# Machine Learning-Powered Course Allocation

**Preprint** · October 2022

DOI: 10.48550/arXiv.2210.00954

Ermis Soumalias
University of Zurich
**7** PUBLICATIONS **10** CITATIONS
SEE PROFILE

Jakob Weissteiner
University of Zurich
**16** PUBLICATIONS **56** CITATIONS
SEE PROFILE

Sven Seuken
University of Zurich
**103** PUBLICATIONS **1,258** CITATIONS
SEE PROFILE

# Machine Learning-powered Course Allocation

**ERMIS SOUMALIAS**[*], University of Zurich & ETH AI Center, ermis@ifi.uzh.ch
**BEHNOOSH ZAMANLOOY**[*], McMaster University, zamanlob@mcmaster.ca
**JAKOB WEISSTEINER**, University of Zurich & ETH AI Center, weissteiner@ifi.uzh.ch
**SVEN SEUKEN**, University of Zurich & ETH AI Center, seuken@ifi.uzh.ch

We introduce a machine learning-powered course allocation mechanism. Concretely, we extend the state-of-the-art *Course Match* mechanism with a machine learning-based preference elicitation module. In an iterative, asynchronous manner, this module generates pairwise comparison queries that are tailored to each individual student. Regarding incentives, our *machine learning-powered course match (MLCM)* mechanism retains the attractive *strategyproofness in the large* property of Course Match. Regarding welfare, we perform computational experiments using a simulator that was fitted to real-world data. Our results show that, compared to Course Match, MLCM increases average student utility by 4%-9% and minimum student utility by 10%-21%, even with only ten comparison queries. Finally, we highlight the practicability of MLCM and the ease of piloting it for universities currently using Course Match.

[*]These authors contributed equally.

## 1 INTRODUCTION

The *course allocation problem* arises when educational institutions assign bundles of courses to students [Budish and Cantillon, 2012]. Each course has a limited number of indivisible seats and monetary transfers are prohibited for fairness reasons. What makes this problem particularly challenging is the combinatorial structure of the students' preferences, as students may view certain courses as complements or substitutes [Budish and Kessler, 2022].

### 1.1 Course Match

The state-of-the-art practical solution to the course allocation problem is the *Course Match (CM)* mechanism by Budish et al. [2017], which provides a good trade-off between efficiency, fairness, and incentives. CM has now been adopted in many universities such as the Wharton School at the University of Pennsylvania and Columbia Business School.

CM uses a simple reporting language to elicit students' preferences over *schedules* (i.e., course bundles). Concretely, CM offers students a graphical user interface (GUI) to enter a *base value* between 0 and 100 for each course, and an *adjustment value* between −200 and 200 for each *pair* of courses. Adjustments allow students to report complementarities and substitutabilities between courses, up to *pairwise* interactions. The total value of a schedule is then the sum of the base values reported for each course in that schedule plus any adjustments (if both courses are in the schedule).

Prior to the adoption of CM in practice, Budish and Kessler [2022] performed a lab experiment to evaluate CM. Regarding efficiency, they found that, on average, students were happier with CM compared to the *Bidding Points Auction* [Sönmez and Ünver, 2010], the previously used mechanism. Regarding fairness, students also found CM fairer. Regarding the reporting language, they found that students were able to report their preferences "accurately enough to realize CM's theoretical benefits." Given these positive findings, Wharton was then the first school to switch to CM.

### 1.2 Preference Elicitation Shortcomings of Course Match

However, Budish et al. [2017] were already concerned that the CM language may not be able to fully capture all students' preferences. Furthermore, they mentioned that some students might find it non-trivial to use the CM language and might therefore make *mistakes* when reporting their preferences. Indeed, the lab experiment by Budish and Kessler [2022] revealed several shortcomings of CM in this regard.

First, students made very limited use of the CM language: on average, students only reported a base value for half of the 25 courses in the experiment. Furthermore, the average number of pairwise adjustments was only 1.08 (out of 300), and the median was 0. This suggests that cognitive limitations negatively affect how well students can report their preferences using the CM language. Second, in addition to not reporting part of their preferences, Budish and Kessler [2022] provided evidence that students are also *inaccurate* when they do report their preferences.

Budish and Kessler [2022] found that both of these reporting mistakes negatively affected the welfare of CM. In their experiment, about 16% of students would have preferred another schedule of courses, with a median utility difference for these schedules of 13%. Thus, preference elicitation in course allocation still remains an important challenge.

### 1.3 Machine Learning-powered Preference Elicitation

To address this challenge, we turn to *machine learning (ML)*. The high-level idea is to train a separate ML model for each student based on that student's reports after using the CM language (i.e., the GUI). These ML models can then be used in an *ML-powered preference elicitation algorithm* that asks each student a sequence of carefully selected queries (thus, enabling them to correct the reporting mistakes they made in the GUI). Based on those queries, the student's ML model is updated in real-time and the next query is generated. At the end, the mechanism allocates schedules to students based on their trained ML models.

With our approach, we build on the ideas developed in a recent stream of papers on ML-powered preference elicitation. Lahaie and Parkes [2004] and Blum et al. [2004] were the first to combine ML and mechanism design, studying the relation between learning theory and preference elicitation. Brero et al. [2017, 2018] were the first to integrate an ML-powered preference elicitation component into a practical combinatorial auction mechanism. They used support vector regression (SVR) to learn bidders' value functions and to iteratively generate new queries in each auction round. In [Brero et al., 2021], the authors proposed the MLCA mechanism and showed that it achieves higher allocative efficiency than the widely-used combinatorial clock auction [Ausubel et al., 2006]. In recent years, there has been a stream of papers further improving the ML capability of the MLCA mechanism, which we discuss in Section 2.

While these works are important pre-cursors to the present paper, there are several noteworthy differences. First, these papers used *value queries* as the interaction paradigm (i.e., asking agents a query of the form "What is your value for bundle {XYZ}"), which would be unnatural in course allocation. Instead, we use *pairwise comparison queries* (i.e., asking students "Do you prefer course schedule A or B?"). Importantly, a pairwise comparison query is a simpler type of query, known to have low cognitive load [Chajewska et al., 2000, Conitzer, 2009]. Second, our goal is to build on top of the CM language; thus, we must be able to handle the *cardinal* input that students provide via the CM reporting language as well as the *ordinal* feedback from answering comparison queries. Third, while an auctioneer can require bidders in an auction to participate in a synchronous way (i.e., submitting a bid in every round), we must allow students to interact with the mechanism in an asynchronous manner (i.e., allowing students to answer a sequence of comparison queries without having to wait on other students). Finally, MLCA could only use the ML models to elicit information, but it could *not* use them to determine the final outcome, as that would often lead to an individual rationality violation. In contrast, in our setting (without monetary transfers), we can *also* use the ML models to determine the final allocation, which leads to additional efficiency gains.[1]

## 1.4 Overview of Contributions

In this paper, we introduce the *machine-learning based course match (MLCM)* mechanism (Section 4). MLCM builds on top of CM by improving its preference elicitation component. Importantly, our design makes the process of upgrading from CM to MLCM particularly simple for the universities and seamless for their students.

First, students use the CM reporting language (i.e., via the same GUI). As in CM, this input is required from all students. Second, MLCM uses these initial reports to train a separate ML model for each student so that it can predict each student's utility for any possible course schedule. Third, MLCM uses an ML-powered preference elicitation algorithm to generate *pairwise comparison queries* that are tailored to each student, and students simply answer which schedule they prefer. Based on this feedback, the ML model is retrained and the next query is generated. Importantly, this phase is *optional* – each student can answer as many such queries as she wants (including none). However, the more queries she answers, the better the ML model will typically approximate her true preferences, which will benefit her in the last phase, where MLCM computes the final allocation based on all ML models (in case a student has answered no queries, then only her GUI reports will be used for the final allocation calculation).

[1]An alternative approach would be to only use the trained ML models to update the student's GUI reports, and to then run the original CM mechanism on the updated GUI reports. See Section 8 for a discussion on this approach.

To understand the theoretical properties of MLCM, we extend the theoretical guarantees of CM to MLCM (Section 4.3). Importantly, we explain why MLCM is also *strategyproof in the large*.[2]

To evaluate MLCM, we introduce a new course allocation simulation framework (Section 5). The first component is a realistic student preference generator, which is designed such that each student's complete preferences can be encoded in a succinct *Mixed Integer Program* (MIP). This allows computing a benchmark allocation given the students' true preferences. The second component models the students' *reporting mistakes* when interacting with the CM language. We calibrate the framework's parameters based on real-world data from Budish and Kessler [2022].

In Section 6, we instantiate the ML model used by MLCM. We show experimentally that the recently introduced *monotone-value neural networks (MVNNs)* [Weissteiner et al., 2022a] exhibit the best generalization performance in our domain, while also being MIP-formalizable, such that the corresponding utility maximization problem can be solved fast enough in practice. Furthermore, we show how the cardinal input from the CM language and the ordinal feedback from the comparison queries can be combined when training neural networks.

In Section 7, we empirically evaluate the performance of MLCM. We find that MLCM significantly outperforms CM in terms of *average student utility* as well as *minimum student utility*, even with only five comparison queries (Section 7.2). Furthermore, we show that these results are robust to changes in students' reporting mistakes (Section 7.3), we show the expected benefit of an individual student unilaterally opting into MLCM (Section 7.4), we show that MLCM also outperforms CM when students have simple additive preferences (Section 7.5), and we show that the runtime of our mechanism scales gracefully in the number of courses (Section 7.6).

In Section 8, we put our results into perspective and discuss alternative approaches one could have taken. Finally, we conclude in Section 9 and discuss interesting avenues for future work.

## 2 RELATED WORK

Our work is related to the research on course allocation and ML-based preference elicitation.

### 2.1 Course Allocation

The course allocation problem is an instance of the *combinatorial assignment problem*, for which several impossibility results establish a tension between welfare, incentives, and fairness. For example, it is known that the only mechanisms for this problem that are ex-post Pareto efficient and strategyproof are dictatorships [Hatfield, 2009, Pápai, 2001].

Multiple empirical studies have pointed out design flaws of course allocation mechanisms used in practice. Budish and Cantillon [2012] showed that the *Harvard Business School (HBS) draft* mechanism [Brams and Straffin Jr, 1979] creates significant incentives for students to misreport, leading to large welfare losses. Similarly, the commonly used *Bidding Points Auction* [Sönmez and Ünver, 2010] implicitly assumes that students have positive value for left-over virtual currency, which harms incentives and ultimately leads to allocations that are neither efficient nor fair.

Motivated by these design flaws, Budish [2011] proposed a new mechanism for the combinatorial assignment problem called *approximate competitive equilibrium from equal incomes (A-CEEI)*. A-CEEI circumvents the impossibility results previously mentioned by making slight compromises in all three of those dimensions of interest. Specifically, A-CEEI is approximately efficient, satisfies desirable fairness criteria (envy bounded by a single good and $(n + 1)$-maximin share guarantee), and is strategyproof in the large [Azevedo and Budish, 2019]. Later, Budish et al. [2017] introduced CM as the practical implementation of A-CEEI. We present A-CEEI and CM in Section 3.

[2]A mechanism is *strategyproof in the large* if, for a large enough number of students and any full-support i.i.d. distribution of opponent reports, reporting truthfully is approximately interim optimal [Azevedo and Budish, 2019].

Diebold et al. [2014] modeled course allocation as a two-sided matching problem where instructors also have preferences over students, which makes the problem quite different from combinatorial assignment. Bichler and Merting [2021] studied the assignment of tutorials for mandatory courses, which is more similar to a scheduling problem.

## 2.2 Machine Learning-based Preference Elicitation

Preference elicitation (PE) using comparison queries has received a lot of attention in the ML community. Bayesian approaches are a natural candidate for PE due to their ability to explicitly model uncertainty of users' utility functions. Chu and Ghahramani [2005] and Bonilla et al. [2010] used Gaussian processes (GPs) to learn and elicit preferences. In particular, Chu and Ghahramani [2005] used GPs for the problem of learning preferences given a (fixed) set of comparison queries. However, they did not answer the question of *which* comparison query to ask. Bonilla et al. [2010] addressed this question by iteratively selecting a pairwise comparison query that maximizes the *expected value of information* (EVOI). However, their approach is impractical in our setting, as the EVOI needs to be analytically calculated for each possible comparison query, thus scaling quadratically in the number of alternatives (course schedules in our case), which is already polynomial in the number of courses. Guo and Sanner [2010] introduced an approximate PE framework for performing efficient closed-form Bayesian belief updates and query selection for a multi-attribute belief state, speeding up the evaluation of EVOI heuristics. However, their approach is inherently limited as it can only model additive utility functions, and is thus not well-suited for course allocation, where students have more complex, non-additive preferences. In general, GPs are also not well suited to our setting due to the high dimensionality of the input space and the integrality constraints that make GP optimization intrinsically difficult.

Ailon [2012] took a different approach, proposing an active learning algorithm that, using binary comparison queries which may be non-transitive, can learn an almost optimal linear ordering of a set of alternatives with an almost optimal query complexity, which is further improved by Ailon et al. [2011]. However, these approaches are impractical for course allocation because they would require more than one hundred thousand queries per student. The reason for this large number of queries is that they do not exploit any notion of similarity between schedules.

Most related to our work is the research on ML-powered combinatorial auctions using SVRs mentioned above (see Section 1.3). Weissteiner and Seuken [2020] extended this work further by using neural networks (NNs), which further increased the efficiency. Weissteiner et al. [2022b] introduced Fourier-sparse approximations for the problem of learning combinatorial preferences. Weissteiner et al. [2022a] introduced *MVNNs*, which are specifically designed to learn *monotone* combinatorial preferences, resulting in a further efficiency increase. Finally, Weissteiner et al. [2023] proposed a *Bayesian optimization-based combinatorial assignment (BOCA)* mechanism which includes a notion of posterior *model uncertainty* [Heiss et al., 2022] to properly balance the *explore-exploit dilemma* during the preference elicitation phase in a principled manner.

## 3 PRELIMINARIES

In this section, we first present our formal model and then review A-CEEI and its practical implementation, CM.

### 3.1 Formal Model

Let $N = \{1, \ldots, n\}$ denote the set of students indexed by $i$, and let $M = \{1, \ldots, m\}$ denote the set of courses indexed by $j$. Each course $j$ has a *capacity* $q_j \in \mathbb{N}_{>0}$. Each student $i$ has a set $\Psi_i \subseteq 2^M$ of *permissible course schedules.* $\Psi_i$ encapsulates both scheduling constraints as well as any student-specific constraints. An indicator vector $x \in \mathcal{X} = \{0, 1\}^m$ denotes a course schedule where $x_j = 1$

iff course $j \in M$ is part of schedule $x$. We let $a = (a_i)_{i \in N} \in \mathcal{X}^n$ denote an *allocation* of course schedules to students, where $a_i$ is the course schedule of student $i$. A course $j$ is *oversubscribed* in an allocation $a$ iff $\sum_{i=1}^{n} a_{ij} > q_j$ and *undersubscribed* iff $\sum_{i=1}^{n} a_{ij} < q_j$. We denote the set of feasible allocations by $\mathcal{F} = \{a \in \mathcal{X}^n : \sum_{i=1}^{n} a_{ij} \leq q_j \ \forall j \in M, a_i \in \Psi_i \ \forall i \in N\}$. Students' preferences over course schedules are represented by their (private) *utility functions* $u_i : \mathcal{X} \to \mathbb{R}_+$, $i \in N$, i.e., $u_i(x)$ represents student $i$'s utility for course schedule $x$.

## 3.2 Approximate Competitive Equilibrium from Equal Incomes (A-CEEI)

Budish [2011] proposed the A-CEEI mechanism as an approximation to the competitive equilibrium from equal incomes (CEEI). A-CEEI simulates a virtual economy where students are assigned budgets that are approximately (but not exactly) equal. To introduce A-CEEI formally, we represent each student $i$'s complete ordinal preferences by $\preceq_i$. Each student $i$ is allocated a budget $b_i \in [1, 1+\beta]$, $\beta > 0$. Next, *approximate market-clearing prices* $p^* \in \mathbb{R}^m_{\geq 0}$ are calculated (where $p^*_j$ is the price for course $j$) such that, when each student $i$ purchases her favorite permissible schedule $a^*_i$ within her budget, the market approximately clears. Formally, given an allocation $a^*$, the *clearing error* $z_j$ for course $j$ and price $p^*_j$ is

$$z_j := \begin{cases} \sum_i a^*_{ij} - q_j & p^*_j > 0, \\ \max\left\{\sum_i a^*_{ij} - q_j, 0\right\} & p^*_j = 0. \end{cases} \tag{1}$$

The *clearing error* of the allocation $a^*$ is defined as $\alpha := \sqrt{\sum_j z_j^2}$. Following Budish [2011], we say that a price vector $p^*$ *approximately clears* the market if $\alpha \leq \sqrt{\sigma m}/2$, with $\sigma = \min\{2k, m\}$, where $k$ is the maximum number of courses in a permissible schedule.[3] Each student $i$ is allocated her utility-maximizing schedule $a^*_i$ that is permissible and within her budget. Formally,

$$a^*_i \in \underset{\preceq_i}{\arg\max} \left[ a_i \in \Psi_i : \sum_j a_{ij} p^*_j \leq b_i \right]. \tag{2}$$

Then $[a^*, b, p^*]$ constitutes an $(\alpha, \beta)$-A-CEEI.

## 3.3 Course Match (CM)

A-CEEI has many attractive properties (see Section 2), but it cannot be directly implemented in practice for multiple reasons. First, A-CEEI assumes access to the students' *full* ordinal preferences. Second, A-CEEI only *approximately* clears the market, which implies that some courses could be oversubscribed (which would violate a hard capacity constraint in many business schools, where seats cannot easily be added to a classroom). Third, the combinatorial allocation problem is PPAD-complete [Othman et al., 2016]; thus, we do not have a polynomial-time algorithm to solve it. To address these challenges, Budish et al. [2017] introduced *Course Match (CM)* as a practical implementation of A-CEEI. In CM, students first report their preferences using the GUI (see Section 1.1); the final allocation is then computed in three stages (see Figure 1).

*Stage 1.* In Stage 1, CM uses *tabu search* [Glover et al., 2018] to find a price vector that constitutes an A-CEEI. To do this, for every price vector examined, for every student, a MIP has to be solved to determine the student's utility-maximizing schedule within her budget.

*Stage 2.* In Stage 2, CM removes oversubscription by iteratively increasing the price of the most oversubscribed course until no oversubscribed courses are left.

[3] Budish [2011] proved that for any $\beta > 0$ such a price vector always exists.

*Stage 3.* In Stage 3, CM reduces undersubscription by first increasing all students' budgets by a fixed percentage and then allowing students, one after the other, to "purchase" courses that still have seats available.

## 4 MACHINE LEARNING-POWERED COURSE MATCH (MLCM)

In this section, we introduce our *ML-powered Course Match* (MLCM) mechanism and discuss its theoretical properties. Here, we describe MLCM for a generic ML model denoted $\mathcal{M}$. In Section 6, we instantiate the ML model using MVNNs.

### 4.1 Details of MLCM

MLCM proceeds in five phases (see Figure 1).

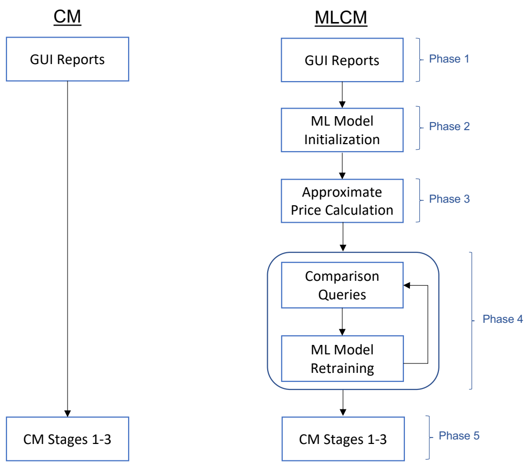

Fig. 1. Schematic overview of CM and MLCM

*Phase 1: Preference Reporting via GUI.* In Phase 1, students initially report their preferences using the same reporting language (and same GUI) as in CM (see Section 1.1). After this phase, each student can decide whether she also wants to use MLCM's ML-based preference elicitation feature (which we will simply call the "ML feature" going forward). If a student decides to "opt out", then MLCM treats that student's preference reports in the same way as CM would, without employing any ML.

*Phase 2: ML Model Initialization.* In Phase 2, for each student $i$ that has not opted out of the ML feature, MLCM creates an initial ML model of her utility function based on her GUI reports from Phase 1. To do so, MLCM creates a *cardinal* training data set $D_{i,\text{card}}$ consisting of $\ell$ schedule-value pairs implied by student $i$'s reports from Phase 1, i.e., $D_{i,\text{card}} = \{(x_{ik}, \hat{u}_i(x_{ik}))\}_{k=1\ldots\ell}$ (see Appendix A for details) and then uses $D_{i,\text{card}}$ to train an initial ML model $\mathcal{M}_i^0 : \mathcal{X} \to \mathbb{R}_+$, where $\mathcal{M}_i^0(x)$ denotes the ML model's prediction of student $i$'s utility for schedule $x$. See Section 6, for how to choose the ML model class.

*Phase 3: Approximate Price Calculation.* Next, MLCM runs Stage 1 of CM to calculate approximately market-clearing prices for all courses. As those prices are not final, but are only used to steer the preference elicitation in Phase 4, one can use less computation time for this step than in Phase 5, accepting a larger clearing error.[4] For those students who have opted out of the ML feature, MLCM uses the values implied by the CM GUI; for every other student $i$, it uses the values predicted by the ML model $\mathcal{M}_i^0$.

*Phase 4: ML-based Preference Elicitation.* In this phase, MLCM uses an ML-powered algorithm to generate a sequence of comparison queries (CQs) for every individual student. The algorithm has three main steps: (1) maintain an ordered list $S_i$ of already queried schedules; (2) use the ML model to determine the next schedule $x$ with the highest predicted value; (3) use "binary search" to generate a sequence of CQs, until the new schedule $x$ can be sorted into $S_i$. Note that students can

[4]If it was desired that students could immediately start answering comparison queries after reporting their initial preferences via the GUI, one could also use last year's prices for Phase 4 (and setting the price of new courses to some average price). Of course, if students' preferences for courses have changed significantly, or if the new courses are particularly popular or unpopular, then using last year's prices would lower the effectiveness of the ML-powered elicitation phase.

**ALGORITHM 1:** ML-powered CQ Generation for Student $i$

**Input**: Initial ML model $\mathcal{M}_i^0$, cardinal data set $D_{i,\text{card}}$, price vector $\hat{p}$, permissible schedules $\Psi_i$, budget $b_i$
**Output**: Updated ML model $\mathcal{M}_i^t$

```
1:  $S_i^0 = \{x \in \arg\max_{x' \in \Psi_i \cap D_{i,\text{card}} : x' \cdot \hat{p} \le b_i} \mathcal{M}_i^0(x')\}$
2:  for $t = 1$ to $\infty$ do
3:      $x \in \arg\max_{x' \in \Psi_i, x' \notin S_i^{t-1} : x' \cdot \hat{p} \le b_i} \mathcal{M}_i^{t-1}(x')$
4:      $CQs^t$ = BinarySearchQueries($S_i^{t-1}, x$)
5:      if student $i$ answered all $CQs^t$ then
6:          $S_i^t$ = Sort($S_i^{t-1}, x$) based on answers to $CQs^t$
7:          $D_{i,\text{ord}}^t$ = All pairwise orderings implied by $S_i^t$
8:          $\mathcal{M}_i^t$ = Train($D_{i,\text{card}}, D_{i,\text{ord}}^t$)
9:      else
10:         $D_{i,\text{ord}}^t = D_{i,\text{ord}}^{t-1} \cup$ {answers to $CQs^t$}
11:         $\mathcal{M}_i^t$ = Train($D_{i,\text{card}}, D_{i,\text{ord}}^t$)
12:         break
13:     end if
14: end for
15: return $\mathcal{M}_i^t$
```

answer as many of these CQs as they like, and stop at any time. Algorithm 1 formally describes the process. In Line 1, we create the initial set of totally ordered schedules $S_i^0$, which only contains the one schedule $x'$ from the cardinal training set with the highest predicted value $\mathcal{M}_i^0(x')$. Then we iteratively repeat the following procedure: In Line 3, we determine the highest-valued schedule $x$ according to $\mathcal{M}_i^{t-1}$ that is not contained in $S_i^{t-1}$, subject to feasibility and budget constraints. In Line 4, we ask student $i$ the CQs to determine $x$'s position in the ordered set $S_i^{t-1}$. If $i$ answered all CQs, then in Line 6, we completely order $S_i^t = S_i^{t-1} \cup \{x\}$ based on her answers, and then in Line 7, we create the ordinal training set $D_{i,\text{ord}}^t$ that contains all pairwise orderings implied by $S_i^t$. Finally, we train $\mathcal{M}_i^t$ on both $D_{i,\text{card}}$ and the ordinal set $D_{i,\text{ord}}^t$. If the student did not answer all CQs, then in Line 10, we append her (partial) answers to the previous ordinal data set $D_{i,\text{ord}}^t$, and then in Line 11, we train $\mathcal{M}_i^t$ using both data sets. This is the final ML model we return in Line 15.

Remark 4.1. *One could envision several alternative query heuristics. We have already conducted experiments using the following alternatives: (1) comparing the currently predicted utility-maximizing bundle only against the best one queried previously, and (2) comparing at each iteration the two schedules with the highest predicted utility. However, both approaches led to worse results compared to Algorithm 1. See Section 9 for possible future work on query generation procedures.*

*Phase 5: Computing the Final Allocation.* Finally, MLCM runs Stages 1–3 of CM to determine the final allocation. For those students who have opted out of the ML feature, MLCM uses the values implied by the CM GUI; for every other student $i$, it uses the utility function implied by $\mathcal{M}_i^t$.

Remark 4.2 (Runtime Considerations and Real-time Interaction). *In MLCM, Phases 3 (approximate price vector calculation) and 5 (final allocation calculation) are computationally expensive, because in those phases, MLCM computes an A-CEEI. However, this is not a concern in practice, because both of those phases do not happen in real-time (and furthermore, this computation is embarrassingly parallelizable). For a large school, one would use a compute cluster to calculate an A-CEEI for both of these phases. Furthermore, in work concurrent to the present paper, Budish et al. [2023] introduced a*

| Preference Model | Courses 1 | 2 | 3 | 4 | Utility Maximizing Schedule $a_1^*$ | Utility $u_1(a_1^*)$ | Answer to CQ |
|---|---|---|---|---|---|---|---|
| $u_1$ | 85 | 70 | 50 | 40 | {1,3} | 135 | |
| $u_1^{GUI}$ | 75 | 77 | 42 | 45 | {2,4} | 110 | |
| $\mathcal{M}_1^{t=0}$ | 75 | 77 | 42 | 45 | {2,4} | 110 | $\{1,4\} \succ \{2,4\}$ |
| $\mathcal{M}_1^{t=1}$ | 80 | 72 | 42 | 45 | {1,4} | 125 | $\{1,3\} \succ \{1,4\}$ |
| $\mathcal{M}_1^{t=2}$ | 80 | 72 | 47 | 40 | {1,3} | 135 | $\{1,4\} \succ \{2,3\}$ |
| $\mathcal{M}_1^{t=3}$ | 80 | 72 | 47 | 40 | {1,3} | 135 | $\{2,3\} \succ \{2,4\}$ |
| $\mathcal{M}_1^{t=4}$ | 80 | 72 | 47 | 40 | {1,3} | 135 | |

Table 1. Worked example illustrating the ML-based preference elicitation algorithm. Each row represents the linear coefficients (corresponding to the base values) that uniquely define the corresponding function, the current utility maximizing schedule $a_1^*$, its corresponding utility $u_1(a_1^*)$ and the answer to the CQ.

*novel algorithm for computing an A-CEEI that is multiple orders of magnitudes faster than the tabu search currently in use, which will likely remove any runtime concerns in the near future.*

*The only phase with real-time interaction is Phase 4 (ML-based Preference Elicitation), which is computationally very cheap for our Algorithm 1. There are two distinct cases. If a student answered a CQ that did not complete an iteration of binary search (Line 4 in Algorithm 1), then generating the next query takes milliseconds. If a student answered a CQ that completed a round of binary search, then for the ML model class we selected, retraining the student's ML model (Line 8 in Algorithm 1) and solving the MIP to determine the bundle for the next iteration of binary search (Line 3 in Algorithm 1) takes less than two seconds, and can be further improved with the use of GPUs (please see Section 6 for details on the selected ML model class). Thus, in terms of real-time interaction, students do not perceive the computational cost, which highlights MLCM's practicability.*

## 4.2 MLCM - A Worked Example

In this subsection, we present a worked example to illustrate MLCM's ML-based preference elicitation algorithm (i.e., *Phase 4*). Additionally, this example provides intuition for how the CQs help the ML model correct students' initial reporting mistakes. Note that we assume that students make no mistakes when answering CQs.

Example 4.3 (Correcting reporting mistakes). *We consider a setting with four courses $M := \{1,2,3,4\}$ with capacity 1 each. There is a single student $N := \{1\}$ who has a budget of $b_1 = 1$ and who wants a schedule consisting of at most two courses (and there are no other student-specific constraints). The prices of the courses are $\hat{p} := (0.6, 0.6, 0.3, 0.3)$, such that the student can afford all bundles of size two, except for $\{1,2\}$. For ease of exposition, we assume that the student has additive preferences and that the GUI only allows for additive preference reports. The whole example is presented in Table 1. The student's true utility function $u_1$ is presented in row 1. The student's GUI reports $u_1^{GUI}$ (which include reporting mistakes) are shown in row 2. As the ML model $\mathcal{M}_1^t$ we use linear regression, such that each linear coefficient can be interpreted as an estimate of the student's base value for a course. Let $\mathcal{M}_1^t(x)$ denote the ML model trained on the cardinal data $D_{1,card}$ as well as $t \in \mathbb{N}_0$ answered CQs.*

*First, given $u_1^{GUI}$, MLCM constructs a cardinal training data set $D_{1,card}$ (see Section A). Next, MLCM fits the linear regression model $\mathcal{M}_1^{t=0}$ on $D_{1,card}$.[5] We see that $\mathcal{M}_1^{t=0}$ is able to perfectly fit*

[5]We train the linear regression models using gradient descent, because (i) we restrict them to be monotone, i.e., their coefficients are constrained to be non-negative, and (ii) this also allows us to train them on both cardinal and ordinal data.

$u_1^{GUI}$. *Next, MLCM generates the first CQ by comparing the two admissible schedules within budget that have the highest and second highest predicted utility with respect to* $\mathcal{M}_1^{t=0}$, *i.e.,* $\{2,4\}$ *with* $\mathcal{M}_1^{t=0}(\{2,4\}) = 122$ *and* $\{1,4\}$ *with* $\mathcal{M}_1^{t=0}(\{1,4\}) = 120$. *When presenting this CQ to the student, she answers that she actually prefers* $\{1,4\}$ *with a true utility of* $u_1(\{1,4\}) = 125$ *over* $\{2,4\}$ *with true utility* $u_1(\{2,4\}) = 110$. *Next, MLCM updates the ordinal training data set as* $D^1_{1,ord} = \{\{1,4\} \succ \{2,4\}\}$ *and retrains the model* $\mathcal{M}_1^{t=1}$ *on the combined ranking and regression loss (see Section 6.3). We see that the regression coefficient for course 1 of* $\mathcal{M}_1^{t=1}$ *increased from 75 to 80 and the coefficient for course 2 decreased from 77 to 72. Now the model* $\mathcal{M}_1^{t=1}$ *correctly predicts the ordinal ranking of the above elicited CQ, i.e.,* $\mathcal{M}_1^{t=1}(\{1,4\}) = 125 > 117 = \mathcal{M}_1^{t=1}(\{2,4\})$ *and thus has corrected the student's initial reporting mistake on her two most preferred courses. We see that after a single CQ, the student's utility for her (predicted) utility maximizing schedule* $u_1(a_1^*)$ *has already increased from* 110 *to* 125. *At this point, the first iteration of "binary search" (Line 4 in Algorithm 1) has been completed (here, only leading to one CQ). For the next CQ, MLCM selects* $\{1,3\}$ *– the schedule with the highest predicted utility that has not been elicited so far. Then MLCM performs binary search again, trying to insert* $\{1,3\}$ *into the already elicited list of schedules (Line 4 in Algorithm 1) resulting in the CQ consisting of the schedules* $\{1,3\}$ *and* $\{1,4\}$.[6] *Since the student prefers the schedule* $\{1,3\}$, *now the complete list of bundles* $\{1,3\}, \{1,4\}$ *and* $\{2,4\}$ *can be sorted and binary search is completed. After retraining with the updated ordinal training data set* $D^2_{1,ord} = \{\{1,4\} \succ \{2,4\}, \{1,3\} \succ \{1,4\}, \{1,3\} \succ \{2,4\}\}$, *the new model* $\mathcal{M}_1^{t=2}$ *represents the correct ordinal ranking of all admissible schedules and thus already found the true utility maximizing schedule* $\{1,3\}$. *The next schedule selected by MLCM is* $\{2,3\} \notin D^2_{1,ord}$ *and binary search is again performed. Since the student's answers do not contradict the model's predicted ordinal ranking, the linear coefficients of* $\mathcal{M}_1^{t=3}$ *and* $\mathcal{M}_1^{t=4}$ *remain unchanged.*

For an example where the student has additionally forgotten some of her base values and how MLCM infers those missing base values, please see Appendix J.

## 4.3 Theoretical Properties of MLCM

Budish [2011] showed that A-CEEI satisfies *envy-bounded by a single good*, $(n+1)$*-maximin share guarantee*, and *Pareto efficiency*. If CM had access to the full true ordinal preferences, then the Stage 1 allocation of CM would also satisfy the same properties. In Appendix B, we prove that the same properties also hold in an *approximate* sense for the Stage 1 allocation of MLCM if the students' preferences are captured *approximately* via the ML models $\mathcal{M}_i$.

*Incentives.* Regarding incentives, Budish [2011] showed that A-CEEI is *strategyproof in the large (SP-L)*. Budish et al. [2017] argued that CM is also SP-L. Their argument proceeds in two steps. First, as the number of students increases, $p^*$ calculated at the end of Stage 2 will be exogenous to the students' reports, and hence the students become price-takers. Second, since the market-clearing error target $\alpha \leq \sqrt{\sigma m}/2$ is independent of the number of students, the probability that Stage 3 affects a student's allocation goes to zero as the market grows. In MLCM, the main difference is that we use the trained ML models instead of the GUI reports when calculating prices. However, students are still price-takers, and the probability that Stage 3 affects an individual student still goes to zero. Therefore, we argue that MLCM is also SP-L.

## 5 COURSE ALLOCATION SIMULATION FRAMEWORK

In this section, we describe our student preference generator (Section 5.1), how we model students' reporting mistakes (Section 5.2), and how we calibrate the parameters to real-world data (Section 5.3). For additional details, please see Appendices C to E.

[6]When performing binary search, we break ties in favor of the schedule with the higher predicted utility.

## 5.1 Student Preference Generator

We have two goals for our student preference generator. First, students' preferences (and their reports) should be realistic, i.e., they should closely match real-world data on the usage of CM. Second, we must be able to encode each student's complete preferences as a MIP so that we can compute an (optimal) benchmark allocation given the students' true preferences.

*Correlation.* One of the key features the simulator must capture is some notion of *correlation* between students' preferences. To this end, we divide courses into *popular* and *unpopular*. Popular courses are those that many (but not necessarily all) students have a high value for. Concretely, for every student $i$, we randomly select a set of *favorite* courses from the set of popular courses. Then, for each course, student $i$'s base value is drawn from some distribution, where the mean of that distribution is high for her favorite courses and low for all others. Note that a smaller number of popular courses implies higher correlation, as more students will have the same favorite courses. Thus, we can use the number of popular courses to control the degree of correlation.

*Complementarities and Substitutabilities.* To model complementarities and substitutabilities between courses, we build on prior experimental work modeling bidders in combinatorial auctions [Goeree and Holt, 2010, Scheffel et al., 2012]. Concretely, following Scheffel et al. [2012], we assume that courses lie on a latent space and that the distance between courses in that space defines the students' view on the complementarity/substitutability of those courses. Figure 2 depicts an example latent space with $M = \{1, \ldots, 30\}$. The set of popular courses are $M_p = \{8, 9, 21, 29\}$ and are marked with a star. Assume that student $i$'s favorite courses are 9 and 21. From those, we randomly draw a set of *centers* – which is course 9 in Figure 2. All courses with $L_1$ distance smaller or equal to 1 from that center form the set of substitutabilities $\{9, 3, 8, 10, 15\}$. All courses with $L_\infty$ distance smaller or equal to 1 from that center (that are not in the set of substitutabilities) and the center 9 form the set of complementarities $\{9, 2, 4, 14, 16\}$. The number of centers and the distances control the *degree* of complementarity and substitutability.

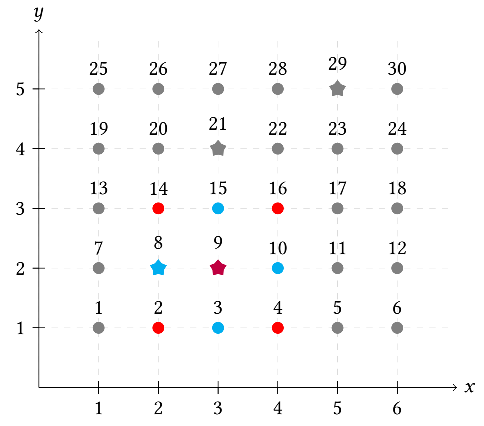

Fig. 2. A latent space with 30 courses.

Of course, one could use other topologies that result in the same or very similar preferences – and for the simulator, only the induced preferences matter. However, using our proposed common latent space enables a concise, mathematically tractable formalization of substitutabilities/complementarities. With this, our framework provides us with granular (and interpretable) control over the degree of complementarities/substitutabilities. As we will show in Section 5.3, this design allows us to produce instances very similar to those observed in practice by Budish and Kessler [2022].

*Total utility calculation.* We calculate a student's utility for a schedule based on her values for single courses and the number of courses from each substitutability/complementarity set in the schedule. In Appendix C, we provide mathematical details for the preference generator. In Appendix D, we provide the corresponding MIP formulation.

## 5.2 Students' Reporting Mistakes

Students can make mistakes when reporting their base value for single courses or when reporting pairwise adjustments. To capture these reporting mistakes, we use the following four parameters:

(1) $f_b \in [0,1]$ denotes the probability that a student forgets one of her *base values*. We assume that a student forgets to report a base value for her lower-valued courses first. For example, if $M = \{1,2,3\}$, and a student values these courses as 2, 5 and 10, she will first forget course 1.
(2) $f_a \in [0,1]$ denotes the probability a student forgets to report one of her adjustments (out of those adjustments for which she has not forgotten to report a base value for either of the courses in the pair). We assume that students forget adjustments uniformly at random.
(3) $\sigma_b \in \mathbb{R}_{\geq 0}$ denotes the standard deviation of the additive Gaussian noise $\mathcal{N}(0, \sigma_b^2)$ with which the students report their *base values.*
(4) $\sigma_a \in \mathbb{R}_{\geq 0}$ controls the support of the multiplicative uniform noise $\mathcal{U}[1-\sigma_a, 1+\sigma_a]$ with which the students report their *adjustment values.*

## 5.3 Calibration of the Simulation Framework

We calibrate the parameters of our framework (i.e., the preference generator and the reporting mistakes) to match the experimental results from Budish and Kessler [2022]. The key metrics from their experiment are:

(1) Students only report a base value for 49.9% of the courses.
(2) The number of courses with a reported value in $[50, 100]$ is approximately equal to the number of courses with a reported value in $[0, 50]$.
(3) Students report between 0 and 10 pairwise adjustments.
(4) Students report an average of 1.08 pairwise adjustments; the median is equal to 0.
(5) Students were asked to compare the schedule they received with several other course schedules. Their answers were consistent with their reported preferences in 84.41% of the cases, and in case of disagreements, the median utility difference, based on their reports, was 13.35%.

In Appendix E Table 5, we provide detailed results of our calibration procedure. In our experiments with 6 popular courses we set $(f_b, f_a, \sigma_b, \sigma_a) = (0.5, 0.4825, 17, 0.2)$, and with 9 popular courses we set $(f_b, f_a, \sigma_b, \sigma_a) = (0.5, 0.48, 23, 0.2)$. With these parameters, we are able to match metrics (1)-(3) exactly. For metric (4), our mean is 10% lower (in one of two settings), while our median is slightly larger (1 instead of 0). Regarding metric (5), both accuracy and scaled median utility difference are within 3 percentage points of the reported one. Thus, our framework produces instances very similar to those described in [Budish and Kessler, 2022]. See Appendix E for details.

# 6 MACHINE LEARNING INSTANTIATION

In this section, we instantiate the ML model used by MLCM.

## 6.1 Machine Learning Model Desiderata

There are three important desiderata for the choice of the ML model class. First, it needs to be sufficiently *expressive* to be able to learn students' complex preferences over course schedules (including complementarities and substitutabilities). Second, we must be able to train the ML model using both the *cardinal* input that students provide via the CM reporting language as well as the *ordinal* input from answering CQs. Third, in Phases 3–5 of MLCM, we must be able to compute each student's utility-maximizing schedule, given the ML model predictions, budget, and prices. More formally:

$$x \in \underset{x \in \Psi_i, x \cdot p \leq b_i}{\arg\max} \ \mathcal{M}_i(x) \tag{3}$$

| Data Type | Ridge | nuSVR-gauss | xgboost | NN | MVNN |
|---|---|---|---|---|---|
| GUI | 34.01±1.18 | 34.85±1.13 | 34.99±1.23 | 34.85±1.12 | 34.52±1.42 |
| 30 RAND VQs | 19.36±1.04 | 21.76±0.81 | 23.01±0.88 | 23.25±0.88 | 18.64±0.99 |
| 50 RAND VQs | 12.88±0.83 | 14.31±0.59 | 15.79±0.54 | 15.71±0.65 | 11.93±0.65 |
| 100 RAND VQs | 9.87±0.68 | 9.40±0.50 | 15.75±0.55 | 9.32±0.56 | 5.84±0.53 |
| 150 RAND VQs | 9.18±0.61 | 7.89±0.44 | 13.72±0.41 | 5.62±0.48 | 3.50±0.38 |

Table 2. Comparison of different ML models. Shown are MAEs on the test set and 95% CIs. Winners marked in grey.

Thus, a key requirement on the ML model $\mathcal{M}_i$ is that Equation (3) can be solved fast enough. For this, we adopt as a requirement that Equation (3) can be translated into a succinct MIP.

## 6.2 Generalization Performance

Next, we compare the generalization performance of different ML models to identify the best one for our domain.

*Experiment Setup.* As in the experiment by Budish and Kessler [2022], we consider a setting with 25 courses. We use our simulator to create 100 instances of student preferences, where we use 20 to tune the ML hyperparameters, and 80 for testing. We use two different types of training data sets. First, we use as the training set the cardinal data $\{D_{i,\text{card}}\}_{i \in N}$ generated using the same procedure as Phase 2 of MLCM (based on the students' GUI reports) (see Appendix A). Second, we use $\{30, 50, 100, 150\}$ *random* value queries as the training data set. To test the trained models, we use the complement of these training sets as the test sets and report the mean absolute error (MAE) and Kendall tau (KT) of each ML model. We use MAE and KT to determine the winners.

As ML models, we consider ridge linear regression (Ridge), nu-support vector regression with Gaussian kernel (nuSVR-gauss), XGBoost, neural networks (NN), and monotone-value neural networks (MVNNs). We provide a detailed description of our hyperparameter tuning procedure and the winner configurations in Appendix F.

*Results.* Table 2 presents MAE results for a setting with 9 popular courses (results for 6 popular courses, and for KT, are qualitatively similar; see Appendices G and H). We see that MVNNs have the best performance across all different training sets with regard to MAE and KT. For the GUI reports, we observe that all ML models perform on par. This can be explained by the fact that each ML model learns (approximately) the same quadratic preferences induced by the GUI on the training set and thus achieves out-of-sample the same test MAE. However, with random values queries as the training set, MVNNs significantly outperform the other models.

## 6.3 Integrating comparison queries into MVNNs

To simultaneously train MVNNs on GUI reports (regression data) and CQs (classification data), we use a method by Sculley [2010] called *combined ranking and regression (CRR).* CRR trains regression models using both regression and classification data. For this, CRR randomly alternates in each gradient step between a regression and a classification loss. Specifically, with probability $\alpha \in [0, 1]$, CRR selects a regression loss $l_{reg}(\cdot)$ and with probability $(1 - \alpha)$ a classification loss $l_{class}(\cdot)$, respectively. In case the classification loss is selected, the sigmoid of the difference between the predicted values for the two schedules included in the comparison query is interpreted as the probability with which the MVNN predicts that one schedule is better than the other, thus resulting in a classification problem. To train MVNNs in our setting, we use CRR with MAE and binary cross entropy as the regression and classification loss, respectively. See Appendix I for details.

### 6.4 MIP-Formalizability of the Utility Maximization Problem

Weissteiner et al. [2022a] already provided a very succinct MIP formulation for MVNNs. It is straightforward to adopt their MIP formulation to the constraints of our setting, making the optimization of the student's utility maximization problem (i.e., eq. (3)) practically feasible.

Given that MVNNs satisfy all three desiderata for the choice of the ML model class we have laid out above (see Section 6.1), we adopt MVNNs as the ML model for MLCM.

## 7 EXPERIMENTAL EVALUATION

In this section, we experimentally evaluate the performance of MLCM using our simulation framework from Section 5. We provide the source code for all experiments in the supplementary material.

### 7.1 Experiment Setup

We consider a setting with 25 courses and 100 students.[7] We consider 6 and 9 popular courses (where 6 popular courses corresponds to extreme correlation). As in the lab experiment of [Budish and Kessler, 2022], every student wants a schedule of (at most) five courses. Thus, for 100 students the total demand is 500 seats. The *supply ratio* (SR) denotes the fraction of the total number of seats to the total demand. Thus, a SR of 1.5 means that there are a total of 750 course seats (split equally among all courses). We consider SRs of 1.25 and 1.5.

To compare CM and MLCM, we assume that both receive the same reports from the CM reporting language, with students' reporting mistakes calibrated as described in Section 5.3. For MLCM, each student additionally answers 1, 5, 10, 15, or 20 CQs. We denote these mechanisms as CM and MLCM (1/5/10/15/20 ML-BASED CQs), respectively. We assume that students make no mistakes when answering CQs.[8] We further consider four benchmarks. The first is CM (No Mistakes), which is CM, but assuming no reporting mistakes. The second is CM$^*$ (Full Preferences), which is a modified version of CM that takes as input the *full* correct cardinal preferences of the students. Importantly, CM$^*$ includes interactions between three or more courses, which cannot be expressed using CM's reporting language. Furthermore, we use a benchmark called MLCM (20 RANDOM CQs), where each student answers 20 *randomly* generated CQs. Finally, we also use random serial dictatorship (RSD) as another benchmark, with the GUI reports as input and the same amount of reporting mistakes.

We use the same 100 instances to test each mechanism. We use a maximum of 10 random restarts for the *tabu search* of Stage 1. Furthermore, we use three hidden layer MVNNs with 20 units per layer such that they provide a good trade-off between generalization performance and computational cost of the utility maximization MIP (Equation (3)). See Appendix K for details on the selected MVNN hyperparameters and the computing infrastructure. Unless otherwise noted, CQs denote *ML-based CQs*.

[7]We selected 25 courses to match the number of courses in the lab experiments of [Budish and Kessler, 2022], allowing us to properly calibrate the mistake profile of the students (see Section 5.3). However, since this calibration does not depend on the number of students, we increased the number of students from about 20 in [Budish and Kessler, 2022] to 100. Consequently, the course capacity increased from approximately 5 to 25, which means that every individual student's reports have a smaller impact on the (over-)demand for individual courses. This makes this setting more realistic and reduces variance during experimentation, while still keeping the runtime for our experiments manageable. Recall that the results from the lab experiments with 20 students per instance extrapolated well to Wharton with over 1700 students.

[8]Recall that pairwise comparison queries are known to have low cognitive load [Chajewska et al., 2000, Conitzer, 2009]. Furthermore, Budish and Kessler [2022] used the students' answers to pairwise comparison queries as *ground truth* for the same problem.

| Mechanism | Avgerage Student Utility |  |  | Minimum Student Utility |  |  | Overs. | Time |
|---|---|---|---|---|---|---|---|---|
|  | Stage 1 | Stage 2 | Stage 3 | Stage 1 | Stage 2 | Stage 3 | Stage 1 | in h |
| CM* (Full Preferences) | 100.0 ± 0.0 | 95.4 ± 0.9 | 98.1 ± 0.7 | 73.2 ± 0.9 | 70.9 ± 1.1 | 72.1 ± 1.0 | 7.2 ± 0.9 | 4.1 |
| CM (No Mistakes) | 98.6 ± 0.3 | 95.3 ± 0.7 | 97.4 ± 0.5 | 72.5 ± 0.9 | 70.5 ± 1.1 | 71.6 ± 1.0 | 5.7 ± 0.8 | 3.7 |
| RSD | - | - | 76.8 ± 0.5 | - | - | 29.8 ± 1.3 | - | 0.0 |
| CM | 79.6 ± 0.5 | 77.9 ± 0.6 | 79.0 ± 0.5 | 42.6 ± 1.2 | 41.5 ± 1.1 | 42.1 ± 1.1 | 2.9 ± 0.5 | 2.0 |
| MLCM ( 1 ML-based CQ) | 79.6 ± 0.5 | 77.4 ± 0.9 | 78.7 ± 0.8 | 41.3 ± 1.1 | 40.1 ± 1.2 | 41.2 ± 1.2 | 3.3 ± 0.5 | 23.2 |
| MLCM ( 5 ML-based CQs) | 83.9 ± 0.5 | 80.7 ± 1.2 | 82.1 ± 1.1 | 46.6 ± 1.3 | 44.0 ± 1.4 | 45.1 ± 1.5 | 3.0 ± 0.5 | 23.5 |
| MLCM (10 ML-based CQs) | 86.3 ± 0.5 | 82.8 ± 1.0 | 84.6 ± 0.9 | 50.5 ± 1.1 | 47.6 ± 1.3 | 48.3 ± 1.3 | 3.7 ± 0.6 | 26.0 |
| MLCM (15 ML-based CQs) | 88.0 ± 0.4 | 84.5 ± 0.9 | 86.4 ± 0.7 | 52.0 ± 1.3 | 49.2 ± 1.4 | 50.2 ± 1.4 | 4.3 ± 0.7 | 27.1 |
| MLCM (20 ML-based CQs) | 89.4 ± 0.5 | 86.4 ± 0.8 | 88.0 ± 0.7 | 53.3 ± 1.3 | 50.5 ± 1.4 | 51.4 ± 1.5 | 3.3 ± 0.5 | 27.4 |
| MLCM (20 Random CQs) | 78.9 ± 0.6 | 77.0 ± 0.8 | 78.2 ± 0.7 | 41.0 ± 1.2 | 40.3 ± 1.2 | 40.6 ± 1.2 | 3.2 ± 0.5 | 25.0 |

Table 3. Comparison of RSD, CM and MLCM (also using random comparison queries) in Stages 1–3 for a supply ratio of 1.25, 9 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

## 7.2 Welfare Results

In Table 3, we present results for SR = 1.25 (which is very close to Wharton's SR; see [Budish and Kessler, 2022]) and 9 popular courses. The results are better for SR = 1.5 and worse for 6 popular courses (see Appendix L). We normalize all results by the average utility of CM* after Stage 1, so that all utility metrics can be reported in percent. The metrics of interest are the *average* and *minimum* student utility after Stage 3 (i.e., for the final allocation). We see that MLCM (10 ML-BASED CQs) significantly outperforms CM, both in average and minimum student utility.[9] In particular, MLCM (10 ML-BASED CQs) increases average utility from 79.0% to 84.6% (a 7.1% increase) and minimum utility from 42.1% to 48.3% (a 14.7% increase). As the number of CQs increases, the performance of MLCM improves further. Compared to CM, MLCM (20 ML-BASED CQs) increases average utility from 79.0% to 88.0% (an 11.4% increase) and minimum utility from 42.1% to 51.4% (a 22.1% increase).

Recall that, if a student answers zero CQs, then MLCM treats her preferences in the exact same way as CM (i.e., only using her GUI reports). Table 3 shows that, if the student answers a single CQ, then the performance of MLCM is statistically on par with CM's performance, and the performance only improves by increasing the number of CQs. Thus, MLCM never performs worse than CM. Additionally, we observe that asking 20 *random* CQs does *not* improve upon the performance of CM. This shows the importance of asking "smart" queries in MLCM. Furthermore, we see that switching from RSD to CM leads to a surprisingly small increase in average student utility, while it does substantially increase minimum student utility. This further highlights the performance of MLCM, since MLCM significantly improves both average and minimum student utility compared to CM. Finally, we observe that the performance difference between CM* (full preferences) and CM (No Mistakes) is not significant. This shows that our choice of hyperparameters for the simulation framework results in preferences that can be captured well by the CM language (which can capture at most pairwise interactions). Thus, the welfare improvements of MLCM over CM are primarily due to its ability to correct students' reporting mistakes, and not due to MLCM's ability to (in principle) capture more than pairwise interactions between courses.

[9]See Appendix M for statistical tests for all such statements.

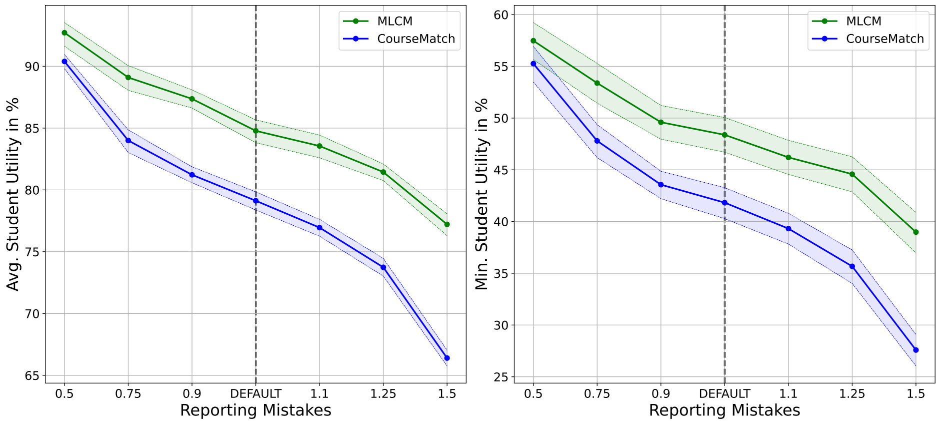

Fig. 3. Reporting mistakes ablation experiment for a supply ratio of 1.25 and 9 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

## 7.3 Reporting Mistakes Ablation Study

We now vary how many reporting mistakes students make. For this, we keep the setting fixed (SR 1.25 and 9 popular courses) and multiply all parameters of the students' mistake profile (i.e., $f_b$, $f_a$, $\sigma_b$ and $\sigma_a$) by a common constant $\gamma$. For $\gamma < 1$, students make fewer mistakes than in the default profile, while for $\gamma > 1$, the opposite is true. Importantly, $\gamma$ does not linearly affect the students' mistakes. For example, for $\gamma = 0.5$, students make about 50% *fewer* mistakes compared to $\gamma = 1$, but the *severity* of these mistakes (utility difference) is only approximately 25% as large (see Appendix E). For each $\gamma$, we run the same 50 instances for CM and MLCM, with each student answering 10 CQs in MLCM.

Figure 3 shows the results of the ablation study (see Appendix N for SR = 1.5 and 6 popular courses). As $\gamma$ increases, the performance of both, CM and MLCM, monotonically decreases. MLCM significantly outperforms CM for all $\gamma \in [0.5, 1.5]$. As $\gamma$ increases, the relative performance gap between the two mechanisms gets significantly larger. For $\gamma = 1.5$, MLCM performs 16.2% and 41.3% better than CM in terms of average and minimum student utility, respectively. Those results could be further improved by retuning MLCM's hyperparameters for each value of $\gamma$.

## 7.4 Should Individual Students Opt Into the ML Feature?

Suppose that MLCM is implemented in practice, and an individual student must decide whether to opt into MLCM's ML feature or not. How much would the student benefit, if (a) no other student opted into the ML feature, or (b) if everyone else also opted in?

*No Other Student to Opt into the ML Feature.* We first consider the scenario where the student in question is considering opting in, and no one else does so. We study this by running both MLCM and CM twice – once where no students have opted in, and once where a single student has done so.[10] We use 20 instances and 100 students per instance. We report averages over those 2000 students. Table 4 shows the results for a SR of 1.25. We observe that the *expected relative gain* from opting in is at least 8.5% (across all settings). Furthermore, the student prefers the "MLCM schedule" in at

[10]Following [Budish and Kessler, 2022], we report results after Stage 1. Furthermore, to make the experiment computationally feasible, for each setting, we use the Stage 1 price vector that would result if no student would opt in (assuming that the student who is considering to opt in is a price taker).

| Setting | | | Preferred Mechanism | | | Gain from Opting Into MLCM | | |
|---|---|---|---|---|---|---|---|---|
| SR | #PoP | #CQs | MLCM | CM | Indiff. | Expected | if pref MLCM | if pref CM |
| 1.25 | 9 | 10 | 72.70% | 5.95% | 21.35% | 11.1% | 16.1% | -10.8% |
| 1.25 | 9 | 15 | 78.75% | 4.25% | 17.00% | 13.5% | 17.6% | -8.6% |
| 1.25 | 9 | 20 | 83.15% | 4.70% | 12.15% | 15.5% | 19.0% | -10.1% |
| 1.25 | 6 | 10 | 67.00% | 9.30% | 23.70% | 8.5% | 13.8% | -8.7% |
| 1.25 | 6 | 15 | 73.20% | 8.05% | 18.75% | 9.8% | 14.4% | -8.3% |
| 1.25 | 6 | 20 | 78.75% | 6.70% | 14.55% | 11.4% | 15.1% | -7.7% |

Table 4. Expected gain of opting into MLCM's ML feature when no other student opts in. Shown are average results across 2000 students per setting (SR,#PoP,#CQs). CIs for all metrics are $\approx 0$.

least 67% of the cases, while she prefers the "CM schedule" in at most 9.3% of the cases.[11] As the number of CQs the student answers increases, the benefit from opting into MLCM's ML feature becomes even larger. Finally, the improvement is larger for more popular courses and for a larger SR (see Appendix O for SR 1.5).

In Appendix O, we perform an analogous experiment for the case when all other students opt into MLCM's ML feature. These results (i.e., the expected utility gains) are almost identical.

## 7.5 Results for Additive Preferences

In this subsection, we test the robustness of our approach to changes in the *true* students' preferences. Specifically, we repeat all experiments described in the previous subsections for the simple case of students having *additive* true preferences.

We use our student preference generator (see Section 5.1 and Appendix C) to generate additive student utility functions and calibrate the mistake profile of the students so that both their accuracy and the reported utility difference in case of disagreements match those determined in the lab experiment of Budish and Kessler [2022] (see Table 36 in Appendix P). With these preferences and mistake profiles, we repeat the welfare experiment described in Section 7.2 for supply ratios of 1.25 and 1.5, the reporting mistakes ablation experiment described in Section 7.3, as well as the experiment investigating the expected gain of a student opting into MLCM's ML feature described in Section 7.4. The results of those experiments are qualitatively very similar to those for the original preferences and can be found in Appendix P. For example, for the most realistic supply ratio of 1.25, MLCM (10 ML-BASED CQs) increases average student utility by 7.0% and minimum student utility by 23.1%. Overall, these results show the robustness of our design and its applicability to a large range of settings.[12]

## 7.6 Scaling to Schools with more Courses

All experiments in Section 7 were performed in settings with 25 courses in order to match the experimental setup in [Budish and Kessler, 2022]. As the number of courses increases, the time required to solve the MIP that determines a student's most preferred schedule at a given price vector, which is required both for the calculation of the final allocation as well as the ML preference

[11]In the lab experiment of [Budish and Kessler, 2022], they compared CM to the previous mechanism used at Wharton, BPA. In those experiments, 42.4% of the students preferred the allocation under CM, 31.8% preferred BPA, while the remaining students were indifferent.

[12]Note that we did not perform hyperparameter optimization for this new setting, but instead we used the hyperparameters determined for the original setting described in Section 7.1, which further illustrates the robustness of our approach.

elicitation according to Algorithm 1, increases. This could raise some concerns about MLCM's applicability to a large school with hundreds of courses.

To test whether this is indeed an issue, we conducted the following experiment: Using our preference generator (Section 5.1), we generated problem instances varying the number of courses from 25 up to 350. For each problem instance, we trained an MVNN on the GUI reports of each student and measured the time required to determine that student's most preferred course schedule for a given price vector. For each number of courses, we tested 100 different students, each on the same 10 random price vectors. In Figure 4, we plot the average MIP solution times as well as their 95% confidence intervals.

From Figure 4, it becomes clear how gracefully the MIP solution time, for our choice of ML model, scales with the number of courses.[13] Concretely, a 14-fold increase in the number of courses, from 25 to 350, does not even lead to a doubling of the average MIP solution time. Thus, the runtime required for finding an A-CEEI would increase by at most a factor of 2, assuming that the number of price vectors evaluated by tabu search remains constant.[14] Regarding the real-time component of MLCM (i.e., generating pairwise comparison queries), note that even with 350 courses, we can still generate the query within at most two seconds (where most of the time is required for retraining the ML model, and not for solving the MIP).

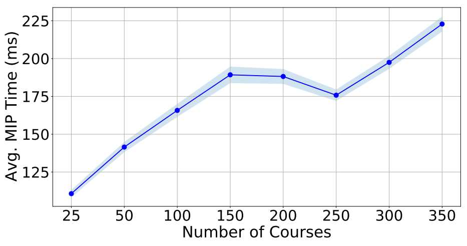

Fig. 4. MIP solution times for MVNNs as a function of the number of courses. Shown are average results over 100 students and 10 random price vectors including 95% CI.

## 8 DISCUSSION

For our most realistic setting, our results show that, compared to CM, MLCM with 10 comparison queries per student increases average student utility by over 7% and minimum student by almost 15%. Furthermore, the a priori expected utility gain of a student answering 10 comparison queries is over 10%. To put these numbers into perspective, consider that CM increased average student utility compared to RSD by less than 3% (while the increase in minimum student utility was large). Given this, we consider the improvements achieved by MLCM to be very large.

To realize these gains, the students incur additional costs, as they have to answer some CQs on top of reporting their preferences to the GUI. However, given that students are already encouraged to input their cardinal values to the GUI for at least 10 courses [Wharton, 2020], and given that pairwise comparison queries have a comparatively lower cognitive cost [Chajewska et al., 2000, Conitzer, 2009], we consider the relative increase in cost justified, given the large utility improvements.

Recall that MLCM uses a separate ML model for each student. An alternative approach would have been to instead use a *single* neural network for all students, leveraging possible similarities between students in the learning task. However, this approach might open new possibilities for

[13]For this test, we used the same MVNN architecture as for all experiments in Section 7, only changing the input layer to support the additional number of courses. Given that the structure of the students' preferences does not become significantly more complicated when increasing the number of courses (as suggested by Wharton's CM user manual, encouraging students willing to take up to 5 courses to report a base value for at least 10 courses, even though the school offers over 300 courses per semester [Wharton, 2020]), our original network architecture is still able to capture the students' preferences.
[14]Additionally, recall that the new algorithm for computing an A-CEEI that was introduced by Budish et al. [2023] (in work concurrent to the present paper) is multiple orders of magnitudes faster than the tabu search currently in use, which will likely remove any runtime concerns regarding finding an A-CEEI in the near future.

strategic manipulations since a student could then influence other students' utility representations via her reports. Additionally, this design would not allow for the asynchronous interaction paradigm between students and the mechanism (where every student can decide when to answer their CQs), which was a major consideration in our design.

An important element of MLCM is that we use the trained ML models, not only to generate CQs in the iterative preference elicitation phase, but also to determine the final allocation. An alternative approach would be to only use the trained ML models to *suggest corrections* to each student's GUI reports, which the student could then either accept or reject. With this approach, one would then run the original CM mechanism with the "corrected" GUI reports to determine the final allocation. While this approach might seem intuitive at first, it has multiple drawbacks. The main one is that this approach would require significantly more interaction by the students with the mechanism: the students would have to first report their preferences to the GUI, then answer some queries, and finally go back to correct their original GUI reports. This last step might be a task that is cognitively too demanding for most students, given that it involves reviewing "suggested corrections" for possibly tens of courses, which are all in conflict with what the student originally reported via the GUI. Additionally, since it requires that students enter their cardinal base values and adjustments a second time, this might introduce new reporting mistakes. Finally, this approach does not address the fact that the CM language may not be able to fully capture every student's preferences.

Some universities might prefer piloting our approach while making minimal changes to an already existing implementation of CM at their institution. To enable this, one could modify MLCM in Phases 3 and 5, *projecting* the learned ML models back into the original GUI language. Consequently, one could then simply use the original CM mechanism to find the A-CEEIs at the end of Phases 3 and 5, without the need for any changes to the core of the CM implementation. However, defining such a projection without a significant performance loss is non trivial but an interesting direction for future work. This approach would also not be able to capture preferences with more than pairwise interactions between courses.

## 9 CONCLUSION

In this paper, we have introduced the *machine learning-powered course match (MLCM)* mechanism. MLCM extends the well-known CM mechanism with an iterative, ML-based preference elicitation component. We have shown experimentally that MLCM significantly improves average and minimum student utility compared to CM across a wide range of settings.

The main challenge was to design a mechanism that can handle both cardinal and ordinal input from students, that does not *require* students to participate in the ML-powered preference elicitation phase, and that can handle asynchronous interactions of students while at the same time keeping computational costs in check.

In contrast to prior work on course allocation, we have focused on alleviating the students' reporting mistakes when declaring their preferences to the mechanism. We have found that, in realistic scenarios, the impact of the students' reporting mistakes on welfare can be even larger than the mechanism choice itself, highlighting the importance of correcting these mistakes. The main driving force behind our results is the careful selection of CQs, which alleviate the students' reporting mistakes in the most important area of the bundle space: bundles within the students' budgets, for which they have a high value.

There are several interesting avenues for future work. First, it seems promising to investigate the design of the query generation procedure in more depth. The most promising approach seems to be to deploy ML models for capturing model uncertainty (e.g., [Heiss et al., 2022]), to capture uncertainty regarding the students' utility for not yet queried course schedules. With such a model in place, more principled query generation procedures based on *expected improvement* and *expected value of information* could be explored.

Second, it would be worthwhile to run a field experiment to empirically evaluate the performance of MLCM. Our design makes it particularly easy for universities to pilot MLCM, given that each student can decide whether to use the ML feature or not. It would be interesting to see how many students would opt into the ML feature and how many comparison queries they would answer.

Third, CM has already been perceived by some students as “black-box”, i.e., non-transparent [Budish and Kessler, 2022]. MLCM’s ML components could potentially amplify that sentiment. From a user experience standpoint, it would be worthwhile to investigate how to best explain to the users what the ML algorithm has learned about them and why (e.g., "because of your reply to query X, we believe that your utility for course A is lower than what you reported in the GUI”). Adding such explainability features to the interface would likely make students more comfortable with the ML feature of the mechanism, increasing its adoption, and thus ultimately increasing welfare.

Finally, we would like to highlight that our ML-powered preference elicitation approach using comparison queries could in principle be applied to a large variety of combinatorial assignment and matching problems. For example, one future application we envision is refugee resettlement. A recent stream of papers has treated refugee resettlement as a matching problem, taking into account the local communities’ preferences as well as the refugees’ preferences [Bansak et al., 2018, Delacrétaz et al., 2019]. Thus, one interesting direction for future work would be to investigate how an ML-powered preference elicitation approach could be used to help refugees better report their preferences to the matching mechanism (e.g., regarding employment opportunities, localities, and proximity to previously resettled family members), ultimately leading to a more efficient and fairer matching between refugees and local communities.

## ACKNOWLEDGEMENTS

We would like to thank Eric Budish, Hongyao Ma, and Yash Kanoria for very helpful discussions on this research project. We are also thankful for the feedback we received from participants at the 2022 INFORMS Workshop on Market Design. We thank the anonymous reviewers for helpful comments. Any errors remain our own. This paper is part of a project that has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation program (Grant agreement No. 805542).

## REFERENCES

Nir Ailon. 2012. An Active Learning Algorithm for Ranking from Pairwise Preferences with an Almost Optimal Query Complexity. *J. Mach. Learn. Res.* 13, 1 (jan 2012), 137–164. 4

Nir Ailon, Ron Begleiter, and Esther Ezra. 2011. A New Active Learning Scheme with Applications to Learning to Rank from Pairwise Preferences. 4

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. Association for Computing Machinery, New York, NY, USA, 2623–2631. 31

Lawrence M Ausubel, Peter Cramton, and Paul Milgrom. 2006. *The clock-proxy auction: A practical combinatorial auction design.* MIT, Cambridge, US. 120–140 pages. 2

Eduardo M Azevedo and Eric Budish. 2019. Strategy-proofness in the large. *The Review of Economic Studies* 86, 1 (2019), 81–116. 3

Kirk Bansak, Jeremy Ferwerda, Jens Hainmueller, Andrea Dillon, Dominik Hangartner, Duncan Lawrence, and Jeremy Weinstein. 2018. Improving refugee integration through data-driven algorithmic assignment. *Science* 359 (01 2018), 325–329. https://doi.org/10.1126/science.aao4408 19

Martin Bichler and Soeren Merting. 2021. Randomized scheduling mechanisms: Assigning course seats in a fair and efficient way. *Production and Operations Management* 30, 10 (2021), 3540–3559. 4

Avrim Blum, Jeffrey Jackson, Tuomas Sandholm, and Martin Zinkevich. 2004. Preference elicitation and query learning. *Journal of Machine Learning Research* 5, Jun (2004), 649–667. 2

Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. 2010. Gaussian Process Preference Elicitation. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1* (Vancouver, British Columbia, Canada) *(NIPS'10)*. Curran Associates Inc., Red Hook, NY, USA, 262–270. 4

Steven J Brams and Philip D Straffin Jr. 1979. Prisoners' dilemma and professional sports drafts. *The American Mathematical Monthly* 86, 2 (1979), 80–88. 3

Gianluca Brero, Benjamin Lubin, and Sven Seuken. 2017. Probably Approximately Efficient Combinatorial Auctions via Machine Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 31, 1 (Feb. 2017), 397–405. https://doi.org/10.1609/aaai.v31i1.10624 2

Gianluca Brero, Benjamin Lubin, and Sven Seuken. 2018. Combinatorial Auctions via Machine Learning-Based Preference Elicitation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden) *(IJCAI'18)*. AAAI Press, Stockholm, Sweden, 128–136. 2

Gianluca Brero, Benjamin Lubin, and Sven Seuken. 2021. Machine learning-powered iterative combinatorial auctions. 2

Eric Budish. 2011. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy* 119, 6 (2011), 1061–1103. 3, 5, 9, 23, 24, 25, 26

Eric Budish, Gérard P Cachon, Judd B Kessler, and Abraham Othman. 2017. Course match: A large-scale implementation of approximate competitive equilibrium from equal incomes for combinatorial allocation. *Operations Research* 65, 2 (2017), 314–336. 1, 3, 5, 9

Eric Budish and Estelle Cantillon. 2012. The multi-unit assignment problem: Theory and evidence from course allocation at Harvard. *American Economic Review* 102, 5 (2012), 2237–71. 1, 3

Eric Budish, Ruiquan Gao, Abraham Othman, Aviad Rubinstein, and Qianfan Zhang. 2023. Practical algorithms and experimentally validated incentives for equilibrium-based fair division (A-CEEI). Unpublished manuscript. 7, 17

Eric Budish and Judd B Kessler. 2022. Can market participants report their preferences accurately (enough)? *Management Science* 68, 2 (2022), 1107–1130. 1, 3, 10, 11, 12, 13, 14, 15, 16, 19, 27, 31, 32, 36, 43, 44

Urszula Chajewska, Daphne Koller, and Ronald Parr. 2000. Making Rational Decisions Using Adaptive Utility Elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, Austin, USA, 363–369. 2, 13, 17

Wei Chu and Zoubin Ghahramani. 2005. Preference Learning with Gaussian Processes. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*. Association for Computing Machinery, New York, NY, USA, 137–144. https://doi.org/10.1145/1102351.1102369 4

Vincent Conitzer. 2009. Eliciting single-peaked preferences using comparison queries. *Journal of Artificial Intelligence Research* 35 (2009), 161–191. 2, 13, 17

David Delacrétaz, Scott Duke Kominers, and Alexander Teytelboym. 2019. *Matching Mechanisms for Refugee Resettlement*. Working Papers 2019-078. Human Capital and Economic Opportunity Working Group. https://ideas.repec.org/p/hka/wpaper/2019-078.html 19

Franz Diebold, Haris Aziz, Martin Bichler, Florian Matthes, and Alexander Schneider. 2014. Course allocation via stable matching. *Business & Information Systems Engineering* 6, 2 (2014), 97–110. 3

Fred Glover, Manuel Laguna, and Rafael Martí. 2018. Principles and strategies of tabu search. In *Handbook of Approximation Algorithms and Metaheuristics, Second Edition*. Chapman and Hall/CRC, London, UK, 361–377. 5

Jacob K Goeree and Charles A Holt. 2010. Hierarchical package bidding: A paper & pencil combinatorial auction. *Games and Economic Behavior* 70, 1 (2010), 146–169. 10, 27

Shengbo Guo and Scott Sanner. 2010. Real-time Multiattribute Bayesian Preference Elicitation with Pairwise Comparison Queries. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 9)*, Yee Whye Teh and Mike Titterington (Eds.). PMLR, Chia Laguna Resort, Sardinia, Italy, 289–296. https://proceedings.mlr.press/v9/guo10b.html 4

John William Hatfield. 2009. Strategy-proof, efficient, and nonbossy quota allocations. *Social Choice and Welfare* 33, 3 (2009), 505–515. 3

Jakob M Heiss, Jakob Weissteiner, Hanna S Wutte, Sven Seuken, and Josef Teichmann. 2022. NOMU: Neural Optimization-based Model Uncertainty. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, Baltimore, USA, 8708–8758. https://proceedings.mlr.press/v162/heiss22a.html 4, 18

Sebastien M. Lahaie and David C. Parkes. 2004. Applying Learning Algorithms to Preference Elicitation. In *Proceedings of the 5th ACM Conference on Electronic Commerce* (New York, NY, USA) *(EC '04)*. Association for Computing Machinery, New York, NY, USA, 180–188. https://doi.org/10.1145/988772.988800 2

Abraham Othman, Christos Papadimitriou, and Aviad Rubinstein. 2016. The complexity of fairness through equilibrium. *ACM Transactions on Economics and Computation (TEAC)* 4, 4 (2016), 1–19. 5

Szilvia Pápai. 2001. Strategyproof and nonbossy multiple assignments. *Journal of Public Economic Theory* 3, 3 (2001), 257–271. 3

Tobias Scheffel, Georg Ziegler, and Martin Bichler. 2012. On the impact of package selection in combinatorial auctions: an experimental study in the context of spectrum auction design. *Experimental Economics* 15, 4 (2012), 667–692. 10, 27

D. Sculley. 2010. Combined Regression and Ranking. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Washington, DC, USA) *(KDD '10)*. Association for Computing Machinery, New York, NY, USA, 979–988. https://doi.org/10.1145/1835804.1835928 12, 35

Tayfun Sönmez and M Utku Ünver. 2010. Course bidding at business schools. *International Economic Review* 51, 1 (2010), 99–123. 1, 3

Jakob Weissteiner, Jakob Heiss, Julien Siems, and Sven Seuken. 2022a. Monotone-Value Neural Networks: Exploiting Preference Monotonicity in Combinatorial Assignment. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. International Joint Conferences on Artificial Intelligence Organization, Vienna, AUT, 541–548. https://doi.org/10.24963/ijcai.2022/77 Main Track. 3, 4, 13

Jakob Weissteiner, Jakob Heiss, Julien Siems, and Sven Seuken. 2023. Bayesian Optimization-based Combinatorial Assignment. *Proceedings of the AAAI Conference on Artificial Intelligence* 37 (2023). 4

Jakob Weissteiner and Sven Seuken. 2020. Deep Learning—Powered Iterative Combinatorial Auctions. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 02 (Apr. 2020), 2284–2293. https://doi.org/10.1609/aaai.v34i02.5606 4

Jakob Weissteiner, Chris Wendler, Sven Seuken, Ben Lubin, and Markus Püschel. 2022b. Fourier Analysis-based Iterative Combinatorial Auctions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. International Joint Conferences on Artificial Intelligence Organization, Vienna, AUT, 549–556. https://doi.org/10.24963/ijcai.2022/78 Main Track. 4

Wharton. 2020. *Course Match User Manual*. Wharton. https://mba-inside.wharton.upenn.edu/wp-content/uploads/2020/08/Course-Match-manual-08152020.pdf 17

## APPENDIX

## A CARDINAL DATASET FROM STUDENTS' GUI REPORTS

In this section, we describe the process of building the cardinal dataset $D_{i,\text{card}}$ for each student $i \in N$, given their GUI reports.

From a conceptual standpoint, our idea for incorporating the students' reports in the learning framework is straightforward: For every student $i$, we want to use their reports to the GUI and create a bootstrap dataset $D_{i,\text{card}}$ where every datapoint is a tuple of the form (bundle, value). $D_{i,\text{card}}$ is then used to initialize $\mathcal{M}_i$.

As explained in Section 3.3, in the current implementation of Course Match, the students have two means at their disposal to express their preferences: base values and pairwise adjustments. Based on the student $i$'s report, there are three distinct sets of bundles we can generate for our dataset $D_{i,\text{card}}$:

- **Bundles of type 1 (T1):** This set specifically includes two kind of bundles:
  (1) Bundles containing a single course, for which the corresponding student has declared a base value.
  (2) Bundles containing two courses for which the corresponding student has declared both base values plus the corresponding pairwise adjustment.

  The value of each such bundle is calculated exactly as in the current Course Match implementation, i.e., by summing the reported base values plus the reported adjustments whose courses are contained in that bundle.

  These bundles constitute the most solid pieces of information in our dataset, as for any bundle of this type, we can be sure that there cannot be any terms that the student did not report, either because they forgot to, or because it was impossible to do so in the CM reporting language.
- **Bundles of type 2 (T2)**: Each bundle of T2 contains five courses, for all of which the student reported a base value but not necessarily all corresponding pairwise adjustments. The value of each such bundle is calculated in the same way as in the current CM implementation, i.e., it is simply the sum of base values plus any adjustments reported by the student for the courses it contains.

  These bundles are important for two reasons: First, they push the distribution of the training set closer to the part of the space that matters in practice, a student participating in the mechanism is allocated a bundle of about five courses; not one or two. Additionally, these bundles implicitly "show" the learning model the combinatorial structure of the students' utility functions. For most ML algorithms tested, training them without bundles of this type lead to significantly worse generalization performance.
- **Bundles of type 3 (T3)**: Each bundle of this type contains five courses, but for at least one of them, the student did not even submit a base value when reporting their preferences in the CM language. For every such course, we impute a base value for it. Specifically, the imputed value is the expected value of the prior value distribution of a course, but now conditioned on the fact that a student did not report a base value for it in the CM language. No additional pairwise adjustments are imputed that a student did not report. Having imputed those base values, the value of each bundle of T3 can be calculated in the same way as for T1 and T2.

  Concretely, suppose that a student didn't report a base value for a course. In the current implementation, this course will be treated as if the student had declared a zero base value for it. Thus, there are two possible explanations for a missing base value: Either the student actually had a zero value for the specific course, or they had a non-zero base value, but they actually forgot to report it. Given the assumption that students tend to forget to report a base value for courses towards the lower-end of their valuation, we model the base value $v$ of a course that wasn't reported by the student as:

$$v = \begin{cases} 0 & \text{w.p. } p_z \\ \sim U[l, h] & \text{w.p. } 1 - p_z \end{cases} \tag{4}$$

And thus its expected value is:

$$\mathbb{E}[v] = (1 - p_z)\left(\frac{h - l}{2} + l\right) = (1 - p_z)\,\frac{h + l}{2}. \tag{5}$$

In order to generate the bundles of T3 in our experiments, we set $p_z = 0$, $l = 0$, and $h$ equal to the lowest non-zero reported base value.

## B THEORETICAL PROPERTIES OF MLCM

In this section, we show that, if the preferences are captured *approximately* via the ML models $\mathcal{M}_i$, then the same theoretical properties as for the CM Stage 1 allocation (i.e, *envy-bounded by a single good*, $(n+1)$-*maxmin share guarantee*, and *Pareto efficiency*) also hold in an *approximate* sense for the Stage 1 allocation of MLCM. First, we recall our notation.

*Notation.* $N$ is the set of students, $n$ is the number of students, $M$ is the set of courses, $m$ is the number of courses, $k$ is the maximum number of courses allowed in a bundle, $u_i(\cdot)$ is the true utility function for student $i$, $x$ is a bundle of courses (i.e. course schedule), $a$ an allocation, and $a_i$ denotes the bundle student $i$ receives in allocation $a$, i.e., student $i$'s allocation. The set of feasible allocations is represented by $\mathcal{F}$. Furthermore, we slightly overload the notation and denote by $a_i$ *both* the indicator vector representing the allocation for student $i \in N$ (i.e., $a_i \in \{0,1\}^m$) as well as the corresponding set (i.e., $a_i \in 2^M$). With this, we denote by $j \in a_i$ that $j \in M$ is contained in $a_i$ and by $a_i \setminus \{j\}$ the allocation $a_i$ without the course $j \in M$ (we also use this for other generic bundles $x$ or $x'$ ).

First, we define our notion of utility function approximation, approximate fairness and welfare.

Definition B.1 ($\varepsilon$-approximation of a utility function). *For $\varepsilon \geq 0$, a function $\hat{u}(\cdot)$ is an $\varepsilon$-approximation of the true utility function $u : \{0,1\}^m \longrightarrow \mathbb{R}$ if*

$$\sup_{x \in \{0,1\}^m} |\hat{u}(x) - u(x)| < \varepsilon. \tag{6}$$

Definition B.2 (Envy $\varepsilon$-bounded by a single good). *An allocation a satisfies envy $\varepsilon$-bounded by a single good if for all $i, i' \in N$*

*(1) $u_i(a_i) \geq u_i(a_{i'}) - \varepsilon$ or*

*(2) There exists some good $j \in a_{i'}$ such that $u_i(a_i) \geq u_i(a_{i'} \setminus \{j\}) - \varepsilon$.*

That is, if student $i$ envies student $i'$, by removing some single good from student $i'$'s bundle we can make $i$'s envy at most $\varepsilon$.

Definition B.3 ($l$-maximin share, Budish [2011]). *Let $Z^*$ denote a $l$-maximin split, i.e.,*

$$Z^* := \underset{\substack{\{z_1,\ldots,z_l\}: z_k \in \mathcal{X} \\ \sum_{k=1}^{l} z_{kj} \leq q_j}}{\arg\max} \left( \min_{k \in \{1,\ldots,l\}} u_i(z_k) \right). \tag{7}$$

*Then, student $i$'s $l$-maximin share $a^{MaxiMin,l,u_i}$ for her given utility function $u_i$ is defined as*

$$a^{MaxiMin,l,u_i} := \underset{z \in Z^*}{\arg\min}\, u_i(z). \tag{8}$$

*In words, $a^{MaxiMin,l,u_i} \in \{0,1\}^m$ is the course schedule an student obtains when she selects the utility maximizing feasible partition consisting of l bundles of the set of all courses given that an adversary assigns her the worst bundle from that proposed partition.*

Definition B.4 ($(l,\varepsilon)$-maximin share). *For any $\varepsilon \geq 0$ student i's $(l,\varepsilon)$-maximin share is any course schedule $a^{MaxiMin,(l,\varepsilon),u_i}$ for which student i has utility at most $\varepsilon$ less than her maximin share, i.e,*

$$u_i(a^{MaxiMin,(l,\varepsilon),u_i}) \geq u_i\left(a^{MaxiMin,l,u_i}\right) - \varepsilon. \tag{9}$$

Definition B.5 ($(l,\varepsilon)$-maximin share guarantee). *Any feasible allocation $a = (a_i)_{i=1}^n \in \mathcal{F} \subset \mathcal{X}^n$ where* all *students $i \in N$ get a bundle $a_i$ they weakly prefer to their $(l,\varepsilon)$-maximin share $a^{MaxiMin,(l,\varepsilon),u_i}$ (w.r.t. their true utility functions, i.e., $\{u_i\}_{i\in N}$) is said to satisfy the $(l,\varepsilon)$-maximin share guarantee.*

In Proposition B.6, we now prove our main theoretical result.

Proposition B.6. *Let $[a^*, b, p^*]$ be an $(\alpha,\beta)$-A-CEEI calculated using the $\varepsilon$-approximation of the true utility functions $\{\hat{u}_i\}_{i\in N}$, then:*

*(1) If $\beta \leq \frac{1}{k-1}$ with k being the maximum number of courses per student, then $a^*$ satisfies envy $2\varepsilon$-bounded by a single good w.r.t. the true utility functions $\{u_i\}_{i\in N}$. Moreover, this bound is tight.*

*(2) If there exists some $\delta \geq 0$ such that $p^* \in \mathcal{P}(\delta, b)$, [15] and $\beta < (1-\delta n)/n(1+\delta)$, then $a^*$ satisfies the $(n+1, 2\varepsilon)$-maximin share guarantee w.r.t. the true utility functions $\{u_i\}_{i\in N}$. Moreover, this bound is tight.*

*(3) Given $p^*$, the true utility of every student $i \in N$ for the bundle she receives in the allocation $a^*$ (i.e., $u_i(a_i^*)$) is within $2\varepsilon$ of her true utility for her most preferred bundle she could afford.*

*(4) Given $p^*$, the allocation $a^*$ is $2\varepsilon$-Pareto efficient w.r.t. the true utility functions $\{u_i\}_{i\in N}$, i.e., $\nexists\ a' \in \mathcal{F}$ such that $\forall i \in N$ it holds that:*

$$u_i(a_i') \geq u_i(a_i^*) - 2\varepsilon. \tag{10}$$

Proof. (1) Let $\hat{u} = (\hat{u}_1, \hat{u}_2, \dots \hat{u}_n)$ be the profile of learned utility functions. Using Theorem 3 of Budish [2011][16], we have that for those learned utility functions the allocation $a^*$ satisfies envy bounded by a single good i.e., for any $i, i' \in S$ either:

(a) $\hat{u}_i(a_i^*) \geq \hat{u}_i(a_{i'}^*)$ or

(b) There exists some good $j \in a_{i'}$ such that $\hat{u}_i(a_i^*) \geq \hat{u}_i(a_{i'}^* \setminus \{j\})$.

Since $\hat{u}_i(\cdot)$ is an $\varepsilon$-approximation of the true utility function $u_i(\cdot)$ we have that in the first case:

$$u_i(a_i^*) + \varepsilon \geq \hat{u}_i(a_i^*) \geq \hat{u}_i(a_{i'}^*) \geq u_i(a_{i'}^*) - \varepsilon \implies u_i(a_i^*) \geq u_i(a_{i'}^*) - 2\varepsilon \tag{11}$$

Similarly in the second case:

$$u_i(a_i^*) + \varepsilon \geq \hat{u}_i(a_i^*) \geq \hat{u}_i(a_{i'}^* \setminus \{j\}) \geq u_i(a_{i'}^*) - \varepsilon \implies u_i(a_i^*) \geq u_i(a_{i'}^* \setminus \{j\}) - 2\varepsilon \tag{12}$$

From Equations (11) and (12) it follows immediately that $a^*$ satisfies envy $2\varepsilon$-bounded by a single good.

Next we provide an example that shows that the bound is tight. Assume that there are 3 courses, $a, b$ and $c$ with capacities $q_a = 2$ and $q_b = q_c = 1$. Moreover, assume that there are 2 students with the following utility functions:

- $u_1(\{a,b\}) = u_1(\{a,b,c\}) = 1$, and 0 for any other bundle.

[15] $\mathcal{P}(\delta, b) = \{p \in [0, \max_i b_i]^m : \sum_j p_j q_j \leq \sum_i b_i(1+\delta)\}$.

[16] To apply Theorem 3 of Budish [2011], we need $\beta \leq \frac{1}{k-1}$.

- $u_2(\{a\}) = u_2(\{b\}) = 0.5 - \varepsilon, u_2(\{a,c\}) = u_2(\{b,c\}) = 0.5 + \varepsilon, u_2(\{a,b\}) = u_2(\{a,b,c\}) = 1$, and 0 for any other bundle.

Take $\varepsilon, \varepsilon' > 0$ and the learned utility functions of the 2 students to be:
- $\hat{u}_1(\{a,b\}) = 1 - \varepsilon, \hat{u}_1\{a,b,c\} = 1$, and 0 for any other bundle.
- $\hat{u}_2(\{a\}) = \hat{u}_2(\{b\}) = \hat{u}_2(\{a,c\}) = \hat{u}_2(\{b,c\}) = 0.5$
  $\hat{u}_2(\{a,b\}) = \hat{u}_2(\{a,b,c\}) = 1$, and and 0 for any other bundle.

Then, a $(0,\beta)$-A-CEEI ( $\beta \leq \frac{1}{k-1} = \frac{1}{2}$) can be formed with the following elements:
- $a_1^* = \{a,b,c\}, a_2^* = \{a\}$
- $b_1 = 1 + \beta, b_2 = 1$
- $p_a^* = \beta, p_b^* = 1, p_c^* = 0.$

In this $(0,\beta)$-A-CEEI, the envy of student 2 with respect to her true utility function is exactly $2\varepsilon$-bounded by a single good:

$$u_2(a_2^*) = 0.5 - \varepsilon = u_2(a_1^* \setminus \{b\}) - 2\varepsilon. \tag{13}$$

Therefore, the bound is tight.

(2) Let $\hat{u} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_n)$ be the profile of learned utility functions. Using Theorem 2 of [Budish, 2011],[17] we have that for those learned utility functions the allocation $a^*$ satisfies the $(n+1)$-maximin share guarantee. Thus, for any $i \in N$:

$$\hat{u}_i(a_i^*) \geq \hat{u}_i\left(a^{\text{MaxiMin},n+1,\hat{u}_i}\right) \tag{14}$$

$$\geq \hat{u}_i\left(a^{\text{MaxiMin},n+1,u_i}\right) \tag{15}$$

$$\geq u_i\left(a^{\text{MaxiMin},n+1,u_i}\right) - \varepsilon \tag{16}$$

where (14) follows from the definition of the $(n+1)$-maximum share guarantee for the A-CEEI w.r.t. the learned utilities $\{\hat{u}_i\}_{i\in N}$ and (16) is true because $\hat{u}_i$ is per assumption an $\varepsilon$-approximation of $u_i$. Then, we have that

$$u_i(a_i^*) + \varepsilon \geq \hat{u}_i(a_i^*) \geq u_i\left(a^{\text{MaxiMin},n+1,u_i}\right) - \varepsilon, \tag{17}$$

and therefore

$$u_i(a_i^*) \geq u_i\left(a^{\text{MaxiMin},n+1,u_i}\right) - 2\varepsilon. \tag{18}$$

Hence the allocation $a^*$ satisfies the $(n+1, 2\varepsilon)$-maximin share guarantee with respect to the true utilities $\{u_i\}_{i\in N}$.

Next, we provide an example that shows this bound is tight. Assume that there are 4 courses, $a, b, c$ and $d$ with capacities $q_a = q_b = q_c = q_d = 1$. Moreover, assume that there are 2 students with utility functions:
- $u_1(\{a\}) = 1, u_1(\{b\}) = 0.5 + \varepsilon, u_1(\{c\}) = 0.5 - \varepsilon, u_1(\{d\}) = 0$ and $u_1(\{b,c\}) = u_1(\{b,d\}) = u_1(\{c,d\}) = 0.5 + \varepsilon$.
- $u_2(\{a\}) = 0.8, u_2(\{a,b\}) = u_2(\{a,d\}) = 0.9, u_2(\{a,b,d\}) = 1$ and 0 for any other bundle.

Furthermore, assume that the learned utility functions of the 2 students are given as follows:
- $\hat{u}_1(\{a\}) = 1, \hat{u}_1(\{b\}) = \hat{u}_1(\{c\}) = \hat{u}_1(\{b,c\}) = \hat{u}_1(\{b,d\}) = \hat{u}_1(\{c,d\}) = 0.5$ and 0 for any other bundle.
- $\hat{u}_2(\cdot) = u_2(\cdot)$.

Then, a $(0, 0.25)$-A-CEEI ($\beta = 0.25 \leq \frac{1-\delta n}{n(1+\delta)} \overset{\delta=0}{=} \frac{1}{2}$) is given by:
- $a_1^* = \{c\}, a_2^* = \{a,b,d\}$
- $b_1 = 1, b_2 = 1 + \beta = 1.25$

[17] To apply Theorem 2 of Budish [2011], we need the existence of $\delta \geq 0$ such that $p^* \in \mathcal{P}(\delta, b)$ and $\beta < (1 - \delta n)/n(1+\delta)$.

- $p_a^* = 1.1, p_b^* = p_c^* = 0.15, p_d^* = 0$.

A maximin $(n+1)$-split for student 1 w.r.t. her true utility function is $\{\{a\}, \{b\}, \{c, d\}\}$ and her true utility for her $(n+1)$-maximin share is $0.5 + \varepsilon$. Hence, the utility of student 1 with respect to her true utility function is exactly $2\varepsilon$ less than her $(n+1)$-maximin share of the endowment:

$$u_1(a_1^*) = 0.5 - \varepsilon \tag{19}$$

$$= u_1\left(a^{\text{MaxiMin},n+1,u_1}\right) - 2\varepsilon. \tag{20}$$

(3) For any $i \in N$, from the definition of the $(\alpha, \beta)$-A-CEEI, it holds that

$$a_i^* = \underset{x \in \{x' \in \mathcal{X} : p^* \cdot x' \leq b_i\}}{\arg\max} \hat{u}_i(x). \tag{21}$$

Therefore,

$$u_i(a_i^*) + \varepsilon \geq \hat{u}_i(a_i^*) \tag{22}$$

$$= \hat{u}_i\left(\underset{x \in \{x' \in \mathcal{X} : p^* \cdot x' \leq b_i\}}{\arg\max} \hat{u}_i(x)\right) \tag{23}$$

$$\geq \hat{u}_i\left(\underset{x \in \{x' \in \mathcal{X} : p^* \cdot x' \leq b_i\}}{\arg\max} u_i(x)\right) \tag{24}$$

$$\geq u_i\left(\underset{x \in \{x' \in \mathcal{X} : p^* \cdot x' \leq b_i\}}{\arg\max} u_i(x)\right) - \varepsilon \tag{25}$$

Hence, we have that

$$u_i(a_i^*) \geq u_i\left(\underset{x \in \{x' \in \mathcal{X} : p^* \cdot x' \leq b_i\}}{\arg\max} u_i(x)\right) - 2\varepsilon. \tag{26}$$

(4) Assume that $a^*$ is not $2\varepsilon$- Pareto efficient. This implies that there exists an $a' \in \mathcal{F}$, which is a $2\varepsilon$-Pareto improvement of $a^*$ in economy $\left(N, M, (q_j^*)_{j=1}^m, (u_i)_{i=1}^n\right)$, i.e., the utility of every student $i \in N$ in allocation $a'$ is more than $2\varepsilon$ higher than her utility in $a^*$. By item 3, for every $i \in N$ it holds that $p^* \cdot a_i' > p^* \cdot a_i^*$. This implies that $\sum_i p^* \cdot a_i' > \sum_i p^* \cdot a_i^*$. This is a contradiction since prices are non-negative and $a^*$ allocates all units of positive-priced goods.

□

Intuitively, item 4 of Proposition B.6 says that if we fix the prices of A-CEEI, the students cannot aggregate their budgets and then spend them in such a way that they are all benefited by more than $2\varepsilon$, w.r.t. their true utility function.

Using the wording of Budish [2011], an implication of item 4 of Proposition B.6 is that the allocation induced by an A-CEEI on the $\varepsilon$-approximate utility functions will not admit any $2\varepsilon$-Pareto-improving trades among the students, but may admit Pareto-improving trades among sets of students and the administrator.

## C STUDENT PREFERENCE GENERATOR

In this section, we describe in detail our proposed student preference generator.

Our design goal for the student preference generator is two-fold: First, it should be realistic. That is, the preferences combined with the modeling of students' mistakes should closely approximate the metrics on reported preferences given in [Budish and Kessler, 2022]. Second, we should be able to formulate the generated preferences in a succinct MIP as this is required by all stages of CM.

We build our preference generator based on the following two assumptions. First, a single student's value for a course schedule depends on their value for every single course and the complementarities/substitutabilities between the courses within a bundle. Second, the high-valued courses amongst students are correlated and fall under the category of popular courses. This is because students have the same reasoning for considering a course to be high-valued. These reasons could be a popular topic, a good instructor, an interest in the same minors, etc.

Thus, a student's utility for a bundle of courses depends on the following two things:

**D1** The *individual value* of each course in that bundle.
**D2** The *complementarities and substitutabilities* between courses in that bundle.

Recall, that $M \neq \emptyset$ and and $N \neq \emptyset$ denote the set of all courses and the set of students, respectively. To imitate the effect of students having correlated high-valued courses, we choose $M_p \subseteq M$ to represent the *popular courses* amongst students and $M_{np} = M \setminus M_p$ to represent the *non-popular courses*. We also assume that the set of high-valued courses for each student is a subset of the popular courses. We call these the student's *favorite courses*. For each student, their *base value* for each course in their favorite and non-favorite courses is drawn independently from $U(l_p, u_p)$ and $U(l_{np}, u_{np})$, respectively, where $U(a, b)$ denotes the uniform distribution on the interval $(a, b)$, $0 \leq l_{np} \leq l_p$, and $0 \leq u_{np} \leq u_p$.

To simulate the substitutabilities and complementarities between courses in students' preferences, we build on the prior work on preference generators for spectrum auctions (i.e., the Local Synergy Value Model (LSVM) [Scheffel et al., 2012] and the Global Synergy Value Model (GSVM) [Goeree and Holt, 2010]).

Specifically, we assume that the courses are arranged in a latent space where the complementarity and substitutability relation between courses is a function of their distance to each other, i.e.,

- If two courses are "too close", then the contents of the courses have too much overlap and hence they are substitutes for each other.
- If they are "close" but not "too close", they are complements.
- If they are "far away", they are neither complements nor substitutes to each other.

More precisely, we assume that $M \subset \mathbb{N}^2$ and that the set of complementarities and substitutabilities are centered around a subset of the popular courses $M_p$, called centers, i.e., $M_c \subseteq M_p$. Then the set of substitutabilities and complementarities around a center point $c \in M_c$ are defined using the $L_1$ and $L_\infty$ distances as follows:

- *Set of substitutabilities:*
$$S_c = \{m \in M | L_1(m, c) \leq r_s\} \tag{27}$$
- *Set of complementarities:*
$$C_c = \{m \in M | L_\infty(m, c) \leq r_c\} \setminus S_c \cup \{c\} \tag{28}$$

where $r_s, r_c \in \mathbb{N}_{>0}$ are the radius of the set of substitutabilities and the radius of the set of complementarities, respectively. The larger these radii, the more complementarities/substitutabilities are present in the students' preferences. We illustrate an example of this latent space in Figure 5.

Figure 5, depicts a latent space of height 5 and width 6 with the courses $M = \{1, \ldots, 30\}$, where we assume that both radii are equal to 1, i.e., $r_s = r_c = 1$ and the set of popular courses is given as $M_p = \{8, 9, 21, 29\}$. Then, the set of substitutabilities and complementarities defined by the center 9 are given by $S_c = \{9, 3, 8, 10, 15\}$ and $C_c = \{9, 2, 4, 14, 16\}$, respectively.

To model the impact of complementarities/substitutabilities on students' preferences, we use piece-wise constant step functions $\psi_c : \{0, \ldots, |C_c|\} \to \mathbb{R}_+$ and $\xi_c : \{0, \ldots, |S_c|\} \to \mathbb{R}_-$ to represent

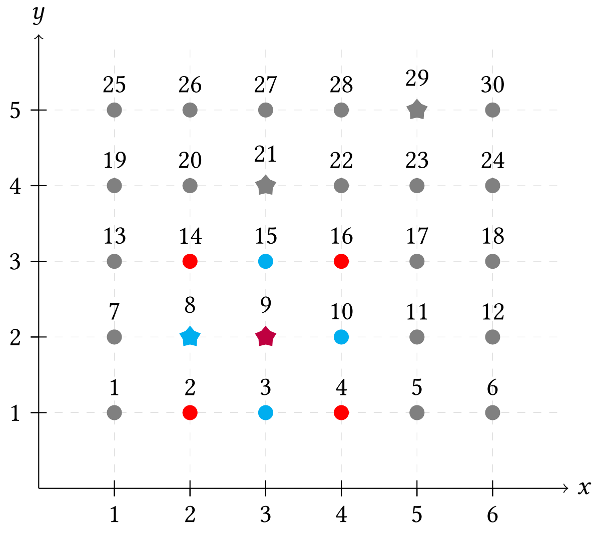

Fig. 5. A latent space with 30 courses.

the multiplicative bonuses and penalty factors for a bundle of courses if its items belong to the set of complementarities $C_c$ or the set of substitutabilities $S_c$. We further assume that $\psi_c$ is monotonically non-decreasing and $\xi_c$ is monotonically non-increasing. For example, given a center $c$, $\psi_c(3)$ represents a student's multiplicative bonus for taking three courses from the set of complementarities $C_c$.

With all these building blocks, the utility of a student $i \in N$ for a bundle $x \in \{0,1\}^m$ is defined as:

$$\begin{aligned} u_i(x) = &\sum_{j \in x} u_i(\{j\}) \\ &+ \sum_{c \in M_c} \sum_{j \in C_c} \psi_c(\tau_c(x)) u_i(\{j\}) \\ &+ \sum_{c \in M_c} \sum_{j \in S_c} \xi_c(\kappa_c(x)) u_i(\{j\}), \end{aligned} \tag{29}$$

where $u_i(\{j\})$ represents the base values of student $i \in N$ for course $j$ and $\tau_c(x) := |\{j \in x : j \in C_c\}|$ is the number of courses that belong both to the bundle $x$ and the set of complementarities at center $c$, i.e., $C_c$. Analogously, $\kappa_c(x) := |\{j \in x : j \in S_c\}|$ is the number of courses that belong both to the bundle $x$ and the set of substitutabilities at center $c$, i.e., $S_c$.

This preference generator enables to efficiently encode students' complementarities/substitabilities into a MIP (see Appendix D). Thus, it allows us to conduct experiments and compare MLCM with the original CM mechanism in a synthetic setting where we have access to the optimal allocations w.r.t. students' true preferences.

## D MIP FOR THE STUDENT PREFERENCE GENERATOR

In this section, we provide a succinct MIP formulation for calculating the optimal allocation for a single student $i$ given her preferences generated by our proposed student preference generator from Appendix C and a price vector $p$. This enables us to compute, for any price vector, the allocation under the students' true preferences. First, we define the individual student optimization problem.

Definition D.1 (Individual student utility optimization problem). *For student $i \in N$, utility function $u_i : \mathcal{X} \rightarrow \mathbb{R}_+$ defined in Equation (29), price vector $p \in \mathbb{R}_+^m$ and budget $b_i \in [1, 1+\beta]$, $\beta > 0$, we define the individual student utility optimization problem as*

$$\underset{\{x \in \Psi_i : x \cdot p \leq b_i\}}{\arg\max} \quad u_i(x) \tag{30}$$

*In words, a solution of Equation (30) maximizes student $i$'s true utility (as defined by our preference generator in Equation (29)) amongst all her permissible course schedules $\Psi_i$ that are affordable at prices $p$ when given a budget $b_i$.*

Next, we provide in Proposition D.2 the equivalent MIP.

Proposition D.2 (Individual student MIP). *Using the notation from Appendix C, and Definition D.1 the individual student utility optimization problem defined in Equation (30) can be equivalently formulated as the following MIP:*

$$\underset{x \in \Psi_i}{\arg\max} \sum_{c \in M_c} \sum_{j \in C_c} \sum_{\tau=1}^{\bar{\tau}} \psi_c(\tau) \cdot G_{c,j,\tau} \cdot u_i(\{j\}) \tag{31}$$

$$+ \sum_{c \in M_c} \sum_{j \in S_c} \sum_{\tau=1}^{\bar{\tau}} \xi_c(\tau) \cdot J_{c,j,\tau} \cdot u_i(\{j\}) \tag{32}$$

$$+ \sum_{j \in [m]} x_j \cdot u_i(\{j\}) \tag{33}$$

*s.t.*

$$\sum_{\tau=1}^{\bar{\tau}} \tau \cdot G_{c,j',\tau} \leq \sum_{j \in C_c} x_j, \ \forall \ j' \in C_c, \forall \ c \in M_c \tag{34}$$

$$\sum_{\tau=1}^{\bar{\tau}} G_{c,j,\tau} \leq 1, \ \forall \ j \in C_c, \forall \ c \in M_c \tag{35}$$

$$\sum_{\tau=1}^{\bar{\tau}} \tau \cdot J_{c,j',\tau} \geq \sum_{j \in S_c} x_j, \ \forall \ j' \in S_c, \forall \ c \in M_c \tag{36}$$

$$\sum_{\tau=1}^{\bar{\tau}} J_{c,j,\tau} \leq 1, \ \forall \ j \in S_c, \forall \ c \in M_c \tag{37}$$

$$\sum_{j \in [m]} p_j \cdot x_j \leq b_i \tag{38}$$

*where*

- *$\bar{\tau}$ is the maximum number of courses in a schedule.*
- *$M_c$ is the set of centers.*
- *$C_c$ is the set of complementarities centered around $c$.*
- *$S_c$ is the set of substitutabilities centered around $c$.*
- *$G_{c,j,\tau}$ denotes the binary variable, whether or not the course indexed by $j$, which belongs to the set of complementarities $C_c$ is in $x$, while in total $\tau$ courses from that set are included in $x$.*
- *$J_{c,j,\tau}$ denotes the binary variable, whether or not the course indexed by $j$, which belongs to the set of substitutabilities $S_c$ is in $x$, while in total $\tau$ courses from that set are included in $x$.*
- *$b_i$ is the budget of student $i$.*

Proof. The first thing we need to show is that the problem is actually a MIP, i.e., all constraints are linear and all variables are either linear or integer. First, note that all variables are either linear or integer by their definition. Moreover, the same is true for all constraints shown, other than the $x \in \Psi_i$ constraint. Next, we show that this constraint can be encoded in a linear way.

There are 2 reasons a schedule $x$ would not be permissible for a student $i$. The first one would be if it contained any courses that the student is not eligible for. Let $E_i$ be the indexes of those courses for student $i$. The second constraint would be if it contained more than one courses that take place at the same time. Let $H$ be the set of all course hours (also known as *time slots*, e.g. Wednesdays 10-11am), and $T_h$ the set of indexes of courses that take place at time $h \in H$. Then, $x \in \Psi_i$ is equivalent to the following two *linear* constraints:

$$\sum_{j \in E_i} x_j \leq 0, \tag{39}$$

$$\sum_{j \in T_h} x_j \leq 1 \ \forall h \in H. \tag{40}$$

Constraint (39) enforces that student $i$ cannot enroll in courses that are not permissible for her, while constraint (40) enforces that she cannot take more than one courses that take place in the same time slot. Now that we have proven that the problem defined above is a valid MIP respecting all constraints, we need to prove that its solution is a utility-maximizing course schedule for student $i \in N$, i.e, is equivalent to Equation (30).

If all binary variables $G_{c,j,\tau}$ and $J_{c,j,\tau}$ are set to the correct value for a given course schedule $x$, then Equations (31) to (33) encode the utility of course schedule $x$ for a given student $i$, as the only additive terms not zeroed-out are those in Equation (29). Thus, to prove the correctness of the MIP, it suffices to prove that those binary variables are correctly set for any valid $x$ and that the budget constraint is satisfied.

Constraint (37) enforces that for any $(c, j) \in M_c \times S_c$ pair at most one of the binary variables $J_{c,j,\tau}$ will be set to 1. This combined with constraint (36) enforce that if $\tau$ elements from set $S_c$ are included in $x$, then for the $J_{c,j,\tau'}$ variable it must hold that $\tau' \geq \tau$. But since we have a maximization problem and the function $\xi_c(\cdot)$ is monotonically non-increasing, the smallest value of $\tau'$ that satisfies the above constraint will be set to one, thus $\tau' = \tau$. Finally, note that if $\xi_c(\tau') = \xi_c(\tau)$ for some $\tau' \geq \tau$, i.e. $\xi_c(\cdot)$ is at a constant part of its support, it can be the case that the maximizer sets $J_{c,j,\tau'} = 1$ instead of the $J_{c,j,\tau'} = 1$. However, since $\xi_c(\tau') = \xi_c(\tau)$, this is not a problem, as the objective value is the same in both cases.

Similarly, constraint (35) enforces that for any $(c, j) \in M_c \times C_c$ pair, at most one of the $G_{c,j,\tau'}$ variables will be set to one. Constraint (34) enforces that if $\tau$ of the elements of $C_c$ are included in $x$, then it must hold that $\tau' \leq \tau$. But since this is a maximization problem and $\psi_c(\cdot)$ is monotonically non-decreasing, the solver will set exactly the variable $G_{c,j,\tau}$ to one where $\tau' = \tau$, as this maximizes the objective value out of all feasible solutions. If $\psi_c(\tau) = \psi(\tau')$, $G_{c,j'_\tau}$ could be set to one instead of $G_{c,j,\tau}$, but the same argument as above again applies.

Finally, constraint (38) enforces that a student cannot take any bundle with a price larger than her budget. □

## E MISTAKE PROFILE CALIBRATION

In this section, we present details on our mistake profile calibration.

*Results.* Table 5 shows detailed results of our calibration procedure. For each row, we multiply the default reporting mistake parameters $f_b, f_a, \sigma_b, \sigma_a$ as defined in Section 5.3 by the common constant $\gamma$. We see that our mistake calibration for the default common mistake constant $\gamma = 1$ (i.e, the rows marked in grey) closely matches the metrics reported in Budish and Kessler [2022].

| Setting | | #Courses with value | | | #Adjustments | | | | Accuracy | If disagreement |
|---|---|---|---|---|---|---|---|---|---|---|
| #Pop | $\gamma$ | $> 0$ | $(0, 50)$ | $[50, 100]$ | Mean | Median | Min | Max | CQs | utility difference in % |
| 9 | 0.5 | 18.75 ± 0.02 | 14.35 ± 0.07 | 4.40 ± 0.07 | 3.72 ± 0.12 | 3.0 | 0 | 17 | 0.93 ± 0.01 | -3.81 ± 0.41 |
| 9 | 0.75 | 15.62 ± 0.02 | 10.31 ± 0.08 | 5.30 ± 0.08 | 2.06 ± 0.09 | 1.0 | 0 | 12 | 0.87 ± 0.01 | -9.41 ± 0.62 |
| 9 | 0.9 | 13.75 ± 0.02 | 7.91 ± 0.09 | 5.84 ± 0.08 | 1.37 ± 0.07 | 1.0 | 0 | 11 | 0.84 ± 0.01 | -13.10 ± 0.72 |
| 9 | 1.0 | 12.49 ± 0.02 | 6.32 ± 0.09 | 6.17 ± 0.09 | 0.98 ± 0.05 | 1.0 | 0 | 10 | 0.81 ± 0.01 | -15.52 ± 0.76 |
| 9 | 1.1 | 11.20 ± 0.02 | 4.71 ± 0.09 | 6.49 ± 0.09 | 0.69 ± 0.04 | 0.0 | 0 | 7 | 0.78 ± 0.01 | -18.11 ± 0.77 |
| 9 | 1.25 | 9.39 ± 0.02 | 2.58 ± 0.08 | 6.81 ± 0.08 | 0.37 ± 0.03 | 0.0 | 0 | 6 | 0.75 ± 0.01 | -21.60 ± 0.80 |
| 9 | 1.5 | 6.20 ± 0.02 | 0.36 ± 0.04 | 5.84 ± 0.04 | 0.10 ± 0.01 | 0.0 | 0 | 3 | 0.66 ± 0.01 | -26.78 ± 0.78 |
| 6 | 0.5 | 18.75 ± 0.02 | 13.71 ± 0.06 | 5.04 ± 0.06 | 3.83 ± 0.12 | 3.0 | 0 | 19 | 0.93 ± 0.01 | -3.11 ± 0.37 |
| 6 | 0.75 | 15.62 ± 0.02 | 10.01 ± 0.07 | 5.61 ± 0.07 | 2.17 ± 0.09 | 2.0 | 0 | 13 | 0.87 ± 0.01 | -8.02 ± 0.57 |
| 6 | 0.9 | 13.75 ± 0.02 | 7.80 ± 0.08 | 5.95 ± 0.07 | 1.47 ± 0.07 | 1.0 | 0 | 10 | 0.85 ± 0.01 | -11.42 ± 0.67 |
| 6 | 1.0 | 12.49 ± 0.02 | 6.33 ± 0.08 | 6.16 ± 0.08 | 1.08 ± 0.06 | 1.0 | 0 | 10 | 0.83 ± 0.01 | -13.34 ± 0.71 |
| 6 | 1.1 | 11.20 ± 0.02 | 4.82 ± 0.08 | 6.38 ± 0.08 | 0.73 ± 0.04 | 0.0 | 0 | 10 | 0.80 ± 0.01 | -15.48 ± 0.75 |
| 6 | 1.25 | 9.39 ± 0.02 | 2.72 ± 0.08 | 6.67 ± 0.08 | 0.41 ± 0.03 | 0.0 | 0 | 10 | 0.76 ± 0.01 | -19.12 ± 0.77 |
| 6 | 1.5 | 6.20 ± 0.02 | 0.38 ± 0.04 | 5.82 ± 0.04 | 0.11 ± 0.02 | 0.0 | 0 | 3 | 0.68 ± 0.01 | -23.83 ± 0.78 |
| [Budish and Kessler, 2022] | | 12.45 | 6.17 | 6.27 | 1.08 | 0 | 0 | 10 | 0.84 | -13.35 ± 0.41 |

Table 5. Mistake profile calibration experiment for several settings defined by the number of popular courses (#Pop) and the common mistake constant $\gamma$ compared to the experimental findings in [Budish and Kessler, 2022]. Our two default settings (i.e., $\gamma = 1$) are marked in grey. 2000 students in total. We show the number of courses with reported value in 3 distinct intervals, the mean, median, minimum and maximum number of adjustments in the students' reports, the accuracy of their reports as determined by asking CQs and the median scaled utility difference between the two schedules in a CQ in case of disagreement between the CQ answer and the reported preferences. Shown are average results and a 95% CI.

## F HYPERPARAMETER OPTIMIZATION

In this section, we present details on our hyperparameter optimization (HPO).

*HPO Method.* The parameter selection for all the models is done using the *optuna* hyperparameter selection algorithm by Akiba et al. [2019]. The number of trials for optuna for all models were equal and was selected to be 100. The range of parameters for each model is presented in Table 6.

| ML Models | Parameter | HPO-Range |
|---|---|---|
| NN, MVNN | Hidden Layers | [1,2,3] |
| | Units/Hidden Layer | [8, 16, 32, 50, 80, 120, 200, 300] |
| | Learning Rate | (1e-4, 1e-2) |
| | Epochs | (80,700) |
| | $l_2$ Regularization | (1e-15, 1e-3) |
| | Batch Size | [4, 8, 16, 30, 50] |
| XGBoost | Eta | ( 1e-5 , 0.2) |
| | Colsample Bytree | (0.4, 0.8) |
| | Gamma | (0, 0.02) |
| | Learning Rate | (0.03, 0.3) |
| | Max Depth | (2,6) |
| | N_Estimators | (10, 200) |
| | Subsample | (0.4, 0.6) |
| nuSVR | Nu | (0, 0.25) |
| | Gamma | (1e-4, 0.9) |
| Ridge | Alpha | (0,1) |

Table 6. HPO-ranges used for the optuna algorithm.

*Experiment Setup.* As in the experiment by Budish and Kessler [2022], we consider a setting with 25 courses. We use our simulator to create 100 instances of student preferences, where we use 20 to tune the ML hyperparameters (HPs). In each of these 20 instances, we use the cardinal data $\{D_{i,\text{card}}\}_{i \in N}$ that MLCM generated based on the students' GUI reports as the training set, which we denote by GUI in the result tables (see Appendix A for details). Second, we use $\{30, 50, 100, 150\}$ *random* value queries (RAND VQs) each consisting of five courses as the training set. We used the MAE on the complement of the training set to select the reported hyper-parameters.

*Results.* In Tables 7 to 11 we present the selected HPs of the respective ML models for 6 popular courses. In Tables 12 to 16 we present the selected HPs of the respective ML models for 9 popular courses.

| Data Type | Hidden Layers | Units per Hidden Layer | Learning Rate | Epochs | L2-Regularization | Batch Size |
|---:|---:|---:|---:|---:|---:|---:|
| GUI | 2 | 80 | 0.00074 | 102 | 0.00033 | 16 |
| 30 RAND VQs | 1 | 200 | 0.00965 | 288 | 0.00062 | 16 |
| 50 RAND VQs | 2 | 120 | 0.00666 | 80 | 0.00079 | 50 |
| 100 RAND VQs | 3 | 300 | 0.00641 | 454 | 0.00100 | 4 |
| 150 RAND VQs | 1 | 50 | 0.00646 | 197 | 0.00031 | 4 |

Table 7. Selected HPs for MVNN for 6 popular courses.

| Data Type | Hidden Layers | Units/Hidden Layer | Learning Rate | Epochs | L2-Regularization | Batch Size |
|---:|---:|---:|---:|---:|---:|---:|
| GUI | 3 | 80 | 0.00710 | 545 | 0.00094 | 16 |
| 30 RAND VQs | 2 | 8 | 0.00573 | 697 | 0.00088 | 4 |
| 50 RAND VQs | 1 | 120 | 0.00956 | 538 | 0.00054 | 4 |
| 100 RAND VQs | 2 | 8 | 0.00821 | 681 | 0.00043 | 8 |
| 150 RAND VQs | 2 | 32 | 0.00989 | 371 | 0.00049 | 4 |

Table 8. Selected HPs for NN for 6 popular courses.

| Data Type | Eta | Colsample Bytree | Gamma | Learning Rate | Max Depth | N_Estimators | Subsample |
|---:|---|---:|---:|---:|---:|---:|---:|
| GUI | 0.03 | 0.59 | 0 | 0.18 | 3 | 159 | 0.44 |
| 30 RAND VQs | 0.13 | 0.64 | 0 | 0.13 | 2 | 140 | 0.50 |
| 50 RAND VQs | 0.11 | 0.72 | 0 | 0.09 | 6 | 146 | 0.45 |
| 100 RAND VQs | 0.17 | 0.47 | 0 | 0.18 | 4 | 170 | 0.59 |
| 150 RAND VQs | 0.03 | 0.77 | 0 | 0.12 | 4 | 200 | 0.45 |

Table 9. Selected HPs for XGBoost for 6 popular courses.

| Data Type | Nu | Gamma |
|---:|---|---:|
| GUI | 0.24 | 0.05 |
| 30 RAND VQs | 0.25 | 0.04 |
| 50 RAND VQs | 0.25 | 0.04 |
| 100 RAND VQs | 0.24 | 0.04 |
| 150 RAND VQs | 0.24 | 0.05 |

Table 10. Selected HPs for nuSVR with Gaussian kernel for 6 popular courses.

## G GENERALIZATION PERFORMANCE RESULTS WITH 25 COURSES AND 6 POPULAR COURSES

*Experiment Setup.* As in the experiment by Budish and Kessler [2022], we consider a setting with 25 courses. We use our simulator to create 100 instances of student preferences, where we use 20 to tune the ML hyperparameters (see Appendix F for details), and 80 for testing. We perform two

| Data Type | Alpha |
|---|---|
| GUI | 0.22 |
| 30 RAND VQs | 0.39 |
| 50 RAND VQs | 0.38 |
| 100 RAND VQs | 0.41 |
| 150 RAND VQs | 0.45 |

Table 11. Selected HPs for Ridge for 6 popular courses.

| Data Type | Hidden Layers | Units/Hidden Layer | Learning Rate | Epochs | L2-Regularization | Batch Size |
|---|---|---|---|---|---|---|
| GUI | 2 | 32 | 0.00979 | 342 | 0.00009 | 8 |
| 30 RAND VQs | 1 | 200 | 0.00998 | 604 | 0.00070 | 30 |
| 50 RAND VQs | 2 | 50 | 0.00586 | 414 | 0.00072 | 4 |
| 100 RAND VQs | 3 | 50 | 0.00949 | 401 | 0.00085 | 8 |
| 150 RAND VQs | 2 | 120 | 0.00165 | 305 | 0.00084 | 4 |

Table 12. Selected HPs for MVNN for 9 popular courses.

| Data Type | Hidden Layers | Units/Hidden Layer | Learning Rate | Epochs | L2-Regularization | Batch Size |
|---|---|---|---|---|---|---|
| GUI | 2 | 300 | 0.00669 | 247 | 0.00099 | 4 |
| 30 RAND VQs | 2 | 300 | 0.00472 | 700 | 0.00096 | 16 |
| 50 RAND VQs | 2 | 8 | 0.00730 | 591 | 0.00048 | 16 |
| 100 RAND VQs | 1 | 50 | 0.00682 | 657 | 0.00078 | 16 |
| 150 RAND VQs | 1 | 300 | 0.00496 | 367 | 0.00075 | 4 |

Table 13. Selected HPs for NN for 9 popular courses.

| Data Type | Eta | Colsample Bytree | Gamma | Learning Rate | Max Depth | N_Estimators | Subsample |
|---|---|---|---|---|---|---|---|
| GUI | 0.00 | 0.52 | 0 | 0.09 | 4 | 165 | 0.56 |
| 30 RAND VQs | 0.18 | 0.68 | 0 | 0.14 | 6 | 112 | 0.40 |
| 50 RAND VQs | 0.06 | 0.57 | 0 | 0.21 | 3 | 85 | 0.54 |
| 100 RAND VQs | 0.10 | 0.64 | 0 | 0.10 | 2 | 189 | 0.42 |
| 150 RAND VQs | 0.05 | 0.68 | 0 | 0.24 | 3 | 93 | 0.43 |

Table 14. Selected HPs for XGBoost for 9 popular courses.

| Data Type | Nu | Gamma |
|---|---|---|
| GUI | 0.22 | 0.05 |
| 30 RAND VQs | 0.24 | 0.04 |
| 50 RAND VQs | 0.24 | 0.04 |
| 100 RAND VQs | 0.13 | 0.05 |
| 150 RAND VQs | 0.25 | 0.04 |

Table 15. Selected HPs for nuSVR with Gaussian kernel for 9 popular courses.

experiments. First, we use the cardinal data $\{D_{i,\text{card}}\}_{i\in N}$ that MLCM generated based on the students' GUI reports as the training set, which we denote by GUI in the result tables (see Appendix A for

| Data Type | Alpha |
|---|---|
| GUI | 0.22 |
| 30 RAND VQs | 0.22 |
| 50 RAND VQs | 0.27 |
| 100 RAND VQs | 0.36 |
| 150 RAND VQs | 0.45 |

Table 16. Selected HPs for Ridge for 9 popular courses.

details). Second, we use $\{30, 50, 100, 150\}$ *random* value queries (RAND VQs) each consisting of five courses as the training set. To test the trained models, we use the complement of the training set as the test set and report the mean absolute error (MAE) and Kendall tau (KT) of each ML model.

*Results.* In Tables 17 and 18, we provide the generalization performance results for the *mean absolute error (MAE)* and the *Kendall tau* rank correlation with 25 courses in total and 6 popular courses.

| Data Type | Ridge | nuSVR | XGBoost | NN | MVNN |
|---|---|---|---|---|---|
| GUI | 37.45± 1.17 | 37.63± 1.13 | 38.38± 1.30 | 38.25± 1.09 | 37.38± 1.45 |
| 30 RAND VQs | 22.35± 1.07 | 26.64± 0.85 | 25.82± 0.86 | 25.05± 0.89 | 20.65± 1.02 |
| 50 RAND VQs | 15.08± 0.92 | 17.01± 0.69 | 19.96± 0.67 | 17.77± 0.77 | 13.46± 0.78 |
| 100 RAND VQs | 11.37± 0.72 | 10.80± 0.60 | 15.74± 0.53 | 11.80± 0.56 | 7.80± 0.57 |
| 100 RAND VQs | 10.56± 0.69 | 9.42± 0.56 | 13.68± 0.42 | 7.81± 0.50 | 4.79± 0.38 |

Table 17. MAE on the test set. 25 courses in total and 6 popular courses. Winners are marked in grey.

| Data Type | Ridge | nuSVR | XGBoost | NN | MVNN |
|---|---|---|---|---|---|
| GUI | 0.34± 0.02 | 0.34± 0.02 | 0.33± 0.02 | 0.31± 0.02 | 0.30± 0.03 |
| 30 RAND VQs | 0.65± 0.02 | 0.61± 0.01 | 0.59± 0.01 | 0.60± 0.01 | 0.68± 0.02 |
| 50 RAND VQs | 0.78± 0.01 | 0.76± 0.01 | 0.70± 0.01 | 0.73± 0.01 | 0.80± 0.01 |
| 100 RAND VQs | 0.83± 0.01 | 0.84± 0.01 | 0.77± 0.01 | 0.83± 0.01 | 0.88± 0.01 |
| 150 RAND VQs | 0.84± 0.01 | 0.86± 0.01 | 0.81± 0.01 | 0.88± 0.01 | 0.93± 0.01 |

Table 18. Kendall tau on the test set (larger is better). 25 courses in total and 6 popular courses. Winners marked in grey.

## H GENERALIZATION PERFORMANCE RESULTS WITH 25 COURSES AND 9 POPULAR COURSES

*Experiment Setup.* We use the exact same experiment setup as described in Appendix G, except for the number of popular courses which we set to 9 when generating students' preferences with our proposed student preference generator defined in Section 5.1 and Appendix C.

*Results.* In Tables 19 and 20, we provide the generalization performance results for the *mean absolute error (MAE)* and the *Kendall tau* rank correlation with 25 courses in total and 9 popular courses.

| Data Type | Ridge | nuSVR | XGBoost | NN | MVNN |
|---|---|---|---|---|---|
| GUI | 34.01± 1.18 | 34.85± 1.13 | 34.99± 1.23 | 34.85± 1.12 | 34.52± 1.42 |
| 30 RAND VQs | 19.36± 1.04 | 21.76± 0.81 | 23.01± 0.88 | 23.25± 0.88 | 18.64± 0.99 |
| 50 RAND VQs | 12.88± 0.83 | 14.31± 0.59 | 15.79± 0.54 | 15.71± 0.65 | 11.93± 0.65 |
| 100 RAND VQs | 9.87± 0.68 | 9.40± 0.50 | 15.75± 0.55 | 9.32± 0.56 | 5.84± 0.53 |
| 150 RAND VQs | 9.18± 0.61 | 7.89± 0.44 | 13.72± 0.41 | 5.62± 0.48 | 3.50± 0.38 |

Table 19. MAE on the test set. 25 courses in total and 9 popular courses. Winners are marked in grey.

| Data Type | Ridge | nuSVR | XGBoost | NN | MVNN |
|---|---|---|---|---|---|
| GUI | 0.35± 0.02 | 0.34± 0.02 | 0.32± 0.02 | 0.30± 0.02 | 0.31± 0.02 |
| 30 RAND VQs | 0.68± 0.02 | 0.65± 0.01 | 0.60± 0.02 | 0.59± 0.01 | 0.69± 0.02 |
| 50 RAND VQs | 0.79± 0.01 | 0.78± 0.01 | 0.74± 0.01 | 0.75± 0.01 | 0.81± 0.01 |
| 100 RAND VQs | 0.84± 0.01 | 0.85± 0.01 | 0.74± 0.01 | 0.86± 0.01 | 0.90± 0.01 |
| 150 RAND VQs | 0.85± 0.01 | 0.87± 0.01 | 0.78± 0.01 | 0.9± 0.01 | 0.94± 0.01 |

Table 20. Kendall tau on the test set (larger is better). 25 courses in total and 9 popular courses. Winners are marked in grey.

## I INTEGRATING COMPARISON QUERIES INTO MVNNS

In this section, we describe the details how we simultaneously train the MVNNs on GUI reports (regression data) and CQs (classification data).

Sculley [2010] proposed a method called *combined ranking and regression (CRR)* to train linear regression models using *both* regression and classification data at the same time (in our application the data generated from the students answers to the GUI is the regression data and the students' answers to comparison queries is the classification data). For this they use a trade-off parameter $\alpha \in [0, 1]$ to combine a regression loss and a ranking loss into one loss. More specifically, given $\alpha$, a convex regression loss function $l_{reg}(\cdot)$, and a convex classification loss function $l_{class}(\cdot)$, we define the hybrid loss $l_h$ as follows:

$$l_h(\cdot) = \alpha l_{reg}(\cdot) + (1 - \alpha) l_{class}(\cdot). \tag{41}$$

Using this hybrid loss, we fit the parameters of the MVNNs using Algorithm 2, which trains the MVNN using Adam on both the data from the GUI (regression data) and the data from the CQs (classification data).

The main idea of Algorithm 2 is to choose the data from GUI or the CQs to update the parameters of MVNN. In Algorithm 2, with probability $\alpha$ a data point from the regression data set is chosen (Line 4). Similarly, a data point from the classification data is chosen with probability $1 - \alpha$ (Line 8). If the data from the CQs is selected, we use the Sigmoid function $f(x) = \frac{1}{1+e^{-x}}$ to convert the real-valued outputs of MVNN$(\cdot)$ to the $[0, 1]$-interval, where the value of $\frac{1}{1+e^{-(\hat{y}_1 - \hat{y}_2)}}$ represents the predicted probability that bundle $x_1$ is more valuable than bundle $x_2$ for the student (Line 13).

**ALGORITHM 2:** Combined ranking and regression for MVNN

```
Input: $\{X_{reg}, y_{reg}\}$, $\{X_{class}, y_{class}\}$
Parameters: $\alpha \in [0,1]$, epochs $t$, learning rate $\eta$
Output: Parameters of trained MVNN
1: $\theta_0 \leftarrow$ initialize parameters of the MVNN($\cdot$)
2: for $i = 1$ to $t$ do
3:     pick $z$ uniformly at random from $[0,1]$
4:     if $z < \alpha$ then
5:         $(x, y) \overset{Unif}{\sim} \{X_{reg}, y_{reg}\}$
6:         $\hat{y} = \text{MVNN}(x)$
7:         $loss = l_{reg}(y, \hat{y})$
8:     else
9:         $(x, y) \overset{Unif}{\sim} \{X_{class}, y_{class}\}$
10:        $(x_1, x_2) = x$
11:        $\hat{y}_1 = \text{MVNN}(x_1)$
12:        $\hat{y}_2 = \text{MVNN}(x_2)$
13:        $\hat{y} = \frac{1}{1+e^{-(\hat{y}_1 - \hat{y}_2)}}$
14:        $loss = l_{class}(y, \hat{y})$
15:    end if
16:    $\theta_i = \text{ADAM}(\theta_{i-1}, loss, \eta)$
17: end for
18: return $\theta_t$
```

ADAM is then used to update the parameters of MVNN. The algorithm terminates after $t$ epochs and returns the selected parameters.

## J INFERRING MISSING BASE VALUES - A WORKED EXAMPLE

In Example J.1, we show how MLCM can infer missing base values, a common error observed in the lab experiment by Budish and Kessler [2022].

| Preference Model | Courses 1 | 2 | 3 | 4 | Utility Maximizing Schedule $a_1^*$ | Utility $u_1(a_1^*)$ | Elicited CQ |
|---|---|---|---|---|---|---|---|
| $u_1$ | 85 | 70 | 40 | 0 | {1,3} | 125 | |
| $u_1^{GUI}$ | 75 | 76 | 0 | 0 | {2} | 70 | |
| $\mathcal{M}_1^{t=0}$ | 75 | 76 | 38 | 38 | {2,3} | 110 | $\{2,3\} \succ \{2,4\}$ |
| $\mathcal{M}_1^{t=1}$ | 75 | 76 | 41 | 33 | {2,3} | 110 | $\{1,3\} \succ \{2,3\}$ |
| $\mathcal{M}_1^{t=2}$ | 78 | 71 | 44 | 36 | {1,3} | 125 | $\{2,3\} \succ \{1,4\}$ |
| $\mathcal{M}_1^{t=3}$ | 78 | 72 | 45 | 32 | {1,3} | 125 | $\{1,4\} \succ \{2,4\}$ |
| $\mathcal{M}_1^{t=4}$ | 78 | 72 | 45 | 32 | {1,3} | 125 | |

Table 21. Worked example illustrating the ML-based preference elicitation algorithm. Each row represents the linear coefficients (corresponding to the base values) that uniquely define the corresponding function, the current utility maximizing schedule $a_1^*$, its corresponding utility $u_1(a_1^*)$ and the answer to the CQ.

Example J.1 (Inferring Missing Base Values). *In this example, we assume that the student forgot to report a base value for courses 3 and 4. This is treated in the original CM reporting language* *as inserting a base value equal to zero in the GUI, i.e.,* $u_1^{GUI}(\{3\}) = u_1^{GUI}(\{4\}) = 0$ *(as can be seen in the second row of Table 21). First, note that our cardinal dataset generation procedure (see Appendix A for details) implies that any schedule for which a student forgot to report a base value has the same initial predicted base value (as defined in Equation (5)), which is not necessarily zero but lower than the base value of any non-forgotten course. This results in* $\mathcal{M}_1^{t=0}(\{3\}) = \mathcal{M}_1^{t=0}(\{4\}) = 38$ *instead of zero.*

*After this point, MLCM proceeds again in the same way as in Example 4.3 until the student stops answering CQs.*

## K WELFARE EXPERIMENTS DETAILS

For all welfare experiments we selected the MVNN hyperparameters shown in Table 22. .

| Hidden Layers | Units per Hidden Layer | Learning Rate | Epochs | L2-Regularization | Batch Size |
|---|---|---|---|---|---|
| 3 | 20 | 0.001 | 400 | 6e-5 | 8 |

Table 22. Selected hyperparameters for MVNNs in MLCM.

*Computing Infrastructure.* All welfare experiments were conducted on a compute cluster running Debian GNU/Linux 10 with Intel Xeon E5-2650 v4 2.20GHz processors with 24 cores and 128GB RAM and Intel E5 v2 2.80GHz processors with 20 cores and 128GB RAM and Python 3.7.10.

## L WELFARE RESULTS FOR OTHER SETTINGS

In this section, we provide additional welfare results for the following three settings defined by a supply ratio (SR) and a number of popular courses (#Pop):

- SR = 1.50 and #Pop = 9 (see Table 23),
- SR = 1.50 and #Pop = 6 (see Table 24),
- SR = 1.25 and #Pop = 9 (see Table 25),

where we use the same experimental setup as described in the main paper in Section 7.2.

*Results.* From Table 23, Table 24, and Table 25, we see that the results (both average and minimum student utility) are better for SR = 1.5 and worse for 6 popular courses but qualitatively similar compared to the results presented in the main paper in Section 7.2.

Namely, for SR = 1.5 and 9 popular courses MLCM (10 ML-BASED CQs) increases average utility from 79.2% to 86.5% (a 8.8% increase) and minimum utility from 41.6% to 50.4% (a 21.2% increase). At the most extreme setting of SR = 1.25 and 6 popular courses, MLCM (10 ML-BASED CQs) increases average utility compared to CM from 83.4% to 86.7% (a 4% increase) and minimum utility from 50.0% to 55.0% (a 10.0% increase). For all settings, as the number of CQs increases, the performance of MLCM improves further.

## M STATISTICAL SIGNIFICANCE TESTS

To test whether our results from Section 7 are statistically significant, we perform a *multivariate analysis of variance (MANOVA)* test, using as the independent variable the mechanism and as dependent variables (a) average student utility, and (b) the minimum student utility. We use as significance level for this test a value of 0.05. The MANOVA test determined that there is a statistically significant treatment effect between the different mechanisms. Given this, we then performed a Post-Hoc tukey test for all pairs of mechanisms. For those tests, we used one dependent

| | Avgerage Student Utility | | | Minimum Student Utility | | | Overs. | Time |
|---|---|---|---|---|---|---|---|---|
| Mechanism | Stage 1 | Stage 2 | Stage 3 | Stage 1 | Stage 2 | Stage 3 | Stage 1 | in h |
| CM* (Full Preferences) | 100.0 ± 0.0 | 98.5 ± 0.3 | 99.7 ± 0.2 | 71.5 ± 1.0 | 71.1 ± 1.0 | 71.4 ± 1.0 | 3.3 ± 0.6 | 3.3 |
| CM (No Mistakes) | 98.4 ± 0.3 | 97.3 ± 0.3 | 98.3 ± 0.3 | 71.2 ± 1.0 | 70.7 ± 1.0 | 71.1 ± 1.0 | 2.5 ± 0.5 | 2.9 |
| RSD | - | - | 78.2 ± 0.5 | - | - | 36.2 ± 1.2 | - | 0.0 |
| CM | 79.3 ± 0.5 | 78.7 ± 0.4 | 79.2 ± 0.5 | 41.7 ± 1.1 | 41.7 ± 1.1 | 41.6 ± 1.1 | 1.3 ± 0.3 | 1.4 |
| MLCM ( 1 ML-based CQ) | 79.2 ± 0.5 | 78.7 ± 0.5 | 79.1 ± 0.5 | 41.0 ± 1.2 | 40.8 ± 1.2 | 40.9 ± 1.2 | 1.2 ± 0.3 | 13.6 |
| MLCM ( 5 ML-based CQs) | 83.8 ± 0.4 | 83.2 ± 0.4 | 83.7 ± 0.4 | 47.3 ± 1.2 | 46.8 ± 1.2 | 47.0 ± 1.1 | 1.1 ± 0.2 | 14.1 |
| MLCM (10 ML-based CQs) | 86.6 ± 0.4 | 86.0 ± 0.4 | 86.5 ± 0.4 | 50.3 ± 1.2 | 50.0 ± 1.2 | 50.4 ± 1.2 | 1.1 ± 0.3 | 12.4 |
| MLCM (15 ML-based CQs) | 88.4 ± 0.4 | 87.8 ± 0.4 | 88.3 ± 0.4 | 51.8 ± 1.3 | 51.4 ± 1.3 | 51.6 ± 1.3 | 1.2 ± 0.3 | 13.2 |
| MLCM (20 ML-based CQs) | 89.3 ± 0.4 | 88.6 ± 0.4 | 89.2 ± 0.4 | 54.2 ± 1.2 | 53.7 ± 1.2 | 53.9 ± 1.2 | 1.3 ± 0.3 | 16.7 |
| MLCM (20 Random CQs) | 78.7 ± 0.5 | 78.2 ± 0.5 | 78.6 ± 0.5 | 41.0 ± 1.2 | 40.8 ± 1.2 | 40.9 ± 1.2 | 1.4 ± 0.3 | 16.7 |

Table 23. Comparison of RSD, CM and MLCM (also using random CQs) in Stages 1–3 for a supply ratio of 1.5, 9 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

| | Avgerage Student Utility | | | Minimum Student Utility | | | Overs. | Time |
|---|---|---|---|---|---|---|---|---|
| Mechanism | Stage 1 | Stage 2 | Stage 3 | Stage 1 | Stage 2 | Stage 3 | Stage 1 | in h |
| CM* (Full Preferences) | 100.0 ± 0.0 | 95.7 ± 0.7 | 98.4 ± 0.5 | 77.5 ± 0.7 | 74.0 ± 0.9 | 75.8 ± 0.9 | 8.7 ± 1.6 | 3.8 |
| CM (No Mistakes) | 98.7 ± 0.4 | 93.7 ± 0.7 | 96.6 ± 0.6 | 76.5 ± 0.8 | 72.7 ± 0.9 | 73.9 ± 0.9 | 10.8 ± 1.5 | 3.2 |
| RSD | - | - | 81.0 ± 0.7 | - | - | 34.8 ± 1.3 | - | 0.0 |
| CM | 83.8 ± 0.6 | 81.3 ± 0.7 | 83.4 ± 0.6 | 49.4 ± 1.4 | 48.1 ± 1.4 | 48.8 ± 1.4 | 5.0 ± 0.9 | 1.4 |
| MLCM ( 1 ML-based CQ) | 83.3 ± 0.6 | 81.1 ± 0.7 | 83.0 ± 0.6 | 50.2 ± 1.3 | 48.3 ± 1.3 | 49.8 ± 1.3 | 4.4 ± 0.8 | 21.4 |
| MLCM ( 5 ML-based CQs) | 87.1 ± 0.6 | 83.7 ± 0.8 | 86.1 ± 0.6 | 55.0 ± 1.3 | 51.4 ± 1.4 | 52.7 ± 1.4 | 5.7 ± 1.0 | 20.8 |
| MLCM (10 ML-based CQs) | 89.2 ± 0.5 | 85.6 ± 0.7 | 88.2 ± 0.5 | 57.7 ± 1.3 | 54.5 ± 1.2 | 55.9 ± 1.3 | 5.5 ± 0.9 | 21.0 |
| MLCM (15 ML-based CQs) | 90.3 ± 0.6 | 86.2 ± 0.8 | 88.9 ± 0.6 | 59.7 ± 1.2 | 55.1 ± 1.4 | 56.9 ± 1.3 | 5.8 ± 1.0 | 22.7 |
| MLCM (20 ML-based CQs) | 90.8 ± 0.5 | 87.2 ± 0.7 | 89.9 ± 0.6 | 59.2 ± 1.3 | 56.4 ± 1.2 | 57.6 ± 1.2 | 5.6 ± 0.9 | 23.3 |
| MLCM (20 Random CQs) | 83.0 ± 0.6 | 80.6 ± 0.6 | 82.5 ± 0.6 | 50.0 ± 1.3 | 48.9 ± 1.4 | 49.5 ± 1.4 | 4.4 ± 0.7 | 22.1 |

Table 24. Comparison of RSD, CM and MLCM (also using random CQs) in Stages 1–3 for a supply ratio of 1.5, 6 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

| | Avgerage Student Utility | | | Minimum Student Utility | | | Overs. | Time |
|---|---|---|---|---|---|---|---|---|
| Mechanism | Stage 1 | Stage 2 | Stage 3 | Stage 1 | Stage 2 | Stage 3 | Stage 1 | in h |
| CM* (Full Preferences) | 100.0 ± 0.0 | 94.4 ± 1.0 | 97.7 ± 0.9 | 78.0 ± 0.7 | 74.0 ± 1.1 | 75.5 ± 1.0 | 13.7 ± 2.1 | 5.0 |
| CM (No Mistakes) | 98.4 ± 0.5 | 92.7 ± 1.3 | 96.0 ± 1.1 | 77.1 ± 0.7 | 73.2 ± 1.3 | 75.1 ± 1.2 | 11.6 ± 2.0 | 4.3 |
| RSD | - | - | 80.5 ± 0.8 | - | - | 31.3 ± 1.4 | - | 0.0 |
| CM | 83.7 ± 0.6 | 81.2 ± 0.7 | 83.4 ± 0.7 | 49.8 ± 1.3 | 49.1 ± 1.2 | 50.0 ± 1.3 | 5.4 ± 0.9 | 2.0 |
| MLCM ( 1 ML-based CQ) | 83.1 ± 0.6 | 80.6 ± 0.8 | 82.4 ± 0.7 | 50.3 ± 1.4 | 48.2 ± 1.3 | 49.1 ± 1.4 | 5.2 ± 0.8 | 27.7 |
| MLCM ( 5 ML-based CQs) | 86.4 ± 0.6 | 82.4 ± 1.2 | 84.6 ± 1.1 | 54.6 ± 1.3 | 51.3 ± 1.5 | 52.1 ± 1.5 | 5.6 ± 0.8 | 30.1 |
| MLCM (10 ML-based CQs) | 88.4 ± 0.6 | 84.2 ± 1.4 | 86.7 ± 1.3 | 56.9 ± 1.4 | 54.0 ± 1.4 | 55.0 ± 1.5 | 6.1 ± 1.1 | 29.8 |
| MLCM (15 ML-based CQs) | 89.6 ± 0.6 | 85.6 ± 1.2 | 88.2 ± 1.1 | 59.1 ± 1.3 | 55.9 ± 1.6 | 58.2 ± 1.6 | 6.0 ± 1.0 | 30.9 |
| MLCM (20 ML-based CQs) | 90.9 ± 0.6 | 86.5 ± 1.2 | 88.9 ± 1.1 | 60.6 ± 1.3 | 56.4 ± 1.7 | 58.9 ± 1.6 | 6.6 ± 1.1 | 31.5 |
| MLCM (20 Random CQs) | 83.0 ± 0.6 | 79.8 ± 0.9 | 81.9 ± 0.8 | 49.8 ± 1.3 | 47.7 ± 1.3 | 48.9 ± 1.4 | 5.7 ± 1.0 | 28.7 |

Table 25. Comparison of RSD, CM and MLCM (also using random CQs) in Stages 1–3 for a supply ratio of 1.25, 6 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

variable (i.e., either average or minimum student utility), with a significance level of 0.025 where the null hypothesis is that there is no treatment effect between Group1 and Group2, i.e., $\mathcal{H}_0 : \mu_{Group1} = \mu_{Group2}$. The results for each setting are provided in Tables 26 to 33.

| group1 | group2 | meandiff | p-adj | lower | upper | reject |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-based CQs) | 0.0563 | 0.0010 | 0.0417 | 0.0708 | True |
| CM | MLCM (20 ML-based CQs) | 0.0899 | 0.0010 | 0.0754 | 0.1044 | True |
| CM | MLCM (20 Random CQs) | -0.0075 | 0.5326 | -0.0220 | 0.0070 | False |
| CM | RSD | -0.0216 | 0.0010 | -0.0361 | -0.0071 | True |
| MLCM (10 ML-based CQs) | MLCM (20 ML-based CQs) | 0.0337 | 0.0010 | 0.0191 | 0.0482 | True |
| MLCM (10 ML-based CQs) | MLCM (20 Random CQs) | -0.0638 | 0.0010 | -0.0783 | -0.0492 | True |
| MLCM (10 ML-based CQs) | RSD | -0.0778 | 0.0010 | -0.0924 | -0.0633 | True |
| MLCM (20 ML-based CQs) | MLCM (20 Random CQs) | -0.0974 | 0.0010 | -0.1119 | -0.0829 | True |
| MLCM (20 ML-based CQs) | RSD | -0.1115 | 0.0010 | -0.1260 | -0.0970 | True |
| MLCM (20 Random CQs) | RSD | -0.0141 | 0.0324 | -0.0286 | 0.0004 | False |

Table 26. Post-hoc tukey test for average student utility. Supply ratio 1.25, 9 popular courses. Significance level for Reject column was set to 0.025.

| group1 | group2 | meandiff | p-adj | lower | upper | reject |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-based CQs) | 0.0627 | 0.0010 | 0.0353 | 0.0900 | True |
| CM | MLCM (20 ML-based CQs) | 0.0937 | 0.0010 | 0.0664 | 0.1210 | True |
| CM | MLCM (20 Random CQs) | -0.0152 | 0.4661 | -0.0425 | 0.0122 | False |
| CM | RSD | -0.1224 | 0.0010 | -0.1498 | -0.0951 | True |
| MLCM (10 ML-based CQs) | MLCM (20 ML-based CQs) | 0.0310 | 0.0069 | 0.0037 | 0.0584 | True |
| MLCM (10 ML-based CQs) | MLCM (20 Random CQs) | -0.0778 | 0.0010 | -0.1052 | -0.0505 | True |
| MLCM (10 ML-based CQs) | RSD | -0.1851 | 0.0010 | -0.2125 | -0.1578 | True |
| MLCM (20 ML-based CQs) | MLCM (20 Random CQs) | -0.1089 | 0.0010 | -0.1362 | -0.0815 | True |
| MLCM (20 ML-based CQs) | RSD | -0.2161 | 0.0010 | -0.2435 | -0.1888 | True |
| MLCM (20 Random CQs) | RSD | -0.1073 | 0.0010 | -0.1346 | -0.0800 | True |

Table 27. Post-hoc tukey test for minimum student utility. Supply ratio 1.25, 9 popular courses. Significance level for Reject column was set to 0.025.

| group1 | group2 | meandiff | p-adj | lower | upper | reject |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-based CQs) | 0.0724 | 0.0010 | 0.0628 | 0.0820 | True |
| CM | MLCM (20 ML-based CQs) | 0.0993 | 0.0010 | 0.0897 | 0.1089 | True |
| CM | MLCM (20 Random CQs) | -0.0066 | 0.2454 | -0.0162 | 0.0030 | False |
| CM | RSD | -0.0108 | 0.0072 | -0.0204 | -0.0013 | True |
| MLCM (10 ML-based CQs) | MLCM (20 ML-based CQs) | 0.0269 | 0.0010 | 0.0173 | 0.0365 | True |
| MLCM (10 ML-based CQs) | MLCM (20 Random CQs) | -0.0790 | 0.0010 | -0.0886 | -0.0694 | True |
| MLCM (10 ML-based CQs) | RSD | -0.0832 | 0.0010 | -0.0928 | -0.0737 | True |
| MLCM (20 ML-based CQs) | MLCM (20 Random CQs) | -0.1059 | 0.0010 | -0.1155 | -0.0963 | True |
| MLCM (20 ML-based CQs) | RSD | -0.1102 | 0.0010 | -0.1197 | -0.1006 | True |
| MLCM (20 Random CQs) | RSD | -0.0043 | 0.6544 | -0.0138 | 0.0053 | False |

Table 28. Post-hoc tukey test for average student utility. Supply ratio 1.5, 9 popular courses. Significance level for Reject column was set to 0.025.

## N MISTAKE ABLATION EXPERIMENT FOR SUPPLY RATIO 1.5

In this section, we present three further reporting mistake ablation experiments for the following choices of supply ratios and number of popular courses, respectively:

- A supply ratio of 1.25 and 6 popular courses (see Figure 6).
- A supply ratio of 1.5 and 9 popular courses (see Figure 7).
- A supply ratio of 1.5 and 6 popular courses (see Figure 8).

| GROUP1 | GROUP2 | MEANDIFF | P-ADJ | LOWER | UPPER | REJECT |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-BASED CQs) | 0.0882 | 0.001 | 0.0627 | 0.1137 | TRUE |
| CM | MLCM (20 ML-BASED CQs) | 0.1229 | 0.001 | 0.0974 | 0.1484 | TRUE |
| CM | MLCM (20 RANDOM CQs) | -0.0066 | 0.900 | -0.0321 | 0.0189 | FALSE |
| CM | RSD | -0.0537 | 0.001 | -0.0792 | -0.0282 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 ML-BASED CQs) | 0.0347 | 0.001 | 0.0092 | 0.0602 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0949 | 0.001 | -0.1204 | -0.0694 | TRUE |
| MLCM (10 ML-BASED CQs) | RSD | -0.1419 | 0.001 | -0.1674 | -0.1164 | TRUE |
| MLCM (20 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.1295 | 0.001 | -0.1550 | -0.1040 | TRUE |
| MLCM (20 ML-BASED CQs) | RSD | -0.1766 | 0.001 | -0.2021 | -0.1511 | TRUE |
| MLCM (20 RANDOM CQs) | RSD | -0.0470 | 0.001 | -0.0725 | -0.0215 | TRUE |

Table 29. Post-hoc tukey test for minimum student utility. Supply ratio 1.5, 9 popular courses. Significance level for REJECT column was set to 0.025.

| GROUP1 | GROUP2 | MEANDIFF | P-ADJ | LOWER | UPPER | REJECT |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-BASED CQs) | 0.0326 | 0.0010 | 0.0125 | 0.0527 | TRUE |
| CM | MLCM (20 ML-BASED CQs) | 0.0551 | 0.0010 | 0.0350 | 0.0752 | TRUE |
| CM | MLCM (20 RANDOM CQs) | -0.0152 | 0.1631 | -0.0352 | 0.0049 | FALSE |
| CM | RSD | -0.0288 | 0.0010 | -0.0489 | -0.0087 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 ML-BASED CQs) | 0.0225 | 0.0079 | 0.0024 | 0.0426 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0478 | 0.0010 | -0.0678 | -0.0277 | TRUE |
| MLCM (10 ML-BASED CQs) | RSD | -0.0614 | 0.0010 | -0.0815 | -0.0414 | TRUE |
| MLCM (20 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0703 | 0.0010 | -0.0903 | -0.0502 | TRUE |
| MLCM (20 ML-BASED CQs) | RSD | -0.0839 | 0.0010 | -0.1040 | -0.0639 | TRUE |
| MLCM (20 RANDOM CQs) | RSD | -0.0137 | 0.2532 | -0.0337 | 0.0064 | FALSE |

Table 30. Post-hoc tukey test for average student utility. Supply ratio 1.25, 6 popular courses. Significance level for REJECT column was set to 0.025.

| GROUP1 | GROUP2 | MEANDIFF | P-ADJ | LOWER | UPPER | REJECT |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-BASED CQs) | 0.0497 | 0.0010 | 0.0191 | 0.0803 | TRUE |
| CM | MLCM (20 ML-BASED CQs) | 0.0886 | 0.0010 | 0.0580 | 0.1192 | TRUE |
| CM | MLCM (20 RANDOM CQs) | -0.0113 | 0.7813 | -0.0419 | 0.0193 | FALSE |
| CM | RSD | -0.1873 | 0.0010 | -0.2180 | -0.1567 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 ML-BASED CQs) | 0.0389 | 0.0016 | 0.0083 | 0.0695 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0610 | 0.0010 | -0.0916 | -0.0304 | TRUE |
| MLCM (10 ML-BASED CQs) | RSD | -0.2370 | 0.0010 | -0.2676 | -0.2064 | TRUE |
| MLCM (20 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0999 | 0.0010 | -0.1305 | -0.0693 | TRUE |
| MLCM (20 ML-BASED CQs) | RSD | -0.2760 | 0.0010 | -0.3066 | -0.2453 | TRUE |
| MLCM (20 RANDOM CQs) | RSD | -0.1761 | 0.0010 | -0.2067 | -0.1454 | TRUE |

Table 31. Post-hoc tukey test for minimum student utility. Supply ratio 1.25, 6 popular courses. Significance level for REJECT column was set to 0.025.

| GROUP1 | GROUP2 | MEANDIFF | P-ADJ | LOWER | UPPER | REJECT |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-BASED CQs) | 0.0479 | 0.0010 | 0.0351 | 0.0607 | TRUE |
| CM | MLCM (20 ML-BASED CQs) | 0.0651 | 0.0010 | 0.0523 | 0.0780 | TRUE |
| CM | MLCM (20 RANDOM CQs) | -0.0088 | 0.2500 | -0.0216 | 0.0041 | FALSE |
| CM | RSD | -0.0241 | 0.0010 | -0.0369 | -0.0113 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 ML-BASED CQs) | 0.0172 | 0.0010 | 0.0044 | 0.0301 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0567 | 0.0010 | -0.0695 | -0.0438 | TRUE |
| MLCM (10 ML-BASED CQs) | RSD | -0.0720 | 0.0010 | -0.0848 | -0.0591 | TRUE |
| MLCM (20 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0739 | 0.0010 | -0.0867 | -0.0611 | TRUE |
| MLCM (20 ML-BASED CQs) | RSD | -0.0892 | 0.0010 | -0.1020 | -0.0764 | TRUE |
| MLCM (20 RANDOM CQs) | RSD | -0.0153 | 0.0037 | -0.0281 | -0.0025 | TRUE |

Table 32. Post-hoc tukey test for average student utility. Supply ratio 1.5, 6 popular courses. Significance level for REJECT column was set to 0.025.

| GROUP1 | GROUP2 | MEANDIFF | P-ADJ | LOWER | UPPER | REJECT |
|---|---|---|---|---|---|---|
| CM | MLCM (10 ML-BASED CQs) | 0.0715 | 0.001 | 0.0434 | 0.0995 | TRUE |
| CM | MLCM (20 ML-BASED CQs) | 0.0882 | 0.001 | 0.0602 | 0.1163 | TRUE |
| CM | MLCM (20 RANDOM CQs) | 0.0076 | 0.900 | -0.0205 | 0.0356 | FALSE |
| CM | RSD | -0.1396 | 0.001 | -0.1676 | -0.1115 | TRUE |
| MLCM (10 ML-BASED CQs) | MLCM (20 ML-BASED CQs) | 0.0167 | 0.388 | -0.0113 | 0.0448 | FALSE |
| MLCM (10 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0639 | 0.001 | -0.0920 | -0.0359 | TRUE |
| MLCM (10 ML-BASED CQs) | RSD | -0.2110 | 0.001 | -0.2391 | -0.1830 | TRUE |
| MLCM (20 ML-BASED CQs) | MLCM (20 RANDOM CQs) | -0.0807 | 0.001 | -0.1087 | -0.0526 | TRUE |
| MLCM (20 ML-BASED CQs) | RSD | -0.2278 | 0.001 | -0.2558 | -0.1997 | TRUE |
| MLCM (20 RANDOM CQs) | RSD | -0.1471 | 0.001 | -0.1752 | -0.1191 | TRUE |

Table 33. Post-hoc tukey test for minimum student utility. Supply ratio 1.5, 6 popular courses. Significance level for REJECT column was set to 0.025.

*Results.* Similarly to the results presented in the main paper in Figure 3 (supply ratio of 1.25 and 9 popular courses), we see that, as $\gamma$ increases, the performance of both, CM and MLCM, monotonically decreases. This should not come as a surprise, as for values of $\gamma$ larger than 1 the students make more mistakes when reporting their preferences to the GUI. MLCM significantly outperforms CM for all $\gamma \in [0.75, 1.5]$. Moreover, as $\gamma$ increases, the relative performance gap of the two mechanisms gets larger. These results further highlight the robustness of our design to changes in the mistake profile of the students.

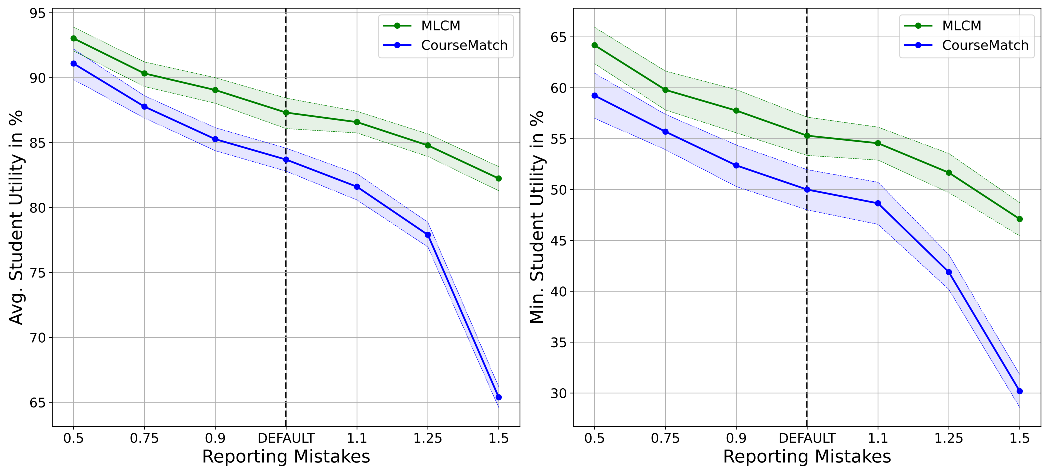

Fig. 6. Reporting mistakes ablation experiment for a supply ratio of 1.25 and 6 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

## O SHOULD INDIVIDUAL STUDENTS OPT INTO THE ML FEATURE?

In this section, we first provide additional results for a supply ratio of 1.5 for the experiment measuring the expected utility gain of a student if she were the only one to opt into MLCM, as described in Section 7.4. We present those results in Table 34. Furthermore, we perform a similar test, measuring the expected gain of a student opting into MLCM, if *every* other student also opted in. We present those results in Table 35.

*No Other Student to Opt into the ML Feature.* Similarly to the results presented in the main paper (see Table 4 for a supply ratio of 1.25), Table 34 shows that for a supply ratio equal to 1.5 the *expected*

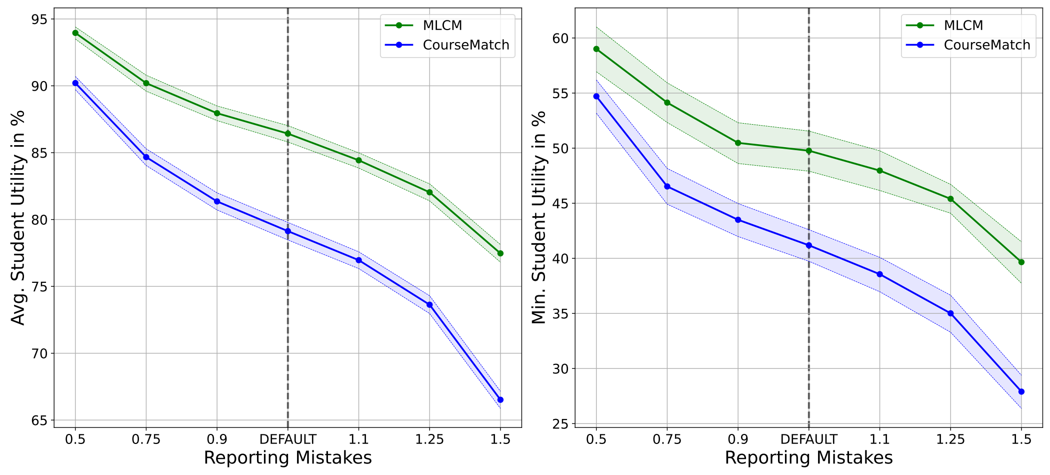

Fig. 7. Reporting mistakes ablation experiment for a supply ratio of 1.5 and 9 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

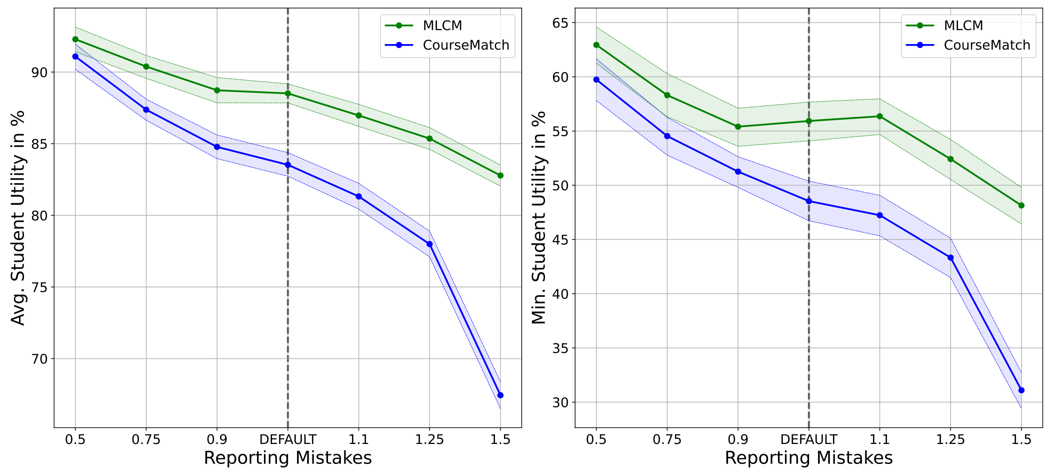

Fig. 8. Reporting mistakes ablation experiment for a supply ratio of 1.5 and 6 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

*relative gain* from opting in is at least 7.9% (across all settings). Furthermore, the student prefers the "MLCM schedule" in at least 68.45% of the cases, while she prefers the "CM schedule" in at most 7.8% of the cases. As the number of comparison queries (CQs) the student answers increases, the benefit from opting into MLCM's ML feature as the first student becomes even larger. Finally, the improvement is larger in for a setting with more popular courses.

*All Other Students to Opt into the ML Feature.* Now suppose that all students have decided to opt into MLCM's ML feature, except for one. How much would that last student benefit if she decided to also "opt in"? As before, we answer this question by running both MLCM and CM twice – once

| Setting | | | Preferred Mechanism | | | Gain from Opting Into MLCM | | |
|---|---|---|---|---|---|---|---|---|
| SR | #PoP | #CQs | MLCM | CM | Indiff. | Expected | if pref MLCM | if pref CM |
| 1.50 | 9 | 10 | 73.85% | 4.70% | 21.45% | 10.9% | 15.5% | -10.9% |
| 1.50 | 9 | 15 | 80.45% | 2.80% | 16.75% | 13.6% | 17.2% | -6.7% |
| 1.50 | 9 | 20 | 84.75% | 2.90% | 12.35% | 15.2% | 18.3% | -9.4% |
| 1.50 | 6 | 10 | 68.45% | 7.80% | 23.75% | 7.9% | 12.5% | -8.5% |
| 1.50 | 6 | 15 | 74.65% | 7.00% | 18.35% | 9.4% | 13.4% | -8.9% |
| 1.50 | 6 | 20 | 78.60% | 6.80% | 14.60% | 10.3% | 13.8% | -7.6% |

Table 34. Expected gain of opting into MLCM's ML feature when no other student opts in. Shown are average results across 2000 students per setting (SR,#PoP,#CQs). CIs for all metrics are ≈ 0.

where all but one student have opted in, and once where *all* students have opted in.[18] We use 20 instances and 100 students per instance. We report averages over those 2000 students. Table 35 shows those results for both supply ratios, 1.25 and 1.5. We see that the *expected relative gain* from opting in is at least 7.6% (across all settings). Furthermore, the student prefers the "MLCM schedule" in at least 64% of the cases, while she prefers the "CM schedule" in at most 11.15% of the cases. As the student answers more CQs, the benefit from opting into MLCM's ML feature becomes even larger. Finally, the improvement is larger for more popular courses and for a larger SR.

| Setting | | | Preferred Mechanism | | | Gain from Opting Into MLCM | | |
|---|---|---|---|---|---|---|---|---|
| SR | #PoP | #CQs | MLCM | CM | Indiff. | Expected | if pref MLCM | if pref CM |
| 1.25 | 9 | 10 | 68.20% | 9.00% | 22.80% | 10.1% | 16.0% | -9.4% |
| 1.25 | 9 | 15 | 74.00% | 7.45% | 18.55% | 12.1% | 17.2% | -8.3% |
| 1.25 | 9 | 20 | 79.50% | 7.10% | 13.40% | 14.4% | 18.8% | -7.0% |
| 1.25 | 6 | 10 | 64.00% | 11.15% | 24.85% | 7.6% | 13.5% | -9.6% |
| 1.25 | 6 | 15 | 69.95% | 10.70% | 19.35% | 8.7% | 13.8% | -8.4% |
| 1.25 | 6 | 20 | 73.05% | 11.15% | 15.80% | 9.9% | 14.9% | -8.0% |
| 1.50 | 9 | 10 | 72.50% | 5.45% | 22.05% | 10.7% | 15.5% | -9.8% |
| 1.50 | 9 | 15 | 79.25% | 3.20% | 17.55% | 13.2% | 17.1% | -8.1% |
| 1.50 | 9 | 20 | 82.35% | 5.05% | 12.60% | 14.6% | 18.3% | -9.1% |
| 1.50 | 6 | 10 | 66.60% | 9.15% | 24.25% | 7.7% | 12.7% | -7.9% |
| 1.50 | 6 | 15 | 71.70% | 9.55% | 18.75% | 8.8% | 13.6% | -9.0% |
| 1.50 | 6 | 20 | 74.90% | 9.85% | 15.25% | 9.6% | 13.9% | -7.8% |

Table 35. Expected gain of opting into MLCM's ML feature when all other students also opt in. Shown are average results across 2000 students per setting (SR,#PoP,#CQs). CIs for all metrics are ≈ 0.

## P ADDITIVE PREFERENCES

In this section, we repeat all of our experiments for the special case of students having *additive* true preferences, i.e., we use our student preference generator (see Section 5.1 and Appendix C) to generate additive/linear true students' utility functions. Again, we calibrated the mistake profile of the students so that both their accuracy and the reported utility difference in case of disagreements match those determined in the lab experiment of Budish and Kessler [2022] (see Table 36).

[18] As before, we report Stage 1 results but now use the fixed price vector that would result if all students chose to opt in.

| Setting | | #Courses with value | | | #Adjustments | | | | Accuracy | If disagreement |
|---|---|---|---|---|---|---|---|---|---|---|
| #Pop | $\gamma$ | $> 0$ | $(0, 50)$ | $[50, 100]$ | Mean | Median | Min | Max | CQs | utility difference in % |
| 9 | 0.5 | 17.20 ± 0.02 | 9.96 ± 0.10 | 7.24 ± 0.09 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.95 ± 0.01 | -1.47 ± 0.31 |
| 9 | 0.75 | 13.39 ± 0.03 | 5.66 ± 0.11 | 7.73 ± 0.11 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.91 ± 0.01 | -6.44 ± 0.73 |
| 9 | 0.9 | 11.03 ± 0.01 | 3.00 ± 0.11 | 8.03 ± 0.11 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.87 ± 0.01 | -11.11 ± 0.89 |
| 9 | 1 | 9.49 ± 0.03 | 1.53 ± 0.09 | 7.96 ± 0.09 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.84 ± 0.01 | -14.03 ± 1.00 |
| 9 | 1.1 | 7.92 ± 0.02 | 0.50 ± 0.06 | 7.42 ± 0.06 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.78 ± 0.01 | -17.70 ± 1.07 |
| 9 | 1.25 | 5.62 ± 0.03 | 0.04 ± 0.01 | 5.58 ± 0.03 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.69 ± 0.02 | -22.18 ± 1.07 |
| 9 | 1.5 | 1.75 ± 0.03 | 0.00 ± 0.00 | 1.75 ± 0.03 | 0.00 ± 0.00 | 0.0 | 0 | 0 | 0.59 ± 0.02 | -17.01 ± 1.36 |
| Budish and Kessler [2022] | | 12.45 | 6.17 | 6.27 | 1.08 | 0 | 0 | 10 | 0.84 | -13.35 ± 0.41 |

Table 36. Mistake profile calibration experiment for *additive* preferences for several settings defined by the number of popular courses (#Pop) and the common mistake constant $\gamma$ compared to the experimental findings in [Budish and Kessler, 2022]. Our default settings (i.e., $\gamma = 1$) is marked in grey. 1000 students in total. We show the number of courses with reported value in 3 distinct intervals, the mean, median, minimum and maximum number of adjustments in the students' reports, the accuracy of their reports as determined by asking CQs and the median scaled utility difference between the two schedules in a CQ in case of disagreement between the CQ answer and the reported preferences. Shown are average results and a 95% CI.

For these new preferences we present

- the main welfare results for supply ratios 1.25 and 1.5 (Table 37 and Table 38),
- the reporting mistakes ablation study for supply ratios 1.25 and 1.5 (Figure 9 and Figure 10),
- the first- and last-student to opt into MLCM experimental study (Table 39 and Table 40).

Overall our results in the following three sections show that MLCM achieves qualitatively the same results when the true students' utilities are restricted to be additive/linear. These results further highlight the robustness of our design, and its adaptability to different environments.

## P.1 Welfare Results for Additive Preferences

In this subsection, we present the main welfare results when students' true utility functions are restricted to be *additive/linear*. Please see Section 7.2 for details on the experiment setup.

In Table 37, we present results for SR = 1.25 (which is very close to Wharton's SR; see [Budish and Kessler, 2022]) and 9 popular courses. The results are qualitatively very similar for a SR of 1.5. We normalize all results by the average utility of CM$^*$ (Full Preferences) after Stage 1. Note that in this setting with additive preferences, the mechanisms CM$^*$ (Full Preferences) and CM (No Mistakes) completely coincide. The metrics of interest are the average and minimum student utility after Stage 3 (i.e., for the final allocation). We see that MLCM (10 ML-BASED CQs) significantly outperforms CM, both in average and minimum student utility. In particular, MLCM (10 ML-BASED CQs) increases average utility from 84.7% to 90.6% (a 7% increase) and minimum utility from 53.3% to 65.6% (a 23% increase). As the number of CQs increases, the performance of MLCM improves further.

In these tables, we show one additional version of MLCM, MLCM-LR. It is the same MLCM mechanism but now the machine learning model of each student is an MVNN with zero hidden layers, thus it can only capture additive preferences (as is the case for this setting). The performance achieved with these networks is on par with our original MLCM design, yet at the same time, the computational cost of calculating a final allocation is the same as for CM, as indicated by the last column of Table 37. Thus, for a setting where the school in question only wishes to capture such preferences, the welfare improvement of MLCM compared to CM comes at no additional computational cost.

| Mechanism | Average Student Utility Stage 1 | Stage 2 | Stage 3 | Minimum Student Utility Stage 1 | Stage 2 | Stage 3 | Oversubs Stage 1 | Time (in h) |
|---|---|---|---|---|---|---|---|---|
| CM* (Full Preferences) | 100.0 ± 0.0 | 89.6 ± 2.5 | 93.0 ± 2.2 | 84.5 ± 0.6 | 74.7 ± 2.3 | 77.3 ± 2.2 | 10.9 ± 1.6 | 2.6 |
| CM (No Mistakes) | 100.0 ± 0.0 | 89.6 ± 2.5 | 93.0 ± 2.2 | 84.5 ± 0.6 | 74.7 ± 2.3 | 77.3 ± 2.2 | 10.9 ± 1.6 | 2.6 |
| CM | 85.8 ± 0.4 | 82.3 ± 0.6 | 84.7 ± 0.4 | 54.9 ± 1.0 | 51.8 ± 1.1 | 53.3 ± 1.0 | 5.2 ± 0.9 | 1.2 |
| MLCM ( 1 ML-based CQs) | 88.7 ± 0.2 | 85.1 ± 0.8 | 87.5 ± 0.6 | 61.3 ± 1.0 | 58.5 ± 1.2 | 60.4 ± 1.2 | 5.6 ± 0.8 | 26.1 |
| MLCM ( 5 ML-based CQs) | 91.3 ± 0.2 | 86.9 ± 1.0 | 89.2 ± 0.9 | 65.0 ± 1.1 | 60.5 ± 1.4 | 62.3 ± 1.3 | 5.9 ± 1.0 | 25.8 |
| MLCM (10 ML-based CQs) | 93.2 ± 0.3 | 88.1 ± 1.2 | 90.6 ± 1.0 | 68.3 ± 1.1 | 62.9 ± 1.5 | 65.6 ± 1.2 | 7.2 ± 1.3 | 25.9 |
| MLCM (15 ML-based CQs) | 94.1 ± 0.2 | 88.9 ± 1.4 | 91.5 ± 1.3 | 69.8 ± 1.0 | 65.0 ± 1.6 | 66.7 ± 1.5 | 6.0 ± 1.0 | 27.3 |
| MLCM (20 ML-based CQs) | 94.9 ± 0.2 | 89.3 ± 1.3 | 92.1 ± 1.1 | 71.1 ± 1.0 | 65.4 ± 1.6 | 67.9 ± 1.4 | 6.7 ± 1.1 | 28.2 |
| MLCM-LR ( 1 ML-based CQ) | 88.6 ± 0.2 | 84.7 ± 0.8 | 87.2 ± 0.6 | 60.5 ± 1.0 | 57.5 ± 1.1 | 58.7 ± 1.1 | 5.8 ± 0.9 | 1.1 |
| MLCM-LR ( 5 ML-based CQs) | 90.7 ± 0.2 | 86.6 ± 0.8 | 89.1 ± 0.6 | 63.4 ± 1.0 | 60.3 ± 1.4 | 61.8 ± 1.2 | 6.6 ± 1.1 | 1.0 |
| MLCM-LR (10 ML-based CQs) | 91.7 ± 0.2 | 87.0 ± 1.3 | 89.5 ± 1.1 | 65.5 ± 1.1 | 61.2 ± 1.8 | 63.2 ± 1.6 | 5.6 ± 0.8 | 1.0 |
| MLCM-LR (10 ML-based CQs) | 93.0 ± 0.2 | 88.2 ± 1.4 | 90.5 ± 1.3 | 67.3 ± 1.1 | 63.2 ± 1.7 | 65.4 ± 1.5 | 6.2 ± 0.9 | 1.0 |
| MLCM-LR (20 ML-based CQs) | 93.8 ± 0.2 | 88.7 ± 1.2 | 91.3 ± 1.0 | 69.1 ± 1.0 | 64.6 ± 1.5 | 66.6 ± 1.4 | 5.6 ± 0.7 | 1.0 |

Table 37. Comparison of CM and MLCM in Stages 1–3 for *additive* preferences, a supply ratio of 1.25, 9 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of course seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

| Mechanism | Average Student Utility Stage 1 | Stage 2 | Stage 3 | Minimum Student Utility Stage 1 | Stage 2 | Stage 3 | Oversubs Stage 1 | Time (in h) |
|---|---|---|---|---|---|---|---|---|
| CM* (Full Preferences) | 100.0 ± 0.0 | 96.4 ± 0.5 | 98.6 ± 0.4 | 84.3 ± 0.6 | 81.0 ± 0.8 | 82.7 ± 0.7 | 12.8 ± 1.8 | 1.8 |
| CM (No Mistakes) | 100.0 ± 0.0 | 96.4 ± 0.5 | 98.6 ± 0.4 | 84.3 ± 0.6 | 81.0 ± 0.8 | 82.7 ± 0.7 | 12.8 ± 1.8 | 1.8 |
| CM | 88.4 ± 0.4 | 84.9 ± 0.6 | 87.4 ± 0.4 | 57.5 ± 1.1 | 54.4 ± 1.1 | 56.7 ± 1.1 | 4.1 ± 0.7 | 1.1 |
| MLCM ( 1 ML-based CQ) | 90.2 ± 0.3 | 87.8 ± 0.4 | 89.7 ± 0.3 | 62.0 ± 1.0 | 61.1 ± 1.0 | 62.1 ± 1.0 | 3.8 ± 0.7 | 20.0 |
| MLCM ( 5 ML-based CQs) | 92.5 ± 0.2 | 89.6 ± 0.4 | 91.7 ± 0.3 | 66.0 ± 1.0 | 63.1 ± 1.1 | 64.3 ± 1.1 | 5.4 ± 0.9 | 21.1 |
| MLCM (10 ML-based CQs) | 93.9 ± 0.3 | 91.0 ± 0.5 | 93.2 ± 0.3 | 68.8 ± 1.0 | 66.3 ± 1.0 | 67.5 ± 1.0 | 5.2 ± 0.8 | 20.9 |
| MLCM (15 ML-based CQs) | 94.8 ± 0.3 | 92.0 ± 0.4 | 94.2 ± 0.3 | 70.6 ± 1.0 | 67.9 ± 1.0 | 69.1 ± 1.0 | 5.0 ± 0.9 | 22.1 |
| MLCM (20 ML-based CQs) | 95.5 ± 0.3 | 92.4 ± 0.5 | 94.5 ± 0.4 | 71.7 ± 1.1 | 69.1 ± 1.0 | 70.3 ± 0.9 | 5.5 ± 0.9 | 25.6 |
| MLCM-LR ( 1 ML-based CQ) | 90.2 ± 0.3 | 87.6 ± 0.4 | 89.5 ± 0.3 | 61.8 ± 1.1 | 60.5 ± 1.1 | 61.5 ± 1.0 | 4.9 ± 0.8 | 0.7 |
| MLCM-LR ( 5 ML-based CQs) | 91.5 ± 0.3 | 89.1 ± 0.4 | 90.9 ± 0.3 | 63.9 ± 1.1 | 61.7 ± 1.2 | 63.1 ± 1.2 | 4.8 ± 0.8 | 0.8 |
| MLCM-LR (10 ML-based CQs) | 92.5 ± 0.2 | 90.2 ± 0.5 | 91.9 ± 0.3 | 65.4 ± 1.1 | 63.6 ± 1.1 | 64.4 ± 1.1 | 4.7 ± 0.8 | 0.8 |
| MLCM-LR (10 ML-based CQs) | 93.7 ± 0.3 | 91.0 ± 0.4 | 93.0 ± 0.3 | 68.1 ± 0.9 | 66.3 ± 0.9 | 66.9 ± 0.9 | 4.7 ± 0.8 | 0.8 |
| MLCM-LR (20 ML-based CQs) | 94.4 ± 0.3 | 92.1 ± 0.4 | 94.0 ± 0.3 | 69.6 ± 0.9 | 67.4 ± 1.1 | 69.1 ± 1.0 | 4.9 ± 0.9 | 0.8 |

Table 38. Comparison of CM and MLCM in Stages 1–3 for *additive* preferences, a supply ratio of 1.5, 9 popular courses, and default parameterization for reporting mistakes. We normalize all results by the average utility of CM* after Stage 1. Shown are averages in % over 100 runs and 95% CIs. Additionally, we present the oversubscription (in number of course seats) after Stage 1 (Overs.) and total runtime (in hours) per run.

## P.2 Reporting Mistakes Ablation Study for Additive Preferences

In this subsection, we present in Figures 9 and 10 the reporting mistakes ablation study when students' true utility functions are restricted to be *additive/linear*. Please see Section 7.3 for details on the experiment setup.

Similar to the results in Section 7.3, as $\gamma$ increases, the performance of both CM and MLCM monotonically decreases. MLCM significantly outperforms CM for all $\gamma \in [0.5, 1.5]$. As $\gamma$ increases, the relative performance gap of the two mechanisms gets significantly larger. Those results could be further improved by retuning MLCM's hyperparameters for each value of $\gamma$.

## P.3 Should Individual Students Opt Into MLCM?

In this section, we present the experimental results measuring a students expected gain from opting into MLCM when students' true utility functions are restricted to be *additive/linear*. Recall, that we use 20 instances and 100 students per instance. We report averages over those 2000 students. Please see Section 7.4 for more details on the experiment setup.

*No Other Student to Opt into the ML Feature.* Table 39 shows the results when all other students opt out of MLCM's ML feature. We observe that the *expected relative gain* from opting in is at least 8.4% (across both supply ratios). Furthermore, the student prefers the "MLCM schedule" in at least 71.5% of the cases, while she prefers the "CM schedule" in at most 6.3% of the cases. As the number

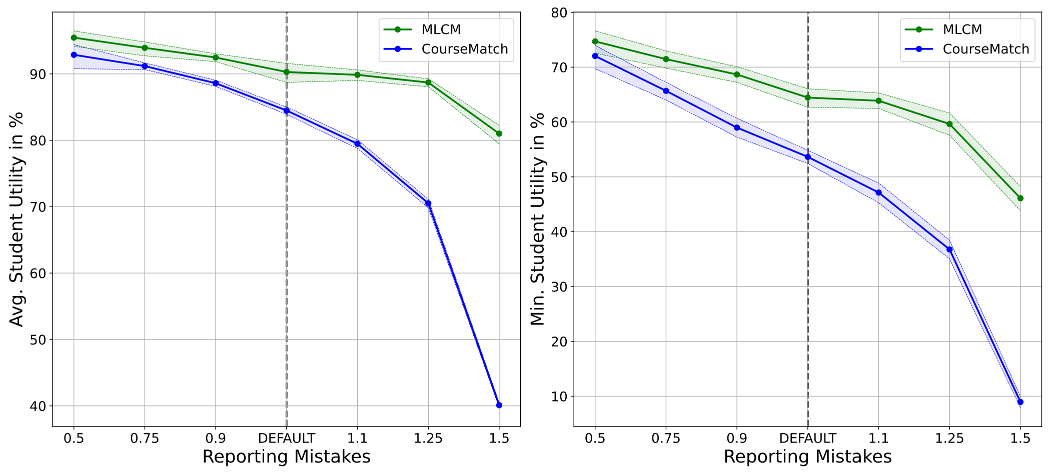

Fig. 9. Reporting mistakes ablation experiment for *additive* preferences for a supply ratio of 1.25 and 9 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

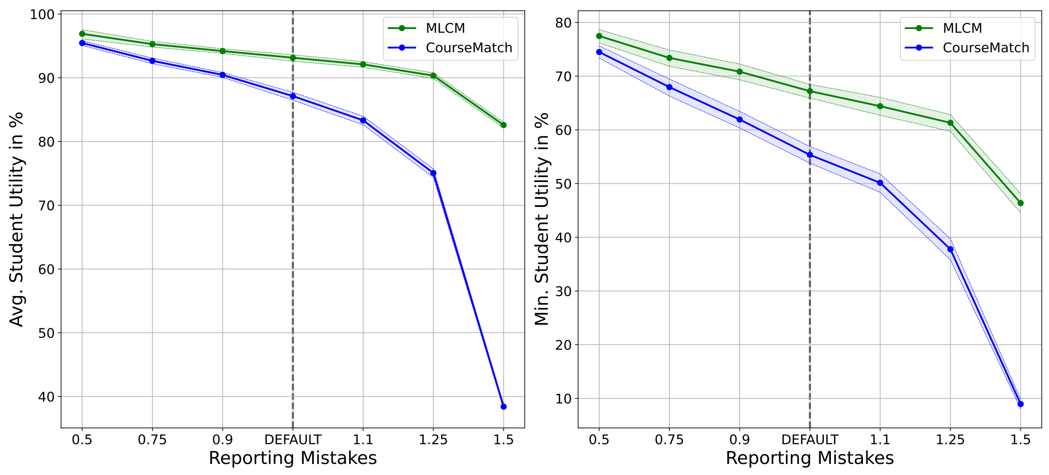

Fig. 10. Reporting mistakes ablation experiment for *additive* preferences for a supply ratio of 1.5 and 9 popular courses. Shown are average results in % for the final allocation over 50 runs including 95% CI.

of CQs the student answers increases, the benefit from opting into MLCM's ML feature becomes even larger.

| Setting | | | Preferred Mechanism | | | Gain from Opting Into MLCM | | |
|---|---|---|---|---|---|---|---|---|
| SR | #PoP | #CQs | MLCM | CM | Indiff. | Expected | if pref MLCM | if pref CM |
| 1.25 | 9 | 10 | 77.60% | 5.75% | 16.65% | 10.98% | 14.54% | -5.46% |
| 1.25 | 9 | 15 | 81.35% | 6.35% | 12.30% | 12.32% | 15.52% | -4.93% |
| 1.25 | 9 | 20 | 85.30% | 6.10% | 8.60% | 13.58% | 16.21% | -4.08% |
| 1.50 | 9 | 10 | 71.55% | 6.30% | 22.15% | 8.42% | 12.30% | -6.02% |
| 1.50 | 9 | 15 | 77.85% | 5.45% | 16.70% | 9.81% | 12.95% | -4.39% |
| 1.50 | 9 | 20 | 81.70% | 6.50% | 11.80% | 10.56% | 13.30% | -4.89% |

Table 39. Expected gain of opting into MLCM's ML feature when no other student opts in for *additive* preferences. Shown are average results across 2000 students per setting (SR,#PoP,#CQs). CIs for all metrics are $\approx 0$.

*All Other Students to Opt into the ML Feature.* Table 40 shows the results when all other students opt into MLCM's ML feature. We observe that the *expected relative gain* from opting in is at least 7.6% (across both supply ratios). Furthermore, the student prefers the "MLCM schedule" in at least 68.5% of the cases, while she prefers the "CM schedule" in at most 8.4% of the cases. As the number of CQs the student answers increases, the benefit from opting into MLCM's ML feature becomes even larger.

| Setting | | | Preferred Mechanism | | | Gain from Opting Into MLCM | | |
|---|---|---|---|---|---|---|---|---|
| SR | #PoP | #CQs | MLCM | CM | Indiff. | Expected | if pref MLCM | if pref CM |
| 1.25 | 9 | 10 | 74.55% | 8.4% | 17.05% | 10.05% | 14.12% | -5.80% |
| 1.25 | 9 | 15 | 77.6% | 8.4% | 14.00% | 10.94% | 14.73% | -5.91% |
| 1.25 | 9 | 20 | 81.7% | 8.15% | 10.15% | 11.79% | 14.91% | -4.67% |
| 1.50 | 9 | 10 | 68.5% | 7.95% | 23.55% | 7.59% | 11.70% | -5.75% |
| 1.50 | 9 | 15 | 73.95% | 8.65% | 17.40% | 8.54% | 12.20% | -5.95% |
| 1.50 | 9 | 20 | 77.8% | 8.65% | 13.55% | 9.06% | 12.19% | -5.56% |

Table 40. Expected gain of opting into MLCM's ML feature when all other students also opt in for *additive* preferences. Shown are average results across 2000 students per setting (SR,#PoP,#CQs). CIs for all metrics are $\approx 0$.