmez, Carbonell, and Bennett [16] show how points near the boundary of this linear classification problem are more representative if they are more tightly clustered. Clustering among data points is another method often used in determining the representativeness of a point. Specifically, if data is unlabeled and belongs to a tight cluster, it generally serves as a good representative of said cluster and labeling such a point will have broader accuracy improvements. Several papers recently [17], and [18] show that different approaches to clustering can identify the best points to learn. In essence, active learning depends on estimating the value of incorporating different pieces of data into a learning task.

Algorithmic Game Theory

This problem relies heavily on algorithmic game theory to transform information into updated beliefs and those beliefs into actions. These in turn result in economic outcomes. There is a large body of literature on such topics (see [19], [20], [21]) and our focus will be on different aspects of games studied in papers such as [22]. We will utilize some of these models and results to evaluate how rational agents behave when buying and selling information, and, additionally will after determining players' preferences, attempt to design information sharing mechanisms that are optimal for one or both parties. Such algorithmic mechanism design has been explored in [23]. In particular, one active area of research we will draw upon is the field of online mechanism design, in which dynamic outcomes from a game are used to design an economic mechanism in which some global behavioral objective is optimized. Algorithms and results in this field are succinctly described in [23] in which the authors describe a theoretical framework in which an online mechanism can be achieved in polynomial time given some reasonable assumptions on the structure of a game. Since we are trying to optimize feedback control, it is additionally useful to consider plant dynamics similar to those in [24] and [25]. Here the author provides an optimal bidding mechanism for advertisements that relies on a randomized online algorithm to update its actions.

In the simplest case, information trading between two parties can be described as a multi armed bandit problem with costly observations. The multi armed bandit problem, in which a player has the choice of some number of of actions, each with observable rewards given according to some unknown distribution, has been thoroughly studied in the statistical learning theory literature [26], [27], [28]. This model describes optimal decision making in the presence of uncertainty and has applications in a variety of different domains [29] [30] [31]. It is a prototypical example of what machine learning experts term the exploration versus exploitation trade off. Initially, when not much is known about the reward for each action, there is an incentive to experiment with different actions to learn their underlying return structure. At a certain point, however, one would want to use such knowledge to pick the best action. In our case however, we consider the case in which observations, i.e. data collection, becomes costly. Several papers recently [32], [33] and [34] have examined this situation given some constraints on the exploration phase of the algorithm. [35] and [36] study the case when observations have fixed costs and costs drawn from some underlying distribution respectively. In the latter case, learning not only has to be done on the reward distribution of each action but also on the cost distribution of observing each action's result. In our case, instead of considering the cost of observations drawn from some distribution, we allow this quantity to be determined as the result of game theoretic dynamics between those sharing and those utilizing information. The question then becomes whether or not the same techniques and algorithms can be applied in this case to efficiently solve this variation on the multi armed bandit problem. Much of the work above describes polynomial time algorithms that converge to optimal solutions in the case where there is only one strategic decision maker. In our case however, the dynamics of two players trading information becomes far more intricate. There must necessarily be considerations on learning budget, stability and truthfulness. In [23] the authors show how truthful mechanisms can be created in an auction type game, results we hope to apply to our information sharing mechanism.

1.3 Contributions of This Thesis

In almost every major industry, due to digitization and cheap sensors if a quantity of interest can potentially be measured and predicted, it is. Modern computational resources and the increasing sophistication of statistical inference algorithms has led to firms getting more value from the data they collect. The prevalence of data sources and the increasing value firms find in it, has led to data increasingly being viewed as an asset.

Currently, there are gross inefficiencies in the way data is traded - both from the perspective of firms buying data, and vendors collecting and selling it. Firms buy data based on intuitions on whether it might be useful in predicting trends they are interested in modeling. For example, within financial firms there are large teams whose sole purpose is to obtain datasets from a variety of sources to value financial instruments. Often they buy data without ever having assessed its predictive quality, out of fear that all other firms are buying it too. Even if it turns out that the data a firm buys is predictive, there is no principled method to put a dollar value on the increase in predictive quality. Vendors (e.g. Reuters, Forrester, Gartner), on the other hand, have no method to value the new, unique data sources increasingly becoming available. Furthermore, vendors have no way of knowing which firms in particular will find these datasets useful. Hence, they keep selling the same type of datasets they have been selling for years to same set of companies leading to is firms and vendors locked in long-term contracts for the same dataset. Given the nature of data-driven decision making, if vendors cannot make informed decisions of what data to collect and who to sell it to, it will lead to less predictive models. Our goal is to create a valuation system, or data marketplace, in which data can be bought and sold.

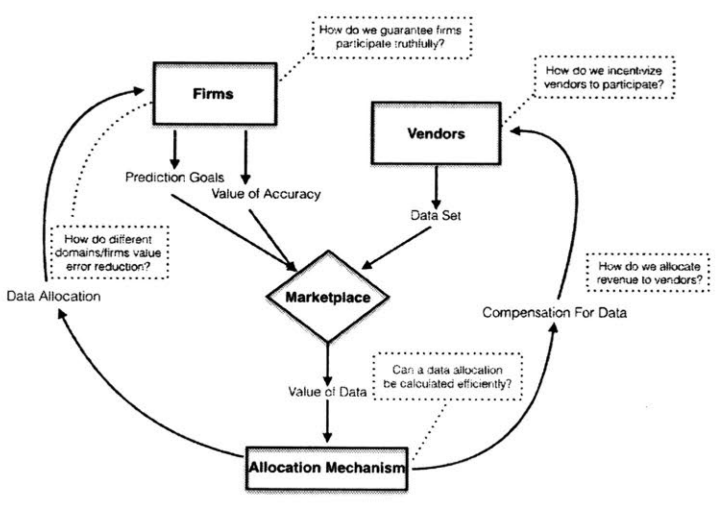

How do we guarantee firms participate truthfully?
Firms
How do we incentivize vendors to participate?
Vendors
Prediction Goals
Value of Accuracy
Data Set
How do different domains/firms value error reduction?
Data Allocation
Marketplace
How do we allocate revenue to vendors?
Compensation For Data
Can a data allocation be calculated efficiently?
Value of Data
Allocation Mechanism

Figure 1-3: A Data Marketplace.

The increasing use of data in modern society is an inevitable product of recent technological advances. From the perspective of those using the data, new algorithms and computers have been huge in advancing their goals, but the current process of learning is becoming hopelessly inefficient. One of the reasons big data is catching on is because it is designed to deal with huge quantities of data through techniques like dimension reduction. However, as powerful as dimension reduction is, at a certain point these firms must face the fact that they will no longer be able to collect an huge amount of complex data and conclude something useful, rather they will need to shift to collecting useful data in a more targeted way. Our system is designed to do just that. Firms who participate will be guaranteed to only receive data that is most useful to their objective, disregarding other information that could be misleading or repetitive. Additionally, while they will pay some fee to collect this information, it will be small compared to the effort required to sift through all the data available to find data they need. Firms also sometimes require information that is difficult to collect either due to privacy concerns, lack of sources or some other complication. In this case, offering a financial incentive to share information in a fair market will make sources of rare data less reticent in releasing their information, increasing the possibilities for firms.

From the individual's perspective, the current system of data acquisition completely disregards our right to privacy. If Google or Equifax or the government can collect information on anyone's preferences and beliefs at their leisure, the social ramifications could be catastrophic and we are already seeing a push towards regulations on privacy. However, a push towards what? It is not as if the economy, which subsists on data, will stop collecting our information. What is needed is a systematic way that data and information can be shared such that all parties benefit. The results proposed in this paper would be such a system.

Upon adoption of such a system, we foresee the following societal benefits:

- Useful data would be more available to those who need it, streamlining learning.
- Individual privacy would be protected.
- There would exist a systematic, unified method to value information.

Apart from the considerable social impact solving this problem will have, there are some fascinating and fundamental technical challenges that need to be tackled along the way, that make this endeavor especially exciting. First, how many times should a vendor replicate a dataset to maximize revenue? - Data shares similarities with digital goods (e.g. songs, mobile apps) where the marginal cost of production is zero. However unlike other digital goods, it is realistic for a firm to derive most of its value from a dataset due to exclusive access, and so replicating data and making it less scarce can severely affect its market value; Second, unlike other economic assets, the value of data to a firm cannot be assessed a priori; the value of data is derived solely from its ability to predict a quantity of interest. In addition, valuations for datasets are inherently combinatorial in nature and given the potential scale of this problem, it is infeasible for any mechanism to go through every possible combination of datasets when matching firms and vendors. Hence any mechanism not only has to match firms and vendors, it must efficiently and accurately assess the predictive quality of various combinations of dataset with respect to the quantities firms are interested in. No mechanism currently used has this capability. Solving either problem above would be a significant technical milestone in of itself, and both are vital in achieving the vision of a data marketplace.

This thesis seeks to make the most general system possible, however, as previously mentioned, it is not possible to quantify the value of data without considering the application domain. For example, information on financial assets might be both structured and valued in a vastly different ways from data on medical procedures. With that said, we think that there are huge possibilities for this technology in the domains of medicine, finance and insurance.

Data is integral to every decision a firm takes. Thus, firms and vendors must value data very carefully. Understanding the fundamental economic forces governing data exchange is an important problem. We hope change how data is traded, away from long-term contracts between the same set of firms and vendors to a more robust data allocation mechanism. This will ensure that firms that derive the most value from a dataset are the ones that get it, and vendors are incentivized to collect unique, predictive datasets. In addition, given how sensitive of a topic privacy has become, only by accurately assessing how valuable the data of an individual is, can we have a rigorous discussion on what data is worth collecting in addition to ensuring that individuals that choose to provide their data are fairly paid for it. To date, there has been no attempt to create a system to fairly value and trade data. Such a system could be applied in a variety of fields from finance and e-commerce to medicine and advertising.

1.3.1 Outline

The structure of the thesis is as follows: in Chapter 2, we introduce the mathematical preliminaries necessary for the analysis and design of data marketplaces. We also introduce several different methods of both valuing and trading data, and discuss their relation to previous work done in combinatorial auctions and algorithmic game theory. In Chapter 3, we develop a method with which to price data to optimize total learning under certain restrictions on learning. Chapter 4 describes an integer programming formulation of our problem and novel relaxations and approximations we may use to solve it. Chapter 5 gives a framework for a profit maximizing data allocation mechanism. We discuss several directions for future work and conclude in Chapter 6.

Chapter 2

Preliminaries

In this chapter we provide the mathematical background that will allow us to value data. It is divided into three main sections: the first section is devoted to machine learning, especially to highlight several algorithms and results in supervised learning. In particular, metrics for measuring error in machine learning will be important in quantifying the value of data, and hence will be covered; the second section covers several major results in algorithmic game theory, focusing in general on mechanism design, combinatorial auctions and associated algorithms; finally, the third section gives a model for the data valuation problem along with preliminary results.

2.1 Machine Learning

In "Computing Machinery and Intelligence" [37], Alan Turing posed the question "Can machines think?" or rather "Can machines do what we can do." This idea was formalized by Tom Mitchel who said "A computer program is said to *learn* from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." This is precisely the goal of machine learning: to provide a computer some experience and use that to learn to do some assigned job better. Machine learning makes data driven predictions or decisions by building a model from input. Applications of this technology are pervasive in the modern world. We find machine learning software reading handwriting [38], identifying spam [39], characterizing social networks, recognizing faces among many other applications. There is currently great excitement in academia and society regarding the possibilities of such technology. However, there are reasons to temper our expectations. Pattern recognition, which lies at the heart of all machine learning problems, is highly nontrivial. It is easy for example, to come to the wrong conclusions given limited or faulty data, as Microsoft learned in 2016 with their chatbot fiasco [40]. Furthermore, current research on the brain suggests that the ways we process data and the way a computer functions are vastly different, so trying to obtain human-type cognition from a machine is probably unreasonable.

With those caveats in place, machine learning has also made great strides in recent years. From Google's reinforcement learning algorithm mastering the notoriously difficult game 'Go', to the emergence of self driving cars, machine learning and artificial intelligence are becoming commonplace in modern life. Machine learning can be separated into several different types of tasks. Supervised learning, unsupervised learning and reinforcement learning. For the most part, this thesis will focus on supervised learning tasks, however, it is useful to understand the flavor of the other two topics as well.

2.1.1 Supervised Learning

Example 1. *Let's consider a fictional email service, "Fahoo," that must design a method for identifying spam emails. Mathematically this means constructing a function $f : \mathbb{R}^n \to \{S, N\}$ that maps the vectorized description of some email to "spam" (S) or "not spam" (N). To do this, we have access to a set of pre-classified emails $\{(e_1, c_1), (e_2, c_2), ..., (e_k, c_k)\}$ with which we can test and tune our function f. For this example we will gloss over exactly how emails are turned into vectors e_1 except to say that these vectors are called "features" and the associated $c_i \in \{S, N\}$ are called their "labels." From this information Fahoo must decide, given some new email $e \in \mathbb{R}^n$, whether e is spam or not.*

The above example is a classic problem in supervised learning. An algorithm is presented with a set of inputs with which it can learn a function f in order to accomplish a task (in this case identifying spam). Supervised learning tasks fall into two main categories: classification and regression. A classification task asks the machine categorize data into a finite number of categories. This could be binary classification such as the spam example above, or it could be something more complicated like identifying a handwritten letter which would belong to a set of size 26. In regression, the prediction can range over a continuous set of values. Examples of a problem in regression might be predicting the rainfall for the next day given current meteorological measurements. In either case, the essence of the problem is to take input data $\{(e_1, c_1), (e_2, c_2), ..., (e_k, c_k)\}$, process this data using an algorithm, and return a function f which makes predictions $f(e) = c$.

At its heart, supervised learning is an optimization problem. In the spam example above we would like to design a function f that mistakes as few inputs as possible. To do so, we might define an error function $\text{error}(f) = \sum_i \mathbb{1}_{f(e_i) \neq c_i}$ where $\mathbb{1}_{f(e_i) \neq c_i}$ is an indicator function.

Anywhere the function f makes a mistake is penalized by the error function. In general, f belongs to a class of functions parametrized by $\alpha \in \mathbb{R}^m$ such that the machine learning algorithm must now solve the optimization problem $\min_\alpha \text{error}(f)$. A classic example of this might be, if $e \in \mathbb{R}^2$ to take a linear function $f(x, \alpha) = \alpha_1 x_1 + \alpha_2 x_2$. This may seem like a restrictive class of functions to consider, however, it can give astonishingly accurate results, particularly when combined with the kernel methods. It is important to note that different algorithms can be trained using different error functions with different results. In general, error functions that are nice, e.g. linear or convex, functions of the parameters α provide nice algorithmic solutions, however, as we will see, this is not always possible.

One problem in supervised learning is that, once given a labeled data set $\{(e_1, c_1), (e_2, c_2), ..., (e_k, c_k)\}$, the algorithm should avoid overfitting. It could be the case that the function we design f works perfectly on the input data $\{(e_1, c_1), (e_2, c_2), ..., (e_k, c_k)\}$, but not well on unknown inputs. We need the function f to generalize well. There are many methods to design such functions, but, without new data, we must have a way to test how well our function f might behave on unknown data. Do do this we employ a common tool in machine learning and statistics known as cross validation. This technique, in essence, chooses a subset S of our training data $\{(e_{S_1}, c_{S_1}), ..., (e_{S_l}, c_{S_l})\}$, uses this subset as the input to our chosen algorithm to design f_S, and then tests the error of f_S on the remainder of the training data. If we do this procedure repeatedly we can measure how well our algorithm does creating functions that generalize well. We will use similar techniques later on to determine the value of data and information to different tasks.

Another difficulty faced by supervised learning is that there is no limit to how large or complex the data can be. Often times this makes it necessary for a preliminary algorithm to transform the data initially before it can be used to train the learning task. Such transformations could come in the form of dimension reduction, Fourier and Laplace transformations, or other methods, depending on the domain. Consider genetic data. The complexity of DNA and the human genome might mean that this information could belong to large dimensional Euclidean space and it is generally difficult to design efficient, meaningful algorithms that make predictions in $\mathbb{R}^n$.

Thankfully, as many modern results like the Johnson-Lindenstrauss theorem show, high dimensional data can be projected onto much smaller subspaces without losing too much information. In the following we give an example of how such dimension reduction techniques can be employed to preprocess data.

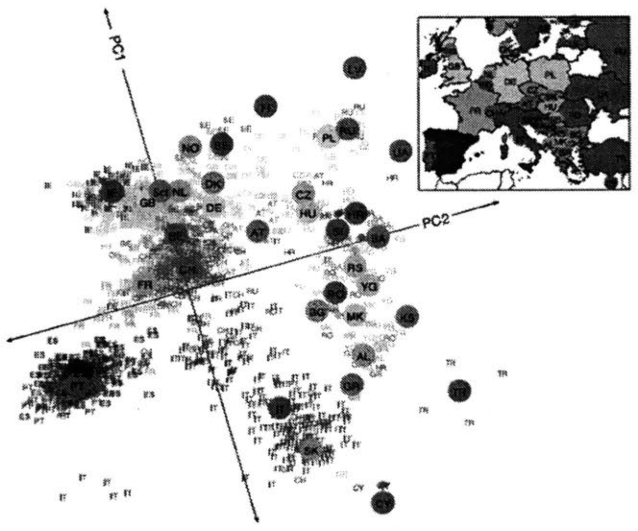


Figure 2-1: Genes Mirror Geography.

Example 2. *A study in 2008 by American and Swiss researchers led by John Novembre at UCLA [41] examined how human genetics relate geographically with one another. They collected blood from over 1,300 individuals from three dozen countries across the whole of Europe. Each of these samples was a 200,000 dimensional vector $v_i \in \mathbb{R}^{200000}$ representing single nucleotide polymorphisms, essentially places in human DNA that commonly differ between populations. Doing analysis in 200,000 dimensional space is difficult and the researchers wanted a way to somehow visualize* *their results. To do this they collected the column vectors $\{v_i\}$ into a matrix A, and then decomposed A into its singular value decomposition $A = UDV^T$. Then, defining the first two columns of U as u_1, u_2 they calculated U_2UDV^T where $U_2 = [u_1, u_2]^T$. Using the well known result from linear algebra that states that this projection optimally preserves variance of the columns of A, this process resulted in a 2 dimensional vector $\tilde{v}_i$ for each individual. That amounts to a dimension reduction from 200,000 dimensions to 2 dimensions. Incredibly, when the new values $\tilde{v}_i$ were plotted in $\mathbb{R}^2$, seen in Figure 2-1, they exactly mirrored the geographical locations of each sample. This is simply one example of the usefulness of pre-processing and the necessity of finding the critical components of data. Had the researchers really tried to do analysis with the original A, they would have been sunk.*

There are numerous different methods for supervised learning, all of which will be compatible with the data marketplace we design, however we will discuss two examples that highlight some important algorithmic problems and properties.

Support Vector Machines

Imagine we have a binary classification task such that we would like to have a function $f : \mathbb{R}^n \to \{-1, 1\}$ where $f(x)$ classifies the feature vector $x \in \mathbb{R}^n$ as belonging to one group or another. One method we might use to do this is called linear classification. Specifically, this method says to parametrize the class of functions $f(x)$ we want to pick from by a vector $\alpha \in \mathbb{R}^{n+1}$ such that $f(x) = \operatorname{sgn}\left(\alpha_{n+1} + \sum\limits_{i=1}^{n} \alpha_i x_i\right)$. The function $\alpha_{n+1} + \sum\limits_{i=1}^{n} \alpha_i$ simply describes a hyperplane in n dimensional space which classifies points based on which "side" they happen to fall on. Now suppose we have a set of labeled points $\{(x_1, y_1), ..., (x_k, y_k)\}$ such that $x_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$.

Given this, we would like to choose the optimal vector α such that f makes the fewest mistakes possible on the training set. To this end, we might penalize the function $f(x, \alpha)$ as we did before, by adding a penalty of one for every misclassified example, however, this approach has several problems. First of all, the loss function is discontinuous and hard to optimize, and second, there could be an infinite number of so-called separating hyperplanes that properly classify the data we are provided with. How should we choose one? For these reasons, we would rather consider the loss function $\frac{1}{k}\left(\sum\limits_{i=1}^{k}(\max\{0, 1 - y_i(\alpha_{n+1} + \sum\limits_{i=1}^{n} \alpha_i)\}\right) + \lambda\alpha^T\alpha$. This gives us the ability to both create a maximally separating hyperplane and to deal with training data that is not linearly separable. At this point, we solve for α using dual convex programming. The dual problem, which relies only

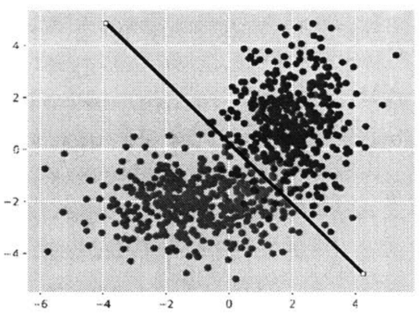


Figure 2-2: Linear Classifier.

on the inner product of the feature vectors, can be manipulated so that we may take non-linear mappings $\phi : \mathbb{R}^n \to \mathbb{R}^m$, apply these to our initial features, and solve the problem in the higher dimensional linear space. Such methods are known as kernel methods and are used extensively. SVMs are but one example of a method of training classifiers, however, in most cases these methods follow the same principles: There is some class of functions f parametrized by α. Given some labeled data $\{(x_1, y_1), ..., (x_k, y_k)\}$ and some loss function, we minimize the loss over the labeled data with respect to α to estimate the true classifier.

Neural Networks

Another interesting class of learning algorithms are known as neural networks. Pioneered in the 1950's and inspired by rudimentary models of the brain, they remained largely forgotten until modern computing power allowed it to reach its full potential.

Neural networks are simply functions $f : \mathbb{R}^n \to \mathbb{R}^m$. What makes them special is that, given any other function $g : \mathbb{R}^n \to \mathbb{R}^m$, we can design a neural network to give us f that approximates g arbitrarily well. This means that even two sets such as those seen in Figure 2-3 can be correctly classified by these methods.

A neural network is a directed graph with nodes and edges consisting of several different components. The first component is the input layer, a set of n nodes, the output layer, a set of m nodes, and some number of hidden layers each with an unspecified number of nodes. Each of these layers are connected to subsequent layers by a number of weighted edges w_{ij}, beginning with the input, then hidden, then output layers. Nodes in the network take in a sum, weighted by the strength w_{ij} of incoming edges, of outputs A_i in the previous layer. This sum is then fed into a function $\phi : \mathbb{R} \to \mathbb{R}$ such that the output of node j becomes $\phi(\theta_j + \sum_i w_{ij} A_i)$ where the weights w_{ij} and the offsets θ_j are all parameters the learning algorithm can adjust. Common functions used for ϕ are sigmoid functions, step functions, or any other function that approximates some sort of threshold. Now, given some set of labeled data $\{(x_1, y_1), ..., (x_k, y_k)\}$ where $x_i \in \mathbb{R}^n$ and $y_i \in \mathbb{R}^m$ we can use an algorithm known as backpropegation to adjust the weights and offsets so that f correctly classifies training data. The backpropegation algorithm is just clever differentiation in combination with gradient descent. Any neural network made up of more than one hidden layer is known as a deep neural network and if can be shown that any function can be arbitrarily well approximated by three hidden layers of arbitrary size.

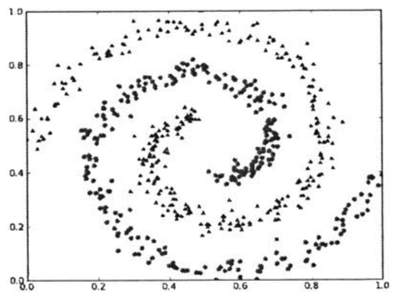


Figure 2-3: Difficult Classification

In each of the problems above, it was assumed that the algorithm was provided with labeled training data. This doesn't account for the problem that useful data is often scarce, noisy and costly to collect. Machine learning algorithms that factor this constraint into their methodologies are called active learning methods. They are particularly important to our work as they focus on only collecting data of the highest quality. This means that by defining a value for data will place this class of algorithms on solid footing, which until now operate mainly based on heuristic estimates of what is "good" data.

Active Learning

Any supervised learning algorithm, including the ones discussed above, rely on previously labeled data. If there is no data to tell a neural network what outputs should be associated with different inputs, you might as well just throw a dart at a board for all the predictive value the function will have. For this reason, data is incredibly important to all of these learning tasks. However, data and information can be both difficult and costly to obtain. Active learning is a subfield of supervised learning that attempts to develop algorithms which take this cost into account. Suppose that we are conducting a study on sexually transmitted diseases in a population of college students. Perhaps our goal is to predict the likelihood a student with particular features and behavior has of coming down with a given infection. To do this, we would need to collect some data on students that both have and don't have the ailment in question. However, due to the sensitivity of such information, subjects would both have a tendency to either lie or not report at all. To encourage them to report their feature $x_i \in \mathbb{R}^n$ and their status $y_i \in \{-1, 1\}$ (infected or not) we as researchers could offer a financial incentive for truthful reports. As researchers, we have a limited budget so we would like to build the most accurate classifier possible while minimizing the money spent on data. Classic approaches to this problem generally obtain data that is the most uncertain to the classifier. In the case of a support vector machine, this might mean labeling feature vectors that lie closest to the separating hyperplane previously given by our learning algorithm. Other variations on this idea include choosing to label data points that, in expectation, improve the performance our algorithm. Specifically, suppose we have some supervised learning algorithm $\mathcal{A}$ that takes in some training set $\{(x_1, y_1), ..., (x_k, y_k)\}$ and produces a function $f(x) \in [0,1]^2$ representing the probability of x being in one category or the other. Suppose further that there is an error function $E(f, \{(x_1, y_1), ..., (x_k, y_k)\})$ such that f satisfies $f = \operatorname*{argmin}_{g} E(g, \{(x_1, y_1), ..., (x_k, y_k)\})$. Now, given this, suppose there exists unlabeled data $\{w_1, ..., w_l\}$ such that, if the true label of w_i is u_i, the total error changes by $\delta(w_i, u_i) = E(f, \{(x_1, y_1), ..., (x_k, y_k)\}) - E(\tilde{f}, \{(x_1, y_1), ..., (x_k, y_k), (w_i, u_i)\})$. Then we can choose the data point w_i that, in expectation $f(w_i)_1\delta(w_i, 1) + f(w_i)_2\delta(w_i, -1)$ improves the performance of the algorithm the most.

Two other important branches of machine learning are unsupervised learning and reinforcement learning. Unsupervised learning takes unlabeled input $\{x_1, ..., x_n\}$ and tries to associate with those points some sort of structure. Clustering is a popular tool used for this purpose and a variety of different problems in network analysis and other fields require machines to "learn" about unlabeled data. Reinforcement learning tries to update some learning function f from data that is fed back into the system as a result of decisions made from previous functions f. A classic example of this is the lauded multi-armed bandit problem which, in its simplest form, states that a gambler has access to m arms $a_1, ..., a_{}$ each paying 1 with probability p_i and 0 with probability $1 - p_i$. The objective of the gambler is to, while playing, identify which arm is best. A classic solution to this problem dictates that one should choose an arm a_i at time t with probability $\frac{e^{w_i(t)}}{\sum_j e^{w_j(t)}}$ where $w_j(t)$ is the number of wins of arm a_j at time t. Over time this policy converges to the optimal policy with bounded regret.

2.2 Algorithmic Game Theory

In their pivotal paper [42], Nisan and Ronen pointed out a growing need to consider algorithmic problems in distributed settings with self interested agents. The most obvious example of such a setting is the internet, in which countless selfish agents interact at all times. Current research in this field focuses on the stability and computability of equilibria as well as the efficiency, in terms of the price of anarchy of such distributed systems. Several different economic applications to this work are gaining importance including complex auctions, targeted advertisement, and a variety of other multi-agent systems. We will draw mainly on the ideas of mechanism design and in particular auction theory as they relate to our learning problem.

2.2.1 Mechanism Design

A blend of economics, game theory and engineering, mechanism design analyzes strategic interaction by designing mechanisms to incentivize players to cooperate for a global objective. In mechanism design problems, games are played by agents that have particular types. Let $\theta_i \in \Theta_i$ denote the type of the ith agent from the total set of possibilities Θ_i. Then, given an outcome of the game $o \in \mathcal{O}$, the utility of each agent can be expressed in terms of a function parametrized by their type. In mathematical terms, if agent i has type $\theta_i \in \Theta_i$, and $o \in \mathcal{O}$ is the outcome of the game, the utility they receive is $u_i(o, \theta_i)$. Agents, or players, then have a set of strategies or actions $s_i(\theta_i) \in \Sigma_i$ they can take to effect the outcome of the game. In addition to playing a single one of these actions, we may consider mixed strategies, where agents play distributions over the set Σ_i representing the probability they play particular actions. The utility of each agent can be rewritten as a function $u_i(o(s_1, s_2, ..., s_n), \theta_i)$ depending on their type and the other players' actions, where $o(s_1, s_2, ..., s_n)$ is the outcome of the game depending on the strategy profile of all the players participating.

Definition 3. *A strategy profile $(s_1, s_2, ..., s_n)$ is called a Nash equilibrium if every agent maximizes their expected utility so that for each i we have $u_i(o(s_i, s_{-i}), \theta_i) \geq u_i(o(\hat{s}_i, s_{-i}), \theta_i), \quad \forall \hat{s}_i \neq s_i$*

Although the idea of the Nash equilibrium is central to game theory, it makes excessively strong assumptions about agents' information about other players. Additionally, there are sometimes multiple equilibria. Various different extensions of this idea have been proposed to solve these issues. Dominant strategy equilibria feature agents who have one strategy that outperforms all others regardless of the other players' actions. In addition there are mixed Nash equilibria and Bayesian Nash equilibria which both behave better than pure Nash equilibria.

Mechanism design is often concerned with dominant strategy equilibria. Consider auction design. The standard problem states that there is some object for sale, and agents $a_1, ..., a_n$ who would like to buy the object. Each agent has their own valuation $v_1, ..., v_n$ for the object and therefore would not like to spend any more than that to obtain it. The problem is to allocate the object to the person who values the object the most. One idea to consider would be to collect how much each agent says they value the object, $\tilde{v_1}, ..., \tilde{v_n}$ (these are known as the bids) and give the object to the agent with the greatest valuation $\tilde{v_i}$ for the price $\tilde{v_i}$. However, this doesn't work, since agents may lie about their valuations in order to increase their utility. Specifically, if the winning agent decreases their bid $\tilde{v_i}$ by ϵ, as long as the bids are reasonable space, he will increase his utility by ϵ, so there is an incentive to lie. Enter the Vickrey auction, the cornerstone of mechanism design and truthful mechanisms. The Vickrey auction, also known as a second price auction, determines that the agent with the highest bid $\tilde{v_i}$ is allocated the object for a price of $\tilde{v_j}$ where $\tilde{v_j}$ is the second highest bid. In this case, if the winning agent changes their bid up or down a small amount, they do not change their utility, unless they drop too far and lose the object entirely, going from positive utility to zero utility. Hence, the winning player has no incentive to lie. If the other players lie, they either continue not to get the object, or get it at too high a price. The structure is elegant in its simplicity and many results in mechanism design were either inspired by or simplify to this kind of mechanism.

Mechanism design, in its simplest form, consists of a social choice function $f : \Theta_1 \times ... \times \Theta_n \to \mathcal{O}$ that describes the best outcome for set a players with different types. A mechanism $\mathcal{M} = (\Sigma_1, ..., \Sigma_n, g(\bullet))$ defines the set of strategies available to each player and a rule $g(\bullet)$ that describes the outcome of each action profile. A mechanism defines the actions available and the outcomes based on those actions.

Definition 4. *We say that a mechanism $\mathcal{M}$ implements the social choice function $f(\theta)$ if, at equilibrium, the outcome predicted by the social choice function is also the outcome at equilibrium of the mechanism.*

Many mechanisms such as auctions rely on agents to report their types $\theta \in \Theta_i$ such that $f(\theta) = o \in \mathcal{O}$ is the optimal social outcome.

Definition 5. *Mechanisms that feature dominant strategies that are truthful are called "truthful".*

The Vickrey auction discussed above is the prototypical example of a truthful mechanism since a simple payoff rule incentivizes agents to report their true value for an object. In later sections, our goal will be to design rules for allocating goods and money such that all parties involved in a transaction are incentivized to reveal their true type.

2.2.2 Combinatorial Auctions

It is important to take a closer look at a particular brand of mechanism design known as combinatorial auctions before we delve more deeply into the problem of valuing data. The purpose of an auction like the second price auction discussed above, is to allocate a good in the most socially optimal way possible. It can be shown that the Vickrey mechanism does just this, by both incentivizing bidders to reveal their true valuations and by giving the object to that agent who values it most. While this is fine when considering selling a single good or service, more complicated auctions might not yield such an elegant result. In particular, auctions involving the sale of a variety of distinct assets pose several interesting challenges to the mechanism designer. First of all, because of complementaries and substitution effects between different goods being sold, bidders have valuations and preferences over subsets of goods. This means that economic efficiency is improved as long as bidders may place offers on subsets of goods. Auctions with this structure are known as combinatorial auctions and are becoming more and more important in the 21st century. Examples of combinatorial auctions include the FFC spectrum auction, auctions for airport time slots [43], and delivery routes [44].

These are auctions in which there are n agents as usual $a_1, ..., a_n$ however, instead of bidding on one object, there are multiple objects $o_1, ..., o_k$ available. Furthermore, the value of an allocation of objects $o_1, .., o_k$ to player a_i is not linear in the objects. In mathematical terms this means that $V_i(o_1, ..., o_k) \neq V_i(o_1, ..., o_{k-1}) + V_i(o_k)$, where $V_i(\bullet)$ gives the value of a subset of objects to agent a_i, and therefore objects may be worth more when paired with other objects. As an example of this one can think of the body of a vehicle and its engine as two objects. Separately they may have low value, but together they have value greater than the sum of their parts. For this reason agents release valuations or bids on subsets of objects and the auction designer must come up with a way to distribute money and objects in such a way that buyers and sellers alike are incentivized to participate fairly. Due to this, bidders must transmit some description of their bids for each subset of items being sold, a computationally complex task. Thus, the bidder must calculate its utility for each of $m!$ different subsets, and then transmit this as an exponentially long vector to the auctioneer. It is clear that this isn't efficient or even feasible as m grows. Many solutions to this problem including bidding languages [45], oracle submission [46], and other bidding restrictions [47] have been proposed. It is well known that this problem is NP-complete in the worst case, however, in recent years several different methods to approximate optimal solutions have been proposed. We can write the problem of multi unit combinatorial auctions as the following integer program.

$$\max \quad \sum_{i=1}^{n} \sum_{S \subseteq M} V_i(S) y(S,i) \tag{2.1}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{S | j \in S} \sum_{i=1}^{n} y(S,i) \leq k_j \quad \forall j \in \{1, ..., m\} \\ & \sum_{S \subseteq M} y(S,i) \leq 1 \quad \forall i \in \{1, ..., n\} \\ & y(S,i) \in \{0,1\} \end{aligned} \tag{2.2}$$

The constraint $y(S,i) \in \{0,1\}$ can be relaxed to $y(S,i) \in [0,1]$ in order to utilize tools from linear programming. Solutions that result from this relaxation may be useful to estimate optimal allocations of goods, and may additionally give information in terms of the dual of the problem. Finally, some work has been done analyzing incentive compatible or truthful mechanisms to elicit bids from participants in such an auction. The classic example of this is the Vickrey-Clark-Groves (VCG) mechanism which is truthful provided optimal solutions to the above integer program can be solved exactly. As this is often intractable, some researchers have proposed other forms of mechanisms.

2.3 Valuing Data

The purpose of this thesis is to define the value of a piece of data, and describe a method through which it can be sold. In fact these two goals are intrinsically intertwined in that the price of data is dependent on some kind of market which in turn is impacted by the definition of the value of data. Dynamics of buying and selling data can take many different forms and in order to consider them in the greatest generality possible, we consider for the remainder of this work the following situation. Suppose we have n firms $f_1, ..., f_n$ who would like to buy information and m vendors of information $v_1, ..., v_m$ each selling data set $d_1, ..., d_m$. Each of the firms would like to estimate the state of a variable $\theta_i \in \Theta_i$, where $\Theta_i \in \mathbb{R}^k$ is convex and there is a function $g_i(\theta)$ for each firm f_i giving them some utility for their estimate of the type θ_i. These functions $g_i(\theta)$ are maximized when θ is the true state θ_i. Each firm has previous information $\mathcal{I}_i(t)$ at time t and an algorithm $\mathcal{A}_i : \mathcal{I} \times d :\rightarrow \hat{\theta}_i$ that uses previous information and some subset $d \in \{d_1, ..., d_m\}$ to update the firm's estimate of θ to $\hat{\theta}_i$. On the other side of the transaction, the vendors v_j gain utility by monetary payment from firms for releasing data, in addition to some feedback effects $b_j(f_1, ..., f_k)$ based upon who gets dataset d_j. Note that vendors may replicate their data to sell it to multiple firms. Finally, depending on the level of information of different firms, individual firms may be penalized by the function $c_i(\hat{\theta}_1, ..., \hat{\theta}_n)$. It is our task to come up with a good method to allocate data to firms and choose how firms pay vendors.

Definition 6. *A* **data valuation** *game consists of the following:*

1. *n firms $\{f_1, ..., f_n\}$ each having information $\{\mathcal{I}_1, ..., \mathcal{I}_n\}$ who wish to estimate the state $\theta_i \in \Theta_i$, where Θ_i is convex, of some variable. Utility is gained through the function $g_i(\hat{\theta})$, where $\hat{\theta}$ is the estimate of the state, such that $g_i(\hat{\theta})$ is globally maximized when the estimate $\hat{\theta}$ is equal to the true state θ_i. Finally, each firm has an algorithm $\mathcal{A}_i$ that takes in $\mathcal{I}_i$ and some datasets $\{d_{i_1}, ..., d_{i_k}\}$ that produces a new estimate of the state $\hat{\theta}'$. Finally, there is a function for each firm representing competition between firms $c_i(\hat{\theta}_1, ..., \hat{\theta}_n)$ so that the total utility of the firm f_i is equal to $g_i(\hat{\theta}) + c_i(\hat{\theta}_1, ..., \hat{\theta}_n)$ minus what they pay for the data they buy.*

2. *The information parametrizing the firms gives them utility for each set of data $S = \{d_{i_1}, ..., d_{i_k}\}$ denoted $V_i(S)$.*

3. *There are m data vendors $\{v_1, ..., v_m\}$ selling data sets $\{d_1, ..., d_m\}$ such that the utility from a transaction is decomposed into the monetary payment, p_j, and the feedback effect $b_j(f_{j_1}, ..., f_{j_l})$ that occurs as the result of selling information d_j to the firms $f_{j_1}, ..., f_{j_l}$.*

To sort through all of the notation introduced above, it is useful to consider an example:

Example 7. *Consider the problem of a ride sharing service like Lyft or Uber trying to estimate the future demand for service at a specific time and a specific place. To make this estimate they will need to collect historical data on users' past demand. Suppose, for this example, there are two firms f_1, f_2 representing Lyft and Uber, who would like to estimate the number of people θ demanding their service on a Saturday night so that they may adjust their prices accordingly. Each firm gains $g_1(\theta), g_2(\theta)$ respectively from these estimates, the more accurate the better, and hence would like to collect as much information as possible. They must weigh this gain against the potential cost of collecting this data. Finally, they may be penalized by a competition function $c_i(\hat{\theta}_1, \hat{\theta}_2)$ dependent on both players estimates. On the other hand, there are m users of these services who may sell their data $\{d_1, ..., d_m\}$ to either (or both) firms at some set prices. Their utility is then this price plus the possible feedback effects of sharing their data. For example, if an individual's data suggests that their demand on Saturday will be extremely high, it is in the best interest of a ride sharing service to increase prices at that time, since this will result in more revenue. In this case, the information this individual shared had a direct effect on their prosperity in the future and hence, this must be factored into their utility. In such a game, the players may decide to buy or not buy at a particular price (in the firms case) or sell or not sell at a particular price (in the case of the vendors.) However, without knowing how the data d_j improves their estimate, how is a firm supposed to decide what price a piece of data is worth? Conversely, without knowing the effects of their data on the estimate $\hat{\theta}$ and the resulting feedback, how do vendors decide at which price to sell? The inherent informational asymmetries of this game make these tasks difficult.*

This example, and preceding definition, suggest that a third party might be necessary in determining the optimal price to buy and sell data, since, the only way the firm can value the data is by processing it, implying that it already received it. Additionally, the only way to determine the feedback effects of sharing information, is to share it to find out how that changes $\hat{\theta}$. These are both unattractive options for all parties involved, and hence, data will be sold at far to high or far to low a price, these prices will be slow to respond to changes in learning or data quality and total welfare will suffer. Hence, we would like to create a mechanism that, given the information of the firms and access to the data sets, assigns an allocation of both data and funds. In this way, it is similar to the idea of a combinatorial auction discussed earlier with some significant variations. First of all, firms do not bid on subsets of data, they provide their information and utility functions and the mechanism decides how much they value these subsets; and second, the datasets can be replicated, removing one of the more difficult constraints of the combinatorial auction problem. The following chapter will be concerned with constructing allocations of data and wealth optimal for all participants.

2.3.1 Individual Data Valuation

Before we develop the methods to do this however, it is necessary to discuss an firm's data valuation. This is the quality of data we would like to describe however value is a nebulous term. In much of the previous research on scoring rules and active learning, the value of data is simply how it increases accuracies of prediction. In this paper, we will often take a similar approach, but it is worth pointing out that information is only useful in a situation where it can be exploited. If I obtain data that accurately predicts the economic growth of Kazakhstan, this isn't useful to me unless I can use that information. In this research we assume that data serves the following role. A firm is playing a game with actions $a \in A$ such that, depending on the state of the game $\theta \in \Theta$, they receive utility $u(a, \theta)$. The goal then, given the firm knows the true state θ^*, becomes to play a that satisfies $a = \operatorname*{argmax}_{a \in A} u(a, \theta^*)$. Before soliciting information, the firm has an estimate of θ informed by either past information other priors. They also have the ability to incorporate new data into this estimate of θ using some predetermined algorithm with the goal of improving utility. The value of information is simply then the difference in utility with and without using the data to estimate θ and obtain optimal actions $a \in A$.

This estimation of value is very dependent on the application. If we are trying to predict the price of a particular stock θ, it is not possible to determine the utility increase of data since we can't tell what θ will be. One solution to this is to obtain data about past stock prices to use current data as a predictive tool. This brings up a subtle but important distinction in how data is used. Some data is used to train the model. In the case of a stock, this might be historical correlations between different data and said stock. This information is used to tune the algorithm that gives the best actions $a \in A$ possible. Additionally data is used as a signal of the current state. In the stock pricing example, this might be current economic indicators or social media sentiment that we use to feed into our algorithm to give us a. Both uses of data increase prediction accuracy, but in different ways. All this raises the question why we don't simply just quantify value as an increase in accuracy. It could be the case that certain mistakes still lead to the right $a \in A$ while other minor mistakes lead to catastrophic loss in utility. So we are not only concerned with how data improves our model, but how that model impacts utility. This distinction is not present in any of the current literature.

We will assume for the remainder of the thesis that the system that allocates data to firms has the ability to calculate functions $V_i(S)$ that quantify the increase in utility of firm f_i obtaining and learning from data $d_j \in S \subseteq \{d_1, ..., d_m\}$. Additionally, as competition between firms and feedback from data to vendors is difficult to model and analyze, we assume that the functions V_i represent only the value firm f_i gets, and the vendors are simply paid some profit. We include an example before moving onto the next chapter.

A Hidden Markov Model

Suppose we are a financial firm tracking the price x_t of an asset over time. We assume that this asset's price is described by some m state hidden Markov model. At the outset, we have a set of priors on the transition probabilities, in the model which we would like to update using past observations. We also have access to historical prices. Observations in this case can be bought at some price p_i. Given no observations, all we have is the historical prices, and hence, can calculate the estimated distribution of prices at time m. Ideally, we would like to get all observations $y_0, ..., y_{m-1}$ so that we can get an accurate estimate of $\mathbb{P}(x_i|y_i)$. If we can get such an estimate, and our previous knowledge of the Markov model gives us $\mathbb{P}(x_{m-1}, x_m)$ then if we can obtain observation y_m, we should be able to accurately estimate the future price x_m. Depending on the prices of the past observations and the states $...x_{-1}, x_0, x_1, ..., x_{m-1}$, it should be optimal to get some subset of observations $y_1, ..., y_{m-1}$ to train the model.

Then, getting y_m gives us a signal about the future value we want to predict. This illustrates the multifaceted uses of data, both as a training tool and as an indicator of the current state. Now, assume that the utility of our firm is described by $-(||x_m - \tilde{x_m}||)^2$ where $\tilde{x_m}$ is our estimate of the future stock price. In this case, since we do not yet have access to the value x_m, what we may do to estimate the value of getting some observations y_S for $S \subseteq [m]$ would be to first get an estimate of $\tilde{x}_i$, for all $i \in S$ without using any observations, then train the model using the data y_S to get new observations $\tilde{x}_i'$. Then we can measure the difference in utility both learning and not learning the data by taking $V(y_s) = \sum_{i \in S} -(||x_i - \tilde{x}_i'||)^2 + (||x_i - \tilde{x}_i||)^2$. Then, this should give some estimate of the value of the subset $y_S \subseteq \{..., y_{-1}, y_0, ..., y_{m-1}\}$. Finally in order to assess the value of the current observation y_m, we use the model, trained using y_S, and measure the increase in utility over time both with and without current observations. In other words, we take

$V(y_m) = \frac{1}{|S|}\left(\sum_{i\in S} -(||x_i - \tilde{x_i}'||)^2 + (||x_i - \tilde{x_i}||)^2\right)$ where $\tilde{x_i}'$ is the value of x_i estimated by the trained model with observation y_i, and $\tilde{x}_i$ is the estimate without this observation.

Hence, the total value of learning subset $y_S \cup y_m$ becomes simply the sum of their two values $V(y_S \cup y_m) = V(y_S) + V(y_m)$. Note that all of these values can be calculated empirically and give an estimate of the increase in utility of the firm by learning data.

Simulations

Simulations using a similar model and four data sets demonstrate that, while initially, data largely improves estimation, additional information has decreasing, or even possibly negative, marginal returns. This is seen in Figure 2-4 where the Hasse Diagram of the partially ordered set defined by inclusion of subsets of all data $\{d_1, d_2, d_3, d_4\}$ is labeled with the accuracy improvements achieved by learning the data corresponding to each node. It is clear that as we look higher in the graph, and therefore learn more data, the accuracy improves.

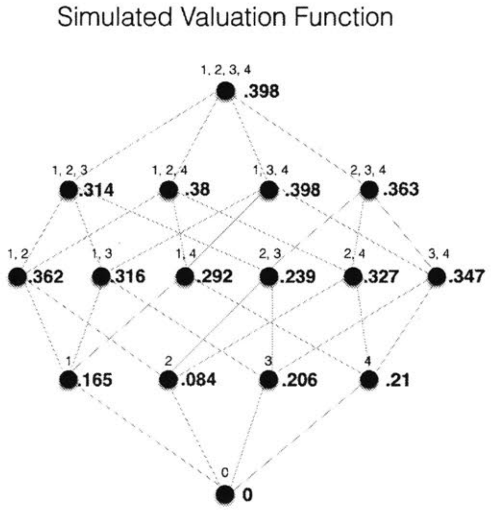


Figure 2-4: Simulated Valuation Function

This is further seen in figure 2-5 where the values $\{V_i(0), V_i(d_1), V_i(d_1 \cup d_2), V_i(d_1 \cup d_2 \cup d_3), V_i(d_1 \cup d_2 \cup d_3 \cup d_4)\}$ are plotted for several different V_i functions. These different V_i's were generated by allowing each agent f_i access to different prior information. It is easily seen though that in general, learning follows an increasing, subadditive trend.

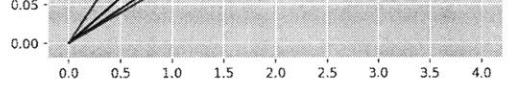


Figure 2-5: Comparison of Different Valuation Functions

Chapter 3

Data Matchings

3.1 Data Matching Mechanisms

In the previous chapter, we defined the problem faced by a mechanism designer valuing and selling data. In this section, we will describe several methods to allocate data that guarantee truthfulness given some structure on data and learning. We later develop and analyze an algorithm to find optimal matchings.

Definition 8. *In a data valuation game, the outcome of playing is called an allocation and can be represented by sets $S_1, ..., S_n \subseteq \{d_1, ..., d_m\}$ representing which data is allocated to which firm, and prices p_{ij} paid by firm i to data set j if $j \in S_i$.*

Our goal is to find data allocations that maximize both profit and learning. This can also be visually represented by a weighted bipartite graph $\mathcal{M}$ such that there are n nodes representing firms, m nodes representing vendors and edges only exist between these groups. The weight of the edge encodes the price paid for the data.

Definition 9. *Given a data valuation game, a* **data matching graph** *is defined as the weighted bipartite graph $\mathcal{M}$ between nodes representing firms and nodes representing vendors such that the weight of each edge represents the amount of money paid by a particular firm for a particular dataset.*

Note that an allocation in the sense of definition 8 corresponds to a unique data matching graph. The challenge of designing a mechanism to value and trade data then becomes finding a good matching $\mathcal{M}$. We need the parties to present their information truthfully, otherwise the value of data will be ill-defined. This means that firms will not be incentivized to alter their information and vendors need to properly represent their data as well. If we design a mechanism in which there is an incentive to lie, then the value of a piece of data in that case is meaningless. Additionally we need to design a mechanism in which there is some incentive to play i.e. participants always have some positive benefit. Finally, there may be many matchings $\mathcal{M}_1, ..., \mathcal{M}_p$ that satisfy the above conditions. To distinguish among them, we need to define some kind of system wide goal such as matchings that maximize total learning or maximize total revenue (the sum of edge weights in the graph.)

Note that, from the perspective of both firms and vendors, the data matching graph contains all the information necessary to determine their utility. Hence, we may rewrite the utility of the firms as simply $V_i(\mathcal{M})$ and of vendors as $u_j(\mathcal{M})$. One important consideration in designing an auction (as this is a highly complex example of an auction) is that the agents who value an object the most receive that object. In our case, the value a firm has for an individual piece of data is linked to what other data they have received, as well as the information of other firms. One method with which to measure this would be to look at the incremental difference of $V_i(\mathcal{M})$ when an edge, representing transmission of a single data set, is removed.

Definition 10. *Suppose that firm f_i and data set d_j are neighbors in the data matching graph $\mathcal{M}$. The incremental value $V_{ij}(\mathcal{M})$ of a dataset d_j to the firm f_i is the difference $V_i(\mathcal{M}) - V_i(\mathcal{M}')$ where $\mathcal{M}'$ is the same as $\mathcal{M}$ minus the edge connecting f_i and d_j.*

This definition works fine when we consider data that firm f_i is already allocated. However, suppose we wanted to determine f_i's incremental value for a piece of data it does not have like d_k. In this case, we may add an edge to $\mathcal{M}$ and do the same procedure, but it is unclear what weight this edge should be. This brings us to our first simplifying assumption on $\mathcal{M}$, namely that edges connected to a dataset d_k must all have the same weight. In fact, this will play a pivotal role in designing a truthful payment mechanism later on. With this in mind, we may define the incremental value of data d_k to firm f_i as the difference $V_{ik}(\mathcal{M}) = V_i(\mathcal{M}') - V_i(\mathcal{M})$ where $\mathcal{M}'$ contains an edge between d_k and f_j whose weight is determined by the algorithm generating matchings.

Now, back to auctions. The goal of an auction is to allocate goods to the firms that value them most. With the above definitions we may define the notion of a stable matching. Let $S_i(\mathcal{M})$ be the set of data sets received by firm f_i in matching $\mathcal{M}$ and likewise let $R_j(\mathcal{M})$ be the set of firms receiving data set d_j in that same matching. For the remainder of this section, we suppose that $k_j = R_j(\mathcal{M})$ is a fixed parameter of the system. So vendors must decide ahead of time exactly how many times to replicate their data.

Definition 11. *Given a data valuation game and an associated data matching graph $\mathcal{M}$, the* *matching $\mathcal{M}$ is called a* **stable matching** *if for every firm i and every data set $j \in S_i(\mathcal{M})$, we have that $V_{ij}(\mathcal{M}) \geq V_{kj}(\mathcal{M})$ for every $k \notin R_j(\mathcal{M})$.*

This means that a matching $\mathcal{M}$ is stable if there is never the case that firm f_i incrementally values d_j more than another firm f_k, and d_j is allocated to f_k and not f_i.

Proposition 12. *If a matching $\mathcal{M}$ is not stable it cannot maximize total learning.*

Proof. Define $\mathcal{V}(\mathcal{M}) = \sum_{i=1}^{n} V_i(\mathcal{M})$ and note that if $\mathcal{M}$ is not stable, it is the case that firm f_i values d_k more than f_j and d_k is only allocated to f_j. Here, if we alter the matching $\mathcal{M}$ to $\mathcal{M}'$ such that instead of an edge between d_k and f_j there is now an edge between d_k and f_i, by definition, the utility $\mathcal{V}(\mathcal{M}) \leq \mathcal{V}(\mathcal{M}')$ and hence $\mathcal{M}$ cannot maximize total learning. □

As the above result shows, if we would like to maximize social welfare in terms of learning (or more generally bidder utility) we must consider only stable matchings.

Before beginning to describe a method to build such matching $\mathcal{M}$ consider a simple example. Suppose we have two firms f_1, f_2, and two data sets d_1, d_2 such that the utility of each firm is given by $u_1(\varnothing) = 0, u_1(d_1) = 1, u_1(d_2) = 1, u_1(d_1 \cup d_2) = 10$ and $u_2(\varnothing) = 0, u_2(d_1) = 2, u_1(d_2) = 2, u_1(d_1 \cup d_2) = 4$.

At this point there are several methods to consider. We might first take a greedy approach, simply adding an edge at the point where the incremental value $V_{ij}(\mathcal{M})$ is the greatest. In this case, we must decide what the weight of the edge should be and at what point to stop adding edges, lest we end up with a complete graph.

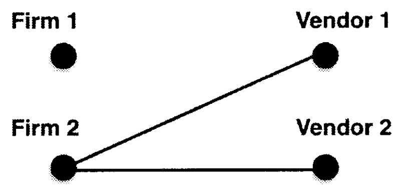


Figure 3-1: A Greedily Constructed Matching.

Doing this and stopping after adding 2 edges gives the following stable graph seen in figure 3-1. However, the high value of giving both d_1 and d_2 in combination to firm f_1 suggests that the matching seen in figure 3-2 would be more profitable. Furthermore, if we continue this greedy process until there are 3 edges, we obtain a graph that is no longer stable, implying that someone, f_1 who values d_2 more than f_2 doesn't get it.

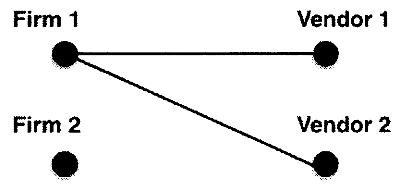


Figure 3-2: A Better Matching.

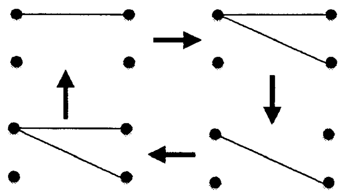


Figure 3-3: A Cycling Algorithm Never Terminates.

We would like to avoid the possibility that this algorithm might cycle between different matchings, never converging to an optimal graph as seen in figure 3-3 and we additionally need to make sure that whichever matching this algorithm terminates at is truthful. If, for example, we construct a graph such as in figure 3-4 where the firms pay exactly their valuations for the data, they will be incentivized to lie about those valuations to the system.

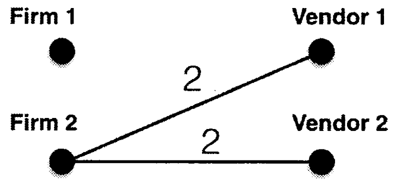


Figure 3-4: An Untruthful Matching.

In this case truthful reporting is no longer a Nash equilibria, a situation we need to avoid. In total, we are looking for an algorithm with the following properties:

- The algorithm to find $\mathcal{M}$ can't get stuck at a solution that is not stable, in the sense defined above.
- The algorithm cannot cycle.
- The algorithm must terminate at a matching $\mathcal{M}$ that incentivizes truthful reporting from firms and vendors.

Note that firms providing the mechanism with information $\{\mathcal{I}_1, ..., \mathcal{I}_n\}$, $g_i(\hat{\theta})$, and $\mathcal{A}_i$, is mathematically the same as simply providing functions $V_{ij}(\mathcal{M})$. Hence for notational brevity we will assume that firms simply report these functions.

Proposition 13. *Consider a data valuation game with n firms and one vendor and one corresponding data set d. Suppose that the vendor has already decided it will replicate the data $1 \leq k < n$ times. In this case, there exists a truthful mechanism to allocate the data.*

Proof. In order to construct such a mechanism, we will generalize the idea of a second price auction, in which the winning bidder must pay the second highest bid for the object in question. Here, we define the k winners to be those k firms $f_1, ..., f_k$ (without loss of generality firms are ordered by $V_1(d) > V_2(d) > ... > V_n(d)$) who have the k highest values $V_i(d)$ for the data set. Each of these firms pay $V_{k+1}(d)$ for the data d. In this case, no firm has an incentive to change their information. To see this, we will consider two case. First, consider a firm f_i who wins the data by submitting information truthfully. Then, suppose they lie about their information, changing their valuation of the data to $\hat{V}_i(d)$. If $\hat{V}_i(d) \geq V_i(d)$, they still win the object and have the same positive utility $V_i(d) - V_{k+1}$ which is the same as if $V_{k+1}(d) < \hat{V}_i(d) < V_i(d)$. Finally, if $V_{k+1}(d) > \hat{V}_i(d)$ the firm f_i gets utility 0. So in any of these cases, there is no incentive to lie about the information firm f_i provides. Second, suppose f_i does not get allocated d when they truthfully provide information. If they lie to get a new valuation $\hat{V}_i(d)$, there are several possibilities: if $\hat{V}_i(d) \leq V_i(d)$, they still lose the object and get utility 0; if $V_k(d) > \hat{V}_i(d) > V_i(d)$, again they get utility 0; and finally if $V_k(d) < \hat{V}_i(d)$, they receive the object and get negative utility $V_i(d) - V_k(d)$. So in any case, there is no incentive to lie, and this is a truthful mechanism. □

Definition 14. *In a data valuation game, we will call the above mechanism a* **generalized second price auction**.

3.1.1 Separable Data

In the case of more than one data set, we need to find the right amount firms should pay for data sets in transactions given by $\mathcal{M}$. To this end, suppose we have a data valuation game with n firms, m vendors and an associated data matching graph $\mathcal{M}$. Suppose further that this graph is stable. As before, let $S_i(\mathcal{M})$ be the set of data sets received by firm f_i in matching $\mathcal{M}$ and let $R_j(\mathcal{M})$ be the set of firms receiving data set d_j in that same matching. Then we may generalize the Vickrey auction discussed in the introduction by setting the price of each piece of data sold to the highest incremental valuation of that data by those who did not receive it. In mathematical terms, the price $p_j(\mathcal{M})$ of data d_j in the matching $\mathcal{M}$ is given by $\max\limits_{k \notin R_j(\mathcal{M})} V_{kj}(\mathcal{M})$. If the set $R_j(\mathcal{M}) = \{f_1, ..., f_n\}$ then we set the price $p_j = 0$. It can be shown that in the case of multiple data sets, this is not necessarily a truthful mechanism. However, by putting certain structure on the way firms value data, this can be remedied.

One major difficulty in trying to sell and value data in a market is that many different pieces of information may contain substitutive value. For example, if a climatologist were looking for data about greater Boston area, the weather reports in Cambridge and Boston would tell them very similar information. Hence, both data sets are not necessary for learning. If we suppose though, that one of the stipulations of the market is that vendors sell differing data sets, we could avoid this problem. What we would like to be able to do is, given the number of times each data set is replicated k_j, auction each good off individually to construct the desired data matching.

Definition 15. *Given a data valuation game with n firms and m vendors, the data sets $d_1, ..., d_m$ are called separable if for every pair of data matchings $\mathcal{M}$, $\mathcal{M}'$, every $i \in [n]$ and every $j \in [m]$ we have $V_{ij}(\mathcal{M}) = V_{ij}(\mathcal{M}')$.*

With this in place, let's consider a greedy method of constructing matchings by choosing a single data set at a time to auction. In this case we order the data sets by the amount of total revenue they receive in a generalized second price auction as above, then iteratively perform these auctions to create the matching. Suppose for the remainder of this section that there are no ties in valuing data i.e. $V_{ij}(\mathcal{M}) \neq V_{kj}(\mathcal{M})$ for all k, i, j and matchings $\mathcal{M}$. This makes tie breaking a non issue. This could also be solved by saying that if firms tie, if in doubt the firm with the smaller index is allocated the good.

Algorithm 1 Elementwise Matching Contstruction

```
Input: $V_1, ..., V_n$: valuation functions; $d_1, ..., d_m$: separable datasets; $k_1, ..., k_m$: number of reproductions of each data set.
1: for $j$ from 1 to $m$ do
2:     Do: Generalized second price auction auction:
3:     Find $k_j$ highest bids $V_{ij}(\emptyset)$. Call the $k_j + 1$th highest bid $V_j^*$.
4:     Add an edge $e_{ij}$ of weight $V_j^*$ for all $i$ with $k_j$ highest bids.
5: Return: Optimal stable matching $\mathcal{M}$ with edges given by the process above.
```

Theorem 16. *Given a data valuation game with separable data, a stable matching $\mathcal{M}$ with payments defined using the generalized second price auction is unique and truthful.*

In order to prove this, we state and prove the following lemmas.

Lemma 17. *In a data valuation game with separable data, a stable matching can be found by individually performing m generalized second price auctions on the single data sets $d_1, ..., d_m$ where values for firms are given simply by $V_{ij}(\emptyset)$, and then combining these auctions to construct a matching. In other words, there will be an edge between f_i and d_j in $\mathcal{M}$ if and only if in the auction for d_j, f_i is allocated the data set.*

Proof. Consider such a matching $\mathcal{M}$. If $\mathcal{M}$ were unstable, it would be the case that for $i \in R_j(\mathcal{M})$ and $k \notin R_j(\mathcal{M})$, $V_{ij}(\mathcal{M}) < V_{kj}(\mathcal{M})$. But, since the data is separable this would imply that $V_{ij}(\emptyset) < V_{kj}(\emptyset)$. This is a contradiction since the generalized second price mechanism on a single data set would not have allocated the data set d_j to f_i instead of f_k if this were the case. $\square$

Lemma 18. *A data valuation game with separable data has a unique stable matching.*

Proof. Note that the number of times each data set is replicated is fixed by the vendors at $k_1, ..., k_m$. Hence, consider any matching $\mathcal{M}'$ different to the matching $\mathcal{M}$ constructed in the previous lemma. Suppose $\mathcal{M}'$ has an edge between f_i and d_j where $\mathcal{M}$ does not. This implies by the above construction that $V_{ij}(\mathcal{M}) < V_{kj}(\mathcal{M})$ for some k who is not allocated the data d_j in $\mathcal{M}'$. Hence $\mathcal{M}'$ cannot be a stable matching. □

Lemma 19. *A data valuation game with separable data is truthful under the generalized second price auction mechanism and the above matching algorithm.*

Proof. This fact follows from the fact that the matching, and hence the utility of each firm, is simply a combination of individual auctions. Hence, by changing their information firm f_i is playing m different auctions in which he will not benefit from changing his strategy as shown in proposition 13. Thus there is no incentive for firms participating in this market mechanism to lie about information they provide to the system. □

In total, these three lemmas prove theorem 16, and furthermore, since there exists a unique equilibrium in this case, the notion of the price, or value, of data is well defined. Related to the idea of separability above is the notion of partially separability data

Definition 20. *Given a data valuation game with n firms and m vendors, the data sets $d_1, ..., d_m$ are called partially separable if there exists a partition $\mathcal{S} = \{S_i\}$ of $\{1, ..., m\}$ such that the function $V_i(d_1, ..., d_m)$ can be decomposed into the sum $\sum_j V_{i,S_j}(d_{S_j})$ where d_{S_j} represents data in the set S_j.*

Now suppose we have a method $\mathcal{A}$ with which to define an optimal matching in a data valuation problem.

Corollary 21. *Given a data valuation game with partially separable data partitioned by $\mathcal{S} = \{S_i\}$ and optimal algorithm $\mathcal{A}$ to assign matchings, an optimal matching for this case can be found by iteratively applying algorithm $\mathcal{A}$ to the subproblems selling data S_i to firms in order to construct a matching for all data $\{1, ..., m\}$.*

Sadly in most cases, the restriction of separability or partial separability is unrealistic, however, we may generalize this method.

3.1.2 ϵ Separability

Definition 22. *Given a data valuation game, data sets $d_1, ..., d_m$ are called ϵ separable if for any matching $\mathcal{M}$, we have $|V_{ij}(\mathcal{M}) - V_{ij}(\emptyset)| < \epsilon$. In other words, data in combination is roughly the same in value when considered individually.*

As a market designer, one can imagine that the system itself gets some percentage of each transaction and hence, would like to maximize the total revenue of any stable matching it constructs. Along these lines, the system would like to maximize the total value $\mathcal{V}$ of matchings it selects.

Definition 23. *The total value $\mathcal{V}(\mathcal{M})$ of matching $\mathcal{M}$ is the total revenue, or sum of all edge weights, in the allocation. The optimal such $\mathcal{V}$ for a data valuation game is called the value of the game.*

In the case of separable data, this is not an issue since there is only a single stable matching. However, if we have a data valuation game with ϵ-separable data, we may still construct a matching, that is not necessarily truthful or stable, but that has nice guarantees on its total value. Before we state this result, we introduce the idea of subadditive learning.

Definition 24. *In a data valuation game, we say that learning is subadditive if for every firm f_i, and every data set d_j, $V_{ij}(\mathcal{M}) < V_{ij}(\mathcal{M}')$ if $S_i(\mathcal{M}') \subseteq S_i(\mathcal{M})$. In other words, the incremental value of data decreases as more data is processed.*

Using these relaxations of the earlier constraints on the value functions, we can modify our original algorithm so that as before, the system individually auctions the data sets, but now the winners pay the alternative amount for data d_j given by $\max\limits_{k \notin R_j(\mathcal{M})} V_{kj}(\mathcal{M})$. It is possible, if ϵ is large or if there is other instability in the system that this matching will not be stable.

Algorithm 2 Modified Elementwise Matching Contstruction

```
Input: $V_1, ..., V_n$: valuation functions; $d_1, ..., d_m$: $\epsilon$ separable datasets; $k_1, ..., k_m$: number of reproductions of each data set.
1: for $j$ from 1 to $m$ do
2:     Do: Generalized second price auction auction on functions $V_{ij}(\emptyset)$:
3:     Find $k_j$ highest bids $V_{ij}(\emptyset)$.
4:     Add an edge $e_{ij}$ for all $i$ with $k_j$ highest bids.
5: Return: Unweighted matching $\mathcal{M}$ with edges specified by the process above.
6: for $j$ from 1 to $m$ do
7:     For each $d_j$, set edge weight of all edges in $\mathcal{M}$ connected to $d_j$ to $\max_{k \notin R_j(\mathcal{M})} V_{kj}(\mathcal{M})$
8: Return: Weighted matching $\mathcal{M}$.
```

In addition to this, we see that data valuation games that are simply perturbed versions of games with separable data inherit many of the nice properties of their more basic predecessors.

Theorem 25. *Given a data valuation game with ϵ-separable data, as long as $\epsilon < \delta(V_1, ..., V_n, d_1, ..., d_m)$ where $\delta(V_1, ..., V_n, d_1, ..., d_m)$ is some parameter of the system, there exists a polynomial time algorithm to find a stable matching.*

Proof. Suppose that using values $V_{ij}(\emptyset)$ we construct the unique matching $\mathcal{M}$ using algorithm 2. In this case, for each data set d_j, we can define the value δ_j to be the difference

$$\delta_j = \min_{\substack{i \in R_j(\mathcal{M}) \\ k \notin R_j(\mathcal{M})}} V_{ij}(\emptyset) - V_{kj}(\emptyset).$$

Then using these parameters we can define $\delta' = \min_{1 \leq j \leq m} \delta_j$ with which we can further define $\delta(V_1, ..., V_n, d_1, ..., d_m) = \delta'/2$. Note then that if $\epsilon < \delta(V_1, ..., V_n, d_1, ..., d_m)$, the matching that algorithm 2 provides remains stable since

$$\min_{\substack{i \in R_j(\mathcal{M}) \\ k \notin R_j(\mathcal{M})}} V_{ij}(\mathcal{M}) - V_{kj}(\mathcal{M}) > \min_{\substack{i \in R_j(\mathcal{M}) \\ k \notin R_j(\mathcal{M})}} V_{ij}(\emptyset) - V_{kj}(\emptyset) - 2\epsilon > 0.$$

This implies that for any firm f_i receiving data d_j in the matching $\mathcal{M}$, we have $V_{ij}(\mathcal{M}) > V_{kj}(\mathcal{M})$ for all $k \notin R_j(\mathcal{M})$, hence the matching $\mathcal{M}$ is stable. □

Thus, given any game, there is a simple way of checking whether or not we may use algorithm 2 to find a matching. We simply need to find the value of $\delta(V_1, ..., V_n, d_1, ..., d_m)$ that belongs to this game and then determine how ϵ-separable the data is. As long as $\epsilon < \delta(V_1, ..., V_n, d_1, ..., d_m)$ the algorithm will converge efficiently to a desirable solution.

These results suggest another benefit of data matching mechanisms. As discussed earlier, it is important for vendors to sell unique data. Otherwise, if all sources of information are essentially the same as far as learning is concerned, competition will drive down prices and there will cease to be an incentive to sell. One method to solve this problem would be to only include data that satisfies the properties stipulated above. In particular, suppose there is a system with firms $f_1, ..., f_n$ and data sets $d_1, ..., d_m$ that satisfy ϵ separability with $\epsilon < \delta(V_1, ..., V_n, d_1, ..., d_m)$. In this case, suppose there is an additional data set d_{m+1} that the system could value and sell as well. Before including it in the transaction however, the mechanism could calculate the new value $\delta(V_1, ..., V_n, d_1, ..., d_m, d_{m+1})$ as well as the new ϵ associated to the game. If ϵ remains smaller than $\delta(V_1, ..., V_n, d_1, ..., d_m, d_{m+1})$, then it makes sense to include the new data d_{m+1}, otherwise it could be discarded. Now, calculating the new ϵ is highly computationally taxing since it involves examining the new values $V_{ij}(\mathcal{M})$ where d_{m+1} is included. However, it turns out this is unnecessary. All that is needed is to use algorithm 2 to get a matching $\mathcal{M}$ with d_{m+1} included and then determine the maximum deviance of $V_{ij}(\mathcal{M})$ from the values $V_{ij}(\emptyset)$. In this sense we can define

$$\tilde{\epsilon} = \max_{\substack{1\leq i\leq n \\ 1\leq j\leq m+1}} |V_{ij}(\mathcal{M}) - V_{ij}(\emptyset)|$$

. If $\tilde{\epsilon} < \delta(V_1, ..., V_n, d_1, ..., d_m, d_{m+1})$ by the same reasoning as in Theorem 25, Algorithm 2 converges in polynomial time to a stable matching. Hence, by selectively choosing the data to include in a transaction, we can guarantee that we may use an efficient algorithm to value and sell data. These results are summarized by the following algorithm.

Algorithm 3 Discriminative Data Selection

```
Input: $V_1, ..., V_n$: valuation functions; $d_1, ..., d_m$: $\epsilon$ separable datasets; $k_1, ..., k_m$: number of reproductions of each data set; $d_{m+1}$, $k + m + 1$: new data set with reproduction limit.
 1: for $j$ from 1 to $m + 1$ do
 2:     Do: Generalized second price auction auction on functions $V_{ij}(\emptyset)$:
 3:     Find $k_j$ highest bids $V_{ij}(\emptyset)$.
 4:     Add an edge $e_{ij}$ for all $i$ with $k_j$ highest bids.
 5: Return: Unweighted matching $\mathcal{M}$ with edges specified by the process above.
 6: for $j$ from 1 to $m$ do
 7:     For each $d_j$, set edge weight of all edges in $\mathcal{M}$ connected to $d_j$ to $\max_{k\notin R_j(\mathcal{M})} V_{kj}(\mathcal{M})$
 8: Return: Weighted matching $\mathcal{M}$.
 9: for $i$ from 1 to $m + 1$ do
10:     $\delta_j = \min_{\substack{i\in R_j(\mathcal{M}) \\ k\notin R_j(\mathcal{M})}} V_{ij}(\emptyset) - V_{kj}(\emptyset)$.
11: $\delta = \left(\min_{1\leq j\leq m} \delta_j\right) /2$
12: $\tilde{\epsilon} = \max_{\substack{1\leq i\leq n \\ 1\leq j\leq m+1}} |V_{ij}(\mathcal{M}) - V_{ij}(\emptyset)|$
13: if $\tilde{\epsilon} < \delta$ then
14:     Include $d_{m+1}$
15: else
16:     Discard $d_{m+1}$
```

3.1.3 Simulations

In order to simulate the operation of the greedy algorithm described in this chapter, we first needed to develop a method with which to construct valuation functions in a way that their separability could be controlled. This was achieved by writing such functions $V(S)$ as functions of indicators $V(z_1, ..., z_m)$ where z_j is 1 if data set j goes to this firm. Then

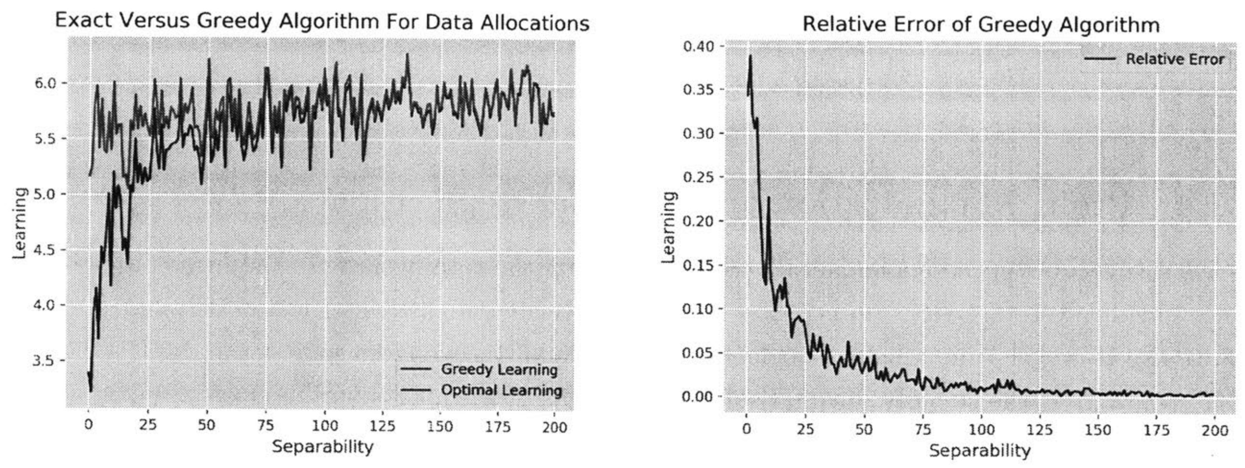


Figure 3-6: Greedy Versus Optimal Learning.

we can simplify this expression by creating a quadratic form $V(z_1, ..., z_m) = z^T Qz$. Note that in the case the data is separable, Q will be a diagonal matrix. Hence, by perturbing diagonal matrices to increasing degrees, we are able to construct valuation functions with a controllable level of separability.

With this, we considered the case of 5 firms $f_1, ..., f_5$ and 4 data sets $d_1, ..., d_4$ such that each firm is assigned a random Q with some level of separability. Then the greedy algorithm described above was executed resulting in some level of global learning and a corresponding matching. Finally, the optimal allocation was calculated with the same associated statistics. We saw that, for moderate amounts of separability, the matchings provided by both mechanisms, seen in Figure 3-5, are not very different. Additionally, as the amount of separability increases (or the matrices Q become closer and closer to diagonal) the graphs in Figure 3-6 show that the learning provided by the greedy algorithm approaches the optimal learning.

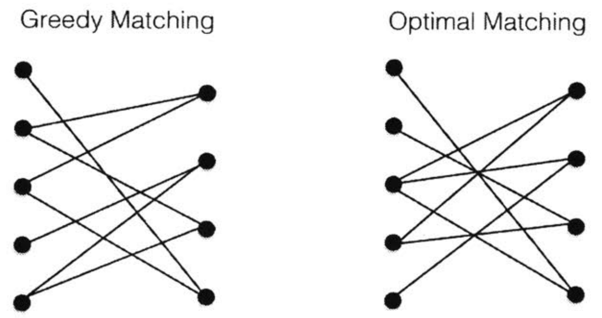


Figure 3-5: Greedy Versus Optimal Matching Given Moderate Separability

These simulations support the results above in that they show how our greedy algorithm works extremely well with separable or near separable data. It also demonstrates that when data is only moderately separable, in our case meaning the matrices Q are only somewhat diagonally dominant, the greedy algorithm still performs well. This implies that in practice, such an algorithm would be very successful in assigning approximately optimal data allocations.

3.1.4 Profit Maximization and Optimal Replication

In the results above, the goal generally was to optimize the total welfare of the firms participating in the system. We examined how several algorithms might construct socially optimal solutions. However, this analysis neglected the utility of the vendors selling the data. One could argue that the socially optimal solution in terms of learning must be close to optimal for vendor's profit as well since, if firms are learning more they are willing to pay more. However, in the design of such algorithms, sometimes vendor's profit falls by the wayside in order to incentivize truthfulness. This isn't a good situation and could drive vendors to leave the market if they are not being fairly compensated for their data. For example, one might consider the case in which a data set is highly valuable to only one firm, and negligibly valuable to the others. Using the mechanism described above, the vendor in this case would not be able to gain much profit from selling this data, despite its value to one firm. In order to fix such a situation, more attention should be paid to the vendor's side of things.

Above we assumed that the vendors have the ability to replicate their data a certain number of times k_j. With something like data, this replication is free, and so it begs the question why vendors wouldn't be willing to replicate their information n times. Broadly, the reason they wouldn't want to do this boils down to supply and demand. If they is a glut of a certain data set it becomes less valuable than if it is scarce even when firms are essentially non competitive. This is clear in the above mechanism where replicating n times would give profit of 0, and in other scenarios. Hence, k_j, the number of times a data set is copied becomes an important parameter with which vendors, or the system, can maximize profit. To see how this might work, consider the following example.

Example 26. *Consider a data valuation game in which there is one data set d_1, and n firms each with value $v_1 > v_2 > \ldots > v_n$ for this information. Suppose we define k to be the number of times that d_1 is copied, and using the generalized second price auction mechanism, we would like to find the k that maximizes profit. In this case, we will calculate $k^* = \operatorname*{argmax}_{k \in [n-1]} v_{k+1}k$. This is all well and good, however, consider the $k^* + 1^{th}$ firm. They now have an incentive to lower their valuation $v_{k^*+1} \to v_{k^*+2} + \epsilon$. In this case, the value in the maximization $k^* = \operatorname*{argmax}_{k \in [n-1]} v_{k+1}k$ becomes $k^* \to k^* + 1$ so that now f_{k^*+1} is included in the allocation of d_1. However, this means that the mechanism is no longer truthful strategy* *dominant.*

As we can see by the above example, changing the parameters of the mechanism in response to firms' reports generally leads to situations that can be exploited by lying. In fact, in a perverse way, your truthfulness depends on the fact that the price of a good is completely independent for your valuation of that good. Otherwise, you may manipulate the system. This seems to suggest that attempting to maximize vendor's profit is entirely pointless if it will always lead to a mechanism that is not truthful, however, we still have a few tricks up our sleeve. Let's stick with the case of a single data set, and change our model slightly so that the vendor now offers a price p for that data set, and firms who have $v_i \geq p$ are allocated the data at that price. We might imagine doing the same sort of optimization over p as we did before with k to maximize profit, but again we see this creates an untruthful mechanism (for identical reasons.) However, in this paradigm we can come up with several work arounds. First, consider the case that the vendor specifies a target profit R they would like to obtain from the transaction. In this case, we can define a truthful mechanism that will achieve this if possible. To do this, we search over the v_i's to find a set $S \subseteq [n]$ with $|S| = k$ such that for all $i \in S$, $v_i \geq R/k$. If such a set can be found, then for the smallest possible k, we allocate the data to these f_i for $i \in S$ for a price R/k. Here, there is no incentive to lie as doing so will not increase your utility.

This is an encouraging result that might suggest that we look for the best R possible, but again, this would run into the same problems we had before. The way to fix this is to partition $[n]$ randomly into two subsets F_1, F_2 such that we may calculate $R_1 = \max_{p \geq 0} p|\{i \in F_1 | v_i \geq p\}|$ and $R_2 = \max_{p \geq 0} p|\{i \in F_2 | v_i \geq p\}|$. Using these two values, we can run the same auction like mechanism above using R_1 on F_2 and R_2 on F_1. It is important to note that either $R_1 \leq R_2$ or $R_2 \leq R_1$. Suppose without loss of generality that $R_1 \leq R_2$, then we are guaranteed to find k firms in F_2 such that $v_i \geq R_1/k$, so we are guaranteed a profit of $\min\{R_1, R_2\}$ from this process. This trick allows us to at least partially optimize the profit gained from selling a single data set in a truthful way, but unfortunately, there is no clear way to generalize this to the case of multiple data sets (unless they are separable.) This drawback aside, we do have the following result:

Theorem 27. *In a data valuation game with one data set and n firms, the mechanism described above achieves expected revenue $\mathbb{E}(Revenue)$ such that*

$$\mathbb{E}(Revenue) \geq OPT/4$$

where OPT is the optimal revenue defined by $OPT = \max_{p \geq 0} p|\{i \in [n] | v_i \geq p\}|$.

Proof. OPT indicates an optimal price p^* for the data, and k firms who buy it. Additionally, after we randomly partition $[n]$ into F_1 and F_2, there are k_1 firms in F_1 who are allocated d_1 and k_2 firms in F_2 who get it. Now suppose we calculate $R_1 = \max p \geq 0 p|\{i \in F_1 | v_i \geq p\}|$ and $R_2 = \max p \geq 0 p|\{i \in F_2 | v_i \geq p\}|$. Here we see that $R_1 \geq p^* k_1$ and likewise $R_2 \geq p^* k_2$. Using this and the fact that $\mathbb{E}(\text{Revenue}) = \mathbb{E}(\min\{R_1, R_2\})$, we have that

$$\begin{aligned}\frac{\mathbb{E}(\text{Revenue})}{OPT} &\geq \frac{\mathbb{E}(\min\{R_1, R_2\})}{kp^*} \\ &\geq \frac{\mathbb{E}(\min\{k_1 p^*, k_2 p^*\})}{kp^*} \\ &\geq \frac{\mathbb{E}(\min\{k_1, k_2\})}{k}\end{aligned}$$

Now note that if we condition on the fact that there are k replications of the data, the expectation $\mathbb{E}(\min\{k_1, k_2\})$ can be bounded inductively. For $k = 1$, $\mathbb{E}(\min\{k_1, k_2\}) = 0$, and for $k = 2$, $\mathbb{E}(\min\{k_1, k_2\}) = 1/2$. Note that for k increasing $\mathbb{E}(\min\{k_1, k_2\}) = M_k = M_k - M_{k-1} + M_{k-1}$ and if we let the increment be defined as $X_k = M_k - M_{k-1}$ we see that $\mathbb{E}(\min\{k_1, k_2\}) = \sum_{i=1}^{k} X_k$. Then if we have i odd we see that $k_1 \neq k_2$ and hence, $X_i = 1/2$ whereas if i is even $X_i \geq 0$. This gives us that

$$\begin{aligned}\mathbb{E}(\min\{k_1, k_2\}) &= \sum_{i=1}^{k} X_i \\ &\geq \frac{k}{2}\frac{1}{2} = \frac{k}{4}\end{aligned}$$

But then, we see that $\frac{\mathbb{E}(\text{Revenue})}{OPT} \geq \frac{\frac{k}{4}}{k}$ and hence that $\mathbb{E}(\text{Revenue}) \geq OPT/4$. □

This is an interesting result in trying to optimally sell one data set, however, it cannot be clearly extended to the more combinatorially complex case of multiple data sets. Still though, it illustrates the importance of maintaining independence between prices and valuations in these sorts of transactions, an observation that will be crucial in much of this thesis.

In this chapter, we saw how to generalize a truthful mechanism in the presence of many data sets. We also demonstrated the difficulty in developing such a mechanism with which to sell data and showed, how truthful profit maximization might be achieved.

Chapter 4

Integer Programming Methods

4.1 Integer Programming and Incentive Compatibility

In mechanism design, the goal is to create a system in which individuals have the incentive to act in a way that optimizes a global variable. In the case of auctions, this welfare can be measured by the total utility of participants. This is usually optimal for the sellers of these goods as well since the more utility bidders receive in the system, the more they are willing to pay. In our case of valuing and selling data, note that if we have independent firms and vendors with no feedback, we may write the problem of finding a suitable matching that maximizes social welfare in terms of learning as the following integer program:

$$\begin{aligned}
\max \quad & \sum_{i=1}^{n} \sum_{S \subseteq M} V_i(S) y(S,i) \\
\text{s.t.} \quad & \sum_{S | j \in S} \sum_{i=1}^{n} y(S,i) \leq k_j \quad \forall j \in \{1, ..., m\} \\
& \sum_{S \subseteq M} y(S,i) \leq 1 \quad \forall i \in \{1, ..., n\} \\
& y(S,i) \in \{0,1\}
\end{aligned}$$

where $V_i(S)$ is the valuation of firm f_i for subset of data $S \subseteq M$, k_j is the reproduction limit of jth data set d_j and $y(S,i)$ is an indicator variable representing which data goes to which firm. This method of optimization is the most common technique in analyzing combinatorial auctions, however in our case we run into several difficulties. First of all, there is no reason that k_j should necessarily be limited or predetermined. We note that in the case that $k_j = n$ the solution of this problem is trivial with every firm receiving every dataset. Furthermore, this optimization problem does not specify payments for goods. This could be solved by taking the dual of the Lagrangian relaxation of the problem above, and interpreting dual variables associated with the first set of constraints as prices for goods. However, without guaranteeing that the relaxation of the problem is integral, there is no guarantee such a method will yield a reasonable solution. Also, the combinatorial complexity of the bids V_i themselves are difficult to work with. Later in this section we propose a method to simplify representational complexity.

Another problem with this formulation is that there isn't an incentive compatible mechanism to induce bidders to bid truthfully. In other words, they could lie about the function V_i to get a better outcome. In order to eliminate this possibility we as the auction designers need to develop a way to design payments to fix this. The most common way this is done is through a payment structure known as a VCG mechanism. Suppose we have solved the integer program above to get a solution y^* and total utility V. Then suppose we solve the same problem without firm k to get an optimal value V_{-k}. In this case, the system would charge agent k the price $V_{-k} - (V - \sum\limits_{S \subseteq M} V_k(S)y^*(S,k))$ which is precisely the difference in total utility of the system (other than firm k) with and without k's participation. In this case agent k's utility becomes $\sum\limits_{S \subseteq M} V_k(S)y^*(S,k) - \left(V_{-k} - (V - \sum\limits_{S \subseteq M} V_k(S)y^*(S,k))\right) = V - V_{-k} \geq 0$ since the addition of agent k can only increase the value of the optimal solution. Finally, this is incentive compatible since if agent k lies, the total value of the allocation calculated will be $V' \leq V$ and his own personal utility will be $V' - V_{-k} \leq V - V_{-k}$ so there is no incentive not to report truthfully. The problem with this is that calculating the prices involves actually solving the above integer program exactly multiple times. Furthermore, there are results showing that if you calculate valuations and solutions approximately, the same VCG mechanism may not be incentive compatible.

One nice result is that in the case of one data set and n firms, the VCG mechanism is exactly the generalized second price auction have previously been using. Additionally, in the case of separable data, the mechanism we use is also a VCG mechanism. For more complicated cases though, there exist fewer results.

4.1.1 Representational Complexity

In combinatorial auctions, the representational complexity of bids is a problem. In our case, this is compounded by the fact that the system itself has to calculate these bids in order to utilize them. This means that, in the case that n and m are even moderately large, the task of solving for the V_i's will become computationally infeasible. Generally in the literature this is solved by either stating that bidders are so called "single minded" in that they only want a single subset S or there is some other structure imposed limiting the complexity of bids. In our case, if we consider the case of separable data above, we see that the value functions $V_i(S)$ are given by $V_i(S) = \sum\limits_{j \in S} \alpha_j$ where α_j are fixed parameters of the system. This is fine if data has no complementary or substitutive value structures, however, this is more than likely too strong an assumption. One extension of this to allow for such complementary effects would be to measure pairwise "complementaryness" of different data sets. In other words, one might calculate $V_i(d_1) + V_i(d_2) - V_i(d_1 \cup d_2)$ to get an estimate of how much two pieces of data overlap. Suppose there were only two data sets d_1, d_2, then each valuation function must take on 4 values for $\emptyset, d_1, d_2, d_1 \cup d_2$. Here represent V_i as the function $V_i(z_1, z_2)$ where z_i is an indicator function associated to including data set i in the allocation to the firm. Then we can write $V_i(z_1, z_2) = \alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 z_1 z_2$ where $\alpha_1 = V_i(1, 0)$, $\alpha_2 = V_i(0, 1)$ and $\alpha_3 = V_i(1, 1) - V_i(1, 0) - V_i(0, 1)$. This example extends to the following proposition:

Proposition 28. *Any valuation function $V_i(z_1, ..., z_m)$, where z_i is an indicator function associated to data set i, can be written as an n degree polynomial in the z_i's where any term with z_i^p for $p > 1$ has coefficient 0.*

Proof. We prove this by induction on the number of data sets m. If $m = 1$ we easily see this is true. Now suppose this holds for $m < k$. When $m = k$, consider such functions $g_S(z_{S_1}, ..., z_{S_{m-1}})$ on all subsets of m of size $m - 1$. If we set the coefficient α corresponding to the term $z_1 z_2 ... z_n$ we can calculate $\alpha = V_i(z_1, ..., z_m) - \sum g_S(z_{S_1}, ..., z_{S_{m-1}})$. Then, combining this with the previously calculated functions (which by construction agree on intersecting terms) we obtain the desired polynomial. $\square$

This gives us another way of describing the valuation functions. Suppose also that the data has some regularities in that learning is submodular. It is reasonable to assume then that taking the above polynomial and getting rid of all terms higher than degree two should reasonably approximate the value firm i has for different sets. This is good since it still captures some of the complementaries and substitution effects of data learning while being reasonably computable. In essence then, our system has to calculate only the individual value of the data sets to the firms and then some pairwise values in order to write down an estimate $\tilde{V}_i$ of the true valuation.

This brings us back to incentive compatibility. Given the above parametrization of the V_i functions, can we design an incentive compatible mechanism that works efficiently? Does this parametrization somehow give the integer program above some structure we may exploit? All of these are intriguing questions.

4.1.2 Iterative Auctions

One method proposed in the combinatorial auction literature to assign an allocations is an iterative auction. There are two main flavors of these methods. The first has bidders assign prices they are willing to pay for different bundles of goods, tentatively creates an allocation, and allows bidders to change their valuation to increase their utility. Here it is the bidders that change how much they value the goods. The other case posts a set of prices p_i for the goods with which bidders can evaluate which bundle is most valuable to them given these prices. They release this optimal bundle to the auctioneer who increases or decreases the price of goods in order to clear the market. This paradigm seems most useful for our purposes and could be an efficient way, particularly given some structure on the valuation functions, of approximating an optimal allocation.

Note that here our goal is to set prices in order to maximize social welfare measured by total learning. There is an intrinsic relationship between the price of a data set and the optimal data bundle demanded by each firm. Assume each firm values data independently of the allocation to others and that there exist functions $V_1, ..., V_n$ describing the utility of each data bundle $S \subseteq \{1, ..., m\}$. In order to formalize the problem, let's consider the following integer program:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \sum_{S \subseteq \{1,...,m\}} V_i(S) y(i,S) \\ \text{s.t.} \quad & \sum_{i=1}^{n} \sum_{S|j \in S} y(i,S) \leq k_j \quad \forall j \in \{1, ..., m\} \\ & \sum_{S \subseteq \{1,...,m\}} y(i,S) \leq 1 \quad \forall i \in \{1, ..., n\} \\ & y(i,S) \in \{0,1\} \quad \forall S, i \end{aligned}$$

This integer program describes the problem of creating a data allocation in order to maximize $\sum_{i=1}^{n} \sum_{S \subseteq \{1,...,m\}} V_i(S) y(i,S)$. Nowhere in this problem are prices mentioned, however, it is well known that the dual of such a problem has dual variables that can be interpreted as prices. There is a problem with this in that the dual of such an integer program will not be an easily solvable optimization in general. To solve this, we introduce the following relaxation of the above integer program:

$$\begin{aligned}
\max \quad & \sum_{i=1}^{n} \sum_{S \subseteq \{1,\dots,m\}} V_i(S) y(i,S) \\
\text{s.t.} \quad & \sum_{i=1}^{n} \sum_{S | j \in S} y(i,S) \leq k_j \quad \forall j \in \{1,\dots,m\} \\
& \sum_{S \subseteq \{1,\dots,m\}} y(i,S) \leq 1 \quad \forall i \in \{1,\dots,n\} \\
& y(i,S) \in [0,1] \quad \forall S, i
\end{aligned}$$

which allows the variables $y(i,S)$ to range between 0 and 1. One possible interpretation of this relaxation is that variables $y(i,S)$ now represent probabilities that firm i will get subset S in some randomly realized allocation. We assume that the expected number of copies of each data set sold is less than or equal to k_j and that total learning, or social welfare is maximized in expectation. Other than that, this relaxation allows us to approximate some optimal solution to the integer program above. This allows us now to consider the following optimization:

$$\begin{aligned}
Z(\lambda) = \max \quad & \sum_{i=1}^{n} \sum_{S \subseteq \{1,\dots,m\}} V_i(S) y(i,S) + \sum_{j=1}^{m} \lambda_j \left(k_j - \sum_{i=1}^{n} \sum_{S | j \in S} y(i,S) \right) \\
\text{s.t.} \quad & \sum_{S \subseteq \{1,\dots,m\}} y(i,S) \leq 1 \quad \forall i \in \{1,\dots,n\} \\
& y(i,S) \in [0,1] \quad \forall S, i
\end{aligned}$$

It is easy to solve this using traditional linear programming techniques in the case that $\lambda = \{\lambda_j\}$ is fixed. It is also useful to rewrite the objective function of this optimization as

$$\sum_{i=1}^{n}\sum_{S\subseteq\{1,\ldots,m\}} V_i(S)y(i,S)+\sum_{j=1}^{m}\lambda_j\left(k_j-\sum_{i=1}^{n}\sum_{S|j\in S} y(i,S)\right)$$
$$=\sum_{j=1}^{m}\lambda_j k_j+\sum_{j=1}^{m}\sum_{i=1}^{n}\sum_{\substack{S\subseteq\{1,\ldots,m\}\\ \text{s.t.}\quad j\in S}}(V_i(S)y(i,S)-\lambda_j y(i,S))$$
$$=\sum_{j=1}^{m}\lambda_j k_j+\sum_{i=1}^{n}\sum_{S\subseteq\{1,\ldots,m\}} y(i,S)\left(V_i(S)-\sum_{j\in S}\lambda_j\right)$$

Now if we interpret the λ_j as prices for the different data sets, the values $V_i(S)-\sum\limits_{j\in S}\lambda_j$ can be interpreted as the profit of firm i from the set S given some set of prices. In this case then, using tools from linear and integer programming, we have that minimizing the above maximization with respect to λ will give the prices associated to the optimal assignment of the $y(i,S)$. Note that given a set of prices λ the firm will want to set $y(S,i)$ to a positive value when the term $V_i(S)-\sum\limits_{j\in S}\lambda_j$ is maximized for all S. In this way, firms choose one (or several) optimal bundles S for each pricing λ, following the discussion earlier in the section. Finally, finding the optimal prices λ simply involves solving $\min\limits_{\lambda\geq 0} Z(\lambda)$ which, since the objective

$$\sum_{j=1}^{m}\lambda_j k_j+\sum_{i=1}^{n}\sum_{S\subseteq\{1,\ldots,m\}} y(i,S)\left(V_i(S)-\sum_{j\in S}\lambda_j\right)$$

is piecewise linear and convex in λ, can be accomplished through the subgradient algorithm. This algorithm states that in order to minimize $Z(\lambda)$, we initialize λ to λ_0, then update $\lambda_t \to \lambda_t+\theta_i(Vy-k)$ where y is the vector of the $y(i,S)$ and V is the $(n2^m)\times m$ matrix with entries in the jth row corresponding to sets S and bidders i that are 0 if $j\notin S$ and is $V_i(S)$ otherwise. k is the vector $[k_1,\ldots,k_m]^T$. This gives us another way to calculate the price of the data sets, however, since the matrix V (and other terms in the optimization) are exponentially large it is still tough from a computational perspective. Additionally, since we're solving the dual to a relaxed integer optimization, the solution will be an upper bound on the actual integer program we would like to solve. However, it gives us a good sense of how valuable different data sets are and in specific cases may give unique optimal prices.

We would like to use the same technique outlined above in a way that utilizes the structure of the approximations for V_i described in the previous section in order to simplify the optimization. Consider the case that the V_i have structure $V_i(z_{i1},\ldots,z_{im})=\sum\limits_{j=1}^{m}\alpha_{ij}z_{ij}$. This models the case in which there are no complementary or substitutive effects between different data sets. Here, we can write the optimization in the form

$$\begin{aligned}
&\max \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{ij} z_{ij} \\
\text{s.t.} \quad &\sum_{i=1}^{n} z_{ij} \leq k_j \quad \forall j \in \{1, ..., m\} \\
&z_{ij} \in \{0,1\} \quad \forall i, j
\end{aligned}$$

This integer program can be transformed in the same way as above to get approximately optimal prices λ_j and is far simpler computationally. However, disregarding the interplay between learning different data in combination loses much of the character of the problem. In order to include this, we could rewrite

$$V_i(z_{i1}, ..., z_{im}) = \sum_{j=1}^{m} \alpha_{ij} z_{ij} + \sum_{j=1}^{m} \sum_{k=j}^{m} \beta_{ijk} z_{ij} z_{ik}$$

where the terms $\beta_{ijk} z_{ij} z_{ik}$ model substitutive or complementary effects of learning both pieces of data. For example, for firm f_i if learning data d_1 and d_2 accomplish roughly the same thing then β_{ijk} should be set to a negative value such that $V_i(1, 1, 0, 0, ..., 0) = \alpha_{i1} + \alpha_{i2} - \beta_{ijk} \approx \max\{\alpha_{i1}, \alpha_{i2}\}$. So the sign and magnitude of β_{ijk} indicates the pairwise substitutive/complementary effects of learning two data sets d_j and d_k. As previously noted, if we assume learning is submodular and hence that a firm won't demand more than a small number of data sets that overlap in information, this structure on V_i should give us an effective way of estimating any such valuation function. With this in place, we can rewrite $V_i(z_{i1}, ..., z_{im})$ as the quadratic form $V_i(z_{i1}, ..., z_{im}) = z_i^T(Q_i)z_i$ where Q_i is the symmetric $m \times m$ matrix with entries $Q_{kj} = \beta_{ijk}/2$ for $k \neq j$ and $Q_{jj} = \alpha_{ij}$. With this, we can rewrite the entire optimization as

$$\begin{aligned}
&\max z^T(Q)z \\
\text{s.t.} \quad &Az \leq k \\
&z_{ij} \in \{0,1\} \quad \forall i, j
\end{aligned}$$

where A is the $(nm) \times m$ matrix whose ith row has $A_{ij} = 1$ for $j = i \bmod n$, k is the vector of data limits $k = [k_1, ..., k_m]^T$, z is the concatenation of the z_is and Q is the block diagonal matrix with blocks Q_i. Note that it isn't overly difficult for a system to empirically calculate the Q_i for each firm by simply "learning" each data set independently and then pairwise.

Definition 29. *In a data valuation problem, call the empirically calculated matrix Q_i which estimates V_i, the quadratic approximation of i's valuation V_i.*

With this, we can state the following theorem:

Theorem 30. *Suppose the matrices Q_i are all strictly diagonally dominant. Then there exists a polynomial time algorithm to approximate optimal prices for data sets $d_1, ..., d_m$.*

Proof of this fact follows from Q, which must positive definite by our assumption, hence, the relaxed quadratic programming problem (and thus its dual) are solvable in polynomial time. This relates to our previous notion of ϵ separable data in that the condition of the theorem essentially means that $V_i(d_1) + V_i(d_2) \approx V_i(d_1, d_2)$ which was what our previous separability notion was trying to formalize. It is an open question how large the gap is between the integral and relaxed solutions of the primal problem, but this process still gives us a way to estimate good prices for data.

Even without the assumptions of this theorem, we could try to use gradient descent to find locally optimal λ. We just don't have as many convergence guarantees.

Using the system above, we are able to efficiently get an estimation of the value of each data set by solving the dual optimization problem. This is nice in that it gives a clear way to value data sets in general, but it does not necessary provide a method which we may use to allocate data. One simple way to do this would be to first, given information from firms and vendors, calculate the prices $\{p_j\}$ using the quadratic approximation technique. Then, given these prices, the system decides which firms get allocated which data by solving the optimization $S_i = \underset{S \in [m]}{\operatorname{argmax}} \left(V_i(S) - \sum_{j \in S} p_j \right)$. In this way, the system allocates data to firms that maximize their net utility. Such an allocation, called a take it or leave it mechanism, will be studied in more detail in the next chapter.

Firm Competition

In the above formulation of the data valuation problem, it was assumed that there exist some exogenous limits on data reproduction. It was also assumed that firms utilities are independent of one another. In reality, this is an assumption we would like to relax to analyze the situation in which one firm's learning impacts another firms utility. One simple way to do this would be to associate a row vector $a_i =< a_{i1}, ..., a_{in} >$ with each firm f_i such that their utility, instead of being simply $V_i(S_i)$ now becomes $\sum_{j=1}^{n} a_{ij} V_j(S_j)$. Now, the entries of a_i encode how much each firm cares about the success of others and this modification makes the game something closer to a zero sum game. Now, note that in the case of no competition, if there are no limits on data reproduction, the socially optimal allocation is to give all data to every firm. In the case there is competition, the integer program becomes

$$\begin{aligned} &\max \sum_{j=1}^{n} \sum_{i=1}^{n} a_{ij} \sum_{S \subseteq [m]} y(j,S) V_j(S) \\ \text{s.t.} \quad &\sum_{S \subseteq [m]} y(j,S) = 1 \quad \forall j \in \{1, ..., n\} \\ &y(j,S) \in \{0,1\} \quad \forall j \in \{1, ..., n\}, S \subseteq [m] \end{aligned}$$

Note that the above problem does not include any constraints on the number of times vendors reproduce data, however, given some structure on the vectors a_i, it is no longer optimal to give all data to everyone.

To see this, consider the case that $a_1 =< \alpha, -\beta, -\beta, ..., -\beta >$ where α, β are very large. With α, β large enough, we see that it becomes socially optimal for vendors to simply replicate data once and give it all to firm f_1. Hence, competition, even in this simple formulation, can endogenously limit reproduction of data.

In this chapter, we described a way to model data valuation as an integer program, and subsequent methods with which to analyze said program. These approximations are shown in many cases to be computationally cheap and in simulations perform well.

Chapter 5

Price Driven Mechanisms

5.1 Greedy Truthful Mechanisms

In the previous chapters, the price of data was determined endogenously to incentivize firms to report truthfully and to maximize social welfare. This paradigm focuses on the firms assuming that vendors have very few degrees of freedom. In addition to these results we would like to understand the dynamics from the vendors perspective as well. The vendors' only degree of freedom is the price they offer their data for. In general, prices are dynamic values updated to optimize profit, not necessarily to incentivize truthfulness, although this can sometimes happen as in the case of the VCG mechanism. Motivated by these considerations, in this chapter we consider other price centric methods of assigning data allocations.

In [48] and [49], the authors describe an iterative pricing method used in combinatorial auctions to assign allocations. In their case, they consider an vector of prices $p = \{p_j\}$ provided exogenously to the system with which the bidders must then decide which bundle of goods optimizes their welfare. Any transaction allocates data sets $S_i \subseteq [m]$ to firm f_i. Assume that the marginal value of any dataset is nonnegative.

Definition 31. *These sets $\{S_1, ..., S_n\}$ are called feasible if, given replication constraints $k_1, ..., k_m$ for each of the data sets, the sets satisfy the constraint that for all $j \in [m]$, $\sum\limits_{i|j\in S_i} 1 = k_j$.*

Note that we could have used the constraint $\sum\limits_{i|j\in S_i} 1 \leq k_j$, however, in our case, with nondecreasing valuations, any socially optimal solution must use all data possible, otherwise, more social welfare could be obtained.

Definition 32. *In a data valuation game, with n firms and m data sets, given prices $p = \{p_j\}$ for each piece of data, firms can calculate $S_i^* = \underset{S\subseteq[m]}{\text{argmax}}\, V_i(S) - \sum\limits_{j\in S} p_j$. If the sets S_i are* *feasible, the prices p and the sets S_i are called a Walrasian equilibrium if $S_i = S_i^*$. If some data set d_k is unallocated, set $p_k = 0$.*

Suppose that the system is trying to calculate the best S_i to increase total learning. Define $OPT(V_i, p, k) = \max_{S_i \text{ feasible}} \sum_{i=1}^{n} V_i(S_i)$. Suppose that social welfare is measured as total learning $\sum_{i=1}^{n} V_i(S_i)$.

Theorem 33. *In a data valuation game any Walrasian equilibrium yields a socially optimal allocation S_i.*

Proof. Consider some set of prices p and corresponding allocations $\{S_i^*\}$. Suppose that $\bar{S}_i$ is the optimal feasible allocation given by $OPT(V_i, p, k)$. For each $i \in [n]$ we have that $V_i(S_i^*) - \sum_{j \in S_i^*} p_j \geq V_i(\bar{S}_i) - \sum_{j \in \bar{S}_i} p_j$. Putting this together for all i gives

$$\begin{aligned}
&\sum_{i=1}^{n} V_i(S_i^*) - \sum_{j \in S_i^*} p_j \geq \sum_{i=1}^{n} V_i(\bar{S}_i) - \sum_{j \in \bar{S}_i} p_j \\
&\sum_{i=1}^{n} V_i(S_i^*) - \sum_{j \in S_i^*} p_j \geq OPT(V_i, p, k) - \sum_{i=1}^{n} \sum_{j \in \bar{S}_i} p_j \\
&\sum_{i=1}^{n} V_i(S_i^*) - \sum_{j=1}^{m} k_j p_j \geq OPT(V_i, p, k) - \sum_{j=1}^{m} k_j p_j \\
&\sum_{i=1}^{n} V_i(S_i^*) \geq OPT(V_i, p, k)
\end{aligned}$$

So $\{S_i^*\}$ must be a socially optimal allocation. □

The trouble is both guaranteeing that such equilibria exist and finding them. Even calculating a single set S_i^* could take exponential time, and that is just one part of this problem. The payment structure may result in negative utility for the firms and hence, would not be useful in our case. Motivated by this framework, the goal is now to create an truthful mechanism that attempts to efficiently maximize both learning and profit. Some of our earlier work utilized a greedy algorithm which auctioned off data sets one at a time. This works truthfully in the case of separable or ϵ separable data, however, this is a very strong assumption. In more general settings, such a greedy method could be manipulated by the bidders who may lie about their valuation functions V_i. In fact, in this case, the mechanism is highly sensitive to the order that data is auctioned, and in general, it is always possible to construct examples in which bidders manipulate their values for data sold early, in order to optimize their wealth with information sold later.

Instead of iteratively selling the data sets, we might consider greedily allocating datasets one at a time, minimizing the risk firms manipulate the system in the method described above. Consider the following. A shopkeeper has m goods $d_1, ..., d_m$ with prices $p_1, ..., p_m$ and over the course of a day, n customers arrive to buy goods. Each customer buys the bundle of goods that optimize their welfare given the posted prices, and the shopkeeper would like to adjust the prices to account for demand. As each customer comes in and buys some subset of goods $S \subseteq \{d_1, ..., d_m\}$, the shopkeeper takes that as a signal of the demand for each good and adjusts prices $p_j \to p_j + \epsilon$ for $j \in S$ and $p_k \to p_k - \delta$ for $\epsilon, \delta > 0$. Hence, by the end of the day, goods that were bought many times should have a higher price than those that weren't, reflecting their higher value. The algorithm we will use to greedily allocate data sets will be along a similar vein as this.

Consider a data valuation game where the data sets are given initial prices $p(0) = \{p_j\}$, and suppose that the optimization problem $S_i^* = \underset{S \subseteq [m]}{\operatorname{argmax}} V_i(z_{iS}) - \sum_{j \in S} p_j$ can be calculated efficiently for each firm. Note that in general this is a strong assumption and will be relaxed in later discussion. With this, we randomly permute the n firms f_i, initialize prices, and set demand increments $\delta, \epsilon > 0$. Imagine in this case that there are no reproduction limits k_j and that data vendors would like to extract as much profit as possible from valuable data. Now, consider the first firm f_1 selected from our random ordering. Given prices, $p(0)$ this firm can calculate $S_1^* = \underset{S \subseteq [m]}{\operatorname{argmax}} V_1(S_1)) - \sum_{j \in S} p_j$, and is allocated the set S_1^*, while paying p_j for each data set d_j such that $j \in S_1^*$. Then we may update the prices $p(0) \to p(1)$ such that if $j \in S_1^*$, $p_j \to p_j + \epsilon$ and if $j \notin S_1^*$, $p_j \to p_j - \delta$. We can do this for each data set in the ordering to obtain an allocation S_i^* for each f_i and corresponding payments to the vendors. Note that, as mentioned above, the more valuable data sets will be bought more often, yielding both higher prices and higher purchase rates and hence at the end of the process will attain the most revenue. Thus the gross revenue of each vendor at the end of the algorithm gives an estimate of the relative value of each data set. This process is summarized by the following:

Theorem 34. *In a data valuation game the Iterative Allocation Mechanism defined above is truthful.*

Proof. To show that this is truthful, suppose a firm provides a valuation function $\tilde{V}_i$ instead of their true valuation V_i. In this case, they will be allocated $\tilde{S}_i$ according to $\tilde{S}_i = \underset{S \subseteq [m]}{\operatorname{argmax}} \tilde{V}_i(z_{iS}) - \sum_{j \in S} p_j$ compared to the set $S_i = \underset{S \subseteq [m]}{\operatorname{argmax}} V_i(z_{iS}) - \sum_{j \in S} p_j$ they are allocated when honest. However, by construction

$$V_i(z_{iS_i}) - \sum_{j \in S_i} p_j \geq V_i(z_{i\tilde{S}_i}) - \sum_{j \in \tilde{S}_i} p_j$$

Algorithm 4 Iterative Allocation Mechanism

```
Input: $V_1, ..., V_n$: valuation functions; $d_1, ..., d_m$: datasets; $\epsilon, \delta > 0$ increments; $p_1, ..., p_m$, initial prices.
1: Randomly permute the set $\{1, ..., n\}$.
2: for $i$ from 1 to $n$ do
3:     Calculate $S_i^* = \underset{S\subseteq[m]}{\text{argmax}}\, V_i(S) - \sum_{j\in S} p_j$
4:     Allocate $S_i^*$ to firm $f_i$
5:     Update prices
6:     $\forall j \in S_i^*$, $p_j \to p_j + \epsilon$
7:     $\forall j \notin S_i^*$, $p_j \to p_j - \delta$.
8: Return: Allocation $\{S_i\}$, prices $p_j(i)$ where $p_j(i)$ is the price paid for $j \in S_i$ by firm $i$.
```

and hence, truthfulness is a dominant strategy. □

Thus, firms who participate in such a system have no incentive to lie about their valuations of any data. Assuming that the maximization $\max_{S\subseteq[m]} V_i(z_{iS}) - \sum_{j\in S} p_j$ can be quickly calculated, this gives us a truthful, efficient way to allocate data among firms for prices that reflect demand. This is already good and yields several other related results. If we initialize prices uniformly, the prices at the end of the process should give an estimate of value for individual data.

Corollary 35. *Consider a data valuation game using the iterative allocation mechanism to assign allocations using uniform initial price $p(0) = 0_n$. Assume that the marginal value of data set d_1 is always greater than C_1 and that the marginal value of every other data set is less than C_2 with $C_2 < C_1$. Finally, suppose $\epsilon = \delta$. In this case, the final price $p(n)$ has the property that $p_1 \geq p_k$ for all $k \neq 1$.*

Proof. We can show this fact by induction on the number of firms. In the case $n = 1$ we see that $p_1 = p_k = \epsilon$ for all $k \in [m]$. Assume this holds for $n < k$ and consider the case $n = k$. By our induction hypothesis we see that $p(k-1)$ has $p_1 \geq p_k$ for all $k \neq 1$. In this case there are two possibilities. If at time $k-1$ we have $p_1 > p_k$ for some $k \neq 1$, then even if d_k is allocated and d_1 isn't we have that $p_1 \geq p_k$ at time k. On the other hand, if at time $k-1$ $p_1 = p_k$ for some $k \neq 1$, then d_k will never be allocated when d_1. To see this, suppose $p_1 = p_k$ and d_k is allocated and d_1 is not. Then if $\tilde{S} = \underset{S\subseteq[m]}{\text{argmax}}\, V_{k-1}(z_{k-1S}) - \sum_{j\in S} p_j$, such that

$1 \notin \tilde{S}$ we have that

$$\begin{aligned}
&V_{k-1}(z_{k-1\tilde{S}}) - \sum_{j\in\tilde{S}} p_j \\
&\leq C_2 - C_1 + V_{k-1}(z_{k-1\tilde{S}\setminus\{k\}\cup\{1\}}) - \sum_{j\in\tilde{S}\setminus\{k\}\cup\{1\}} p_j \\
&< V_{k-1}(z_{k-1\tilde{S}\setminus\{k\}\cup\{1\}}) - \sum_{j\in\tilde{S}\setminus\{k\}\cup\{1\}} p_j
\end{aligned}$$

which violates the maximality of $\tilde{S}$ and therefore if d_k is allocated, d_1 must be allocated as well. However in this case at time k, $p_1 \geq p_k$ as well, finishing the proof. □

Note that the same result holds if $\delta = 0$. Additionally, as was the case of our previous greedy algorithm on separable data, we have that there is a relationship between the resulting allocation and socially optimal allocations.

Corollary 36. *Consider a data valuation game using the iterative allocation mechanism to assign allocations with initial price $p(0)$ and suppose $\epsilon = \delta = 0$. In this case, if we consider the resulting allocation $\{S_i\}$ of data sets to firms and define k_j to be the number of times d_j is allocated, we have that $\{S_i\}$ maximizes social welfare under replication restrictions $\sum_{i|j\in S_i} 1 = k_j$ for all $j \in [m]$.*

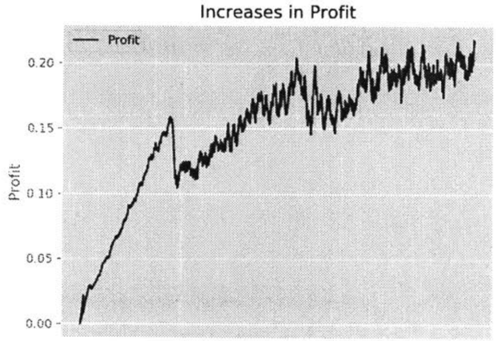
Proof. Proof of this follows from the fact that the prices remain fixed and each firm is allocated their optimal set S_i under these prices, hence $p(0)$ and S_i form a Walrasian equilibrium. □

Simulations show that with a large number of firms, over time prices settle down into equilibrium. Additionally, the total profit passing through the system increases steadily. This is seen in Figure 5-1.

Simulations using general valuation functions also exhibit similar profit increases using this method. Figure 5-2 was generated by first creating a model as in the end of Chapter 2. Then, valuation functions V_i were calculated and the iterative allocation mechanism was implemented. The graph clearly shows that over time, prices adjust in order to maximize profit.

Even with all of these results, calculating the maximization problem

$$S_i^* = \underset{S\subseteq[m]}{\operatorname{argmax}}\, V_i(z_{iS}) - \sum_{j\in S} p_j$$



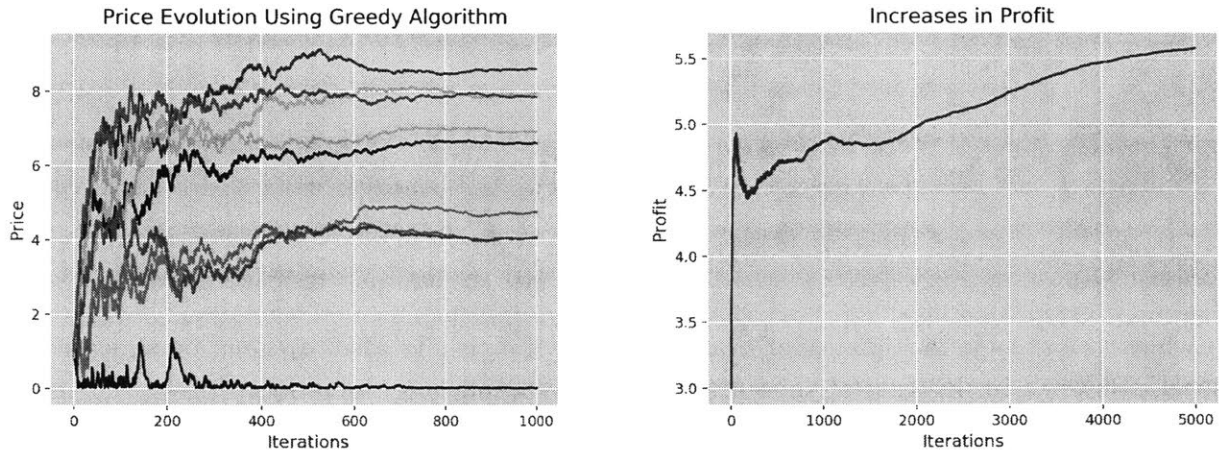


Figure 5-1: Prices and Profit Over Time

is generally hard. We would like to introduce a framework to simplify this component of the algorithm. With this in mind, note that in Walrasian equilibria, it was assumed that data vendors artificially limit the supply of all data sets by raising prices in order to increase their revenue.

If a price p_j were so low that each firm is allocated d_j, the vendor might want to raise the price to increase profit possibly reducing the number of times it is sold. Here prices and supply are inversely correlated. Additionally it can be shown that using a VCG mechanism if $k_j = n$ for all $k \in [m]$ each person be allocated all data and would pay nothing. This is clearly suboptimal for vendors. Thus it is in their interest to limit access to their data. In the previous cases this limitation was provided by the number of times a data set was replicated, however, it could equally have been done by only partially allocating data to firms. In other words if the data set d_j consists of thousands of labeled points and firm f_i is allocated $z_{ij} = \frac{3}{5}$, they would be randomly assigned 60 percent of the data in d_j. This effectively smooths out the function V_i in such a way that the optimization $S_i^* = \underset{S \subseteq [m]}{\operatorname{argmax}} V_i(z_{iS}) - \sum_{j \in S} p_j$ becomes more tractable.

Definition 37. *A data valuation game with fractional allocations is a data valuation game in which an allocation randomly assigns a given percentage of a data set d_j to a firm f_i.*

In these games it is reasonable to assume that due to the randomness in the assignment of data, the functions $V_i(z_{i1}, ..., z_{im})$, where $z_{ij} \in [0,1]$, are differentiable. Furthermore, due to the fact that additional data gives diminishing accuracy returns, we may assume that the V_i's are strictly concave. With these two assumptions in place, instead of the hard combinatorial maximization problem we had for firms before, we can now calculate, given a set of prices $p = \{p_j\}$, the maximization $\max_{z_i \in [0,1]^m} V_i(z_{i1}, ..., z_{im}) - \sum_{j=1}^{m} z_{ij}p_j$. By assumption the objective $V_i(z_{i1}, ..., z_{im}) - \sum_{j=1}^{m} z_{ij}p_j$ is concave for any p over the compact set $[0,1]^n$ and thus has a unique maximizer. More over, this maximization can be done in a short amount of time using either KKT conditions or something like stochastic gradient ascent. Thus, we can now run our greedy truthful algorithm efficiently. There is one caveat to this and that is when the prices are updated, they should take into account the amount of the data set that was allocated. In other words, we set $\delta = 0$ and for all prices, $p_j \to p_j + z_{ij}\epsilon$ after data is allocated to firm f_i.

Algorithm 5 Efficient Iterative Allocation

```
Input: $V_1, ..., V_n$: differentiable strictly concave valuation functions; $d_1, ..., d_m$: datasets; $\epsilon > 0$ increment; $p_1, ..., p_m$, initial prices.
1: Randomly permute the set $\{1, ..., n\}$ via $\sigma$.
2: for $i$ from 1 to $n$ do
3:     Calculate $z^*_{\sigma(i)} = \operatorname{argmax}_{z_{\sigma(i)} \in [0,1]^m} V_i(z_{\sigma(i)1}, ..., z_{\sigma(i)m}) - \sum_{j=1}^{m} z_{\sigma(i)j}p_j$
4:     Allocate $z^*_{\sigma(i)}$ to firm $f_{\sigma(i)}$ and update prices $p_j \to p_j + z^*_{\sigma(i)j}\epsilon$
5: Return: Allocation $\{z_i\}$, prices $p_j(i)$ where $p_j(i)$ is the price paid for the percentage $z_{ij}$ of data $d_j$ by firm $i$.
```

Note that the above algorithm, due to the randomized sampling, is only truthful in expectation, however, this is a small price to pay for the huge increase in efficiency.

5.2 Competitive Pricing

The problem of evaluating the proper price for data and information is difficult on many levels. One reason for this difficulty is that it is difficult for firms and vendors alike to predict the outcome of processing data and information without having access to it. Without buying the information, one cannot know how much it's worth. It would be akin to someone buying property blindfolded, knowing only after the transaction occurs how much the home is worth. However, it might be possible to have a third party do such a valuation (which is what the above work is concerned with.) Also, one might be able to evaluate the value of a data set by examining a proxy dataset that has similar properties. In this case, the notion of differential privacy would become highly invaluable. Another reason why valuing data is so difficult has to do with the combinatorial complexity of the problem. Even if firms and vendors knew exactly how much obtaining data would help them, it is unclear how competition between vendors affects prices. This section intends to focus on precisely this problem. In other words, assuming firms know exactly how they would benefit from information, what should prices be set to in equilibrium?

One interesting property of data is its low marginal cost for reproduction. In this respect, it is similar to other electronic goods such as software. There is some economic literature dealing with how to price such goods [50] but no current research considers competition between different vendors. One could imagine selling a single data set d_1 to n firms $f_1, ..., f_n$ who each have value $V_1 > V_2 > ... > V_n$ for that data. If the data set's price is set to $p \in \mathbb{R}_{\geq 0}$ then any firm f_i with $V_i > p$ will buy the data, yielding a profit of $\sum\limits_{i|V_i > p} p$. In this case, the optimization problem on the side of the vendor selling the data becomes $\max\limits_{p \in \mathbb{R}} \sum\limits_{i|V_i > p} p$ and solving for the optimal $p^* \in \mathbb{R}$ yields the correct price for the data set d_1. Specifying such a p^* automatically results in an allocation in which all f_i with $V_i > p$ receive the data.

The story gets more complicated when more data sets are considered. Consider the same example as above, however with the addition of another data set d_2. Here, we suppose that each firm has value $V_i(\kappa) \in \mathbb{R}_{\geq 0}$ for different combinations of the two data sets. V_i takes the subset $\kappa \subseteq \{1, 2\}$ that f_i is allocated. Now, suppose that there exist prices $p = [p_1, p_2]$ for the two data sets. Each firm, in order to maximize utility, will optimize the following expression $\max\limits_{\kappa \subseteq \{1,2\}} V_i(\kappa) - \sum\limits_{j \in \kappa} p_j$ in terms of κ given the prices p_1, p_2. Knowing this will be the behavior of the firms and leveraging the fact that the values $V_i(\kappa)$ remain fixed allows the vendors of the two data sets to adjust their prices accordingly. In particular, denote the optimal subset of $\{1, 2\}$ chosen by firm f_i given value function V_i and prices $p = [p_1, p_2]$ as $\kappa(V_i, p_1, p_2)$. Then note that each vendor of the data sets would like to optimize $\max\limits_{p_j \in \mathbb{R}} \sum i | j \in \kappa(V_i, p_j, p_{-j}) p_j$. Then we define a price of data sets d_1 and d_2 to be a Nash equilibrium $p \in \mathbb{R}^2$. We may, if we would like, alter the above optimization to include a lower bound on possible prices offered, representing the cost of data production and processing. Regardless, even in the case with two data sets, it is clear this problem is becoming more and more computationally challenging.

We may now state the problem in full generality. Suppose we have n firms and m vendors

Definition 38. *Define a* **competitive pricing game** *as a game such that there are m vendors of data sets $d_1, ..., d_m$ who may set prices $p_1, ..., p_m$ for this information. Additionally, there exist firms with valuation functions $V_i(\kappa) \in \mathbb{R}_{\geq 0}$ that dictate how much utility firms obtain when they are allocated some subset $\kappa \subseteq (\{1, ..., m\})$ of the data. With* *this, given actions* $p = [p_1, ..., p_m]$*, vendor* j *obtains utility* $\sum_{i | j \in \kappa(V_i, p_j, p_{-j})} p_j$ *where* $\kappa(V_i, p) = \operatorname*{argmax}_{\kappa \subseteq \{1,...,m\}} V_i(\kappa) - \sum_{j \in \kappa} p_j$.

Note that the only agents in this game are the vendors who determine the price of their data set. The firms have fixed responses given fixed prices and are hence non strategic. This leads us to our definition for competitive pricing.

Definition 39. *Given an competitive pricing game, the* **competitive prices** *for data* $d_1, ..., d_m$ *are the actions of the data vendors at equilibrium.*

In other words, data should be priced in such a way that no individual vendor selling a data set has the incentive to change their price. If such an equilibrium can be found, it should give an idea of the value of particular pieces of information in the market structure. However, converging to such an equilibrium is not trivial. There is no clear algorithm to perform the optimization necessary, and worse, there is the possibility of reaching multiple equilibria. From the brief discussion above, this may not be a problem since the value of one good might be intrinsically linked with the price of another good. For example if d_1 is only really useful with d_2 and d_2's price p_2 is incredibly high, then the d_1 will not be as valuable. This game models the dynamics from the vendor's perspective and gives a sense of how complementary and substitutive effects among different data sets influence competition and can therefore impact prices. To see how such a system might work in a simple case consider the following.

Example 40. *Suppose we have separable data* $d_1, ..., d_m$ *and firms* $f_1, ..., f_n$ *with valuations* $V_i(d_j)$ *for each firm* f_i *and data set* d_j*. In this case, given a set of prices* $p = \{p_i\}$*, the optimization performed by the firms buying some bundle of data becomes*

$$\begin{aligned}\kappa(V_i, p) &= \operatorname*{argmax}_{\kappa \subseteq \{1,...,m\}} V_i(\kappa) - \sum_{j \in \kappa} p_j \\ &= \operatorname*{argmax}_{\kappa \subseteq \{1,...,m\}} \sum_{j \in \kappa} V_i(d_j) - p_j \\ &= \{j \in \{1, ..., m\} \ s.t. \ V_i(d_j) - p_j > 0\}.\end{aligned}$$

In this case, $\kappa(V_i, p)$ *is quite easy to calculate and additionally, optimizing prices becomes an easy task as well since for each vendor, given a set of prices* $p = \{p_i\}$*, their optimization becomes*

$$\max_{p_j \geq 0} \sum_{i | j \in \kappa(V_i, p_j, p_{-j})} p_j$$
$$= \max_{p_j \geq 0} \sum_{i | V_i(d_j) - p_j > 0} p_j$$

and the optimizations of the different vendors become totally decoupled. Hence, once each vendor calculates $p_j^* = \operatorname{argmax}_{p \in \mathbb{R}} \sum_{i | V_i(d_j) - p > 0} p$, an optimal Nash equilibria is found with competitive prices given by $p^* = \{p_j^*\}$. Note that this is similar to the greedy algorithm used earlier in this chapter as it simply optimizes a value for a single data set at once.

Note that in this example, it only took one pass through the different data sets, to price them accurately. In the case of non separable data, once the first set of prices is determined the vendors might continue to individually adjust their prices. This individual adjustment takes a particular form. Given prices $p = \{p_j\}$, vendor v_j has the choice of changing p_j, all else being equal. Here, as p_j ranges, we see that each firm f_i buys d_j once p_j crosses some threshold $t_{ij}(p_{-j})$. This threshold $t_{ij}(p_{-j})$ can be calculated as the p_j that satisfies

$$\operatorname*{argmax}_{\substack{\kappa \subseteq \{1,\ldots,m\} \\ \text{s.t. } j \in \kappa}} V_i(\kappa) - \sum_{k \in \kappa} p_k$$
$$= \operatorname*{argmax}_{\substack{\kappa \subseteq \{1,\ldots,m\} \\ \text{s.t. } j \notin \kappa}} V_i(\kappa) - \sum_{k \in \kappa} p_k.$$

Hence, given prices $p = \{p_j\}$, each vendor can individually maximize p_j by taking $p_j^* = \operatorname{argmax}_{p_j \in \mathbb{R}} \sum_{i | p_j \leq t_{ij}(p_{-j})} p_j$.

There is no guarantee that such an algorithm will converge to a set of prices. In essence such a system is just a best response algorithm which is efficient if the t_{ij} values can be calculated or estimated quickly.

In this chapter, we've seen how to construct efficient algorithms to allocate data in a way that maximizes profit. In addition, we have proposed an simulated a model for competition between information sources. The notion of Walrasian equilibria and the structure of these kinds of auctions are a good description for this problem and we think would readily extend to the more complicated setting

Chapter 6

Conclusion

In this thesis, we studied different methods to value data and information. Up to now, there did not exist a systematic way to describe the economics of data or a way to value information. The models, methods and results in this work are a step towards such a system. If those who use information don't have access to a method to value new information, they will not be able to fully realize their potential to learn and adapt. By the same token, if those selling their data cannot price it, they will not be fairly compensated. We have seen that by putting reasonable structures on how firms learn using data sets, we can calculate optimal fair allocations of information, revealing the value of different sources of data. Such allocations may be constructed such that each participant has no incentive to lie and every incentive to participate. This has clear implications in several different domains including finance, insurance, and retail where private information is at a premium and has the potential to significantly increase profits. Our system is superior to those that exist currently which rely upon blindly choosing collections of data to buy with the hope of improving predictions and decision making in the future.

In the modern world, sometimes it feels like big brother is always watching. The main difference between Orwellian notion of a surveillance state and our current climate is that the entities collecting our information are firms like Google, Facebook and Amazon who use it for monetary gain. It stands to reason that for their gain we should be compensated for the loss of our individual privacy. Such compensation, as described in the results of this research, would both encourage those with sensitive information to share it, and dissuade firms from collecting too much information and infringing unnecessarily on our privacy. In this way, we think that this research has policy implications for governments who would like to preserve privacy while maintaining economic innovation.

6.1 Directions for Future Work

While the results in this thesis represent a huge leap forward in understanding the economics of data and information, there are still some holes in our current knowledge. It would be useful to understand how competition between firms impacts data valuation. In particular, if different firms would like to monopolize information or simply limit the information gained by competitors, the problem now isn't simply about their learning, but the learning of others. A general model for this interaction is elusive, however, in a simple case one could imagine that firms get utility based on their own learning and are penalized linearly for the learning of the other firms. Mathematically, if each firm's learning is described by the functions $V_i(S_i)$ for $i \in [n]$, the total utility of firm i can be written as $U_i(S_1, ..., S_n) = V_i(S_i) + \sum_{j \neq i} \alpha_{ij} V_j(S_j)$. The α_{ij}'s could be thought of as representing the amount firm f_i cares about the success of firm f_j. In this way, the model for utility is similar to a weighted zero sum game. One question becomes, if there is some structure on the functions $V_i(S_i)$ and the matrix $A = \{\alpha_{ij}\}$, is it possible to extend our previous analysis. Note that this is a simple case of competition between firms and understanding the most general case is an even tougher challenge. It would also be interesting to analyze the case in which there was some kind of feedback between the firm's learning and the vendors. In this case, we might say that if by sharing data, you expose yourself to greater cost in the future, you as a data vendor should be compensated commensurately. It would also be useful to think about dynamics of data valuation over time. In reality, goals and priors of firms are changing constantly, therefore the value of information must change accordingly. Finally, one aspect of the data valuation problem glossed over in this work was the actual algorithm used to incorporate new information with existing priors. Generally it was assumed that the firm came in with this algorithm however, perhaps they simply come in with the goal of learning and the system we have described not only allocates them data, but recommends the best algorithm they can use to learn.

Bibliography

[1] G. W. Brier, "Verification of forecasts expressed in terms of probability," *Monthey Weather Review*, vol. 78, no. 1, pp. 1–3, 1950.

[2] A. P. Dawid, "Present position and potential developments: Some personal views: Statistical theory: The prequential approach," *Journal of the Royal Statistical Society. Series A (General)*, pp. 278–292, 1984.

[3] J. Besag, P. Green, D. Higdon, and K. Mengersen, "Bayesian computation and stochastic systems," *Statistical science*, pp. 3–41, 1995.

[4] I. J. Good, "Rational decisions," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 107–114, 1952.

[5] C. Genest and J. V. Zidek, "Combining probability distributions: A critique and an annotated bibliography," *Statistical Science*, pp. 114–135, 1986.

[6] T. Fissler, J. F. Ziegel, *et al.*, "Higher order elicitability and osband?s principle," *The Annals of Statistics*, vol. 44, no. 4, pp. 1680–1707, 2016.

[7] I. Steinwart, C. Pasin, R. Williamson, and S. Zhang, "Elicitation and identification of properties," in *Conference on Learning Theory*, pp. 482–526, 2014.

[8] R. Frongillo and I. A. Kash, "Vector-valued property elicitation," in *Conference on Learning Theory*, pp. 710–727, 2015.

[9] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Machine learning*, vol. 15, no. 2, pp. 201–221, 1994.

[10] W. Chu, M. Zinkevich, L. Li, A. Thomas, and B. Tseng, "Unbiased online active learning in data streams," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 195–203, ACM, 2011.

[11] A. Beygelzimer, S. Dasgupta, and J. Langford, "Importance weighted active learning," in *Proceedings of the 26th annual international conference on machine learning*, pp. 49–56, ACM, 2009.

[12] B. Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, no. 55-66, p. 11, 2010.

[13] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 3–12, Springer-Verlag New York, Inc., 1994.

[14] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of machine learning research*, vol. 2, no. Nov, pp. 45–66, 2001.

[15] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proceedings of the twenty-first international conference on Machine learning*, p. 79, ACM, 2004.

[16] P. Donmez, J. Carbonell, and P. Bennett, "Dual strategy active learning," *Machine Learning: ECML 2007*, pp. 116–127, 2007.

[17] P. Donmez and J. G. Carbonell, "Proactive learning: cost-sensitive active learning with multiple imperfect oracles," in *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 619–628, ACM, 2008.

[18] S. Dasgupta and D. Hsu, "Hierarchical sampling for active learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 208–215, ACM, 2008.

[19] J. O. Berger, *Statistical decision theory and Bayesian analysis.* Springer Science & Business Media, 2013.

[20] D. Fudenberg and J. Tirole, "Game theory, 1991," *Cambridge, Massachusetts*, vol. 393, p. 12, 1991.

[21] V. Krishna, *Auction theory.* Academic press, 2009.

[22] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory.* SIAM, 1998.

[23] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*, vol. 1. Cambridge University Press Cambridge, 2007.

[24] N. Karlsson, "Control problems in online advertising and benefits of randomized bidding strategies," *European Journal of Control*, vol. 30, pp. 31–49, 2016.

[25] N. Karlsson and J. Zhang, "Applications of feedback control in online advertising," in *American Control Conference (ACC), 2013*, pp. 6008–6013, IEEE, 2013.

[26] R. Agrawal, M. Hedge, and D. Teneketzis, "Asymptotically efficient adaptive allocation rules for the multiarmed bandit problem with switching cost," *IEEE Transactions on Automatic Control*, vol. 33, no. 10, pp. 899–906, 1988.

[27] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[28] R. D. Kleinberg, "Nearly tight bounds for the continuum-armed bandit problem," in *Advances in Neural Information Processing Systems*, pp. 697–704, 2005.

[29] B. Awerbuch and R. D. Kleinberg, "Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pp. 45–53, ACM, 2004.

[30] S. Pandey, D. Chakrabarti, and D. Agarwal, "Multi-armed bandit problems with dependent arms," in *Proceedings of the 24th international conference on Machine learning*, pp. 721–728, ACM, 2007.

[31] M. Babaioff, Y. Sharma, and A. Slivkins, "Characterizing truthful multi-armed bandit mechanisms," in *Proceedings of the 10th ACM conference on Electronic commerce*, pp. 79–88, ACM, 2009.

[32] J.-Y. Audibert and S. Bubeck, "Best arm identification in multi-armed bandits," in *COLT-23th Conference on Learning Theory-2010*, pp. 13–p, 2010.

[33] S. Bubeck, R. Munos, and G. Stoltz, "Pure exploration in multi-armed bandits problems," in *International conference on Algorithmic learning theory*, pp. 23–37, Springer, 2009.

[34] S. Guha, K. Munagala, and P. Shi, "Approximation algorithms for restless bandit problems," *Journal of the ACM (JACM)*, vol. 58, no. 1, p. 3, 2010.

[35] L. Tran-Thanh, A. Chapman, J. E. Munoz De Cote Flores Luna, A. Rogers, and N. R. Jennings, "Epsilon–first policies for budget–limited multi-armed bandits," 2010.

[36] M. Feldman, A. Kalai, and M. Tennenholtz, "Playing games without observing payoffs.," in *ICS*, pp. 106–110, 2010.

[37] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[38] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Muller, E. Sackinger, P. Simard, *et al.*, "Comparison of classifier methods: a case study in handwritten digit recognition," in *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on*, vol. 2, pp. 77–82, IEEE, 1994.

[39] T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to spam filtering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10206–10222, 2009.

[40] R. Price, "Microsoft is deleting its ai chatbot?s incredibly racist tweets," *Business Insider*, 2016.

[41] J. Novembre, T. Johnson, K. Bryc, Z. Kutalik, A. R. Boyko, A. Auton, A. Indap, K. S. King, S. Bergmann, M. R. Nelson, *et al.*, "Genes mirror geography within europe," *Nature*, vol. 456, no. 7218, p. 98, 2008.

[42] N. Nisan and A. Ronen, "Algorithmic mechanism design," in *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pp. 129–140, ACM, 1999.

[43] S. J. Rassenti, V. L. Smith, and R. L. Bulfin, "A combinatorial auction mechanism for airport time slot allocation," *The Bell Journal of Economics*, pp. 402–417, 1982.

[44] C. G. Caplice, *An optimization based bidding process: A new framework for shipper-carrier relationships.* PhD thesis, Massachusetts Institute of Technology, 1996.

[45] N. Nisan, "Bidding and allocation in combinatorial auctions," in *Proceedings of the 2nd ACM conference on Electronic commerce*, pp. 1–12, ACM, 2000.

[46] T. Sandholm, S. Suri, A. Gilpin, and D. Levine, "Cabob: A fast optimal algorithm for combinatorial auctions," in *International Joint Conference on Artificial Intelligence*, vol. 17, pp. 1102–1108, LAWRENCE ERLBAUM ASSOCIATES LTD, 2001.

[47] M. H. Rothkopf, A. Pekeč, and R. M. Harstad, "Computationally manageable combinational auctions," *Management science*, vol. 44, no. 8, pp. 1131–1147, 1998.

[48] A. Roth, "Algorithmic game theory." University Lecture, 2017.

[49] A. Mu'Alem and N. Nisan, "Truthful approximation mechanisms for restricted combinatorial auctions," *Games and Economic Behavior*, vol. 64, no. 2, pp. 612–631, 2008.

[50] H. R. Varian, "Buying, sharing and renting information goods," *The Journal of Industrial Economics*, vol. 48, no. 4, pp. 473–488, 2000.