Designing Incentive Compatible Payment Rules for Combinatorial Auctions with Structural SVMs

A thesis presented

by

Pichayut Jirapinyo

To

Computer Science

in partial fulfillment of the honors requirements

for the degree of

Bachelor of Arts

Harvard College

Cambridge, Massachusetts

March 31, 2011

Abstract

Combinatorial auctions have a wide range of real-world applications; yet, designing combinatorial auction mechanisms that simultaneously possess good economic properties and computational tractability remains a major challenge. An auction mechanism consists of an allocation rule and a payment rule. We propose a new framework that uses Structural SVMs to design a payment rule for any given allocation rule. Besides being tractable, the payment rule produced by an exact classifier is both strategyproof and individually rational. Unlike the VCG payment rule, our framework does not require an optimal allocation rule, an NP-hard problem, in order to obtain a strategyproof mechanism. Our experiments show that the payment rules generated from our framework enjoy a low level of ex post regret and approximate well-known strategyproof payment rules, such as VCG and second price, reasonably well. In addition, applying our framework to an allocation rule with no corresponding strategyproof payment rule does not result in additional performance loss.

Acknowledgments

I thank my thesis advisor David Parkes for being a great advisor and an encouraging mentor. Without his generous support and guidance, this thesis could not have been possible. His insightful suggestions and willingness to guide me through issues large and small have profoundly influenced the direction of this project. I thank John Lai for helpful discussions and help with various concepts, programming, and countless other crucial aspects of the project. I thank Paul Dütting, Felix Fischer, and Ben Lubin for great discussions of the theoretical aspects of the project in our weekly meetings. I thank Andrew Mao for his server. Furthermore, I thank Kun Phalitnonkiat, Edmund Soriano, Yod Wattanaprakornkul, and John Tan for helpful suggestions. Lastly, I thank my family and friends for their encouragement and support.

Contents

Chapter 1

Introduction

> *This problem [of combinatorial auctions] has direct applications, may be viewed as a general abstraction of complex resource allocation, and is the paradigmatic problem on the interface of economics and computer science.*
>
> Blumrosen and Nisan [4]

Combinatorial auctions (CAs) allow multiple items to be auctioned simultaneously and permit participants to express preferences over different subsets of these items. The complex set of possible strategies for a bidder allows for the expression of such intricate interrelations between different items in the auctions as complementarity and substitutability. Many real-world resource allocation problems require that type of expressibility as the value of an item often depends on other items. For instance, in spectrum auctions of US Federal Communications Commission (FCC), licenses for transmitting signals over different wavelengths across multiple geographical areas are being auctioned, and these licenses can be complementary and substitutable. Other applications of combinatorial auctions include airport time slot auctions, and railroad segment auctions [6]. Despite the utility of combinatorial auctions, designing CA mechanisms that are both computationally and economically

desirable remains a major challenge within the field of computational mechanism design.

Combinatorial auction mechanisms involve two steps: first, optimally determining winners of different items ("winner determination problem" or "allocation rule"), and second, deciding how much each winner has to pay for the allocated items ("payment rules"). One hallmark mechanism is the Vickrey-Clarke-Groves (VCG) mechanism, which proposes a payment rule that when coupled with a socially optimal allocation rule, guarantees that bidders truthfully reveal their private information about their preferences. This property, called *incentive compatibility* (IC) or *strategyproofness* (SP), is important within mechanism design, as it allows the auctioneer access to truthful information and reduces coordination problems. Nevertheless, two problems prevent VCG mechanisms from being useful in practice. First, despite its provision of socially optimal and incentive compatible solutions, VCG mechanisms are known to generate poor revenues, and are vulnerable to certain types of manipulation, such as collusion among bidders [1]. A more serious problem with VCG mechanisms is that they require optimal allocations in order that the mechanisms become incentive compatible.

Optimally allocating items to bidders in the combinatorial auction setting is known to be NP-Complete, since the problem can be formulated as an integer programming problem similar to *weighted set packing*, a known NP-Complete problem [23]. One of the possible workarounds is to find an approximately optimal mechanism. Unfortunately, finding a solution with total social welfare within a factor of $m^{\frac{1}{2}-\epsilon}$ of that of an optimal solution, where m is the number of items and $\epsilon > 0$, is known to also be NP-Complete, based on NP-Completeness of the approximation of clique size [8, 4]. Therefore, research has focused on studying special cases such as auctions with Walrasian equilibrium [3] and single-minded CAs [13], and on employing heuristic algorithms [4]. Without optimality, VCG payment rules cannot be used with these algorithms to generate incentive compatible mechanisms.

In this thesis, we propose a new framework that, given any allocation rule, uses machine learning to generate a corresponding incentive compatible payment rule for that particular

allocation rule. The allocation rule is used to generate training instances, from which *Structural Support Vector Machines* (Structural SVMs) [25, 9] are used to learn a classifier for the allocation rule. Then, we derive a payment rule from the learned classifier. An exact classifier produces a fully incentive compatible and individually rational payment rule for the given allocation rule. A primary advantage of this approach over the VCG payment rule is that the optimality of an allocation rule is not required to get an incentive compatible mechanism. Rather, the more general property of monotonicity suffices. In addition, the framework functions equally well when an allocation rule does not have a corresponding incentive compatible payment rule, by seeking a payment rule that minimizes a regularized upper bound on empirical regret for truthful bidding.

This thesis is conducted, in part, for a paper submitted to the 12th ACM Conference on Electronic Commerce 2011, "Payment Rules for Combinatorial Auctions via Structural SVMs" by Dütting, Fischer, Jirapinyo, Lai, Lubin, and Parkes. This thesis discusses the theoretical model presented in the paper and further experiments with the model in various settings, in order to verify the framework and generate insights into how the model works in practice.

1.1 Related Work

With the intractability of combinatorial auctions, a prominent research direction has been to identify different auction mechanisms that compromise certain properties such as optimality and incentive compatibility, while preserving computational tractability. Many approximately optimal and heuristic mechanisms have been proposed for combinatorial auctions. For instance, Lavi [12] studies worst-case optimality of polynomial time mechanisms. Lehmann et al. [13] proposes a $\sqrt{m}$ – approximatation greedy mechanism for a special case of combinatorial auctions where each bidder can bid on up to one bundle of items. In the general setting of combinatorial auctions, linear programming relaxation

(LPR) is known to generate optimal outcomes in some settings [4]. In practice, many types of auctions are employed in place of general CAs, such as Simultaneous Multiple-Round (SMR) auctions and Package Bidding used in FCC Spectrum Auctions.[1]

Focusing primarily on designing payment rules for given allocation rules is a relatively new research direction that this thesis will explore. Enforcing certain desirable properties, Parkes et al. [19] investigates different payment rules for a given allocation rule in the context of combinatorial exchanges. Sharing similar methodologies, but differing in substance, Lahaie [10, 11] employs the technique of using a "kernel trick" to work with nonlinear pricing in combinatorial auctions. Nonetheless, his work differs from ours in many ways. First, Lahaie formulates the problem of identifying market clearing prices as a quadratic optimization program, as opposed to a learning problem as we propose in this thesis. In addition, Lahaie works with market clearing prices rather than with the space of items and bundles. Lastly, his framework is limited to utilitarian goals, leaving incentive compatibility as an indirect result of the model. On the other hand, full incentive compatibility is explicitly enforced by the model presented in this thesis.

On the machine learning side, the core of the framework presented in this thesis is a multiclass Support Vector Machine. The framework uses a Support Vector Machine (SVM) to learn from training instances generated by an allocation rule. Since an allocation rule of a combinatorial auction outputs more than two classes (i.e., all possible bundles of goods), the framework requires an SVM that can handle multiple classes. Nonetheless, SVMs are fundamentally binary classifiers. Earlier efforts to extend SVMs to work with multiple classes included two types of models: one-versus-all and one-versus-one. In a one-versus-all (OVA) model, m separate SVM models are learned for m different classes. Each of the sub-models then defines decision boundaries between that particular class and all other classes. A winning class is chosen based on margins. A one-versus-one (OVO) model, on the other hand, builds $\frac{m(m-1)}{2}$ classifiers for all pairs of classes. Then, a voting algorithm is used to

[1] More information can be found at http://wireless.fcc.gov/auctions.

determine a winning class [17]. Weston and Watkins [26] extend traditional SVMs to work with multiple classes by adding constraints proportional to the number of classes; as a result, only a single classifier is needed to classify multiple classes. Crammer and Singer [5] further increase efficiency of multiclass SVMs. Tsochantaridis et al. [25] generalize multiclass SVMs to work with more complex datasets, and call it *Structural SVMs*. Nonetheless, it is unclear which approach is best for multiclass classification. For instance, Rifkin and Klautau [20] argue that properly-tuned OVA classifiers can be as good as other approaches in terms of accuracy. In the context of this thesis, it is important to note that the framework presented here can be formulated as OVA, OVO, and Structural SVMs. Nevertheless, we have chosen to use Structural SVMs to formulate the framework for a couple of reasons. First, Structural SVMs require training only one classifier per setting, removing logistical costs of preparing datasets for and training multiple, different sub-classifiers for each setting. Moreover, with Structural SVMs, we do not have to deal with normalizing different classifiers to the same scale.

1.2 Summary of Contributions

The theoretical and empirical contributions of this thesis are as follows:

- This thesis presents a new framework for designing a strategyproof (SP) payment rule for any given allocation rule, granted that the allocation rule can be made SP. An exact SVM classifier produces a payment rule that is fully SP and individually rational (IR). The advantage of this framework over existing methods lies in its applicability to any type of allocation rule, whereas for example, the hallmark VCG rule is SP only when coupled with optimal allocation rules.

- In addition, a connection is made between training error and economic regret. Specifically, minimizing regularized empirical error also minimizes a regularized upper bound on empirical regret. This implies that when a classifier is inexact or an allocation rule

cannot be made SP, the resulting payment rule still seeks to minimize an upper bound on empirical regret for truthful bidding.

- We use benchmark value distributions and allocation rules to experiment with and verify our framework. In the single item auction setting, agents on average experience ex post regret of less than 0.2% of agents' valuation range. In the *single-minded* CA setting where each agent can bid on at most one bundle, average ex post regret is less than 1.5%.[2] Lastly, in the *multi-minded* CA setting where each agent bids on a fixed number of bundles, easy value distributions produce average ex post regret of less than 0.3%, whereas difficult distributions yield average ex post regret of up to 2% of agents' valuation range.

- The payment rules produced by our framework approximate well-known SP payment rules, such as second price, VCG, and an SP greedy mechanism for single-minded CAs. The root mean square errors of the predicted payments compared to the outputs of well-known IC payment rules are in most cases less than 0.15, with agents' valuation range at (0, 1].

- Individual rationality violation rates of our payment rules are less than 4% for single item auctions, between 0% and 7% for single-minded CAs, less than 6% for multi-minded CAs with easy value distributions, and between 1% and 16% for multi-minded CAs with difficult value distributions.

- Value distributions in which goods are substitutes are the most difficult setting for our framework, as correct allocations in this setting tend to be skewed toward smaller bundles. In this setting, classification accuracies range from 70% to 92%. On the other hand, value distributions with complementary goods are the easiest setting, as most allocations are in the full package of goods. In this setting, classification accuracies are greater than 90%.

[2]The numbers reported in this section are based on the second attribute vector χ_2, which outperforms χ_1 in most cases.

- One of the main components of the framework is a feature mapping that maps agents' valuations and allocated bundles onto a higher dimensional space, in the process defining the hypothesis space for the shapes of possible payment functions. In this thesis, two types of feature mappings are experimented with—one precludes hypotheses that generalize the knowledge gained from one bundle to another, while the other allows for this. In our experiments, the two mappings perform equally well when the bundle space is small. The second mapping outperforms the first one, as the bundle space gets larger.

- Our initial experiments with an allocation rule that does not have a corresponding SP payment rule show that classification accuracy, distribution of ex post regret, and IR violation rate in this setting are in similar ranges as those in a setting with an SP payment rule, providing initial evidence that applying our framework to a non-SP allocation rule does not result in additional performance loss.

1.3 Outline

The remainder of this thesis is organized as follows. Chapter 2 discusses important notation related to combinatorial auctions and introduces the theoretical framework as well as proofs of its various properties. In Chapter 3, we present the design of the experiments that are employed to yield insights into how the model works in practice. Computational considerations related to implementing the theoretical model are discussed. The three types of auction settings experimented with—single item auctions, single-minded combinatorial auctions, and multi-minded combinatorial auctions—are also explained in details. Chapter 4 presents and discusses the results of the experiments. Chapter 5 concludes and offers directions for future work.

Chapter 2

Designing Payment Rules with Structural SVMs

The issue of designing combinatorial auctions that simultaneously have good computational and economic properties is not trivial. In this chapter, I present the theoretical aspect of a model for designing payment rules based on the concept of *Structural Support Vector Machines* (Structural SVMs). The approach yields payment rules that are both tractable and strategyproof. In addition, we also establish a connection between empirical error of the learning problem and ex post regret of the auction mechanism.

Section 2.1 explores definitions and notations related to combinatorial auctions (CAs), as well as various properties that are important to the framework. Section 2.2 summarizes structural SVMs. Lastly, Section 2.3 introduces the Structural SVM approach to designing strategyproof payment rules, as well as related proofs.

2.1 Combinatorial Auctions

In a combinatorial auction, m distinct items are being auctioned, and n agents place offers on various subsets of these items. Each agent (or "bidder") has a valuation $\mathbf{x}_i \in \mathbb{R}_+^{2^m} = B$ over all possible subsets (also called "bundles" and "packages") of items. Let $Y = \{0,1\}^m$ define the set of possible allocations, where $\mathbf{y}_j = 1$ if and only if item j is included in the bundle. Let $\mathbf{x}_i[\mathbf{y}]$ be agent i's valuation of bundle $\mathbf{y}$. We require that an agent's valuation is monotonic, meaning that $\mathbf{x}_i[\mathbf{y}] \geq \mathbf{x}_i[\mathbf{y}']$ for all $\mathbf{y}' \subseteq \mathbf{y}$, and normalized, meaning that $\mathbf{x}_i[\mathbf{0}] = 0$. We use $\mathbf{x} \in B^n = X$ to represent all agents' valuations $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, and $\mathbf{x}_{-i} \in B^{n-1}$ to represent all agents' values other than agent i's, $(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n)$.

An auction mechanism consists of two components. First, the winner determination algorithm specifies an *allocation rule*, which determines the assignment of packages of items (including the empty bundle) to agents given agents' valuations. We assume that the allocation rule is symmetric with respect to agents, and therefore without loss of generality, we define various things and derive the theoretical framework from the viewpoint of agent 1. Second, a *payment rule* dictates how much agent 1 has to pay for the allocated package. Taken together, the allocation rule and payment rule defines an auction mechanism.

Definition 2.1.1 (Allocation Rule)**.** A function $g : B \times B^{n-1} \to Y$, which outputs the allocation to agent 1 given agent 1's valuation and the other agents' valuations, defines an *allocation rule*.

Definition 2.1.2 (Payment Rule)**.** A function $t : B \times B^{n-1} \times Y \to \mathbb{R}$, which outputs the payment collected from agent 1 given agent 1's valuation, the other agents' valuations, and agent 1's allocated bundle, defines a *payment rule*.

Definition 2.1.3 (Auction Mechanism)**.** An *auction mechanism* is defined as a pair of an allocation rule g and a payment rule t: (g, t).

Another important assumption we make about the allocation rule is that it satisfies

consumer sovereignty. This property helps establish the connection between training error and economic regret.

Definition 2.1.4 (Consumer Sovereignty)**.** A decision rule g with range Y satisfies consumer sovereignty if for every $\mathbf{x}_{-1} \in B^{n-1}$ and every $\mathbf{y} \in Y$, there exists some $\mathbf{x}_1 \in B$ for which $g(\mathbf{x}_1, \mathbf{x}_{-1}) = \mathbf{y}$.

Consumer sovereignty implies that with a sufficiently high valuation, agent 1 can possibly be allocated any bundle the agent wants, regardless of other agents' valuations. This property of a mechanism implies that optimizing for agent 1 over the space of agent 1's valuations is equivalent to optimizing for agent 1 over the space of bundles, as for every $\mathbf{x}_{-1}$ and every $\mathbf{y}$, there exists $\mathbf{x}_1$ that agent 1 can submit and receive bundle $\mathbf{y}$. Since agent 1 is asked to submit $\mathbf{x}_1$, from the viewpoint of agent 1, to minimize the agent's own economic regret is to optimize over the space of agent 1's valuation. Nevertheless, the training problem of the framework presented in Section 2.3 optimizes for agent 1 over the space of bundles. This requirement bridges the gap between the optimization problem of the training problem and agent 1's economic interest. Consumer sovereignty is a strong but reasonable property. A weaker requirement will likely suffice, but we require this property for simplicity.

In addition, three concepts of auction mechanisms are particularly important to the discussion of the new framework presented in this thesis.

2.1.1 Strategyproofness

We assume *quasi-linear* utility functions for agents, and as such, the utility to agent i given that the agent is allocated $\mathbf{y}$ and charged z is $\mathbf{x}_i[\mathbf{y}] - z$. Given auction mechanism (g, t) and other agents' valuations $\mathbf{x}_{-1}$, the utility to agent 1 is:

$$u_1(\mathbf{x}_1) = \mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - t(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1})] \tag{2.1}$$

A property that is central to our work is *incentive compatibility* (IC). A mechanism is considered incentive compatible, if all of the agents have no incentive to deviate from truthful

reports of their actual valuations. Given auction mechanism (g, t) that is symmetrical with respect to agents, we define incentive compatibility from the perspective of agent 1 as follows:

Definition 2.1.5 (Incentive Compatibility)**.** Agent-symmetric mechanism (g, t) is *incentive-compatible* (IC) if and only if

$$\mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - t(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1})) \geq \mathbf{x}_1[g(\mathbf{x}'_1, \mathbf{x}_{-1})] - t(\mathbf{x}'_1, \mathbf{x}_{-1}, g(\mathbf{x}'_1, \mathbf{x}_{-1})),$$

for every $\mathbf{x}_1 \in B$, every $\mathbf{x}'_1 \in B$, and every $\mathbf{x}_{-1} \in B^{n-1}$.

With this property, agent 1 cannot improve its utility by changing the report of its valuation, no matter what the other agents do. This is the definition of *dominant-strategy incentive compatibility* (DSIC) or *strategyproofness* (SP). Hence, we use the terms incentive compatibility and strategyproofness interchangeably in this thesis.

The definition of IC leads to two issues of particular interest. The first issue is the characteristics of allocation rules that would have corresponding, incentive compatible payment rules. The framework for deriving payment rules presented in following sections does not make any assumption about the input allocation rules in terms of incentive compatibility. If a given allocation rule has a corresponding IC payment rule, then the framework would in principle learn the payment rule that when coupled with the allocation rule, yields an IC mechanism. At the same time, if a given allocation rule does not have a corresponding IC payment rule, the framework would still work, but the resulting payment rule would not make the mechanism fully incentive compatible. Rather, the payment rule in that setting would minimize a regularized upper bound on empirical regret for truthful bidding.

To characterize so-called "SP-capable" rules, it is important to note the concept of *weak monotonicity* (W-MON). The W-MON property of mechanism g requires that if $g(\mathbf{x}_i, \mathbf{x}_{-i}) = a \neq b = g(\mathbf{x}'_i, \mathbf{x}_{-i})$, then $\mathbf{x}_i[a] - \mathbf{x}_i[b] \geq \mathbf{x}'_i[a] - \mathbf{x}'_i[b]$. In other words, W-MON implies that if the outcome changes from a to b as a result of a change of a single agent's valuation, then that agent must have increased its valuation of the new outcome b relative to its valuation

of the old outcome a [18]. Weak monotonicity is a *necessary* condition for the mechanism to be capable of being made strategyproof [22]. Various domains have been considered in order to analyze sufficient conditions for specific cases, and generalized monotonicity properties are found to be *sufficient* [22, 2]. For the interest of the framework presented here, we are going to merely note that monotonic allocation rules can be made strategyproof, when coupled with appropriate payment rules.

One can also consider a direct characterization of strategyproofness in terms of g and t. From the perspective of agent 1, two sufficient conditions are required for mechanism (g, t) to be strategyproof [18]:

$$t(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}) = p(\mathbf{x}_{-1}, \mathbf{y}), \quad \forall \mathbf{x}_1 \in B, \forall \mathbf{x}_{-1} \in B^{n-1}, \forall \mathbf{y} \in Y \quad \text{and} \tag{2.2}$$

$$g(\mathbf{x}_1, \mathbf{x}_{-1}) \in \arg\max_{\mathbf{y}' \in Y} \left[\mathbf{x}_1[\mathbf{y}'] - t(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}') \right], \quad \forall \mathbf{x}_1 \in B, \forall \mathbf{x}_{-1} \in B^{n-1} \tag{2.3}$$

where $p(\mathbf{x}_{-1}, \mathbf{y})$ is a price function that does not depend on $\mathbf{x}_1$. Equation 2.2 implies that given the allocation, the price imposed on agent 1 must not depend on agent 1's own valuation. Equation 2.3 seeks to optimize the allocation from the viewpoint of agent 1. (With symmetry, the allocation must simultaneously optimize for each and every agent.)

It is also important to note the uniqueness of strategyproof payment rules. Let mechanism (g, t) be an incentive compatible mechanism. Mechanism (g, t') is incentive compatible if and only if for some function $h : \mathbf{x}_{-1} \to \mathbb{R}$, we have that $t'(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}) = t(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}) + h(\mathbf{x}_{-1})$ for every $\mathbf{x}_1 \in B$, $\mathbf{x}_{-1} \in B^{n-1}$, and $\mathbf{y} \in Y$ [18]. In the context of this thesis, we normalize payment for the empty bundle to be zero for individual rationality (the concept explored in Section 2.1.3). With normalized prices, we can derive the following result:

Proposition 1 (Uniqueness of normalized, incentive compatible payment rules)**.** *Given allocation rule g, normalized payment rule t that makes auction mechanism (g, t) incentive compatible is unique.*

Proof. Assume for a contradiction that the normalized payment rule that makes mechanism

(g, t) incentive compatible is not unique. Thus, there exist two distinct IC payment rules t and t'. As both of them are IC, $t'(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}) = t(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}) + h(\mathbf{x}_{-1})$ for every $\mathbf{x}_1 \in B$, $\mathbf{x}_{-1} \in B^{n-1}$, and $\mathbf{y} \in Y$. When $\mathbf{y} = \mathbf{0}$, we have that $t'(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{0}) = t(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{0}) = 0$, as both t and t' are normalized. Therefore, $h(\mathbf{x}_{-1}) = 0$, and $t = t'$. □

2.1.2 Ex Post Regret

In reality, we cannot always get a fully SP payment rule for a couple reasons. First, since our framework is based on a machine learning method, the resulting classifier is not always exact. Second, if the given allocation rule is not monotonic, then the framework would certainly not learn an SP rule. With these reasons, we are interested in a reasonable metric that could measure the performance of a not fully SP mechanism regarding its strategyproofness. A standard metric for this purpose is regret.[1]

Definition 2.1.6 (Ex Post Regret)**.** The *ex post regret* to agent 1 for truthfully bidding $\mathbf{x}_1$ in auction (g, t), given that the other agents bid $\mathbf{x}_{-1}$ is:

$$\begin{aligned} &\mathit{regret}_{g,t}(\mathbf{x}_1, \mathbf{x}_{-1}) = \\ &\quad \max_{\mathbf{x}'_1 \in B} \big(\mathbf{x}_1[g(\mathbf{x}'_1, \mathbf{x}_{-1})] - t(\mathbf{x}'_1, \mathbf{x}_{-1}, g(\mathbf{x}'_1, \mathbf{x}_{-1}))\big) - \left(\mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - t(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1}))\right). \end{aligned}$$

The ex post regret measures the loss in utility to agent 1 for bidding truthfully as opposed to what the agent could have obtained given knowledge of the other agents' bids (hence *ex post*). In the case that the mechanism (g, t) is strategyproof, Definition 2.1.5 tells us that the $\mathbf{x}'_1$ that maximizes the first term of the regret formula would be equal to $\mathbf{x}_1$, and thus the ex post regret to agent 1 would be zero.

Consider the joint distribution $\mathbf{x}_{-1} \sim P_{\mathbf{x}_{-1}|\mathbf{x}_1}$ on the others' agents given that they bid truthfully and that agent 1's valuation is $\mathbf{x}_1$, the *expected ex post regret* to agent 1 for

[1] For an extended discussion of the topic, see Chapter 4 of [15].

bidding truthfully is

$$E_{\mathbf{x}_{-1}\sim P_{\mathbf{x}_{-1}|\mathbf{x}_1}}\left[regret_{g,t}(\mathbf{x}_1,\mathbf{x}_{-1})\right]. \tag{2.4}$$

2.1.3 Individual Rationality

Another important property of a mechanism is *individual rationality* (IR). IR ensures that all agents gain non-negative utility from the mechanism, and therefore participate in the mechanism on a voluntary basis.

Definition 2.1.7 (Individual Rationality)**.** Agent-symmetric mechanism (g,t) is (ex post) *individually rational* if for every $\mathbf{x}_1 \in B$ and $\mathbf{x}_{-1} \in B^{n-1}$,

$$\mathbf{x}_1[g(\mathbf{x}_1,\mathbf{x}_{-1})] \geq t(\mathbf{x}_1,\mathbf{x}_{-1},g(\mathbf{x}_1,\mathbf{x}_{-1})).$$

This property means that agent 1 always gets non-negative utility by participating in the mechanism, regardless of its own valuation and the other agents' valuations. Definition 2.1.7 is defined from the perspective of agent 1, but by symmetry, it is applied to every agent. This requirement is important in designing real-world mechanisms, as agents would not willingly participate in mechanisms that would yield them negative utility.

2.2 Structural Support Vector Machines

Before we present the framework, we need to familiarize ourselves with *Structural Support Vector Machines* (Structural SVMs), the mechanics behind the framework. The Structural SVM is a generalized form of multiclass support vector machines that can handle more complex outputs, such as trees, sequences, and sets [25, 9]. Let $\{(\mathbf{x}^1,\mathbf{y}^1),(\mathbf{x}^2,\mathbf{y}^2),\ldots,(\mathbf{x}^\ell,\mathbf{y}^\ell)\}$, where $(\mathbf{x}^j,\mathbf{y}^j) \in X \times Y$, be training instances. The goal of the Structural SVM is to learn a classifier

$$h : X \to Y. \tag{2.5}$$

The model then defines the *discriminant function* as the score of class $\mathbf{y}$ for input $\mathbf{x}$:

$$f(\mathbf{x}, \mathbf{y}) = \mathbf{w} \cdot \psi(\mathbf{x}, \mathbf{y}), \tag{2.6}$$

where $\psi(\mathbf{x})$ is a feature map of $\mathbf{x}$ and $\mathbf{y}$, and $\mathbf{w}$ is a parameter vector. The classifier h then is defined as

$$h(\mathbf{x}) = \arg\max_{\mathbf{y} \in Y} f(\mathbf{x}, \mathbf{y}). \tag{2.7}$$

The definitions here lead directly to a hard-margin formulation of the learning problem by fixing a margin of all training examples at 1 and using the norm of $\mathbf{w}$ as a regularizer:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2, \text{ s.t. } f(\mathbf{x}^j, \mathbf{y}^j) - f(\mathbf{x}^j, \mathbf{y}) \geq 1, \quad \forall j, \mathbf{y} \neq \mathbf{y}^j. \tag{2.8}$$

From Equation 2.6, we get the training problem

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2, \text{ s.t. } \mathbf{w} \cdot \psi(\mathbf{x}^j, \mathbf{y}^j) - \mathbf{w} \cdot \psi(\mathbf{x}^j, \mathbf{y}) \geq 1, \quad \forall j, \mathbf{y} \neq \mathbf{y}^j. \tag{2.9}$$

With inconsistent training data, a soft-margin learning model is employed. Since different inaccurate predictions vary, a *loss function* $\Delta : Y \times Y \to \mathbb{R}$ is used instead of a fixed margin. The value of $\Delta(\mathbf{y}^j, \mathbf{y})$ is equal to zero if $\mathbf{y}^j = \mathbf{y}$, and is greater than zero otherwise. The soft-margin, n-slack formulation of the learning problem with margin-rescaling is

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{j=1}^{\ell} \xi_j, \\ \text{s.t.} \quad & \mathbf{w} \cdot \psi(\mathbf{x}^j, \mathbf{y}^j) - \mathbf{w} \cdot \psi(\mathbf{x}^j, \mathbf{y}) \geq \Delta(\mathbf{y}^j, \mathbf{y}) - \xi_j, \quad \forall j, \mathbf{y} \neq \mathbf{y}^j. \end{aligned}$$

Not only does the Structural SVM allow for more complex outputs, but the model also utilizes a cutting plane model, which significantly reduces the number of constraints being considered during the learning process. In the next section, we introduce an approach to design strategyproof payment rules using Structural SVMs.

2.3 Designing Payment Rules with Structural SVMs

The goal of the framework presented in this section is to employ Structural SVMs to design payment rules that would be strategyproof when coupled with allocation rules. The main insight here is the similarity and connection between the decision boundaries of an exact Structural SVM classifier and the defining properties of strategyproofness of auction mechanisms. In this section, I first introduce the framework, then prove that the model actually produces an SP payment rule given an allocation rule, and finally present some other desirable properties of the model.

2.3.1 The Framework

First, we generate ℓ training instances $\{(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \ldots, (\mathbf{x}^\ell, \mathbf{y}^\ell)\}$ from a distribution $(x, y) \sim P(X, Y)$, where $P(X, Y)$ is the distribution induced by distribution $P_{\mathbf{x}}$ of agents' valuations and allocation rule g. Specifically, we first generate agents' valuations $\mathbf{x}^j \in X$ for each of the training instances. Each instance $\mathbf{x}^j$ is drawn from some distribution $P_{\mathbf{x}}$ and consists of n valuation vectors, where each of the n vectors represents an individual agent's valuation $\mathbf{x}^j = (\mathbf{x}_1^j, \mathbf{x}_2^j, \ldots, \mathbf{x}_n^j) = (\mathbf{x}_1^j, \mathbf{x}_{-1}^j)$. Given an allocation rule g, we then generate ℓ training instances, $\{(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \ldots, (\mathbf{x}^\ell, \mathbf{y}^\ell)\}$, where $\mathbf{y}^j = g(\mathbf{x}_1^j, \mathbf{x}_{-1}^j)$.

Following the Structural SVM framework, we use the training data to learn parameter vector $\mathbf{w}$ for a classifier,

$$h_{\mathbf{w}} : X \to Y \text{ s.t. } h_{\mathbf{w}}(\mathbf{x}) = \arg\max_{\mathbf{y} \in Y} f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}).$$

We use the following function as the *discriminant function*:

$$f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = w_1 \mathbf{x}_1[\mathbf{y}] + \mathbf{w}_{-1}^T \psi(\mathbf{x}_{-1}, \mathbf{y}), \tag{2.10}$$

where $\mathbf{w} = (w_1, \mathbf{w}_{-1})$ and $\psi(\mathbf{x}_{-1}, \mathbf{y})$ is a feature map of the other agents' valuations $\mathbf{x}_{-1}$ and bundle $\mathbf{y}$. Note that the discriminant function is linear in agent 1's valuation of $\mathbf{y}$ and nonlinear in the other agents' valuations. As it will become clear in the proof section, this

custom discriminant function enables us to make a connection between the learning problem and strategyproofness of combinatorial auctions. Using this custom discriminant function, we follow the rest of the process as described in Section 2.2, and obtain the training problem:

$$\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{\ell}\sum_{k=1}^{\ell}\xi^k \tag{2.11}$$

$$\begin{aligned}
\text{s.t. } & w_1\big(\mathbf{x}_1^1[\mathbf{y}^1] - \mathbf{x}_1^1[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^1,\mathbf{y}^1) - \psi(\mathbf{x}_{-1}^1,\mathbf{y})\big) \geq \Delta(\mathbf{y}^1,\mathbf{y}) - \xi^1, \quad \forall \mathbf{y}\in Y \\
& \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
& w_1\big(\mathbf{x}_1^\ell[\mathbf{y}^\ell] - \mathbf{x}_1^\ell[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^\ell,\mathbf{y}^\ell) - \psi(\mathbf{x}_{-1}^\ell,\mathbf{y})\big) \geq \Delta(\mathbf{y}^\ell,\mathbf{y}) - \xi^\ell, \quad \forall \mathbf{y}\in Y
\end{aligned}$$

where $\Delta(\mathbf{y},\mathbf{y})$ is a loss function. In the context of this thesis, we use a fixed margin loss function, $\Delta(\mathbf{y},\mathbf{y}') = 0$ if $\mathbf{y} = \mathbf{y}'$, and 1 otherwise. The actual learning process follows the standard one of Structural SVMs. Once we get classifier $h_{\mathbf{w}}(\mathbf{x})$, we finally define *payment rule* $t_{\mathbf{w}}$ for allocation rule g as:

$$t_{\mathbf{w}}(\mathbf{x}_1,\mathbf{x}_{-1},\mathbf{y}) = -\frac{\mathbf{w}_{-1}^T}{w_1}\psi(\mathbf{x}_{-1},\mathbf{y}), \tag{2.12}$$

where the assumption is made that $w_1 > 0$. This assumption will be experimentally validated in Sections 3 and 4. When classifier $h_{\mathbf{w}}$ is exact, auction mechanism $(g, t_{\mathbf{w}})$ is strategyproof. In addition, we can define a normalized version of $t_{\mathbf{w}}$:

$$\bar{t}_{\mathbf{w}}(\mathbf{x}_1,\mathbf{x}_{-1},\mathbf{y}) = t_{\mathbf{w}}(\mathbf{x}_1,\mathbf{x}_{-1},\mathbf{y}) - t_{\mathbf{w}}(\mathbf{x}_1,\mathbf{x}_{-1},\mathbf{0}). \tag{2.13}$$

When classifier $h_{\mathbf{w}}$ is exact, auction mechanism $(g, \bar{t}_{\mathbf{w}})$ is strategyproof and individually rational.

2.3.2 Exact Classifier

We first consider $t_{\mathbf{w}}$ and its strategyproofness property. With the assumption that $w_1 > 0$, we can rewrite $h_{\mathbf{w}}(\mathbf{x})$ as

$$h_{\mathbf{w}}(\mathbf{x}) = \arg\max_{\mathbf{y}\in Y} \frac{1}{w_1} f_{\mathbf{w}}(\mathbf{x},\mathbf{y}) \tag{2.14}$$

$$= \arg\max_{\mathbf{y}\in Y} [\mathbf{x}_1[\mathbf{y}] - t_{\mathbf{w}}(\mathbf{x}_1,\mathbf{x}_{-1},\mathbf{y})]. \tag{2.15}$$

With $t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y})$ as the payment rule, note that $\frac{1}{w_1} f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = \mathbf{x}_1[\mathbf{y}] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y})$ is the utility to agent 1 for being allocated $\mathbf{y}$. Therefore, the classifier $h_{\mathbf{w}}(\mathbf{x})$ optimizes for agent 1. In addition, by design, the payment collected from agent 1 is also independent of agent 1's own valuation. Thus, this framework presents a strategyproof payment rule for allocation rule g. Following, we formally prove this notion:

Theorem 1. *Given allocation rule g, an exact classifier $h_{\mathbf{w}}$ defines a payment rule $t_{\mathbf{w}}$ that makes auction mechanism $(g, t_{\mathbf{w}})$ strategyproof.*

Proof. From Equation 2.12, the payment rule satisfies the first requirement of strategyproofness that the payment is independent of an agent's own valuation (Equation 2.2) by construction. For the second requirement that the mechanism has to optimize for individual agents (Equation 2.3), assume for a contradiction that there exists some $\mathbf{x}$, some $\mathbf{y}' \neq g(\mathbf{x}) = \mathbf{y}$ for which $\mathbf{x}_1[\mathbf{y}'] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}') > \mathbf{x}_1[\mathbf{y}] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y})$. But then, we have $h_{\mathbf{w}}(\mathbf{x}) \neq \mathbf{y}$ and the classifier is not exact. □

The strategyproofness property of $\bar{t}_{\mathbf{w}}$ directly follows, as the $t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{0})$ term used to normalize is independent of agent 1's value (the first requirement of SP) and is being applied to all possible $\mathbf{y}$ equally (preserving $t_{\mathbf{w}}$'s fulfillment of the second requirement of SP). Following is the proof showing that $\bar{t}_{\mathbf{w}}$ is individually rational.

Theorem 2. *Given allocation rule g, an exact classifier $h_{\mathbf{w}}$ defines a payment rule $\bar{t}_{\mathbf{w}}$ that makes auction mechanism $(g, \bar{t}_{\mathbf{w}})$ individually rational.*

Proof. Assume for a contradiction that $(g, \bar{t}_{\mathbf{w}})$ is not individually rational. From Definition 2.1.7, there exists $\mathbf{x}_1 \in B$ and $\mathbf{x}_{-1} \in B^{n-1}$ that satisfies

$$\mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - \bar{t}_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}, \mathbf{x}_{-1})) < 0.$$

Since $h_{\mathbf{w}}$ is exact, we get that $g(\mathbf{x}) = h_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}) = \arg\max_{\mathbf{y} \in Y} [\mathbf{x}_1[\mathbf{y}] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y})]$.

This equation implies that

$$\begin{aligned} \mathbf{x}_1[g(\mathbf{x})] - t_{\mathbf{w}}(\mathbf{x}, g(\mathbf{x})) &\geq \mathbf{x}_1[\mathbf{y}] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}), \qquad \forall \mathbf{y} \in Y \\ &\geq \mathbf{x}_1[\mathbf{0}] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{0}) \\ \mathbf{x}_1[g(\mathbf{x})] - t_{\mathbf{w}}(\mathbf{x}, g(\mathbf{x})) + t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{0}) &\geq 0 \qquad (2.16) \\ \mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - \bar{t}_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}, \mathbf{x}_{-1})) &\geq 0. \end{aligned}$$

Equation 2.16 follows the fact that we normalize agent's utility for non-allocation to zero. Hence, we get a contradiction. □

2.3.3 Inexact Classifier

Most of the time, we will not be able to get an exact classifier from the learning problem. Nevertheless, an inexact hypothesis still enjoys some desirable properties. First, we observe a connection between the error of the classifier and the expected ex post regret of agent 1. We define the following loss function,

$$\Delta_{\mathbf{x}}(\mathbf{y}, \mathbf{y}') = \frac{1}{w_1}[f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}') - f_{\mathbf{w}}(\mathbf{x}, \mathbf{y})], \qquad (2.17)$$

where $\Delta_{\mathbf{x}}(\mathbf{y}, \mathbf{y}') = 0$ when $\mathbf{y} = \mathbf{y}'$ and is otherwise weakly positive since $w_1 > 0$ (by assumption). Now we consider hypothesis space $\mathcal{H}_\psi$ of multi-class classifiers given feature mapping ψ. The *generalization error* for classifier $h_{\mathbf{w}} \in \mathcal{H}_\psi$ and distribution $P(X, Y)$ induced by $P_{\mathbf{x}}$, given loss function $\Delta_{\mathbf{x}}$, is then

$$R_P(h_{\mathbf{w}}) = \int_{X \times Y} \Delta_{\mathbf{x}}(\mathbf{y}, h_{\mathbf{w}}(\mathbf{x})) \mathrm{d}P(X, Y). \qquad (2.18)$$

The connection between generalization error and expected ex post regret can be made as follows.

Theorem 3. *Given allocation rule g and feature mapping ψ, the classifier $h_{\mathbf{w}} \in \mathcal{H}_\psi$ that minimizes generalization error $R_P(h_{\mathbf{w}})$ defines a payment rule that minimizes expected ex post regret.*

Proof. The regret faced by agent 1 for bidding truthfully on instance $\mathbf{x}_1$ given mechanism $(g, t_{\mathbf{w}})$ and other agents' bids $\mathbf{x}_{-1}$ is

$$
\begin{aligned}
&\textit{regret}_{g,t_{\mathbf{w}},\mathbf{x}_{-1}}(\mathbf{x}_1) \\
&\quad = \max_{\mathbf{x}_1' \in \mathbb{R}^{2^m}} \big(\mathbf{x}_1[g(\mathbf{x}_1', \mathbf{x}_{-1})] - t_{\mathbf{w}}(\mathbf{x}_1', \mathbf{x}_{-1}, g(\mathbf{x}_1', \mathbf{x}_{-1}))\big) - (\mathbf{x}_1[g(\mathbf{x})] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1}))) \\
&\quad = \max_{\mathbf{y}' \in Y} \big(\mathbf{x}_1[\mathbf{y}'] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, \mathbf{y}')\big) - (\mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] - t_{\mathbf{w}}(\mathbf{x}_1, \mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1}))) \qquad (2.19) \\
&\quad = \max_{\mathbf{y}' \in Y} \left(\mathbf{x}_1[\mathbf{y}'] + \frac{\mathbf{w}_{-1}^T}{w_1}\psi(\mathbf{x}_{-1}, \mathbf{y}')\right) - \left(\mathbf{x}_1[g(\mathbf{x}_1, \mathbf{x}_{-1})] + \frac{\mathbf{w}_{-1}^T}{w_1}\psi(\mathbf{x}_{-1}, g(\mathbf{x}_1, \mathbf{x}_{-1}))\right) \\
&\quad = \frac{1}{w_1} f_{\mathbf{w}}(\mathbf{x}, h_{\mathbf{w}}(\mathbf{x})) - \frac{1}{w_1} f_{\mathbf{w}}(\mathbf{x}, g(\mathbf{x})) \qquad (2.20) \\
&\quad = \Delta_{\mathbf{x}}(g(\mathbf{x}), h_{\mathbf{w}}(\mathbf{x})) = \Delta_{\mathbf{x}}(\mathbf{y}, h_{\mathbf{w}}(\mathbf{x})),
\end{aligned}
$$

where $\mathbf{y} = g(\mathbf{x})$ is the target class given input $\mathbf{x}$. Eq. (2.19) follows from the assumption of consumer sovereignty, which ensures that for every input $\mathbf{x}_{-1}$ and every $\mathbf{y}'$ there exists some $\mathbf{x}_1'$ that agent 1 can submit and receive package $\mathbf{y}'$ under the allocation rule g. Therefore, minimizing generalization error $R_P(h_{\mathbf{w}})$ is equivalent to minimizing the expected ex post regret with respect to distribution P. □

Lastly, we establish a relationship between the ex post regret on a training instance and SVM training error.

Theorem 4. *Given allocation rule g and solution $(\mathbf{w}^*, \xi^*)$ to the structural SVM classification, the ex post regret on training example $(\mathbf{x}^k, \mathbf{y}^k)$ is bounded above by $\frac{1}{w_1^*}\xi^k$.*

Proof. From the learning problem (2.11), we observe that

$$\xi^k = \max_{\mathbf{y}' \in Y} \left[\Delta(\mathbf{y}^k, \mathbf{y}') + f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') - f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}^k)\right].$$

As $\Delta(\mathbf{y}, \mathbf{y}') \in \{0, 1\}$, we derive that

$$
\begin{aligned}
\xi^k &\geq \max_{\mathbf{y}' \in Y} \left[f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') - f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}^k)\right] \\
\frac{1}{w_1^*}\xi^k &\geq \max_{\mathbf{y}' \in Y} \left[\frac{1}{w_1^*} f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') - \frac{1}{w_1^*} f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}^k)\right] = \textit{regret}_{g,t}(\mathbf{x}_1^k, \mathbf{x}_{-1}^k)
\end{aligned}
$$

The last step comes from the proof of Theorem 3. □

It is noteworthy that instead of using $(g, \bar{t}_{\mathbf{w}})$, we can also adopt $(h_{\mathbf{w}}, \bar{t}_{\mathbf{w}})$ as an auction mechanism. Mechanism $(h_{\mathbf{w}}, \bar{t}_{\mathbf{w}})$ will always be strategyproof and individually rational. Nevertheless, since $h_{\mathbf{w}}$ is inexact, the allocation rule might be infeasible by allocating the same item to multiple agents.

Chapter 3

Experimental Design

In the previous chapter, the theoretical aspect of a new framework for designing strategyproof payment rules for combinatorial auctions is presented. Nevertheless, many details still need to be filled in, before the framework is operational in practice. This chapter presents design considerations for the experiments aimed at substantiating the use of Structural SVMs in designing strategyproof payment rules. In doing so, the framework is tested in three auction settings—single item auctions, single-minded CAs, and multi-minded CAs—that are progressively more difficult in terms of computation, and more sophisticated in terms of agents' possible actions.

The structure of this chapter is as follows. First, Section 3.1 introduces the three auction settings as well as their corresponding value distributions and winner determination algorithms. Taken together, a value distribution and a winner determination rule determine distribution $P(X, Y)$ for training and testing hypotheses. Section 3.2 presents the details of feature maps (ψ) and kernels of the training problem. Lastly, Section 3.3 lists out other practical details that are crucial to running the experiments.

3.1 Auction Settings

3.1.1 Single Item Auctions

We begin the experimental section with simplistic, non-combinatorial single item auctions, which provide initial insights into other, more complex settings. In this setting, one item is being auctioned, agent i places bid $x_i \in \mathbb{R}$ for the item, and n agents are present in the auction.

Allocation rule $g(x_1, \mathbf{x}_{-1}) = 1$ if agent 1 is allocated the item, and zero otherwise. With two classes, the learning problem for this setting can easily be formulated as a binary classification SVM. Nonetheless, for consistency, the Structural SVM formulation is employed in this setting. Given distribution $P_{\mathbf{x}}$ and allocation rule g, training set $\{(\mathbf{x}^1, y^1), \dots (\mathbf{x}^\ell, y^\ell)\}$ is generated from $y^j = g(x_1^j, \mathbf{x}_{-1}^j)$.

The learning problem for the single item setting derived from (2.11) is:

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{\ell}\sum_{k=1}^{\ell}\xi^k \tag{3.1}$$

$$\begin{aligned} \text{s.t. } & w_1\big(x_1^1[y^1] - x_1^1[y]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^1, y^1) - \psi(\mathbf{x}_{-1}^1, y)\big) \geq \Delta(y^1, y) - \xi^1, \quad \forall y \in \{0,1\} \\ & \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ & w_1\big(x_1^\ell[y^\ell] - x_1^\ell[y]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^\ell, y^\ell) - \psi(\mathbf{x}_{-1}^\ell, y)\big) \geq \Delta(y^\ell, y) - \xi^\ell, \quad \forall y \in \{0,1\}, \end{aligned}$$

where $x_1[y] = x_1$ if $y = 1$ and zero otherwise, and loss function $\Delta(y', y) = 1$ if $y' \neq y$ and zero otherwise. Given exact classifier $h_{\mathbf{w}} = \arg\max_{y \in \{0,1\}} f_{\mathbf{w}}(x, y)$, $\bar{t}_{\mathbf{w}}(x_1, \mathbf{x}_{-1}, y) = -\frac{\mathbf{w}_{-1}^T}{w_1}\left(\psi(\mathbf{x}_{-1}, 1) - \psi(\mathbf{x}_{-1}, 0)\right)$ defines a strategyproof, individually rational payment rule for allocation rule g.

Value Distribution

Next, we need to define $P(X, Y)$ for generating training and test instances. This section defines $P_{\mathbf{x}}$ used for the single item setting:

- **Uniform**: The bid of an agent is uniformly drawn from $(0, 1]$.

Winner Determination Algorithm

Once we have agents' valuations $\mathbf{x}$, allocation y is generated from allocation rule g: $y = g(x_1, \mathbf{x}_{-1})$. This section defines allocation rule g for this setting:

- **Optimal**: The agent with the highest bid wins. The corresponding strategyproof payment function of the optimal allocation rule for single item auctions is the second-price payment function: the winner pays the price of the second highest bid. With the uniqueness of strategyproof payment rules (Proposition 1), the exact classifier in this case is the second-price payment function.

  Hypothesis 1. *Payment rule $\bar{t}_{\mathbf{w}}$ derived from classifier $h_{\mathbf{w}}$ trained on the optimal allocation rule for a single item auction approximates the second-price payment rule.*

3.1.2 Single-minded Combinatorial Auctions

Our first foray into combinatorial auctions is with *single-minded* CAs, in which m items are auctioned, and each of the n agents requests a single bundle of items [13]. An agent i's valuation $\mathbf{x}_i$ is represented by (S_i, v_i), where agent i values bundle $S_i \in \{0, 1\}^m = Y$ at $v_i \in \mathbb{R}$. From the viewpoint of agent 1, agent 1 can be allocated either S_1 or the empty bundle.

Given training set $\{(\mathbf{x}^1, \mathbf{y}^1), \ldots, (\mathbf{x}^\ell, \mathbf{y}^\ell)\}$, the learning problem for the single-minded

CA case derived from (2.11) is:

$$\min_{\mathbf{w},\xi\geq 0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{\ell}\sum_{k=1}^{\ell}\xi^k \tag{3.2}$$

$$\begin{aligned}
\text{s.t. } & w_1\big(\mathbf{x}_1^1[\mathbf{y}^1] - \mathbf{x}_1^1[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^1,\mathbf{y}^1) - \psi(\mathbf{x}_{-1}^1,\mathbf{y})\big) \geq \Delta(\mathbf{y}^1,\mathbf{y}) - \xi^1, \quad \forall \mathbf{y}\in\{S_1^1,\mathbf{0}\} \\
& \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
& w_1\big(\mathbf{x}_1^\ell[\mathbf{y}^\ell] - \mathbf{x}_1^\ell[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^\ell,\mathbf{y}^\ell) - \psi(\mathbf{x}_{-1}^\ell,\mathbf{y})\big) \geq \Delta(\mathbf{y}^\ell,\mathbf{y}) - \xi^\ell, \quad \forall \mathbf{y}\in\{S_1^\ell,\mathbf{0}\},
\end{aligned}$$

where $\mathbf{x}_1[\mathbf{y}] = v_1$ if $\mathbf{y} = S_1$ and zero otherwise. An important requirement for strategyproofness in this context is *admissibility*. As the single-minded CA setting is just a special case of the multi-minded CA, we discuss the issue of admissibility in details in Section 3.1.3, when we discuss multi-minded CAs. Meanwhile, it is important to note that for each line of constraints in (3.2), we iterate over $\mathbf{y} \in Y$ instead of $\mathbf{y} \in \{S_1^k, \mathbf{0}\}$ in our experiments to get around the issue of admissibility.

Value Distributions

Regarding single-minded CAs, various distributions introduced by Sandholm [23] are widely used in CA literature. In our experiments, two of the distributions presented by Sandholm are used to generate agents' valuations $\mathbf{x}$ for the single-minded CA setting.

- **Uniform**: For each agent's preference (S_i, v_i), pick $|S_i|$ uniformly from $[1, m]$. $|S_i|$ goods are selected using a uniform distribution. Valuation v_i is drawn from $(0, \frac{|S_i|}{m}]$.
- **Decay**: For each agent's preference (S_i, v_i), first assign one random item, and repeatedly add a new random item until $rand(0, 1)$ exceeds α. Valuation v_i is drawn from $(0, \frac{|S_i|}{m}]$. Sandholm [23] concludes that $\alpha = 0.75$ generates the hardest dataset (at least for his algorithms). α is fixed at 0.75 in this thesis.

Winner Determination Algorithms

For the single-minded CA setting, we learn and test models using two types of allocation rules, which lead to two distinct strategyproof payment functions.

- **Optimal**: The optimal allocation rule seeks to maximize social welfare by solving the following integer linear programming (ILP) problem:

$$\max \sum_{i\in[1,n],S\in Y} a_{i,S} v_i(S) \tag{3.3}$$

$$\text{s.t.} \sum_{i\in[1,n],S|j\in S} a_{i,S} \leq 1 \qquad \forall j \in [1,m] \tag{3.4}$$

$$\sum_{S\in Y} a_{i,S} \leq 1 \qquad \forall i \in [1,n] \tag{3.5}$$

$$x_{i,S} \in \{0,1\} \qquad \forall i \in [1,n], \forall S \in Y \tag{3.6}$$

  where $a_{i,S}$ equals 1 if agent i receives bundle S, and zero otherwise. Condition 3.4 ensures that each item is allocated at most once, whereas Condition 3.5 implies that each agent receives up to one bundle.

  The corresponding strategyproof payment function is the Vickrey-Clarke-Groves (VCG) payment function, which states that each of the agents pays the opportunity cost it imposes on the other agents [18]. Let $T(\mathbf{x}_1, \mathbf{x}_{-1}) = \sum_{i\in[2,n],S\in Y} a_{i,S} v_i(S)$ be the maximum total welfare excluding the utility to agent 1. VCG payment rule t_{VCG} is defined as $T(\mathbf{0}, \mathbf{x}_{-1}) - T(\mathbf{x}_1, \mathbf{x}_{-1})$. This calculation can be done by solving two ILP problems. As normalized and strategyproof payment rules are unique for given allocation rules, if exact classifier $h_{\mathbf{w}}$ is to output strategyproof payment rule $\bar{t}_{\mathbf{w}}$, $\bar{t}_{\mathbf{w}}$ must be the VCG payment rule.

  Hypothesis 2. *Payment rule $\bar{t}_{\mathbf{w}}$ derived from classifier $h_{\mathbf{w}}$ trained on the optimal allocation rule for a single-minded CA approximates the VCG payment rule.*

- **Greedy**: Lehmann et al. [13] presents strategyproof mechanism for single-minded CAs based on a greedy algorithm. Allocation rule g_{greedy} is defined as:
  - Calculate norm value $v_i/|S_i|^{\frac{1}{2}}$ for agent i.
  - Sort agents by norm values in non-increasing order. Iterate through this sorted list, and assign a requested bundle to each of the agents if possible.

  Lehmann et al. [13] proves that g_{greedy} approximates the optimal allocation within a factor of $\sqrt{m}$, the lower limit of an approximation algorithm [8, 4].

  In addition, payment rule t_{greedy} to be coupled with the allocation rule is defined as follows. The price collected from allocated agent i is computed by determining the set of agents who would have been allocated, if agent i had not been part of the auction. Among these agents, pick the maximal norm c. Agent i pays $c \times |S_i|^{\frac{1}{2}}$. Lehmann et al. [13] proves that mechanism $(g_{\text{greedy}}, t_{\text{greedy}})$ is strategyproof.

  Hypothesis 3. *Payment rule $\bar{t}_{\mathbf{w}}$ derived from classifier $h_{\mathbf{w}}$ trained on greedy allocation rule g_{greedy} for a single-minded CA approximates greedy payment rule t_{greedy}.*

3.1.3 Multi-minded Combinatorial Auctions

The last setting in our experiments is the multi-minded combinatorial auction. In this setting, each agent is allowed to bid some constant number of bundles b, but can be granted up to one bundle. Note that if $b = 2^m$, we have the full version of combinatorial auctions; as such, multi-minded combinatorial auctions allow for very complex sets of strategies.

At the same time, multi-minded CAs have a few advantages over general CAs. First, multi-minded CAs allow for a concise representation of an agent's valuation. Instead of an explicit, exponential representation for every possible bundle, an agent's valuation in the multi-minded CA setting can be represented as $\mathbf{x}_i = \{(S_1, v_1), (S_2, v_2), \ldots, (S_b, v_b)\}$, where agent i values bundle $S_j \in Y$ at $v_j \in \mathbb{R}$, for every $j \in [1, b]$. Second, the inner product of two valuations can be computed in polynomial time, leading to efficient kernel computation.

Given distribution $P_{\mathbf{x}}$ and allocation rule g, training set $\{(\mathbf{x}^1, \mathbf{y}^1), \ldots, (\mathbf{x}^\ell, \mathbf{y}^\ell)\}$ can be generated. Then, the learning problem for the multi-minded CA case derived from (2.11) is:

$$\begin{aligned} \min_{\mathbf{w},\xi\geq 0} \ & \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{\ell}\sum_{k=1}^{\ell}\xi^k \qquad (3.7) \\ \text{s.t.} \ & w_1\big(\mathbf{x}_1^1[\mathbf{y}^1] - \mathbf{x}_1^1[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^1, \mathbf{y}^1) - \psi(\mathbf{x}_{-1}^1, \mathbf{y})\big) \geq \Delta(\mathbf{y}^1, \mathbf{y}) - \xi^1, \quad \forall \mathbf{y} \in Y^1 \\ & \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ & w_1\big(\mathbf{x}_1^\ell[\mathbf{y}^\ell] - \mathbf{x}_1^\ell[\mathbf{y}]\big) + \mathbf{w}_{-1}^T\big(\psi(\mathbf{x}_{-1}^\ell, \mathbf{y}^\ell) - \psi(\mathbf{x}_{-1}^\ell, \mathbf{y})\big) \geq \Delta(\mathbf{y}^\ell, \mathbf{y}) - \xi^\ell, \quad \forall \mathbf{y} \in Y^\ell \end{aligned}$$

where given that $\mathbf{x}_1^k = \{(S_1^k, v_1^k), (S_2^k, v_2^k), \ldots, (S_b^k, v_b^k)\}$, $Y^k = \{S_1^k, S_2^k, \ldots, S_b^k\}$ is the set of bundles that agent 1 is interested in. This presents another advantage of multi-minded CAs, as the number of constraints per one training instance is equal to b rather than 2^m.

Since the constraints of the learning problem exclude bundles that are outside of "target" bundles of agent 1, an important requirement for strategyproofness in this setting, as well as in the single-minded setting, is that for every bundle $\mathbf{y}' \notin Y^k$, there exists a bundle $\mathbf{y} \in Y^k$ such that $f_{\mathbf{w}}(\mathbf{x}^k, \mathbf{y}') \leq f_{\mathbf{w}}(\mathbf{x}^k, \mathbf{y})$. This requirement prevents agent 1 from deviating by reporting a bundle that is outside of its target bundles. When $\mathbf{y} \not\subseteq \mathbf{y}', \forall \mathbf{y} \in Y^k$, there is no incentive for agent 1 to deviate, as agent 1 would not gain any utility from $\mathbf{y}'$ and hence the empty zero, of which price is normalized to be zero, will certainly be weakly better. The case that is of particular concern is when $\mathbf{y} \subseteq \mathbf{y}'$, where $\mathbf{y} \in Y^k$ and $\mathbf{y}$ is the maximal such bundle in an agent's multi-minded set. (Such a bundle always exists since the empty bundle is in Y^k.) By definition of multi-minded, $\mathbf{x}_1[\mathbf{y}'] = \mathbf{x}_1[\mathbf{y}]$, so if the price of $\mathbf{y}'$ is lower than that of $\mathbf{y}$, agent 1 can improve its outcome by reporting $\mathbf{y}'$. It suffices to address this concern in terms of *admissibility*:

Definition 3.1.1 (Admissibility)**.** A discriminant function is *admissible* if $\mathbf{w}_{-1}^T\psi(\mathbf{x}, \mathbf{y}') \leq \mathbf{w}_{-1}^T\psi(\mathbf{x}, \mathbf{y})$, so that the payment is greater for every bundle $\mathbf{y}' \geq \mathbf{y}$, for any $\mathbf{x}$.

In addition to ensuring strategyproofness, admissibility also helps extend Theorem 4 to

the single-minded and multi-minded CA settings.

Corollary 1. *Given allocation rule g for the multi-minded CA problem, and admissible solution $(\mathbf{w}^*, \xi^*)$ to the structural SVM classification, the ex post regret on training example $(\mathbf{x}^k, \mathbf{y}^k)$ is bounded above by $\frac{1}{w_1^*}\xi^k$.*

Proof. Admissibility ensures that irrelevant bundles can be ignored, and hence:

$$\max_{\mathbf{y}' \in Y} \left[\frac{1}{w_1^*} f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') \right] = \max_{\mathbf{y}' \in Y^k} \left[\frac{1}{w_1^*} f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') \right].$$

From (3.7), we have that

$$\begin{aligned} \xi^k &= \max_{\mathbf{y}' \in Y^k} \left[\Delta(\mathbf{y}^k, \mathbf{y}') + f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') - f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}^k) \right] \\ &= \max_{\mathbf{y}' \in Y} \left[\Delta(\mathbf{y}^k, \mathbf{y}') + f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}') - f_{\mathbf{w}^*}(\mathbf{x}^k, \mathbf{y}^k) \right], \end{aligned}$$

where the second equation is derived from admissibility. The rest of the proof follows the proof of Theorem 4 □

Enforcing admissibility in the learning constraints is crucial as it is one of the requirements for strategyproofness. In this thesis, we iterate each of the learning constraints over Y instead of Y^k. This achieves the same effect as directly enforcing admissibility, at the expense of computational tractability of the model. Replacing the full iteration over all $\mathbf{y} \in Y$ with additional constraints on admissiblity is an important area of future work.

Value Distributions

In designing datasets for multi-minded auctions, two main concerns arise. First, we need to take into consideration complementarity and substitutability of different goods. An agent can directly express complementarity by the way the agent bids different bundles. The issue of substitutability is slightly more complicated. In regular formulations of CAs, an agent is allowed to receive more than one bundle. As the utility of a combined bundle of substitute

goods to the agent is lower than the sum of the utility of individual items, the problem of calculating total utility to the agent arises. Fujishima et al. [7] propose the use of *dummy goods* to enforce substitutability. In our case, since the formulation of our approach allows each agent up to one bundle, substitutability can easily be expressed in bids in a similar manner as complementarity. Second, we need to ensure *free disposal*, which means that if for any two bundles S_i, S_j requested by an agent and $S_i \subseteq S_j$, then the value of S_i must not be greater than that of S_j. This requirement helps ensure that all bundles requested by an agent are meaningful.

Lubin and Parkes [16] propose a modified version of the Sandholm distributions used in our single-minded CA experiments for their experiments with combinatorial exchanges. With slight modifications to their distributions, we get two distributions for multi-minded combinatorial auctions[1]:

- **Uniform**: This distribution is comparable to Sandholm's uniform distribution for single-minded CAs, and works as follows:
  - Assign a uniform random common value $0 < c(g) \leq 1$ to every good g, and a uniform random private value $0 < y_i(g) \leq 1$ to agent i. The value agent i places on good g is $w_i(g) = \beta c(g) + (1 - \beta)y_i(g)$, for some $\beta \in [0, 1]$.
  - Select the number of goods in each bid uniformly from $[1, m]$. (Each agent places b bids.)
  - For each bid, goods are drawn with a uniform distribution. The value of each bid B to agent i is $\left(\frac{\sum_{g \in B} w_i(g)}{m}\right)^{\zeta}$, for some ζ. Note that $\zeta > 1$ implies that goods are complements, whereas $\zeta < 1$ means that goods are substitutes. Also, the denominator normalizes all valuations to the $(0, 1]$ range.

[1]Note that for multi-minded CAs, the Combinatorial Auction Test Suite (CATS) provides five groups of benchmark distributions that are inspired by real-world scenarios, such as path auctions, and airport slot auctions [14]. Nevertheless, due to the nature of the CATS distributions, we cannot specify a fixed number of agents in an auction. As the current formulations of attribute vectors as explained in Section 3.2 do not support varying the number of agents within the same SVM classifier, we are unable to use CATS for our purpose.

  - Correct free disposal. For every (S_i, v_i) and (S_j, v_j) requested by an agent, if $S_i \subseteq S_j$ and $v_j < v_i$, then assign v_i to v_j.

- **Decay**: In a similar manner, the decay distribution here is intended to be compared to Sandholm's decay distribution for single-minded CAs. The procedure is similar to the uniform distribution for multi-minded CAs as mentioned above, except for the second step. Instead of using a uniform distribution to determine the number of goods in each bundle, the number of desired goods in the bundle is determined by starting with 1 and repeatedly incrementing the number until $rand(0, 1)$ exceeds α. Similar to the single-minded case, we fix α at its hardest value, 0.75.

Winner Determination Algorithms

For the multi-minded CA setting, we train and test models on two allocation rules:

- **Optimal**: The optimal allocation is computed by solving an integer linear programming problem to maximize social welfare. The formulation of the ILP problem is similar to (3.3). The corresponding strategyproof payment function is the VCG payment function, which can be solved through two ILP problems as explained in Section 3.1.2. Due to Proposition 1, exact classifier $h_{\mathbf{w}}$ trained on the optimal allocation rule is the VCG payment function.

  Hypothesis 4. *Payment rule $\bar{t}_{\mathbf{w}}$ derived from classifier $h_{\mathbf{w}}$ trained on the optimal allocation rule for a multi-minded CA approximates the VCG payment rule.*

- **Greedy**: To the best of our knowledge, there are no known greedy allocation rules and payment functions that are incentive compatible and work with multi-minded CAs. As such, we devise a greedy allocation rule from the one by Lehmann et al. [13] in Section 3.1.2. First, we treat all of the bids from different agents equally and rank the bids based on the norm $v_i/|S_i|^{\frac{1}{2}}$. Then, as we iterate through the sorted list

to assign requested bundles, we check that the agent who requests that bundle must have not been allocated any bundle yet, in addition to checking that the items in the bundle are not yet allocated. Since this allocation rule does not have a corresponding strategyproof payment function, we hope to get insights into how the model works with allocation rules without SP payment rules.

Parameters of Interest

In addition to basic parameters of the structural SVM, the multi-minded CA setting introduces two extra parameters: ζ and β. As mentioned above, ζ determines complementarity and substitutability of different goods. In our experiments, we use 0.5, 1, and 1.5 separately for ζ. β represents how correlated agents' valuations of items are across agents, with 1 being perfectly correlated. We experiment with 0.1, 0.3, 0.5, 0.7, and 0.9 as values of β.

3.2 Feature Map

Before the framework presented in the previous chapter would actually work in practice, we need to specify its last component—feature map $\psi(\mathbf{x}_{-1}, \mathbf{y})$. As mentioned in Section 2.3.1, the role of the feature map is to map $\mathbf{x}_{-1}$ (values of all agents other than agent 1) and $\mathbf{y}$ (allocation to agent 1) into a higher dimensional space and in the process define the hypothesis space for price functions over values and bundles. The feature map can be decomposed into two components: an attribute vector (χ) and a feature expansion (ϕ) that corresponds to a Kernel (K) in the dual space:

$$\psi(\mathbf{x}_{-1}, \mathbf{y}) = \phi(\chi(\mathbf{x}_{-1}, \mathbf{y})) \tag{3.8}$$

The attribute vector (χ) combines $\mathbf{x}_{-1}$ and $\mathbf{y}$ into one vector, and the kernel (ϕ) transforms the vector into a higher dimensional space. Choosing appropriate attribute vectors and kernels is both an art and science, and two kinds of attribute vectors are tested in this thesis.

3.2.1 Attribute Vectors

As previously mentioned, attribute vectors combine $\mathbf{x}_{-1}$ and $\mathbf{y}$ into one vector. There are many ways to combine the two components, but in this thesis, we experiment with two kinds.

The first attribute vector (χ_1) separates $\mathbf{x}_{-1}$ based on the $\mathbf{y}$ value. The vector is a concatenated vector of zero vectors and the $\mathbf{x}_{-1}$ vector, with $\mathbf{x}_{-1}$ at the $\mathbf{y}^{\text{th}}$ slot of the vector:

$$\chi_1(\mathbf{x}_{-1}, \mathbf{y}) = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \dots \\ \mathbf{x}_{-1} \\ \dots \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{3.9}$$

This vector essentially assumes that there is zero interaction between different $\mathbf{y}$ values. In other words, knowledge gained from one $\mathbf{y}$ value does not contribute anything to other $\mathbf{y}$ values. This is quite a strong assumption, as there should be symmetry among different $\mathbf{y}$ values. Since this attribute vector uses only data from one $\mathbf{y}$ value to learn the model for that particular $\mathbf{y}$, this vector would likely require more training data.

The second attribute vector (χ_2) that we experiment with allows for more interaction between $\mathbf{x}_{-1}$ and $\mathbf{y}$:

$$\chi_2(\mathbf{x}_{-1}, \mathbf{y}) = \begin{bmatrix} \mathbf{x}_2 \setminus \mathbf{y} \\ \mathbf{x}_3 \setminus \mathbf{y} \\ \dots \\ \mathbf{x}_n \setminus \mathbf{y} \end{bmatrix} \tag{3.10}$$

where $\mathbf{x}_j \setminus \mathbf{y}$ is the projection of the valuation of agent j onto a good space without bundle $\mathbf{y}$ being available. For example, if $\mathbf{x}_j = \{(\{A, B\}, \$5), (\{B, C\}, \$4)\}$ and $\mathbf{y} = \{A, D\}$, then

$\mathbf{x}_j \setminus \mathbf{y} = \{(\{A, B\}, \$0), (\{B, C\}, \$4)\}$, since A is no longer available to agent j. This is a reasonable attribute vector, as $\mathbf{y}$ is being allocated to agent 1 and not available to other agents. At the same time, the effects of allocating $\mathbf{y}$ to agent 1 on other agents should determine whether agent 1 will be allocated $\mathbf{y}$.

It is important to note that both attribute vectors only work with a constant number of agents. This property limits the kind of datasets we can work with, and requires that we train a separate model for every number of agents. Designing an attribute vector that does not depend on the number of agents is an important area of future work.

3.2.2 Kernel

Fundamentally, an SVM is a linear classifier, whereas in many cases, datasets are not linearly separable. One common solution to the problem is to map the data in the original space onto a higher dimensional space in such a way that preserves relevant dimensions of data points, before applying a linear separator in that space. We can achieve this by using a feature expansion $\phi(\mathbf{x})$ to map $\mathbf{x}$ into a higher dimensional space, except that ϕ is not always efficient or possible to directly compute. Instead, we work in the dual space, where only the inner product of two attribute vectors is required for computation. The idea is to replace the inner product with a kernel function $K(\mathbf{x}, \mathbf{x}')$, which corresponds to some ϕ in the primal space. This technique is referred to as "the kernel trick" [17].

Initially, two kernels are considered: polynomial kernel and Gaussian *radial basis function* (RBF). The polynomial is defined as follows:

$$K_{\text{polynomial}}(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^p, \tag{3.11}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are two attribute vectors and $p \in \mathbb{Z}^+$ is the degree of the polynomial. This kernel function corresponds to a polynomial feature expansion in the primal space, with all polynomial terms up to the p^{th}degree.

The RBF kernel is defined as:

$$K_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right), \tag{3.12}$$

where $\sigma \in \mathbb{R}^+$ [24]. In the context of this thesis, we use $\gamma = \frac{1}{2\sigma^2}$ for notational simplicity and consistency with the SVM-Struct package used in our experiments. Simplifying the RBF formula, we get that:

$$K_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = exp\left(-\gamma\left(\|\mathbf{x}\|^2 + \|\mathbf{x}'\|^2 - 2\mathbf{x}^T\mathbf{x}'\right)\right), \tag{3.13}$$

where $\gamma \in \mathbb{R}^+$. The feature map corresponding to the RBF kernel is infinite dimensional. In our preliminary tests, RBF performs significantly better than the polynomial one. Due to limited computational resources, we choose to continue the experiments with only RBF as the kernel.

3.2.3 Inner Product of Attribute Vectors

From Equations 3.11 and 3.13, we can see that both polynomial and RBF kernels require inner products for computation. As such, an important requirement of attribute vectors is that there is a way to efficiently compute the inner product of two vectors.

For the first attribute vector (χ_1), we can calculate the inner product as follows:

$$\left\langle \chi_1(\mathbf{x}_{-1}, \mathbf{y}), \chi_1(\mathbf{x}'_{-1}, \mathbf{y}') \right\rangle = \mathbf{I}_{\mathbf{y}=\mathbf{y}'} \sum_{i=2}^{n} \left\langle \mathbf{x}_i, \mathbf{x}'_i \right\rangle, \tag{3.14}$$

where $\mathbf{I}_p = 1$ if p is true and zero otherwise.

The process for the second attribute vector (χ_2) is as follows:

$$\left\langle \chi_2(\mathbf{x}_{-1}, \mathbf{y}), \chi_2(\mathbf{x}'_{-1}, \mathbf{y}') \right\rangle = \sum_{i=2}^{n} \left\langle \mathbf{x}_i \setminus \mathbf{y}, \mathbf{x}'_i \setminus \mathbf{y} \right\rangle \tag{3.15}$$

The next step is to compute $\langle \mathbf{x}_i, \mathbf{x}'_i \rangle$, which is required for computing the inner products of both of the attribute vectors. (The calculation of $\langle \mathbf{x}_i \setminus \mathbf{y}, \mathbf{x}'_i \setminus \mathbf{y} \rangle$ can be done in a similar manner as that of $\langle \mathbf{x}_i, \mathbf{x}'_i \rangle$.) First, we consider the single-minded setting, which is a special

case of the formula for the multi-minded case. Let $\mathbf{x}_i$ corresponds to a set S_i valued at v_i, and $\mathbf{x}'_i$ to a set S'_i valued at v'_i. For each subset S of all m goods, if $S_i \subseteq S$ and $S'_i \subseteq S$, then $v_i v'_i$ contributes to $\mathbf{x}_i^T \mathbf{x}'_i$. Since there are $2^{m-|S_i \cup S'_i|}$ sets that satisfy this condition, we have:

$$\mathbf{x}_i^T \mathbf{x}'_i = v_i v'_i 2^{m-|S_i \cup S'_i|}. \tag{3.16}$$

The general formula for the multi-minded case is as follows. The proof of the lemma can be found in the appendix.

Lemma 1. *Consider a multi-minded CA and two bid vectors $\mathbf{x}_i$ and $\mathbf{x}'_i$ corresponding to sets $S = \{S_1, \ldots, S_s\}$ and $S' = \{S'_1, \ldots, S'_t\}$ with associated values $v_1, \ldots, v_s$ and $v'_1, \ldots, v'_t$. Then,*

$$\mathbf{x}_i^T \mathbf{x}'_i = \sum_{T \subseteq S, T' \subseteq S'} \Big((-1)^{|T|+|T'|} \cdot (\min_{S_k \in T} v_k) \cdot (\min_{S'_j \in T'} v'_j) \cdot 2^{m-|(\bigcup_{S_k \in T} S_k) \cup (\bigcup_{S'_j \in T'} S'_j)|} \Big). \tag{3.17}$$

Proof. The contribution of a particular bundle U of items to the inner product is equal to the product of $\mathbf{x}_i$'s valuation of U and $\mathbf{x}'_i$'s valuation of U, or $(\mathbf{x}_i[U] \cdot \mathbf{x}'_i[U])$. In a multi-minded CA, $\mathbf{x}_i[U] = \max_{S_k \in S, S_k \subseteq U} v_k$. Thus, the inner product is equal to

$$\mathbf{x}_i^T \mathbf{x}'_i = \sum_{U \in Y} \Big((\max_{\substack{S_k \in S \\ S_k \subseteq U}} v_k) \cdot (\max_{\substack{S'_j \in S' \\ S'_j \subseteq U}} v'_j) \Big).$$

The maximum-minimums identity (see e.g. [21]) states that for any set $\{x_1, \ldots, x_n\}$ of n numbers, $\max\{x_1, \ldots, x_n\} = \sum_{Z \subseteq X} ((-1)^{|Z|+1} \cdot (\min_{x_i \in Z} x_i))$. We can rewrite the two terms in the previous equations as:

$$\max_{\substack{S_k \in S \\ S_k \subseteq U}} v_k = \sum_{\substack{T \subseteq S \\ \bigcup_{S_k \in T} S_i \subseteq U}} \Big((-1)^{|T|+1} \cdot (\min_{S_k \in T} v_k) \Big) \quad \text{and} \quad \max_{\substack{S'_j \in S' \\ S'_j \subseteq U}} v'_j = \sum_{\substack{T' \subseteq S' \\ \bigcup_{S'_j \in T'} S'_j \subseteq U}} \Big((-1)^{|T'|+1} \cdot (\min_{S'_j \in T'} v'_j) \Big).$$

The inner product can thus be written as

$$\mathbf{x}_i^T \mathbf{x}'_i = \sum_{U \in Y} \sum_{\substack{T \subseteq S, T' \subseteq S' \\ \bigcup_{S_k \in T} S_k \subseteq U \\ \bigcup_{S'_j \in T'} S'_j \subseteq U}} \Big((-1)^{|T|+|T'|} \cdot (\min_{S_k \in T} v_k) \cdot (\min_{S'_j \in T'} v'_j) \Big).$$

Lastly, for given $T \subseteq S$ and $T' \subseteq S'$, there are exactly $2^{m-|(\bigcup_{S_k \in T} S_k) \cup (\bigcup_{S'_j \in T'} S'_j)|}$ bundles U such that $\bigcup_{S_k \in T} S_k \subseteq U$ and $\bigcup_{S'_j \in T'} S'_j \subseteq U$, and we obtain

$$\mathbf{x}_i^T \mathbf{x}'_i = \sum_{T \subseteq S, T' \subseteq S'} \Big((-1)^{|T|+|T'|} \cdot (\min_{S_k \in T} v_k) \cdot (\min_{S'_j \in T'} v'_j) \cdot 2^{m-|(\bigcup_{S_k \in T} S_k) \cup (\bigcup_{S'_j \in T'} S'_j)|} \Big). \quad \square$$

If S and S' have constant size as in the single-minded and multi-minded CA cases, then the sum on the right hand side of (3.17) ranges over a constant number of sets and can be computed efficiently.

3.3 Logistics of the Experiments

3.3.1 Practical Details of the Models

The *SVM-Struct* package is used to run all of the experiments. The package provides functionality to train and test a structural SVM model as described by Joachims et al. [9] and as summarized in Section 2.2. We use the margin-rescaling implementation within SVM-Struct. Note that we currently use the default method to enumerate the most preferred bundle of an agent, which iterates through all of the possible bundles. As the number of bundles is exponential in the number of goods, this is likely to become intractable. Finding a more efficient method to search over bundles is another important area of future work.

Using SVM-Struct

Using the provided APIs, we specified three main components of the package: loss, attribute vector, and kernel functions. The loss function is specified as:

$$\Delta_{\mathbf{x}}(\mathbf{y}, \mathbf{y}') = \mathbf{I}_{\mathbf{y} = \mathbf{y}'}$$

The attribute vector and kernel functions are defined as in Section 3.2. Nevertheless, one more step is required before the kernel becomes fully functional. Currently, the kernel

function as described in Section 3.2 takes in $\chi(\mathbf{x}_{-1}, \mathbf{y})$, meaning that we have to find a way to incorporate $\mathbf{x}_1[\mathbf{y}]$ into the kernel function in such a way that $\mathbf{x}_1[\mathbf{y}]$ would remain linear in the kernel formula.[2] This is rather a trivial matter, since in general if there is kernel function K in the dual space that corresponds to feature expansion $\phi(\mathbf{w})$ in the primal space, then a kernel function for $[z, \phi(\mathbf{w})]$ would be just $z * z' + K(\mathbf{w}, \mathbf{w}')$.

In our case, we have standard RBF kernel $K(\chi(\mathbf{x}_{-1}, \mathbf{y}), \chi(\mathbf{x}'_{-1}, \mathbf{y}'))$, which corresponds to $\phi(\chi(\mathbf{x}_{-1}, \mathbf{y}))$ in the primal space. The new kernel function that incorporates $\mathbf{x}_1$ would be:

$$K'((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) = \mathbf{x}_1[\mathbf{y}] * \mathbf{x}'_1[\mathbf{y}'] + K(\chi(\mathbf{x}_{-1}, \mathbf{y}), \chi(\mathbf{x}'_{-1}, \mathbf{y}')). \tag{3.18}$$

Computing Payments from Dual Formula

Since we work in the dual space, we have to be able to compute payments from the dual formula of a trained classifier. After running the learning algorithm, we get a classifier written in the dual form as follows (SV refers to the set of support vectors):

$$\begin{aligned} y(\mathbf{x}) &= \arg\max_{\mathbf{y} \in Y} f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) \\ &= \arg\max_{\mathbf{y} \in Y} \sum_{t \in SV} \alpha_t \mathbf{y}_t K'((\mathbf{x}_t, \mathbf{y}_t), (\mathbf{x}, \mathbf{y})) + \theta_0 \end{aligned} \tag{3.19}$$

Considering some $\mathbf{y}^*, \mathbf{y}^* \in Y$, the discriminant function for the $\mathbf{y}^*$ value is as follows:

$$f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}^*) = \sum_{t \in SV} \alpha_t \mathbf{y}_t K'((\mathbf{x}_t, \mathbf{y}_t), (\mathbf{x}, \mathbf{y}^*)) + \theta_0 \tag{3.20}$$

From Equation 3.18, we replace K' with K:

$$\begin{aligned} f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}^*) &= \sum_{t \in SV} \alpha_t \mathbf{y}_t \left(\mathbf{x}_{t,1}[\mathbf{y}_t] * \mathbf{x}_1[\mathbf{y}^*] + K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*)) \right) + \theta_0 \\ &= \left(\sum_{t \in SV} \alpha_t \mathbf{y}_t \mathbf{x}_{t,1}[\mathbf{y}_t] \right) \mathbf{x}_1[\mathbf{y}^*] + \sum_{t \in SV} \alpha_t \mathbf{y}_t K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*)) + \theta_0 \end{aligned}$$

[2] Recall from Equation 2.10 in Section 2.3.1 that the discrimination function in use is $f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) = w_1 \mathbf{x}_1[\mathbf{y}] + \mathbf{w}_{-1}^T \psi(\mathbf{x}_{-1}, \mathbf{y})$.

From Equation 2.10, $f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}^*) = \mathbf{w}_1\mathbf{x}_1[\mathbf{y}^*] + \mathbf{w}_{-1}^T\psi(\mathbf{x}_{-1}, \mathbf{y}^*)$:

$$\begin{aligned}
&\mathbf{w}_1\mathbf{x}_1[\mathbf{y}^*] + \mathbf{w}_{-1}^T\psi(\mathbf{x}_{-1}, \mathbf{y}^*) \\
&\qquad = \left(\sum_{t\in SV} \alpha_t \mathbf{y}_t \mathbf{x}_{t,1}\right)\mathbf{x}_1[\mathbf{y}^*] + \sum_{t\in SV} \alpha_t \mathbf{y}_t K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*)) + \theta_0
\end{aligned}$$

As such, we get the two following relationships:

$$w_1 = \sum_{t\in SV} \alpha_t \mathbf{y}_t \mathbf{x}_{t,1}[\mathbf{y}_t] \tag{3.21}$$

$$\mathbf{w}_{-1}^T\psi(\mathbf{x}_{-1}, \mathbf{y}^*) = \sum_{t\in SV} \alpha_t \mathbf{y}_t K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*)) + \theta_0$$

From Equation 2.12, the unnormalized payment rule is: $t_{\mathbf{w}}(\mathbf{x}_{-1}, \mathbf{y}) = -\frac{\mathbf{w}_{-1}^T}{w_1}\psi(\mathbf{x}_{-1}, \mathbf{y}^*)$. Hence, in the dual form, the unnormalized payment to agent 1 given that the agent is allocated $\mathbf{y}^*$ is:

$$t_{\mathbf{w}}(\mathbf{x}_{-1}, \mathbf{y}^*) = -\frac{\sum_{t\in SV} \alpha_t \mathbf{y}_t K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*))}{w_1} - \frac{\theta_0}{w_1} \tag{3.22}$$

The normalized payment rule can be derived directly from Equation 2.13:

$$\bar{t}_{\mathbf{w}}(\mathbf{x}_{-1}, \mathbf{y}^*) = -\frac{\sum_{t\in SV} \alpha_t \mathbf{y}_t \left[K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{y}^*)) - K(\chi(\mathbf{x}_{t,-1}, \mathbf{y}_t), \chi(\mathbf{x}_{-1}, \mathbf{0}))\right]}{w_1} \tag{3.23}$$

3.3.2 Instance-based Normalization

In addition, we test an optimization technique called "instance-based normalization," in hopes that it will help improve performances of learned hypotheses. Specifically, we normalize each instance of training and test sets so that the value of the highest bid except for agent 1's bids within each individual instance is at exactly 1, before passing it to the learning algorithm or classifier. Then, we de-normalize the values and solutions back to the former scale afterwards.

3.3.3 Training, Tuning, Testing

For each setting, two parameters need to be tuned. The first parameter is the C value of the optimization problem. As part of the objective function of the learning problems, C is the trade-off between empirical regret and regularization. The second parameter that needs to be adjusted for each model and each dataset is the γ value of the RBF kernel. After some preliminary testing, $(C, \gamma) = \{10^4, 10^5\} \times \{10^{-2}, 10^{-1}, 1\}$ is used during the tuning process.

In each setting, three training sets and three validation sets are generated. 200 training instances of $(\mathbf{x}^j, \mathbf{y}^j)$ are initially generated for each training set, but then for each of the 200 instances, we derive n instances by switching the order of agents in such a way that each of the n agents gets to be ahead of the list and hence be agent 1 exactly once. We then order the other agents on the list randomly. Hence, each training set contains $200 \times n$ instances. In a similar manner, each validation set contains $1000 \times n$ instances. We use these validation sets to tune C and γ. Finally, we generate an additional "golden pair" of training and test sets for each of the settings, and report performances based on these golden pairs.

3.3.4 Metrics

In this thesis, we employ four kinds of metrics to measure performances of the learned hypotheses and to generate insights into the outputs of the models:

0/1 Classification Accuracy

0/1 classification accuracy refers to the accuracy of the trained model in predicting the bundle allocated to agent 1 of each instance in the golden test set. Specifically, with ℓ instances in the test set, the formula of *0/1 classification accuracy* is:

$$\frac{\sum_{k=1}^{\ell} \mathbf{I}(h_{\mathbf{w}}(\mathbf{x}^k) = \mathbf{y}^k)}{\ell} \tag{3.24}$$

Even though this metric might not yield much insight into how the learned hypothesis performs as a strategyproof payment rule, the fact that it is a standard way to directly measure the performance of an SVM leads us to use it as the main metric in tuning parameters.

Ex post Regret

Ex post regret is a good measure of whether the payment rule obtained from the trained hypothesis is strategyproof. Specifically, an SP payment rule should introduce zero regret to participating agents. Let $\mathbf{y} = g(\mathbf{x}_1, \mathbf{x}_{-1})$ be the bundle allocated to agent 1. If classifier $h_{\mathbf{w}}$ correctly predicts a bundle $\mathbf{y}$, the agent experiences no regret. On the other hand, if the model predicts $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{y}' \neq \mathbf{y}$, then according to the payment rule, the agent would be better off with $\mathbf{y}'$ than with $\mathbf{y}$. As such, the agent experiences regret that is equal to the difference in utility surplus from being allocated $\mathbf{y}'$ instead of $\mathbf{y}$. The regret for an individual agent is measured as follows:

$$\mathbf{x}_1[\mathbf{y}'] - t_{\tilde{\mathbf{w}}}(\mathbf{x}, \mathbf{y}') - (\mathbf{x}_1[\mathbf{y}] - t_{\tilde{\mathbf{w}}}(\mathbf{x}, \mathbf{y})) \tag{3.25}$$

In this project, we employ two metrics to measure the expected ex post regret of the trained models. The first metric is *average ex post regret*, which is the arithmetic mean of ex post regret experienced by all of the agents in the test set:

$$\frac{\sum_{k=1}^{\ell} \mathbf{x}_1^k[\mathbf{y}'] - t_{\tilde{\mathbf{w}}}(\mathbf{x}, \mathbf{y}') - (\mathbf{x}_1^k[\mathbf{y}] - t_{\tilde{\mathbf{w}}}(\mathbf{x}, \mathbf{y}))}{\ell} \tag{3.26}$$

Nevertheless, as Lubin and Parkes [16] point out, examining the distribution of regret across all of the agents can yield insights into different mechanisms. As such, the second metric used is the *95^{th}percentile ex post regret*. This metric looks at the 95^{th}percentile regret across all of the agents in the test set instead of the average regret.

Payment Difference

All of the auction settings in this thesis except for the multi-minded CA with a greedy allocation rule have corresponding strategyproof payment rules. In these auctions, we can compare the payments predicted by the trained hypotheses to the actual payments as computed from the payment rules associated with the winner determination algorithms. The primary metric we use for reporting payment difference is the *root mean square error* (RMSE) over all allocated agents. Let $p(\mathbf{x}_{-1}, \mathbf{y})$ denote the strategyproof payment, and A the subset of test instances with agent 1's being allocated. The payment RMSE metric is defined as:

$$\sqrt{\frac{\sum_{k \in A}(\bar{t}_{\mathbf{w}}(\mathbf{x}_1^k, \mathbf{x}_{-1}^k, \mathbf{y}^k) - p(\mathbf{x}_{-1}^k, \mathbf{y}^k))^2}{|A|}} \tag{3.27}$$

In addition, we also employ two secondary metrics that help facilitate the process of interpreting the data. The first of the two metrics is *average payment difference*, which is the arithmetic mean of payment error:

$$\frac{\sum_{k \in A}(\bar{t}_{\mathbf{w}}(\mathbf{x}_1^k, \mathbf{x}_{-1}^k, \mathbf{y}^k) - p(\mathbf{x}_{-1}^k, \mathbf{y}^k))}{|A|} \tag{3.28}$$

The second metric is *average absolute payment difference*, which is defined as:

$$\frac{\sum_{k \in A}\left|\bar{t}_{\mathbf{w}}(\mathbf{x}_1^k, \mathbf{x}_{-1}^k, \mathbf{y}^k) - p(\mathbf{x}_{-1}^k, \mathbf{y}^k)\right|}{|A|} \tag{3.29}$$

Individual Rationality Violation

The last metric that is monitored during testing is the *individual rationality violation* rate. Individual rationality (IR) is violated when an agent is forced to pay more than the value of items allocated to the agent. Note that the normalized payment rule $\bar{t}_{\mathbf{w}}$ is guaranteed to be IR only if classifier $h_{\mathbf{w}}$ is exact. The IR violation rate is measured as:

$$\frac{\sum_{k=1}^{\ell} \mathbf{I}(t_{\tilde{\mathbf{w}}}(\mathbf{x}^k, \mathbf{y}^k) > \mathbf{x}_1^k[\mathbf{y}^k])}{\ell} \tag{3.30}$$

Chapter 4

Experimental Results

In this chapter, I present the results of the experiments described in Section 3.1 of the previous chapter. To reiterate the goals of the experiments, we are interested in validating the framework and specifically the four hypotheses presented there, and in understanding how various components of the model and experimental settings affect performance in practice. Specifically, in each experimental setting, we investigate strategyproofness of learned payment rules by examining *ex post regret*, benchmark the learned rules against well-known strategyproof rules, and study how performance is affected by attribute vectors, value distributions, normalization, and other elements of the experiments.

This chapter is divided into four main sections to reflect three auction settings with strategyproof payment rules and one setting without a strategyproof payment rule. In each section, I summarize major results and trends, before discussing and interpreting the results.

4.1 Single-item Auctions

In the single-item auction setting, we test the two types of attribute vectors (χ_1 and χ_2) as defined in Section 3.2.1, and with and without an optimization technique of normalizing individual instances as explained in Section 3.3.2.

Agents	0/1Accuracy		Regret		99Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	97.3	99.5	0.0005	0.0000	0.0226	0.0000	0.023	0.003	2.70	0.30
3	97.2	96.8	0.0005	0.0006	0.0197	0.0255	0.030	0.034	1.13	1.57
4	95.5	95.6	0.0016	0.0020	0.0559	0.0722	0.052	0.053	2.40	3.40
5	95.1	95.3	0.0020	0.0018	0.0679	0.0594	0.059	0.064	2.32	3.00
6	94.9	94.9	0.0022	0.0023	0.0736	0.0727	0.067	0.069	2.73	3.30

Agents	0/1Accuracy		Regret		99Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	96.8	99.0	0.0007	0.0001	0.0276	0.0001	0.026	0.007	3.25	0.00
3	95.4	97.1	0.0017	0.0006	0.0584	0.0253	0.046	0.027	3.27	2.73
4	96.6	95.8	0.0012	0.0020	0.0440	0.0663	0.039	0.055	0.78	1.82
5	95.0	95.4	0.0022	0.0023	0.0715	0.0757	0.058	0.057	2.94	0.92
6	96.7	95.5	0.0011	0.0018	0.0377	0.0612	0.044	0.055	1.33	1.92

Table 4.1: Performance of the two attribute vectors in single item auctions. (Top: without normalization, Bottom: with instance-based normalization.)

4.1.1 Main Results

Table 4.1 presents the five primary performance metrics. Ranging from mid to high 90s, 0/1 classification accuracies ("0/1 Accuracy") reflect the high quality of the learning algorithm in the single item auction setting. In addition, classification accuracy highly correlates with average ex post regret ("Regret"), 99$^{\text{th}}$percentile ex post regret ("99Regret")[1], and payment root mean square error ("PriceRMSE"), which together determine the quality of payment rules. The correlation between the first four metrics implies that better learned hypotheses are also better payment rules. IR Violation rates ("IR Violate"), on the other hand, are out of sync with the other metrics. We tackle this issue in the subsequent discussion section.

Considering that agent valuations are uniformly drawn from $(0, 1]$, average ex post regret is very low, at roughly 0.2% of the valuation range. Recall Equation 2.20 within the proof of Theorem 3, regret arises only when the instance is misclassified, so it is not surprising that

[1]In the single item setting, with classification accuracies above 95% in most cases, the 95$^{\text{th}}$percentile ex post regret is zero in most cases. Therefore, we report the 99$^{\text{th}}$percentile regret instead.

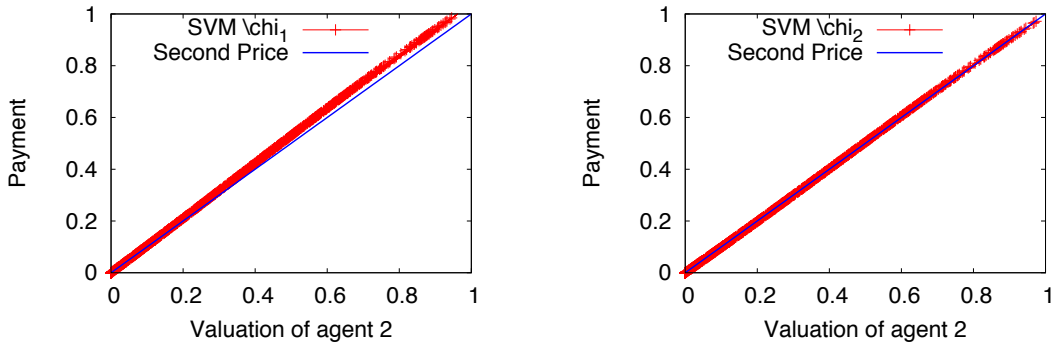

Figure 4.1: Agent 1's predicted payments vs. second price payments in 2-agent single item auctions, without instance-based normalization. (Left: χ_1, Right: χ_2)

average ex post regret is very low given that misclassification rates for all cases are less than 5%. The 99$^{\text{th}}$percentile ex post regret, on the other hand, shows that when a hypothesis misclassifies, the agent can experience a normalized regret of up to 7%. Although the 7% ex post regret might seem high, it is important to note that the 99$^{\text{th}}$percentile metric essentially acts as the upper limit of ex post regret an agent could expect to experience. Therefore, the learned payment rules in the single item setting perform well with respect to strategyproofness.

Now, we investigate further into the type of payment rules obtained from the framework. Hypothesis 1 predicts that payment rule $\bar{t}_{\mathbf{w}}$ from classifier $h_{\mathbf{w}}$ that is trained on the optimal allocation rule for a single item auction approximates the second price payment rule. Back to Table 4.1, payment RMSE comparing predicted payments from $\bar{t}_{\mathbf{w}}$ with second price payments range between 0.01 and 0.06. Later in Table 4.2, we observe that average absolute payment differences ("Average Absolute Payment Diff") are between 0.01 and 0.05 for all single item auction settings. The absolute payment diff metric implies that on average, predicted payments are about $1-5\%$ of agents' valuation range off second price payments, providing strong evidence in support of the similarity between the learned payment rules and the second price rule. To further visualize our payment rules, Figure 4.1 plots the

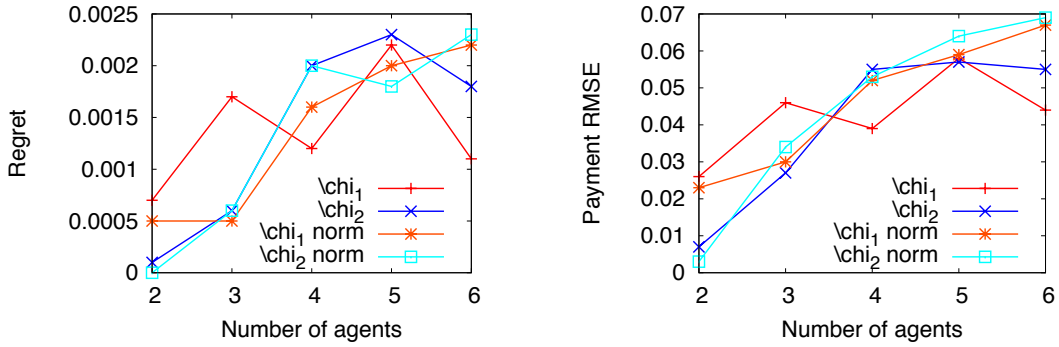

Figure 4.2: Regret and Payment RMSE trends for both attribute vectors, and with and without instance-based normalization in single item auctions.

predicted payments against the second price payments in the setting with two agents. Note that the second price payment rule in this scenario is merely the valuation of agent 2. As seen in both graphs, our models approximate the second price payment rule very well. Taken together, the empirical data provides strong support for Hypothesis 1.

Lastly, we examine how various components affect performance. Figure 4.2 illustrates the overall trends of average ex post regret and payment RMSE, as the number of agents increases. The first observation is that performance suffers as the number of agents increases. For instance, average ex post regret increases from the $[0.0000, 0.0007]$ range for 2 agents to the $[0.0011, 0.0018]$ range for 6 agents. We hypothesize that the decrease in performance is because we do not tune models along the dimension of the size of a training set. In addition, the two graphs also compare the performance of the two attribute vectors with and without normalizing individual instances ("norm"). With respect to attribute vectors, χ_2 performs better than χ_1 in the 2-agent setting, but with more agents, the two attribute vectors yield comparable results.

The effect of normalization is more volatile. Figure 4.2 shows that the performance of normalized models relative to those of non-normalized models tend to be unpredictable. Plotting predicted payments against "correct" second-price payments, Figure 4.3 offers

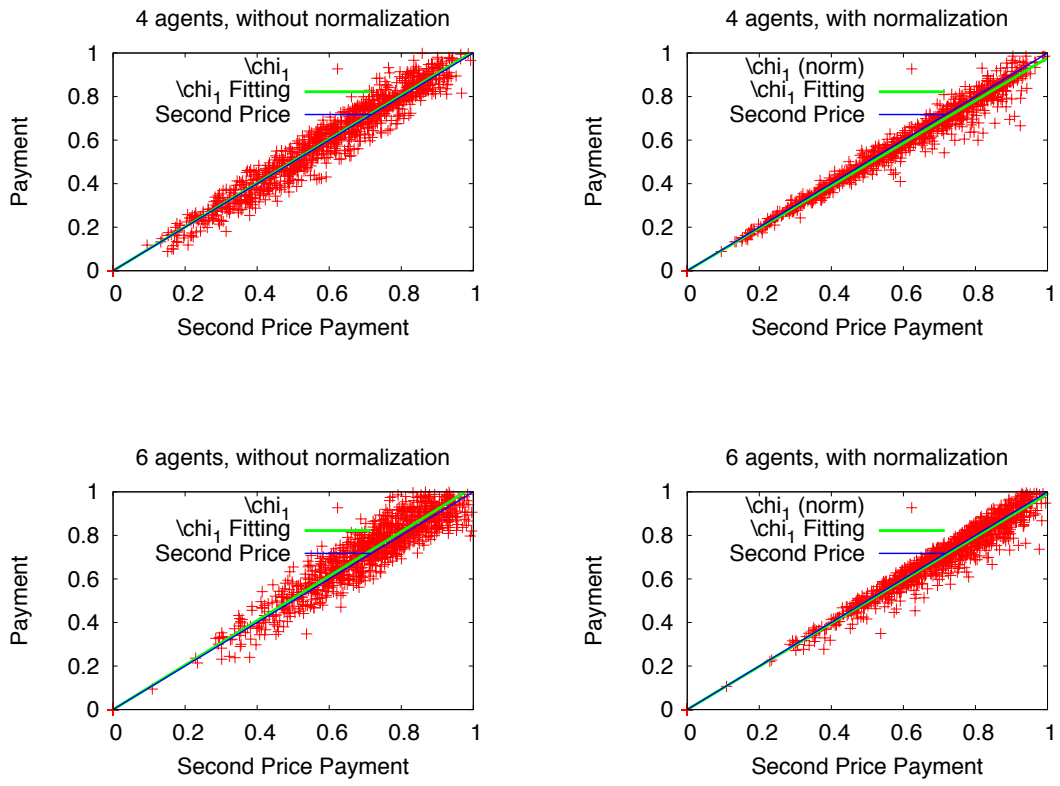

Figure 4.3: Predicted vs. second price payments in single item auctions, using χ_1. (Left: without normalization, Right: with normalization, Top: 4 agents, Bottom: 6 agents.)

a closer look of how instance-based normalization affects performance. With normalized models, predicted payments tend to concentrate better along the second price payment line. In addition, whereas in the non-normalized cases, payment errors are generally independent of correct payments, the range of payment errors in normalized cases is directly proportional to correct payments, and hence to the highest valuation of the other agents ($\max_{i \neq 1} \mathbf{x}_i$); i.e., lower valuations of other agents lead to lower predicted payment errors. This observation implies that the normalized models return the same range of outputs for all of the normalized test instances, and once de-normalized, these instances are in proportion to the original output range.

Agents	Average Payment Diff				Average Absolute Payment Diff			
	χ_1	χ_2	χ_1 norm	χ_2 norm	χ_1	χ_2	χ_1 norm	χ_2 norm
2	0.0186	0.0011	0.0211	-0.0058*	0.0186	0.0024	0.0211	0.0058
3	-0.0002	0.0057	0.0119	0.0201	0.0235	0.0272	0.0369	0.0222
4	0.0021	0.0214	-0.0133*	-0.0021*	0.0416	0.0393	0.0256	0.0399
5	0.0305	0.0338	0.0127	-0.0193*	0.0482	0.0518	0.0456	0.0386
6	0.0164	0.0184	-0.0069*	-0.0029*	0.0532	0.0541	0.0321	0.0447

Table 4.2: Average payment difference across all settings of single item auctions. (* is when a normalized model have a lower IR violation rate that its non-normalized counterpart of the same attribute vector.)

4.1.2 Discussion

Taken together, these numbers provide strong initial evidence in favor of the proposed framework. Classification accuracies are high, while ex post regret and payment difference RMSE are low. Not only do the models in the single item auction setting perform well in terms of strategyproofness, but they also approximate the second price payment rule very well. However, there are two issues that still require more investigation. The first issue is the individual rationality violation rate, and the second issue is possible, structural biases in our payment functions.

Table 4.2 presents average payment difference ("Average Payment Diff") and average absolute payment difference ("Average Absolute Payment Diff") in all of the settings. The first thing we notice is the relationship between IR violation rates and average payment difference. Specifically, holding the number of agents and the type of attribute vector constant, a higher average payment difference perfectly translates into a higher IR violation rate. In all of the six settings that normalized models enjoy lower IR violation rates than their non-normalized counterparts (marked with *), the normalized models also enjoy lower average payment difference. For comparison, in the other settings, all of the normalized models have higher average payment difference. As average payment diff implies the direction of payment bias of a learned model, lower average payment difference suggests a lower

positive bias in predicted payments. The fact that IR is violated if and only if a payment is incorrectly predicted to be above agent 1's valuation implies that the IR violation rate of a payment function is proportionate to the percentage of test instances that the payment function overestimates prices for. Therefore, even though a non-biased payment rule does not imply zero IR violation, a more negatively biased payment rule would likely enjoy a lower IR violation rate. Therefore, we have reduced the question of the volatility of IR violation to the issue of biases in our payment functions.

As for the issue of payment biases, two trends emerge from Table 4.2. First, nine out of the ten non-normalized models are positively biased (i.e., having positive average payment difference). Second, seven out of the ten normalized models are less positively biased than their non-normalized counterparts (i.e., having lower average payment difference). We believe that both trends possibly reflect structural biases within the experimental framework, either in the training model or the value distribution. This issue will require further investigation.

4.2 Single-minded Combinatorial Auctions

In the single-minded CA setting, we continue to experiment with both χ_1 and χ_2, and with and without normalization. The number of items is fixed at 5, and the number of agents is varied between 2 and 6. As mentioned in Section 3.1.2, we use two value distributions: uniform and decay, and two allocation rules: optimal and greedy.

4.2.1 Main Results

Tables 4.3 and 4.4 show the five primary performance metrics under optimal and greedy allocations. Only results from normalized models are presented here, since the results from both types of models are comparable. The effect of normalization will be discussed toward the end of this section. Across all auction settings, classification accuracies are in the

Agents	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	84.2	98.0	0.0081	0.0011	0.0517	0.0000	0.091	0.040	6.70	0.35
3	82.8	91.3	0.0120	0.0047	0.0865	0.0292	0.113	0.076	7.87	4.43
4	83.3	90.5	0.0173	0.0064	0.1252	0.0431	0.162	0.097	9.20	5.03
5	83.2	89.2	0.0243	0.0086	0.1878	0.0641	0.221	0.122	12.00	6.46
6	85.2	89.5	0.0224	0.0093	0.1860	0.0674	0.224	0.136	10.77	6.10

Agents	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	87.2	97.5	0.0057	0.0005	0.0275	0.0000	0.073	0.028	5.75	1.35
3	81.7	93.3	0.0184	0.0036	0.1196	0.0138	0.161	0.080	9.03	2.80
4	83.6	89.0	0.0178	0.0103	0.1314	0.0745	0.167	0.137	8.88	5.88
5	87.7	91.1	0.0153	0.0091	0.1203	0.0597	0.178	0.145	7.76	6.98
6	88.1	90.8	0.0189	0.0096	0.1563	0.0776	0.215	0.147	8.00	7.07

Table 4.3: Performance of the two attribute vectors in single-minded CAs, using optimal allocation, 5 items, and instance-based normalization. (Top: Uniform, Bottom: Decay Datasets.)

Agents	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	87.3	92.2	0.0061	0.0045	0.0350	0.0287	0.078	0.065	5.90	3.15
3	80.8	86.8	0.0133	0.0116	0.0900	0.0976	0.123	0.117	9.50	8.53
4	78.0	88.3	0.0428	0.0095	0.3093	0.0808	0.295	0.124	16.70	5.88
5	85.1	87.1	0.0150	0.0155	0.1118	0.1239	0.165	0.166	8.98	8.74
6	87.8	89.9	0.0138	0.0106	0.1053	0.0773	0.182	0.155	6.55	7.07

Agents	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
2	87.3	92.8	0.0063	0.0040	0.0400	0.0193	0.073	0.064	7.70	3.55
3	85.3	89.8	0.0113	0.0110	0.0800	0.0901	0.124	0.146	8.47	5.80
4	86.5	88.2	0.0153	0.0126	0.1152	0.0979	0.166	0.147	8.75	7.85
5	87.4	89.6	0.0125	0.0090	0.0955	0.0684	0.163	0.128	7.46	5.26
6	90.7	91.4	0.0112	0.0087	0.0795	0.0610	0.182	0.145	5.55	4.72

Table 4.4: Performance of the two attribute vectors in single-minded CAs, using greedy allocation, 5 items, and instance-based normalization. (Top: Uniform, Bottom: Decay Datasets.)

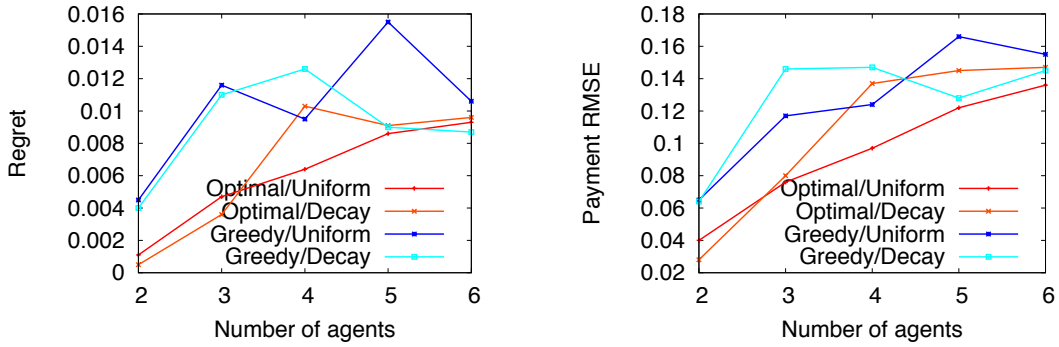

Figure 4.4: Regret and Payment RMSE trends for single-minded CAs with the uniform and decay distributions and the optimal and greedy allocations. (Using χ_2 and normalization.)

high 80s to low 90s for χ_2 and in the low 80s for χ_1. Classification accuracy, average ex post regret, 99^{th}percentile regret, payment RMSE, and IR violation are highly correlated. Across the two value distributions and two allocation rules, there are no major differences in performance. Figure 4.4 plots average ex post regret and payment RMSE across these different settings. The optimal allocation rule is slightly easier to learn than the greedy allocation rule in most cases, whereas the uniform distribution is slightly easier that its decay counterpart in most cases. Note that in the 4-agent, greedy, uniform case, the performance of χ_1 is at odds with the general trends. Specifically, classification accuracy drops slightly from its neighboring cases, whereas regret is about three times, and price RMSE and IR violation rates is about twice as much as those of its neighboring cases. This performance lag does not seem to be systematic, as all of the metrics bounce back to their original levels, when one more agent is added to the setting. Therefore, we believe that this setting likely suffers from some idiosyncratic factors during the training or testing processes, and hence discard this setting as an outlier in our discussion of results.

From Tables 4.3 and 4.4, average ex post regret ranges from 0.1% to 1% for χ_2 and to 2% of agents' valuation range for χ_1. Considering that average ex post regret for the single item setting ranges from 0% to 0.2%, regret in this setting is almost an order of magnitude

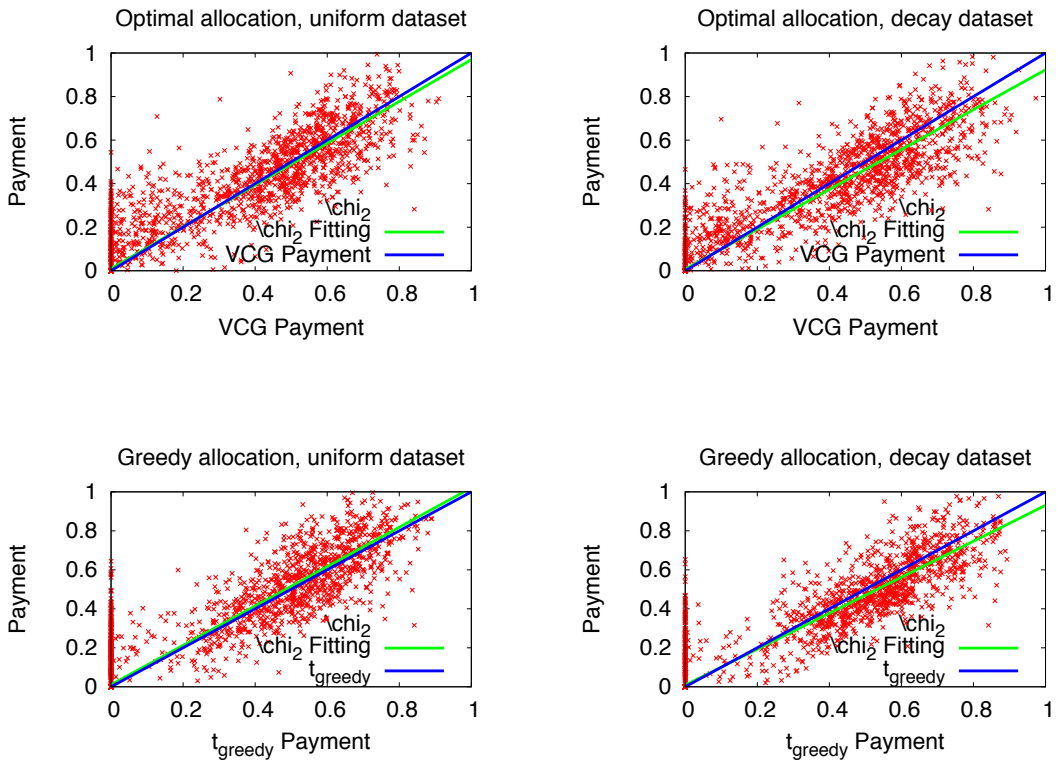

Figure 4.5: Predicted vs. benchmarks payments of 6-agent, 5-item single-minded CAs, using χ_2 and normalization.

higher that the single item case. Nonetheless, 1% expected ex post regret is still small. The 95^{th}percentile ex post regret, on the other hand, ranges from 4% to 19% for χ_1 and from 0% to 12% of agents' valuation range for χ_2. Even though these numbers seem high, the fact that less than 5% of the population experience ex post regret greater than 10% of the valuation range suggests that the predicted payment rules still perform reasonably well in terms of strategyproofness.

In terms of the characteristics of payment rules, Hypothesis 2 predicts that payment rules from classifiers trained on the optimal allocation rule in the single-minded CA setting approximate the VCG payment rule. From Table 4.3, the payment RMSE benchmarking

Agents	Average Payment Diff				Average Absolute Payment Diff			
	ψ_1	ψ_2	ψ_1 norm	ψ_2 norm	ψ_1	ψ_2	ψ_1 norm	ψ_2 norm
2	0.0104	-0.0316	0.0336	0.0010	0.0360	0.0320	0.0440	0.0149
3	0.0597	0.0408	0.0601	-0.0060	0.0978	0.0821	0.1079	0.0543
4	0.0778	0.0537	0.0341	-0.0043	0.1307	0.1006	0.1270	0.1009
5	0.1052	0.1009	0.0687	0.0736	0.1519	0.1282	0.1375	0.1103
6	0.0777	0.0853	0.0472	0.0683	0.1542	0.1222	0.1734	0.1155

Table 4.5: Average payment difference across all settings of single-minded CAs, using optimal allocation and decay distribution.

Agents	Average Payment Diff				Average Absolute Payment Diff			
	ψ_1	ψ_2	ψ_1 norm	ψ_2 norm	ψ_1	ψ_2	ψ_1 norm	ψ_2 norm
2	0.0005	0.0018	0.0308	-0.0107	0.0384	0.0378	0.0459	0.0352
3	0.0145	-0.0271	0.0286	-0.0338	0.0914	0.0990	0.0929	0.1025
4	0.0341	-0.0828	0.0449	0.0373	0.1185	0.1334	0.1193	0.1073
5	0.1022	0.0510	0.0314	0.0093	0.1583	0.1428	0.1262	0.0975
6	0.0535	0.1611	0.0149	0.0071	0.1485	0.1832	0.1418	0.1097

Table 4.6: Average payment difference across all settings of single-minded CAs, using greedy allocation and decay distribution.

predicted payments against VCG payments ranges from 0.03 to 0.22. As shown in Table 4.5[2], average absolute payment difference ranges from 0.01 to 0.17 across all models and settings, reflecting that predicted payments on average differ from VCG payments by 1%-17% of agents' valuation range. Furthermore, the top two graphs of Figure 4.5 plot the predicted payments tested on an unseen test set against the VCG payments in the 6-agent, 5-item settings. The fitting lines are close to the VCG payment rule for both distributions. Even though the predicted payments are dispersed, the figure suggests that with more training instances, the learned payment rule is likely to converge to the VCG payment rule.

As for the greedy allocation rule, Hypothesis 3 predicts that payment rules from classifiers trained on this allocation rule in the single-minded CA setting approximates t_{greedy}. From

[2]Note that only results from the decay distribution are shown here, since they are representative of those from both distributions.

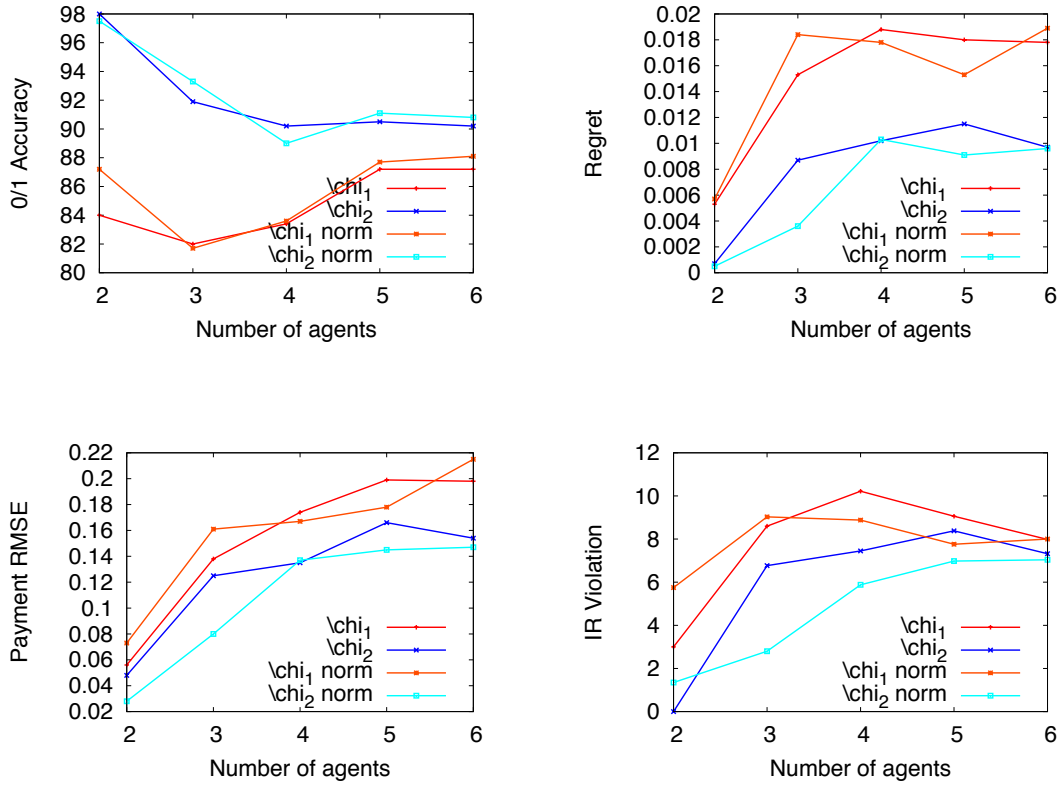

Figure 4.6: Performance of single-minded CAs, using the optimal allocation and decay dataset: 0/1 Accuracy, Regret, Payment diff, IR violation trends for χ_1, χ_2, and with and without instance-based normalization.

Table 4.4, the payment RMSE benchmarking predicted payments against t_{greedy} payments ranges from 0.06 to 0.29. Similar to that of the optimal allocation rule, average absolute payment difference ranges from 3.5% to 18.3% of agents' valuation range across all models and settings, as partially shown in Table 4.6[3]. The bottom two graphs of Figure 4.5 plot the predicted payments on an unseen test set against the t_{greedy} payments in the 6-agent, 5-item settings. The fitting lines of both distributions approximate the t_{greedy} payment rule well, showing evidence in support of Hypothesis 3.

[3]Note that only results from the decay distribution are shown here, since they are representative of those from both distributions.

Finally, we explore the effect of different components of the experiments on performance. Figure 4.6 plots 0/1 classification accuracy, average ex post regret, payment difference, and IR violation with χ_1 and χ_2, and with and without normalizing individual instances. As seen in this figure, χ_2 consistently performs better than χ_1 in all of the four metrics in most of the cases. As for instance-based normalization, it is unclear whether normalization improves performance in general. However, normalized models consistently outperform non-normalized models in terms of IR violation.

4.2.2 Discussion

The errors in terms of classification, regret, and payments in the single-minded CA case are significantly higher than those in the single item auction setting. There are two possible explanations for the performance lag. First, due to limited computing resources[4], the training sets used in training single-minded models are of the same size as those used in single item models. With increasing complexity of the single-minded CA setting, performance is likely to suffer. Second, it is important to note that the value distributions used in the single-minded CA case are different from the one used in the single item experiments. This discrepancy could potentially explain the performance gap. The second hypothesis will be discussed in details in the discussion section of the multi-minded CA experiments.

It is also important to consider the continuing trends of payment biases. In the single-minded CA case, 85% of the non-normalized models from both attribute vectors, both allocation rules, and both distributions have positive biases (i.e., positive average payment difference). This number is comparable to that of the single item case, which is at 90%. Second, 65% of the normalized models from all the settings of single-minded CAs are less positively biased than their non-normalized counterparts (i.e., having less average payment diff). Although the second trend is not as strong, it explains why normalized models enjoy

[4]Training time for single-minded CAs with the current setup is in the order of hours. Whereas most cases take less than an hour, some of the larger cases can take up to 2-3 hours.

lower IR violation rates in most of the settings.

4.3 Multi-minded CAs with an SP payment rule

For the multi-minded CA setting, we decided to drop the non-normalized models due to limited computing resources. As noted in the two previous sections, results from non-normalized and normalized models are comparable, except that predicted payments from normalized models tend to be more concentrated and slightly less positively biased. These characteristics propel us to only experiment with normalized models in the multi-minded CA setting. In addition to the two attribute vectors, we are also interested in the effects of varying ζ, which determines the level of complementarity and substitutability, and β, which determines the correlation between agents' private values. In this section, we focus on the optimal allocation rule, which can be made strategyproof. In the next section, we study the greedy allocation rule, which does not have a strategyproof payment rule.

4.3.1 Main Results

Tables 4.7 and 4.8 provide the primary metrics for our experiments with a multi-minded CA, varying the number of agents and the degree of complementarity and substitutability (ζ) and fixing β at 0.5. Similar to the single-minded CA case, there is a correlation between classification accuracy, average ex post regret, 95^{th}percentile ex post regret, payment RMSE, and IR violation.

Classification accuracies range from 16% to 93% for χ_1 and 70% to 96% for χ_2. In the easiest setting ($\zeta = 1.5$ and decay dataset), classification accuracies are in the 90s for both attribute vectors. Therefore, in easy settings, the performance of multi-minded CAs is better than that of single-minded CAs. In other settings, however, performance is either comparable or worse. This observation suggests that value distributions have large effects on performance. We will further investigate this issue toward the end of this section. Also,

(Agents, ζ)	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
(2, 0.5)	72.8	92.3	0.0145	0.0017	0.0940	0.0093	0.110	0.035	6.5	2.3
(3, 0.5)	53.7	74.1	0.0354	0.0180	0.1555	0.1151	0.144	0.114	19.4	14.1
(4, 0.5)	56.2	70.4	0.0411	0.0236	0.1898	0.1370	0.146	0.116	25.1	15.4
(5, 0.5)	64.0	74.0	0.0376	0.0214	0.1939	0.1393	0.146	0.111	23.0	13.9
(6, 0.5)	69.0	76.1	0.0285	0.0204	0.1648	0.1354	0.117	0.103	16.7	13.8
(2, 1.0)	80.9	94.3	0.0104	0.0005	0.0706	0.0015	0.055	0.018	8.8	1.4
(3, 1.0)	75.9	84.0	0.0135	0.0050	0.0907	0.0373	0.060	0.042	12.8	6.5
(4, 1.0)	72.9	80.5	0.0180	0.0078	0.1096	0.0550	0.060	0.042	20.9	10.8
(5, 1.0)	69.8	75.5	0.0205	0.0100	0.1194	0.0672	0.061	0.039	20.1	11.7
(6, 1.0)	74.8	80.3	0.0206	0.0098	0.1319	0.0704	0.056	0.040	21.8	14.2
(2, 1.5)	89.2	95.7	0.0032	0.0003	0.0212	0.0000	0.027	0.016	3.7	1.4
(3, 1.5)	84.1	89.6	0.0052	0.0021	0.0368	0.0181	0.036	0.025	8.1	5.4
(4, 1.5)	87.6	91.0	0.0047	0.0018	0.0331	0.0117	0.034	0.023	6.5	4.1
(5, 1.5)	91.0	93.0	0.0033	0.0019	0.0214	0.0077	0.036	0.025	6.5	3.4
(6, 1.5)	90.6	90.8	0.0044	0.0024	0.0284	0.0177	0.037	0.026	6.6	4.8

Table 4.7: Performance of multi-minded CAs, using the optimal allocation and uniform distribution. (With 5 items, 3 bundles, $\beta = 0.5$, instance-based normalization.)

(Agents, ζ)	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2	χ_1	χ_2
(2, 0.5)	70.7	91.9	0.0138	0.0011	0.0847	0.0085	0.102	0.022	8.0	1.1
(3, 0.5)	54.7	74.7	0.0370	0.0161	0.1717	0.1022	0.140	0.107	18.3	10.8
(4, 0.5)	65.8	75.7	0.0310	0.0188	0.1658	0.1232	0.145	0.123	17.7	15.0
(5, 0.5)	66.1	75.7	0.0340	0.0186	0.1851	0.1226	0.140	0.112	20.7	12.2
(6, 0.5)	16.2	70.1	0.1185	0.0240	0.2755	0.1447	0.120	0.111	18.2	16.3
(2, 1.0)	85.1	93.8	0.0058	0.0006	0.0425	0.0028	0.039	0.019	4.8	1.9
(3, 1.0)	78.9	85.2	0.0113	0.0045	0.0802	0.0337	0.056	0.038	11.5	6.5
(4, 1.0)	73.9	78.0	0.0148	0.0096	0.0964	0.0666	0.056	0.043	14.7	8.4
(5, 1.0)	79.3	81.4	0.0136	0.0085	0.0970	0.0614	0.056	0.040	14.8	10.1
(6, 1.0)	75.5	78.8	0.0213	0.0106	0.1410	0.0755	0.061	0.041	22.0	15.2
(2, 1.5)	90.9	94.8	0.0026	0.0006	0.0140	0.0003	0.032	0.019	3.2	1.9
(3, 1.5)	93.0	94.7	0.0021	0.0008	0.0089	0.0007	0.032	0.024	3.8	1.4
(4, 1.5)	92.3	93.4	0.0023	0.0011	0.0125	0.0043	0.036	0.023	3.7	1.9
(5, 1.5)	92.8	94.3	0.0017	0.0009	0.0092	0.0031	0.032	0.025	3.4	3.5
(6, 1.5)	94.1	94.9	0.0021	0.0008	0.0042	0.0002	0.035	0.017	3.0	3.0

Table 4.8: Performance of multi-minded CAs, using the optimal allocation and decay distribution. (With 5 items, 3 bundles, $\beta = 0.5$, instance-based normalization.)

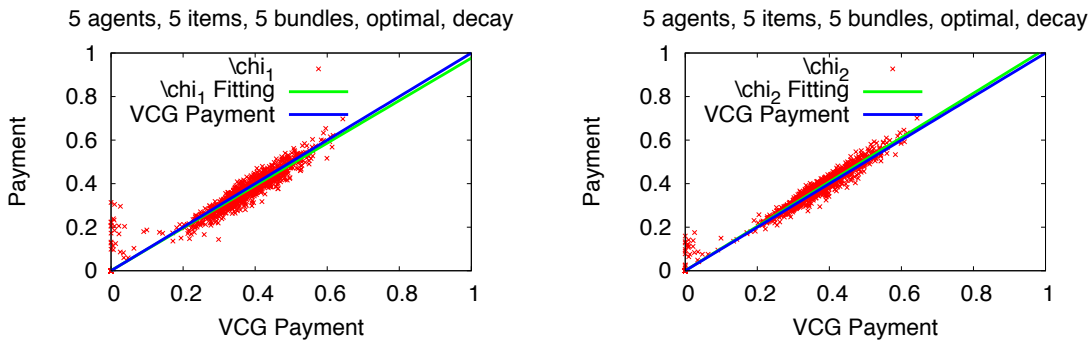

Figure 4.7: Predicted vs. VCG payments for multi-minded CAs with 5 agents, 5 items, 5 bundles, $\zeta = 1.5$, and $\beta = 0.5$. (Left: χ_1, Right: χ_2.)

note that the performance of χ_2 drops significantly, when $\zeta = 0.5$ and the number of agents is 6. Since the performance lag in this case does not seem to be part of the macro trends, we discard the instance in our current analysis of general trends and will further investigate this issue in the subsequent discussion section.

As shown in Tables 4.7 and 4.8, average ex post regret ranges from 0.002 to 0.041 for χ_1 and from 0.001 to 0.024 for χ_2. Relative to agents' valuation range, the average ex post regret is somewhere between 0.1% and 4%. These numbers are only slightly higher than those of single-minded CAs. As for the 95^{th}percentile ex post regret, the regret ranges from 0.01 to 0.19. These numbers are in line with those of single-minded CAs. These numbers suggest that the predicted payment rules perform well as strategyproof payment rules.

As for the characteristics of the payment rules themselves, Hypothesis 4 predicts that in the multi-minded CA setting, payment rules from classifiers trained on the optimal allocation rule approximate the VCG payment rule. As shown in Tables 4.7 and 4.8, payment RMSEs ("PriceRMSE") benchmarking predicted payments against VCG payments are in the same range as payment RMSEs in the single-minded setting. Figure 4.7 plots the predicted payments against the VCG payments in a setting with 5 bundles and 5 items. The fitting lines for both attribute vectors approximate the VCG payment rule well. Therefore,

β	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2
0.1	92.5	94.5	0.0035	0.0013	0.0186	0.0019	0.045	0.031	4.2	2.3
0.3	93.6	94.6	0.0027	0.0015	0.0084	0.0021	0.040	0.028	4.0	2.8
0.5	92.8	94.3	0.0017	0.0009	0.0092	0.0031	0.032	0.025	3.4	3.5
0.7	90.1	91.9	0.0024	0.0014	0.0132	0.0089	0.030	0.022	4.3	4.0
0.9	85.6	89.2	0.0029	0.0012	0.0216	0.0093	0.024	0.015	8.1	5.1

Table 4.9: Performance of multi-minded CAs, varying β. (Using optimal allocation, decay dataset, 5 agents, 5 items, 3 bundles, $\zeta = 1.5$, and instance-based normalization.)

Bundles	0/1Accuracy		Regret		95Regret		PriceRMSE		IR Violate	
	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2
2	92.0	94.1	0.0033	0.0012	0.0192	0.0028	0.039	0.025	5.8	1.1
3	92.8	94.3	0.0017	0.0009	0.0092	0.0031	0.032	0.025	3.4	3.5
4	93.2	93.9	0.0020	0.0013	0.0077	0.0040	0.031	0.024	4.1	3.8
5	92.3	94.5	0.0022	0.0011	0.0094	0.0022	0.030	0.019	2.7	3.6

Table 4.10: Performance of multi-minded CAs, varying the number of bundles requested per agent. (Using the optimal allocation, decay dataset, 5 agents, 5 items, $\beta = 0.5$, $\zeta = 1.5$, and instance-based normalization)

empirical evidence suggests that the learned payment rules approximate the VCG payment rule, as proposed by Hypothesis 4.

Lastly, we examine the effect of various elements of the models and experiments on performance. Tables 4.7 and 4.8 shows that attribute vector χ_2 consistently outperforms χ_1, similar to single-minded CAs. In addition, ζ, the degree of complementarity, has significant effects on performance. Specifically, the models trained on the highest degree of complementarity ($\zeta = 1.5$) perform significantly better than those trained on the other two values of ζ. At the same time, performance suffers in settings with substitute goods ($\zeta = 0.5$). We further investigate this issue in the following discussion section.

Table 4.9 presents the results of different settings varying the correlation between agents' values (β). Performance slightly drops, as there is a higher level of correlation between

agents' values (i.e., higher β). We hypothesize that this is because with a higher degree of correlation, different solutions to an allocation problem yield closer results, and learning problems become more difficult. Finally, we experiment with different numbers of requested bundles by single agents (b). The results are presented in Table 4.10. As seen here, none of the performance metrics are affected by the increasing number of bundles. This observation implies that our framework is able to scale with respect to performance in terms of the number of bundles. Note that we do not scientifically measure training time in this project, but from observation, training time does increase significantly with larger cases.

4.3.2 Discussion

Overall, our models work as well in approximating the VCG payment rule in the multi-minded CA setting as in the single-minded CA setting. Nevertheless, the significant performance gap between different value distributions resulted from different levels of complementarity and substitutability in the multi-minded setting requires further investigation. Table 4.11 presents empirical results for different values of ζ. For each ζ value, the left column shows the distribution of sizes of bundles allocated to agent 1 according to the optimal allocation rule. The table presents data from the uniform distribution, meaning that sizes of requested bundles are uniformly distributed between 1 and 5. More than half of allocated bundles in the substitutes setting ($\zeta = 0.5$) contain fewer than 4 items, compared to 17.2% and 2.4% of allocated bundles in the settings with $\zeta = 1.0$ and $\zeta = 1.5$ respectively. This observation reflects that with substitutability between goods, most items are allocated in smaller packages, since agents value larger bundles less.

With $\zeta = 1.0$, we notice that more than 10% of output bundles contain all of the goods. With the uniform distribution, approximately 20% of all requested bundles contain all items, meaning that roughly half of the agents who request the bundle with all items are allocated that bundle. With no complementarity or substitutability between goods, there is no advantage in allocating smaller or larger bundles in terms of pure values; however,

Size of bundles	$\zeta = 0.5$		$\zeta = 1.0$		$\zeta = 1.5$	
	Allocated	Accuracy	Allocated	Accuracy	Allocated	Accuracy
0	46.2%	80.2%	68.0%	84.3%	77.9%	95.6%
1	27.4%	71.7%	8.6%	42.1%	1.8%	28.1%
2	15.6%	65.8%	4.7%	49.4%	0.3%	46.7%
3	7.6%	67.2%	3.9%	53.9%	0.3%	64.7%
4	3.0%	64.7%	4.7%	63.1%	2.7%	84.3%
5	0.2%	30.0%	10.2%	70.9%	17.0%	90.6%
Total	100.0%	74.0%	100%	75.5%	100%	93.0%

Table 4.11: Distribution of sizes of allocated bundles according to the optimal allocation rule and classification accuracy per each size of bundles for multi-minded CAs, using the uniform distribution, 5 agents, 5 items, 3 bundles per agent, $\beta = 0.5$, and χ_2.

allocating the bundle with all items ensures that all goods are allocated. Therefore, we observe that there is a bias toward the bundle with all items. This effect is amplified with complementarity between items. Specifically, 17.0% of output bundles contain all goods, or about 85% of those who request the full bundle as one of their bundles receive the full bundle. Bundles of other sizes excluding non-allocation constitute roughly 5% of allocated bundles. Therefore, it is clear that with complementarity, it is better to allocate larger bundles.

For each ζ value, the right column in Table 4.11 shows the classification accuracy of our models given that the correct classification is a bundle of that size. As seen here, our models are mediocre classifiers for most bundle sizes in the settings with $\zeta = 0.5$ and $\zeta = 1.0$. However, for $\zeta = 1.5$, our models achieve 90% accuracy rates in two largest cases: non-allocation and allocation of the full bundle. In other words, with low ζ values, the models have to learn to work well with all bundle sizes, whereas with $\zeta = 1.5$, the learning problems are able to concentrate their efforts in learning to classify two classes: the zero and complete bundles. This observation explains why our models are able to achieve classification accuracy of more than 90% in the $\zeta = 1.5$ cases and of lower than 75% in the $\zeta = 0.5$ cases.

Type of classification errors	$\zeta = 0.5$	$\zeta = 1.0$	$\zeta = 1.5$
Should have been allocated	47.77%	42.45%	47.73%
Should not have been allocated	35.29%	43.67%	49.15%
Predicted bundle $\subset$ correct bundle	1.77%	4.90%	1.14%
Correct bundle $\subset$ predicted bundle	2.77%	5.39%	0.85%
Predicted bundle intersects correct bundle	2.31%	1.71%	0.85%
Predicted and correct bundles unrelated	8.17%	1.31%	0.28%
Predicted bundle not requested	1.93%	0.57%	0.00%

Table 4.12: Breakdown of classification errors by type for multi-minded CAs with different ζ values.

Furthermore, we analyze the types of misclassifications experienced in different settings of multi-minded CAs. Table 4.12 presents the breakdown of classification errors by type, further confirming our interpretation of Table 4.11. First, the fact that 8% of classification errors in the substitute good setting (compared to 1.31% and 0.28% in the other settings) occur with no connections between predicted and correct bundles suggests that the models in this setting struggle to learn correct decision boundaries between different requested bundles. In the setting with $\zeta = 1.0$, the third and fourth error types stand out, relative to those of the other settings. Both of these classification types imply that the problem with the models in this setting is allocating bundles that are either too large or too small. Given that there is no complementarity and substitutability between goods, it is reasonable to expect that the models consistently assign packages that are too large and too small without any bias for either of the directions. Another explanation is that without complementarity and substitutability, multiple solutions could yield very similar results, making the learning problem more difficult. Lastly, in the setting with $\zeta = 1.5$, almost 97% of the errors come from the first two types, which are errors between allocating and not allocating. This observation suggests that the primary issue facing the models in this setting is deciding between two classes.

Recall from Section 4.2 that most of the models in the single-minded setting are able to achieve classification accuracy within the range of $80-95\%$. This range is almost 10% lower

Size of bundles	Optimal		Greedy	
	Allocated	Accuracy	Allocated	Accuracy
0	73.0%	94.2%	73.7%	94.4%
1	5.4%	51.5%	5.2%	34.0%
2	3.2%	59.7%	2.9%	51.0%
3	3.7%	77.7%	3.9%	61.3%
4	5.7%	84.6%	6.1%	73.6%
5	9.1%	89.9%	8.1%	91.2%
Total	100.0%	89.2%	100%	87.1%

Table 4.13: Distribution of sizes of allocated bundles according to the optimal and greedy allocation rules and classification accuracy per each size of bundles for single-minded CAs, using the uniform distribution, 5 agents, 5 items, and χ_2.

than those of the single item setting and the multi-minded setting with complementarity ($\zeta = 1.5$). Table 4.13 presents the distribution of sizes of allocated bundles according to the optimal and greedy allocation rules as well as classification accuracy per each bundle size in the single-minded setting. As seen here, the distribution of bundle sizes in this case is very similar to that of the multi-minded setting with $\zeta = 1.0$ shown in Table 4.11. Unlike those in the multi-minded setting, the value distributions in the single-minded setting do not assume underlying values for individual items; yet, the distributions in the single-minded setting draw an agent's valuation of bundle S_i uniformly from $(0, \frac{|S_i|}{m}]$, where m is the total number of goods. In other words, an expected valuation of a bundle is linear to the number of items in the bundle. Therefore, on average the value distributions of the single-minded case approximate a value distribution with no complementarity or substitutability between goods. Since the single-minded setting is simpler than the multi-minded case, it is not surprising that our models in the single-minded case perform better than those in the multi-minded case without complementarity and substitutability ($\zeta = 1.0$), whereas the difficulty of non-complementary goods settings explains why the models perform worse than the multi-minded models with complementarity ($\zeta = 1.5$).

In addition, we revisit the outlier case mentioned at the beginning of the results section. Recall from Table 4.8 that with $\zeta = 0.5$, the decay distribution, and 6 agents, the performance of attribute vector χ_1 drops significantly. Specifically, classification accuracy is at 16.2%, whereas average ex post regret is three times as much as those of its neighboring cases. Inspecting the training data used in the case reveals that the training data does not contain the instances where the full bundle is allocated. This is reasonable, as Table 4.11 points out that with $\zeta = 0.5$, the full bundle is rarely allocated. As such, the models trained in this setting have not been exposed to the full bundle. During testing, the χ_2 model from which we report results handles the situation well, by rarely allocating the full bundle (0.35% of the test instances). The χ_1 model, on the other hand, hands out the full bundle to 73% of the test instances. Consider our initial observation that χ_1 does not permit any interaction between different types of bundles ($\mathbf{y}$). When a type of bundle is missing from training data, the χ_1 model trained on such data will not be able to properly handle the missing bundle type. χ_2 models, on the other hand, are able to infer knowledge for handling unseen bundles from other known bundle types. In addition, this observation also suggests that with a larger bundle space, the performance gap between χ_1 and χ_2 increases, as χ_1 is unable to apply knowledge learned about one bundle type to other bundle types. This fact likely explains why χ_1 and χ_2 perform equally well in the single item case, but χ_2 significantly outperforms χ_1 in larger, more complex settings.

Lastly, we examine the assumption that the linear coefficient, w_1, of agent 1's valuation in the discriminant function is positive. Out of the 1,761 successfully trained models in the optimal, multi-minded CA setting, only 10 models have negative w_1 values. In addition, 8 of these 10 models are χ_1 models, and 9 out of the 10 models are from $\zeta = 0.5$ cases. The fact that more than 99% of the models have positive w_1 values suggests that our assumption about positivity of w_1 is reasonable. Besides, most of the models with negative w_1 values are cases where training is considered more difficult than usual (i.e., either using attribute vector χ_1 or setting ζ at 0.5).

In conclusion, the analysis in this section suggests that settings with substitute goods are the most difficult one for our framework, whereas those with complementary goods are the easiest setting. Moreover, χ_2 models outperform their χ_1 counterparts because χ_2 allows for interactions between different bundle types—models with χ_1 completely break, when some bundle types are missing from training data. It is also important to note that with limited computing resources, we do not tune the models on the dimension of training set sizes. Therefore, it is possible that larger training sets could lead to a significant improvement in performance in all settings.

4.4 Multi-minded CAs without an SP payment rule

In this last experimental setting, we match our framework with a greedy allocation rule, which has no corresponding strategyproof payment rule. The goal of this experiment is to show that the framework extends robustly to a non-strategyproof allocation rule.

Table 4.14 presents the results from this setting, varying the number of agents and the degree of complementarity of goods (ζ). Note that the payment RMSE metric is missing here, as this setting does not have a strategyproof payment rule to benchmark the results against. Excluding the 6-agent and χ_1 case with $\zeta = 0.5$, classification accuracies range from 54% to 95%, a similar range as that of the strategyproof multi-minded setting. The difference in performance across different value distributions is also in line with that in the previous setting. The similarity between classification accuracy of this setting and that of the previous setting implies that what the learning algorithm in this setting is about to achieve is comparable to that in a setting with a strategyproof payment rule.

In addition, Table 4.14 also shows that the distribution of ex post regret in this setting is similar to that in the previous setting. Specifically, average ex post regret ranges from 0.3% to 3.6% of agents' valuation range, whereas the 99^{th}percentile ex post regret ranges from 0.1% to 10.3% of the valuation range. These ex post regret values are slightly less

(Agents, ζ)	0/1Accuracy		Regret		95Regret		IR Violate	
	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2	ψ_1	ψ_2
(2, 0.5)	66.0	88.0	0.0154	0.0026	0.0858	0.0191	10.6	2.6
(3, 0.5)	54.7	74.8	0.0357	0.0167	0.1567	0.1074	24.9	13.3
(4, 0.5)	64.3	76.0	0.0274	0.0185	0.1404	0.1246	16.6	13.2
(5, 0.5)	59.8	69.6	0.0338	0.0244	0.1615	0.1396	17.7	17.7
(6, 0.5)	27.1	74.7	0.0982	0.0208	0.2810	0.1352	17.1	13.8
(2, 1.0)	87.5	94.8	0.0057	0.0007	0.0428	0.0007	5.1	2.0
(3, 1.0)	78.4	82.1	0.0134	0.0058	0.0993	0.0423	13.9	8.5
(4, 1.0)	78.0	82.7	0.0142	0.0072	0.1029	0.0526	13.6	8.8
(5, 1.0)	73.6	79.6	0.0142	0.0081	0.0943	0.0577	14.3	11.0
(6, 1.0)	74.7	77.0	0.0137	0.0200	0.0890	0.1310	16.8	21.6
(2, 1.5)	92.2	92.5	0.0025	0.0013	0.0107	0.0080	3.7	6.0
(3, 1.5)	90.2	91.6	0.0028	0.0015	0.0162	0.0112	5.5	4.8
(4, 1.5)	91.9	92.0	0.0022	0.0016	0.0118	0.0103	5.3	4.3
(5, 1.5)	92.2	94.5	0.0027	0.0011	0.0141	0.0017	5.8	2.1
(6, 1.5)	92.9	94.8	0.0033	0.0011	0.0149	0.0007	5.8	3.6

Table 4.14: Performance of multi-minded CAs with the greedy allocation, using the decay distribution, 5 items, 3 bundles, instance-based normalization.

than those of the previous setting. Moreover, the IR violation rates of this and the previous settings are also within the same range.

The similarity of the results of this setting and the previous strategyproof setting suggests that our framework does not suffer any additional performance loss in a setting without a strategyproof payment rule. In the future, it would also be interesting to further investigate the type of payment rules trained in such a setting.

Chapter 5

Conclusions

We have presented the theoretical framework to design a strategyproof payment rule for any given allocation rule in the context of combinatorial auctions. The advantage of this framework over other existing methods including VCG-inspired payment rules is its applicability to any kind of allocation rule as long as the allocation rule satisfies a form of monotonicity and hence has a corresponding strategyproof payment rule. The only requirement for the framework to function is that training instances can be generated based on the allocation rule. We accomplish this by establishing a connection between decision boundaries of Structural Support Vector Machines and various economic properties of combinatorial auctions. In particular, minimizing regularized empirical error also minimizes a regularized upper bound on empirical regret. An exact classifier is proven to provide a strategyproof and individually rational payment rule.

Through our experiments, we show that the framework is able to approximate well-known strategyproof payment rules well for both optimal and greedy allocation rules under benchmark value distributions, in all three auction settings—single item auctions, single-minded CAs, and multi-minded CAs. The root mean square errors comparing payments from our models to those from known SP rules are in most cases under 0.15, with agents'

valuation range of $(0, 1]$.[1] With respect to strategyproofness, agents on average experience *ex post* regret for bidding truthfully of less than 2.5% of their valuation range in all cases. Individual rationality violation rates are under 4% for single item auctions, under 7% for single-minded CAs, less than 6% for multi-minded CAs with easy value distributions, and between 1% and 16% for multi-minded CAs with difficult value distributions.

In the multi-minded combinatorial auction setting, the methodology works particularly well with complementary goods and not so well with substitute goods. This is because in the substitutes case, correct allocations tend to be skewed toward smaller bundles, whereas most allocations are in the full package in the complements case. Furthermore, we observe that the non-normalized models tend to be positively biased, but the bias is sufficiently small in most cases. Lastly, our initial experiments with an allocation rule that does not have a corresponding SP payment rule reveal that applying our framework to a non-SP allocation rule does not lead to additional performance loss.

A primary concern with the framework is its scalability. Once trained, a classifier produces a payment rule that is polynomial-time in the number of support vectors. Given that the numbers of support vectors are in the thousands in our experiments, the payment rules of our framework are quite fast. Yet, the time it takes to train each classifier presents a bottleneck of the system. We observe that training time of larger and more complex cases is significantly longer than that of easier settings. In large cases, training can take more than 6 hours. We can argue that if we are to use this methodology in practice, spending hours, or even days, training a model should be considered an acceptable, one-time cost. Nevertheless, the long training time severely limits what we can do in our experiments.

A few things can be done to improve computational tractability of the model. First, note that we currently enumerate all interesting bundles in search for a bundle with the highest score during both training and predicting processes. Looking for a more efficient algorithm for determining the most preferred bundle given a hypothesis and agents' valuations is an

[1] The numbers reported in this chapter are based on attribute vector χ_2.

important step to reduce computational cost. Furthermore, recall that for each training instance, we create constraints for all possible bundles instead of just interesting bundles, to get around the issue of *admissibility*. Imposing admissibility, hence, is another important step toward making the framework more computationally efficient.

In terms of future directions, there are many aspects of the framework that could be further improved. First, we currently make an assumption that a well-learned model would yield a positive value of w_1, the coefficient of agent 1's valuation. As this assumption is central to the established connection between the learning problem and strategyproofness, enforcing the positivity of w_1 as an additional constraint is crucial. This task is not trivial, as the constraint has to be formulated in the dual space. Second, although we have shown that exact classifiers yield individually rational payment rules, exact classifiers are unlikely to be achieved in practice. As one of the most desirable properties within the field of mechanism design, methods to directly achieve individual rationality on every instance are of great interest.

Another important area of future work is to improve the attribute vector. As seen in the experiments, using attribute vectors with different structures yields significantly different results in terms of performance. Therefore, there are likely some other attribute vectors that can better capture defining features of agents' valuations and produce even better results. Moreover, both of the attribute vectors in this thesis are dependent of the number of agents and the number of items. Having to learn new payment rules for different numbers of agents is inefficient, and finding an agent-independent attribute vector is a topic for future work. Similarly, an attribute vector that can absorb different numbers of items would also make the process much more efficient.

Bibliography

[1] L. M. Ausubel and P. Milgrom. The lovely but lonely Vickrey auction. In P. Cramton, Y. Shoham, and P. Steinberg, editors, *Combinatorial Auctions*, chapter 1, pages 17-40. MIT Press, 2006.

[2] S. Bikhchandani, S. Chatterji, R. Lavi, A. Mu'alem, N. Nisan, and A. Sen. Weak monotonicity characterizes deterministic dominant-strategy implementation. *Econometrica*, 74(4):1109-1132, 2006.

[3] S. Bikhchandani and J. Ostroy. The Package Assignment Model. *Journal of Economic Theory*, 107:377-406, 2002.

[4] L. Blumrosen and N. Nisan Combinatorial Auctions. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 11, pages 267-300. Cambridge University Press, 2007.

[5] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based machines. *Journal of Machine Learning Research*, 2: 265-292, 2001.

[6] S. de Vries and R. V. Vohra. Combinatorial Auctions: A Survey. *Artificial Intelligence*, 15:284-309, 2003.

[7] Y. Fujishima, K Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Proceedings of IJCAI '99*, pages 548-553, 1999.

[8] J. Håstad. Clique is hard to approximate to within $n^{1-\epsilon}$. *Acta Mathematica*, 182, 1999.

[9] T. Joachims, T. Hofmann, Y. Yue, and C. Yu. Predicting structured objects with support vector machines *Communications of the ACM*, 52(11):97-104, Nov, 2009.

[10] S. Lahaie. Kernel method for market clearing. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 208-213, 2009.

[11] S. Lahaie. Stability and incentive compatibility in a kernel-based combinatorial auction. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 811-816, 2010.

[12] R. Lavi. Computationally efficient approximation mechanisms. In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 12, pages 301-330. Cambridge University Press, 2007.

[13] D. Lehmann, L. I. Oćallaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM*, 49:577-602, 2002.

[14] K. Leyton-Brown, M. Pearson, and Y. Shoham. Towards a universal test suite for combinatorial auction algorithms. In *Proceedings of the 2nd ACM Conference on Electronic Commerce (ACM-EC)*, pages 66-76, 2000.

[15] B. Lubin. *Combinatorial Markets in Theory and Practice: Mitigating Incentives and Facilitating Elicitation*. PhD thesis, Computer Science Department, School of Engineering and Applied Science, Harvard University, Cambridge, MA, 2010.

[16] B. Lubin and D. C. Parkes. Quantifying the strategyproofness of mechanisms via metrics on payoff distributions. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 74-81, 2009.

[17] C. D. Manning, P. Raghavan and H. Schütze, Support vector machines and machine learning on documents. In *Introduction to Information Retrieval*, chapter 15, pages 319-348. Cambridge University Press. 2008.

[18] N. Nisan. Introduction to Mechanism Design (for Computer Scientists). In N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 12, pages 209-242. Cambridge University Press, 2007.

[19] D. C. Parkes, J. Kalagnanam, and M. Eso. Achieving budget-balance with vickrey-based payment schemes in exchanges. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1161-1168, 2001.

[20] R. Rifkin and A. Klautau. In Defense of One-Vs-All Classification. *Journal of Machine Learning Research*, 5:101-141, 2004.

[21] S. M. Ross. *A first course in Probability*. Prentice Hall, New Jersey, 6th edition, 2002.

[22] M. E. Saks and L. Yu. Weak monotonicity suffices for truthfulness on convex domains. In *Proceedings of the 6th ACM Conference on Electronic Commerce (ACM-EC)*, pages 286-293, 2005.

[23] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1-54, Jan. 2002.

[24] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6: 1453-1484, 2005.

[26] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the European Symposium on Artificial Neural Networks*, pp. 219-224, 1999.