A Detailed Literature Review

In this section, we first summarize the related works in Predict+Optimize, and then summarize other works related to learning unknowns in optimization problems, but outside of Predict+Optimize.

A.1 Predict+Optimize

As mentioned in Section 1, prior works all focus on the case where all unknown parameters are revealed simultaneously. Most of them have focused on the regime where the unknown parameters only appear in the objective and use the regret proposed by Elmachtoub et al. [6] as the loss function. Since the regret loss is usually not (sub-) differentiable, and gradient-based methods do not apply, they proposed ways to overcome the non-differentiability of the regret. Elmachtoub et al. [6] propose a differentiable surrogate function for the regret function, while Wilder et al. [26] relax the integral objective in constrained optimization and solve a regularized quadratic programming problem. Mandy and Guns [15] focus on mixed integer linear programs and propose an interior point based approach. In addition to computing the (approximate) gradients of the regret function or approximations of it, another way to deal with the non-differentiability of the regret is to exploit the structure of optimization problems to train models without computing gradients. Demirović et al. [5] investigate problems amenable to tabular dynamic programming and propose a coordinate descent method to learn a linear prediction function. Hu et al. [13] extend their framework, to enable problems solvable with a recursive or iterative algorithm to be tackled in Predict+Optimize. Guler et al. [8] proposes a divide and conquer algorithm, extending the work of Demirović et al. [5] in a different manner so that the algorithm can deal with problems with the linear objective function.

As for Predict+Optimize with unknown parameters also in constraints, Hu et al. [11] first propose the post-hoc regret loss function and a framework for packing and covering LPs with unknown parameters in both objectives and constraints. They [12] further advocate a conceptually simpler framework, which enable solving MILPs with unknown constraints. Besides, there are also works applying Predict+Optimize to a wide range of real-world problems, including maritime transportation [25], last-mile delivery [4], and trading in renewable energy [24].

A.2 Decision-Focused Learning

Now we summarize other works related to learning unknowns in optimization problems, particularly those outside of Predict+Optimize. These works can be placed into two categories.

One category considers learning unknown parameters but with very different goals and measures of loss. For example, CombOptNet [20] and Nandwani et al. [18] focus on learning parameters so as to make the predicted optimal solution (Stage 0 estimated solution in the proposed framework) as close to the true optimal solution x^* as possible in the solution space/metric. However, these works also assume that all unknown parameters are revealed simultaneously, and thus cannot be applied to applications where unknown parameters are revealed progressively over several stages. Furthermore, experiments in Two-Stage Predict+Optimize [12] show that these other methods yield worse predictive performance when evaluated on the post-hoc regret.

Another category gives ways to differentiate through LPs or LPs with regularizations, as a technical component in a gradient-based training algorithm [2, 26, 1]. While our proposed algorithms in Section 4.1 and Appendix C use the methods of Hu et al. [11, 12] and Mandi and Guns [15] to perform gradient computations, we could in principle use any of the aforementioned other works. However, we point out that our main contribution is not the gradient computation method but the two training algorithms of the set of NNs. Nonetheless, experiments in Two-Stage Predict+Optimize framework [12] demonstrate that the gradient calculation method they used (which we also use) performs at least as well in post-hoc regret performance as other gradient methods, while being (significantly) faster. This is the reason we follow Hu et al.'s method and implementation.

A.3 (Multi-Stage) Stochastic Programming

As mentioned in Section 1, while stochastic programming and Predict+Optimize are related frameworks, the technical challenges are very different. The most important difference is that Predict+Optimize is a supervised learning problem, whereas stochastic programming is unsupervised learning. In Predict+Optimize frameworks, the true parameters (which need prediction) are always associated with relevant features that help prediction. On the other hand, stochastic programming frameworks have no such features, and typically assume that the entire distribution over the unknown parameters is given to the algorithm — in practice, the distribution needs to be estimated from historical data over the unknown parameters, which is an unsupervised density estimation problem.

Due to the different starting assumptions, Predict+Optimize and stochastic programming formulate optimization problems rather differently. In stochastic programming, since the assumption is that the full parameter distribution is given, the optimization problem (or problems, across stages) would explicitly include the expectation operator in the objective — the goal is to solve for optimization decisions so that the expected objective, with expectation taken over the parameter distribution, is maximized/minimized. Predict+Optimize frameworks approach this rather differently: while the goal is still to optimize the expected objective, the optimization problems themselves are phrased such that they take predicted parameters, and the problem asks for the optimal decisions assuming the predicted parameters. It then becomes the goal of the *learning algorithm* to learn to make predictions from features, such that the expected objective is optimized overall. This is achieved via empirical risk minimization over training data, which we assume are samples from the underlying (feature,parameter) joint distribution.

We also note the dimensionality of the objects being learnt in the different frameworks. In stochastic programming, the entire distribution over the unknown parameters needs to be learnt. On the other hand, in Predict+Optimize, we learn a mapping from features to predicted parameters, which, under smoothness assumptions or bounded model complexity assumptions (e.g. by restricting to using a fixed neural network architecture), can effectively be regarded as a (much) lower dimensional object than learning an entire distribution over unknown parameters.

B A Detailed Example for Multi-Stage Predict+Optimize Framework

In this section, we use the hospital scenario, i.e., the nurse rostering problem (NRP), mentioned in Section 1 as a running example for the Multi-Stage Predict+Optimize framework described in Section 3.1.

Here we describe the NRP in detail. A hospital needs to make nurses schedule for the whole week (7 days) two weeks beforehand so that the nurses can be well prepared for the work and also plan for their leisure activities. The goal of the hospital is to minimize the total costs for hiring nurses and meet the patients' demands.

There are full-time nurses in the hospital. If there are too many patients and the hospital's nurses are understaffed, the hospital can temporarily hire some extra nurses at a higher salary. Since the number of patients that will come in each shift on each day is unknown two weeks beforehand, the hospital needs to predict the number of patients to make a schedule for the full-time nurses and plan to hire extra nurses. The hospital will learn the predictor based on historical hospital records, considering features such as time of year, day of the week and temperature.

To provide better service to patients, the hospital has an appointment system that requires patients to schedule an appointment in advance to receive medical care. Reservations for the next day close the night before. At this point, the hospital knows the precise number of patients for each shift of the current day. Therefore, at the night of day $(t-1)$, i.e., Stage t $(1 \leq t \leq 7)$, the true numbers of patients for each shift of the current day are revealed.

Now we show the running example for the Multi-Stage Predict+Optimize framework. Examples 1 and 2 are examples for Stage 0 and Stage t (for $1 \leq t \leq T$) respectively.

Example 1. *Suppose there are n full-time nurses, 7 days, and 3 working shifts per day. Full-time nurses are entitled to take a rest: day-off shift. The decision variables are: 1) a Boolean vector $x \in \{0,1\}^{n\times 7\times 3}$, where $x_{i,j,k}$ represents that whether nurse i is assigned to shift k in day j, and 2) an integer vector $\sigma \in \mathbb{N}^{7\times 3}$, where $\sigma_{j,k}$ represents the number of extra nurses hired in shift k day j. Let $d_{j,k}$ denote the number of patients in shift k day j, m_i denote the number of patients that the nurse i can serve per shift, c_i denote the payment of the nurse i per shift, e_s denote the number of patients that each extra nurse can serve per shift, and e_c denote the payment of each extra nurse per shift. The unknown parameters are $\boldsymbol{d} \in \mathbb{N}^{7\times 3}$.*

Consider the time that the schedules need to be made as Stage 0. The hospital learns the predictor and uses the estimated number of patients $\hat{\boldsymbol{d}}^{(0)}$ to optimize for that week's schedule. The Stage 0 OP, the NRP using the estimations, can be formulated as:

$$\hat{x}^{(0)}, \hat{\sigma}^{(0)} = \arg\min_{x,\sigma} \sum_{i=1}^{n} c_i \sum_{j=1}^{7} \sum_{k=1}^{3} x_{i,j,k} + e_c \sum_{j=1}^{7} \sum_{k=1}^{3} \sigma_{j,k} \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{n} m_i x_{i,j,k} + e_s \sigma_{j,k} \geq \hat{d}^{(0)}_{j,k}, \quad \forall j \in \{1, \ldots, 7\}, k \in \{1, 2, 3\} \tag{2}$$

$$\sum_{k=1}^{4} x_{i,j,k} = 1, \quad \forall i \in \{1, \ldots, n\}, j \in \{1, \ldots, 7\} \tag{3}$$

$$x_{i,j,3} + x_{i,j+1,1} \leq 1, \quad \forall i \in \{1, \ldots, n\}, j \in \{1, \ldots, 6\} \tag{4}$$

$$1 \leq \sum_{j=1}^{7} x_{i,j,4} \leq 2, \quad \forall i \in \{1, \ldots, n\} \tag{5}$$

$$x \in \{0, 1\}, \quad \sigma \geq 0 \tag{6}$$

where Equation (1) represents the objective, which is to minimize the total costs for hiring full-time nurses and extra nurses; Equation (2) ensures that the schedule can satisfy the patient demand under each shift; Equation (3) ensures that each full-time nurse will be scheduled for exactly one shift each day; Equation (4) ensures that no full-time nurse will be scheduled to work a night shift followed immediately by a morning shift; and Equation (5) ensures that each full-time nurse gets one or two day-off shifts per week.

After Stage 0, the schedules for day 1 are hard commitments and cannot be changed, i.e., $\hat{x}^{(0)}_0 = \{x_{i,1,k} \mid \forall i \in \{1, ..., n\}, k \in \{1, 2, 3\}\}$, whereas the rest of the decisions are soft commitments.

Example 2. ***(Continued)*** *At the night of day $t-1$, i.e., Stage t (for $1 \leq t \leq 7$), the reservations for the next day close, and the true numbers of patients for the three shifts of the next day $\theta_t = (d_{t,1}, d_{t,2}, d_{t,3}) \in \mathbb{N}^3$ are revealed. By Stage t, all the true numbers of patients for the prior $t-1$ days are also revealed. The number of patients for the later $7-t$ days are still uncovered and are estimated as $\hat{\boldsymbol{\theta}}^{(t)} = (\hat{\theta}^{(t)}_{t+1}, \ldots, \hat{\theta}^{(t)}_T)$, where $\hat{\theta}^{(t)}_i = (\hat{d}^{(t)}_{i,1}, \hat{d}^{(t)}_{i,2}, \hat{d}^{(t)}_{i,3}) \in \mathbb{N}^3$ represents the numbers of patients on day i estimated on day t.*

Hard commitments contain two parts: 1) the schedule for the day t and the prior $t-1$ days, and 2) the number of extra nurses hired in the prior $t-1$ days, i.e., here $x[1:t-1]$ represents $\{x_{i,j,k} \mid \forall i \in \{1, ..., n\}, j \in \{1, ..., t\}, k \in \{1, 2, 3\}\} \cup \{\sigma_{j,k} \mid \forall j \in \{1, ..., t-1\}, k \in \{1, 2, 3\}\}$. The hospital may update the predictions and reschedule for the later $(7-t)$ days. But such rescheduled leads to extra costs for hiring full-time nurses, which are recorded by the penalty function $Pen(\hat{x}^{(t-1)} \rightarrow x, \boldsymbol{\theta}[1:t])$. The more temporarily the shift is rescheduled, the larger the increase in the costs. For simplicity, we assume that the extra cost is linear in the original cost for hiring each full-time nurse. In this scenario, the penalty function can be formulated as $Extra(\hat{x}^{(t-1)} \rightarrow x)$:

$$Extra(\hat{x}^{(t-1)} \rightarrow x) = \sum_{i=1}^{n} \sum_{j=1}^{7} \sum_{k=1}^{3} Extra(\hat{x}^{(t-1)} \rightarrow x)_{i,j,k}$$

where the (i, j, k)-th item in the penalty function is:

$$Extra(\hat{x}^{(t-1)} \rightarrow x)_{i,j,k} = \begin{cases} (T - j + t)\rho_i c_i, & x_{i,j,k} > \hat{x}^{(t-1)}_{i,j,k} \\ 0, & x_{i,j,k} \leq \hat{x}^{(t-1)}_{i,j,k} \end{cases}$$

As mentioned in Section 3.1, the Stage t optimization problem modifies the original Para-OP by: 1) introducing the penalty term capturing the cost of changing the Stage $t-1$ solution $\hat{x}^{(t-1)}$ to the new Stage t solution x to the objective:

$$\hat{x}^{(t)}, \hat{\sigma}^{(t)} = \arg\min_{x,\sigma} \sum_{i=1}^{n} c_i \sum_{j=1}^{7} \sum_{k=1}^{3} x_{i,j,k} + e_c \sum_{j=1}^{7} \sum_{k=1}^{3} \sigma_{j,k} + \sum_{i=1}^{n} \sum_{j=1}^{7} \sum_{k=1}^{3} Extra(\hat{x}^{(t-1)} \rightarrow x)_{i,j,k}$$

and 2) introducing the constraint that hard commitments from prior stages cannot be modified in the current Stage t:

$$x_{i,j,k} = \hat{x}^{(t-1)}_{i,j,k}, \quad \forall i \in \{1, ..., n\}, j \in \{1, ..., t\}, k \in \{1, 2, 3\}\}$$
$$\sigma_{j,k} = \hat{\sigma}^{(t-1)}_{j,k}, \quad \forall j \in \{1, ..., t-1\}, k \in \{1, 2, 3\}$$

Besides, for the first constraint in Equation (2), the Stage 0 predicted parameters $\hat{\boldsymbol{d}}^0$ are replaced by $(d_{1,1}, \ldots, d_{t,3}, \hat{d}^{(t)}_{t+1,1}, \ldots, \hat{d}^{(t)}_{7,3})$:

$$\sum_{i=1}^{n} m_i x_{i,j,k} + e_s \sigma_{j,k} \geq d_{j,k}, \quad \forall j \in \{1, \ldots, t\}, k \in \{1, 2, 3\}$$
$$\sum_{i=1}^{n} m_i x_{i,j,k} + e_s \sigma_{j,k} \geq \hat{d}^t_{j,k}, \quad \forall j \in \{t+1, \ldots, 7\}, k \in \{1, 2, 3\}$$

The four constraints from Equation (3) to Equation (6) keep the same in the Stage t (for $1 \leq t \leq 7$) optimization.

C Gradient Computations in Sequential Coordinate Descent

The post-hoc regret used to train NN_t can be written as:

$$PReg(\hat{\boldsymbol{\theta}}^{(t)}, \boldsymbol{\theta}[t+1:T]) = obj(\hat{x}^{(T)}, \boldsymbol{\theta}) - obj(x^*(\boldsymbol{\theta}), \boldsymbol{\theta}) + \sum_{i=t}^{T} Pen_i(\hat{x}^{(i-1)} \to \hat{x}^{(i)}, \boldsymbol{\theta}[1:i]) \quad (7)$$

Noting the second term is independent of $\hat{\boldsymbol{\theta}}^{(t)}$ and hence NN_t, the gradient with respect to an edge weight w_e in NN_t is

$$\frac{\mathrm{d}\, PReg}{\mathrm{d}w_e} = \frac{\mathrm{d}\, obj(\hat{x}^{(T)}, \boldsymbol{\theta})}{\mathrm{d}w_e} + \sum_{i=t}^{T} \frac{\mathrm{d}\, Pen_i(\hat{x}^{(i-1)} \to \hat{x}^{(i)}, \boldsymbol{\theta}[1:i])}{\mathrm{d}w_e}$$

By the law of total derivative, we can expand this to

$$\frac{\mathrm{d}\, PReg}{\mathrm{d}w_e} = \frac{\mathrm{d}\, obj(\hat{x}^{(T)}, \boldsymbol{\theta})}{\mathrm{d}\hat{x}^{(T)}} \frac{\mathrm{d}\hat{x}^{(T)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}} \frac{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}}{\mathrm{d}w_e} + \sum_{i=t}^{T} \left(\frac{\partial\, Pen_i}{\partial \hat{x}^{(i-1)}}\bigg|_{\hat{x}^{(i)}} \frac{\mathrm{d}\hat{x}^{(i-1)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}} + \frac{\partial\, Pen_i}{\partial \hat{x}^{(i)}}\bigg|_{\hat{x}^{(i-1)}} \frac{\mathrm{d}\hat{x}^{(i)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}} \right) \frac{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}}{\mathrm{d}w_e}$$

Similar to the gradient computation in Section 4.1, the term $\frac{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}}{\mathrm{d}w_e}$ is calculated via standard backpropagation, while the terms $\frac{\mathrm{d}\, obj(\hat{x}^{(T)}, \boldsymbol{\theta})}{\mathrm{d}\hat{x}^{(T)}}$, $\frac{\partial\, Pen_i}{\partial \hat{x}^{(i-1)}}\big|_{\hat{x}^{(i)}}$ and $\frac{\partial\, Pen_i}{\partial \hat{x}^{(i)}}\big|_{\hat{x}^{(i-1)}}$ are calculable when the objective and penalty functions are smooth. The only non-trivial calculation is for $\frac{\mathrm{d}\hat{x}^{(i)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}}$ for all $i \in [t:T]$.

Recall that $\hat{x}^{(i)}$ is computed from the Stage i optimization problem, as a deterministic function of $\hat{x}^{(i-1)}$ and $\hat{\boldsymbol{\theta}}^{(t)}$, while $\hat{x}^{(i-1)}$ itself also depends on $\hat{\boldsymbol{\theta}}^{(t)}$. We thus further decompose $\frac{\mathrm{d}\hat{x}^{(i)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}}$ into the following recursive computation

$$\frac{\mathrm{d}\hat{x}^{(i)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}} = \frac{\partial \hat{x}^{(i)}}{\partial \hat{x}^{(i-1)}}\bigg|_{\hat{\boldsymbol{\theta}}^{(t)}} \frac{\mathrm{d}\hat{x}^{(i-1)}}{\mathrm{d}\hat{\boldsymbol{\theta}}^{(t)}} + \frac{\partial \hat{x}^{(i)}}{\partial \hat{\boldsymbol{\theta}}^{(t)}}\bigg|_{\hat{x}^{(i-1)}}$$

Calculating $\frac{\partial \hat{x}^{(i)}}{\partial \hat{x}^{(i-1)}}\big|_{\hat{\boldsymbol{\theta}}^{(t)}}$ and $\frac{\partial \hat{x}^{(i)}}{\partial \hat{\boldsymbol{\theta}}^{(t)}}\big|_{\hat{x}^{(i-1)}}$ requires differentiating through a MILP. So instead of directly using MILP formulations for the Stage t optimization problems, we use the convex relaxation by Hu et al. [12], which in turn adapts the approach of Mandi and Guns [15].

D Details for Case Studies

In this section, we give a detailed description for the other two benchmarks used in Section 5.

D.1 Production and Sales Problem

We first demonstrate, using the example of the production and sales problem, how our framework can tackle LPs. An oil company intends to develop a production and sales plan for the upcoming T quarters/months. The goal is to maximize the profits, i.e., the sales revenues minus the production costs, under the constraints that the amount of oil product sold each quarter/month cannot exceed the customer demands. The production cost and sales price for each quarter/month are known, but the customer demand is revealed only at the beginning of each quarter/month after the company receives the orders. The company will estimate the customer demands based on historical sales records, considering features such as oil type, oil consumption of different areas, and so on.

The decision variables are: 1) a real vector $x \in \mathbb{R}^T$, where x_i represents the amount of product produced in month i, and 2) a real vector $y \in \mathbb{R}^T$, where y_i represents the amount of product sold in month i. Let p_i denote the unit profit of selling product in month i, c_i denote the unit cost of producing product in month i, d_i denote the customer demands in month i. The unknown parameters are $\boldsymbol{d} \in \mathbb{R}^T$.

At Stage 0, i.e., the time that the production and sales plan needs to be made, there is no order yet and the customer demands are unknown. The company learns the predictor and uses the predicted demands $\hat{\boldsymbol{d}}^{(0)}$ to make the plan. The Stage 0 OP can be formulated as:

$$\hat{x}^{(0)}, \hat{y}^{(0)} = \arg\max_{x,y} \sum_{i=1}^{T} p_i y_i - \sum_{i=1}^{T} c_i x_i \tag{8}$$

$$\text{s.t.} \quad y_i \leq \hat{d}_i^{(0)}, \quad \forall i \in \{1, \dots, T\} \tag{9}$$

$$y_i \leq \sum_{j=1}^{i-1} x_j - \sum_{j=1}^{i-1} y_j, \quad \forall i \in \{1, \dots, T\} \tag{10}$$

$$x \geq 0, \quad y \geq 0 \tag{11}$$

where Equation (8) represents the objective, for maximizing the profits, i.e., the sales revenues minus the production costs; Equation (9) ensures that the amount of oil product sold each quarter/month will not exceed the customer demands; Equation (10) ensures that the amount of oil product sold each quarter/month will not exceed the inventory at that quarter/month.

At the beginning of each quarter/month, the company receives orders, and the demand for that quarter/month is revealed. We assume that the beginning of each quarter/month is one stage. Then, by Stage t ($1 \leq t \leq T$), all the true demands for the prior $(t-1)$ quarters/months as well as the t quarter/month are revealed. The demands for the later $(T-t)$ quarters/months are still uncovered and are estimated as $\hat{\boldsymbol{\theta}}^{(t)} = (\hat{\theta}_{t+1}^{(t)}, \dots, \hat{\theta}_T^{(t)})$, where $\hat{\theta}_i^{(t)} = \hat{d}_i^{(t)}$ represents the demand of quarter/month i estimated on quarter/month t. The production and sales for the quarter/month t and the prior $(t-1)$ quarters/months have already happened and cannot be changed, which are committed variables. There is no penalty function in this scenario. Therefore, the Stage t OP can be formulated as:

$$\begin{aligned}
\hat{x}^{(t)}, \hat{y}^{(t)} = \arg\max_{x,y} & \sum_{i=1}^{T} p_i y_i - \sum_{i=1}^{T} c_i x_i \\
\text{s.t.} \quad & y_i \leq d_i, \quad \forall i \in \{1, \dots, t\} \\
& y_i \leq \hat{d}_i^{(t)}, \quad \forall i \in \{t+1, \dots, T\} \\
& y_i \leq \sum_{j=1}^{i-1} x_j - \sum_{j=1}^{i-1} y_j, \quad \forall i \in \{1, \dots, T\} \\
& x_i = \hat{x}_i^{(t-1)}, y_i = \hat{y}_i^{(t-1)}, \quad \forall i \in \{1, \dots, t-1\} \\
& x \geq 0, \quad y \geq 0
\end{aligned}$$

D.2 Investment Problem

In the second experiment, we showcase our framework on an MILP. The unknown parameters appear in both the objective and constraints. A person wants to make an investment plan for buying several types of financial products next year to maximize the investment profit, under limited capital. The investment profit contains 3 parts: 1) the interest gained from the products owned, 2) the market prices of the products owned at the end of the year, and 3) profits from selling products minus costs for buying products minus transaction fees. The capital for the whole year is given. However, the market price of each product in each quarter/month is revealed only at the beginning of the quarter/month, and the interest of owning each product in each quarter/month is revealed only at the end of the quarter/month. The person will estimate the market prices and interests based on past experiences, considering features such as the product type, the financial condition and operational capabilities of the company to which the product belongs, and so on.

Suppose there are T quarters/months, and N investment products. The decision variables are: 1) a natural vector $x \in \mathrm{N}^{T\times N}$, where $x_{i,j}$ represents the number of product j on hand at the end of quarter/month i, 2) an integer vector $y \in \mathrm{Z}^{(T-1)\times N}$, where $y_{i,j}$ represents the number of product j bought or sold in quarter/month i, and 3) a natural vector $z \in \mathrm{N}^{(T-1)\times N}$, where $z_{i,j}$ represents whether the transaction fee is paid for product j in month i. Let $p_{i,j}$ denote the interest of product j in month i, $w_{i,j}$ denote the market price of product j in month i, C denote the capital for the whole year.

We assume that the end of quarter/month t, i.e., the beginning of quarter/month $(t+1)$, is Stage t. At Stage 0, i.e., the beginning of quarter/month 1, the person can buy some products without paying a transaction fee. The market price of each product at this time is known, i.e., $w_1 = (w_{1,1}, \ldots, w_{1,N})$ are given. The unknown parameters in this OP are $\boldsymbol{p} \in \mathbb{R}^{T\times N}$ and $\boldsymbol{w} = (w_2, \ldots, w_T) \in \mathbb{R}^{(T-1)\times N}$. At the beginning of each subsequent quarter/month, the person can buy more products or sell the products owned but needs to pay a transaction fee. For simplicity, we assume that the transaction fee for buying/selling product i in quarter/month j is linear in the market price of product i in quarter/month j. Here, the linearity factor is independent of the request. That is, if the person buys/sells k number of product i in quarter/month j, the person has to spend $k\sigma w_{ij}$, where $\sigma \geq 0$ is a non-negative tunable scalar parameter, and we call it transaction factor.

At Stage 0, i.e., the beginning of quarter/month 1, the person uses the predicted interests $\hat{\boldsymbol{p}}^{(0)}$ and market prices $\hat{\boldsymbol{w}}^{(0)}$ to make the plan. The Stage 0 OP can be formulated as:

$$\hat{x}^{(0)}, \hat{y}^{(0)}, \hat{z}^{(0)} = \arg\max_{x,y,z} obj(\hat{\boldsymbol{p}}^{(0)}, w_1, \hat{\boldsymbol{w}}^{(0)}, x, y, z) \tag{12}$$

$$\text{s.t.} \quad \sum_{j=1}^{N} w_{1,j} x_{1,j} \leq C, \tag{13}$$

$$\begin{array}{l} \sum_{j=1}^{N} w_{1,j} x_{1,j} \\ + \sum_{i=2}^{t} \sum_{j=1}^{N} \sigma \hat{w}_{i,j}^{(0)} z_{i,j} \\ + \sum_{i=2}^{t} \sum_{j=1}^{N} \hat{w}_{i,j}^{(0)} y_{i,j} \end{array} \leq C, \quad \forall t \in \{2, \ldots, T\} \tag{14}$$

$$x_{i,j} = y_{i,j} + x_{(i-1),j}, \quad \forall i \in \{2, \ldots, T\}, j \in \{1, \ldots, N\} \tag{15}$$

$$z_{i,j} \geq y_{i,j}, \quad \forall i \in \{2, \ldots, T\}, j \in \{1, \ldots, N\} \tag{16}$$

$$z_{i,j} \geq -y_{i,j}, \quad \forall i \in \{2, \ldots, T\}, j \in \{1, \ldots, N\} \tag{17}$$

where

$$\begin{aligned} & obj(\hat{\boldsymbol{p}}^0, w_1, \hat{\boldsymbol{w}}^0, x, y, z) \\ &= \sum_{i=1}^{T} \hat{p}_{i,j}^{(0)} x_{i,j} + \sum_{j=1}^{N} \hat{w}_{T,j}^{(0)} x_{T,j} - (\sum_{j=1}^{N} w_{1,j} x_{1,j} + \sum_{i=2}^{T} \sum_{j=1}^{N} \sigma \hat{w}_{i,j}^{(0)} z_{i,j} + \sum_{i=2}^{T} \sum_{j=1}^{N} \hat{w}_{i,j}^{(0)} y_{i,j}) \end{aligned}$$

represents the objective, which is to maximize the investment profit; Equations (13) and (14) ensure that the money spent on buying products and transaction fees will not exceed the capital available; Equations (15), (16), and (17) formulate the relationships among three decision variables x, y, and z.

At Stage t, i.e., the end of quarter/month t, the interest of owning each product in quarter/month t as well as the market price of each product revealed. Then, by Stage t $(1 \leq t \leq T)$, all the true market prices for the prior t quarters/months, as well as the $(t+1)$ quarter/month, are revealed. Besides, all the true interests for the prior t quarters/months are also revealed. But the market prices for the later $(T-t-1)$ quarters/months and the interests for the later $(T-t)$ are still uncovered and are estimated as $\hat{\boldsymbol{w}}^{(t)} = (\hat{w}_{t+2}^{(t)}, \dots \hat{w}_T^{(t)})$ and $\hat{\boldsymbol{p}}^{(t)} = (\hat{p}_{t+1}^{(t)}, \dots \hat{p}_T^{(t)})$, where $\hat{w}_i^{(t)}$ and $\hat{p}_i^{(t)}$ represents the market price and the interest of quarter/month i estimated on quarter/month t. The investment decisions x, y, z for the prior t quarters/months have already happened and cannot be changed, which are committed variables. There is no penalty function in this scenario.

$$
\begin{aligned}
\hat{x}^{(t)}, \hat{y}^{(t)}, \hat{z}^{(t)} = \arg\max_{x,y,z} obj(\boldsymbol{p}[1:t] \oplus \hat{\boldsymbol{p}}^{(t)}, w_1, \boldsymbol{w}[2:t+1] \oplus \hat{\boldsymbol{w}}^{(t)}, x, y, z) \\
\text{s.t.} \ & \sum_{j=1}^{N} w_{1,j} x_{1,j} \leq C, \\
& \begin{aligned} & \textstyle\sum_{j=1}^N w_{1,j}x_{1,j} \\ & + \textstyle\sum_{i=2}^k \sum_{j=1}^N \sigma w_{i,j} z_{i,j} \\ & + \textstyle\sum_{i=2}^k \sum_{j=1}^N w_{i,j} y_{i,j} \end{aligned} \quad \leq C, \quad \forall k \in \{2, \dots, t\} \\
& \begin{aligned} & \textstyle\sum_{j=1}^N w_{1,j}x_{1,j} \\ & + \textstyle\sum_{i=2}^{t+1} \sum_{j=1}^N \alpha w_{i,j} z_{i,j} + \sum_{i=t+2}^{k} \sum_{j=1}^N \alpha \hat{w}_{i,j}^{(t)} z_{i,j} \\ & + \textstyle\sum_{i=2}^{t+1} \sum_{j=1}^N w_{i,j} y_{i,j} + \sum_{i=t+2}^{k} \sum_{j=1}^N \hat{w}_{i,j}^{(t)} y_{i,j} \end{aligned} \quad \leq C, \quad \forall k \in \{t+1, \dots, T\} \\
& x_{i,j} = y_{i,j} + x_{(i-1),j}, \quad \forall i \in \{2, \dots, T\}, j \in \{1, \dots, N\} \\
& z_{i,j} \geq y_{i,j}, \quad \forall i \in \{2, \dots, T\}, j \in \{1, \dots, N\} \\
& z_{i,j} \geq -y_{i,j}, \quad \forall i \in \{2, \dots, T\}, j \in \{1, \dots, N\} \\
& x_{i,j} = \hat{x}_{i,j}^{(t-1)}, \quad \forall i \in \{1, \dots, t\}, j \in \{1, \dots, N\} \\
& y_{i,j} = \hat{y}_{i,j}^{(t-1)}, z_{i,j} = \hat{z}_{i,j}^{(t-1)}, \quad \forall i \in \{2, \dots, t\}, j \in \{1, \dots, N\}
\end{aligned}
$$

where

$$
\begin{aligned}
& obj(\boldsymbol{p}[1:t] \oplus \hat{\boldsymbol{p}}^{(t)}, w_1, \boldsymbol{w}[2:t+1] \oplus \hat{\boldsymbol{w}}^{(t)}, x, y, z) \\
&= \sum_{i=1}^{t} \sum_{j=1}^{N} p_{i,j} x_{i,j} + \sum_{i=t+1}^{T} \sum_{j=1}^{N} \hat{p}_{i,j}^{(t)} x_{i,j} - \sum_{j=1}^{N} w_{1j} x_{1j} - (\sum_{i=2}^{t+1} \sum_{j=1}^{N} \alpha w_{i,j} z_{i,j} + \sum_{i=t+2}^{T} \sum_{j=1}^{N} \alpha \hat{w}_{i,j}^{(t)} z_{i,j}) \\
&\quad - (\sum_{i=2}^{t+1} \sum_{j=1}^{N} w_{i,j} y_{i,j} + \sum_{i=t+2}^{T} \sum_{j=1}^{N} \hat{w}_{i,j}^{(t)} y_{i,j}) + \sum_{j=1}^{N} \hat{w}_{Tj}^{(t)} x_{Tj}
\end{aligned}
$$

E Hyperparameters for the Experiments

The methods of k-NN, RF, NN, Baseline, SCD and PCD have hyperparameters, which we tune via cross-validation: for k-NN, we try $k \in \{1, 3, 5\}$; for RF, we try different numbers of trees in the forest $\{10, 50, 100\}$; for NN, Baseline, and PCD, we treat the optimizer, learning rate, and epochs as hyperparameters. For Baseline, SCD and PCD, we treat the optimizer, learning rate, the early-cut-off value of log barrier regularization term (μ), and epochs as hyperparameters.

Table 4 show the final hyperparameter choices for the three problems: 1) production and sales problem, 2) investment problem, and 3) nurse rostering problem.

Ridge, k-NN, CART and RF are implemented using *scikit-learn* [21]. The neural network is implemented using *PyTorch* [19]. To compute the two stages of optimization at *test time* for our method, and to compute the optimal solution of an (MI)LP under the true parameters, we use the *Gurobi* MILP solver [9].

F Detailed Experiment Results

F.1 Production and Sales Problem

Table 5 reports the mean post-hoc regrets and standard deviations across 30 simulations for all training methods on the production and sales problem. Compared among standard regression models, NN

Table 4: Hyperparameters of the experiments on the three problems.

Model	Hyperaprameters		
	Production and sales	Investment	Nurse rostering
Baseline	optimizer: optim.Adam; learning rate: 1×10^{-7}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-6}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-6}; $\mu = 10^{-8}$; epochs=20
SCD	optimizer: optim.Adam; learning rate: 1×10^{-7}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-6}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 5×10^{-7}; $\mu = 10^{-7}$; epochs=20
PCD	optimizer: optim.Adam; learning rate: 1×10^{-7}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-6}; $\mu = 10^{-8}$; epochs=20	optimizer: optim.Adam; learning rate: 5×10^{-7}; $\mu = 10^{-7}$; epochs=20
NN	optimizer: optim.Adam; learning rate: 1×10^{-5}; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-5}; epochs=20	optimizer: optim.Adam; learning rate: 1×10^{-3}; epochs=20
k-NN	k=5		
RF	n estimator=100		

Table 5: Mean post-hoc regrets and standard deviations of all methods for the production and sales problem.

Price group	Low-profit		High-profit	
Stage num	4	12	4	12
SCD	**293.78±99.21**	**488.72±127.62**	**505.24±89.55**	**887.38±250.55**
PCD	297.34±107.44	495.21±122.42	520.76±92.20	905.61±255.99
Baseline	305.26±100.88	515.80±137.67	526.77±104.99	935.03±263.47
NN	355.56±103.78	637.77±199.25	561.36±96.49	997.44±273.91
Ridge	390.88±114.89	648.60±214.69	612.49±109.62	1017.41±277.01
knn	368.20±111.34	663.96±208.51	591.47±97.87	1050.42±296.83
CART	485.73±152.05	873.85±279.68	763.88±136.37	1345.56±337.05
RF	375.18±114.23	644.63±204.50	567.35±94.16	1021.51±274.34
TOV	1615.75±675.77	7344.78±2290.04	5007.09±976.65	21066.00±4159.56

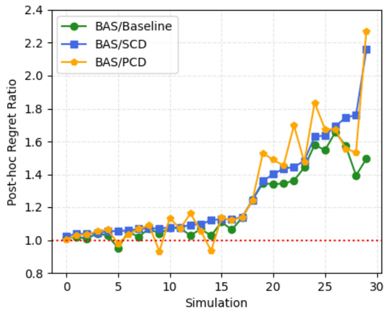
performs well and achieves the best performance in all cases listed in Table 5, while Ridge and RF also have decent performances.

Table 6 shows improvement ratios among the proposed 3 algorithms and BAS. "A vs B" refers to the improvement in the percentage of method A over method B. Take "Baseline vs BAS" as an example, the improvement percentage of the baseline over BAS is $(355.56-305.26)/355.56\times 100\% = 14.15\%$ when $T = 4$ in the low-profit price group. Comparing numbers in "SCD VS BAS", "PCD VS BAS", and "Baseline VS BAS" when stage num = 4 and 12, we can see that the advantages of the proposed methods over BAS are more distinct when the number of stages is larger. Additionally, comparing numbers in "SCD VS Baseline" and "PCD VS Baseline" when stage num = 4 and 12, we also note that the advantages of SCD and PCD over the Baseline are more distinct when the number of stages is larger.

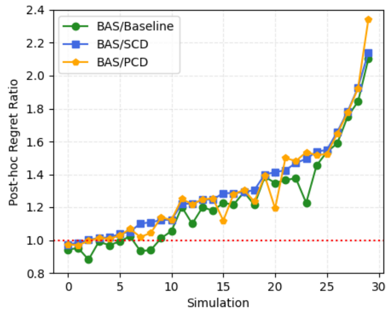
Table 6: Improvement ratios among Baseline, SCD, PCD, and standard regression models for the production and sales problem.

Price group	Stage num	SCD VS BAS	PCD VS BAS	Baseline VS BAS	SCD VS Baseline	PCD VS Baseline	SCD VS PCD
Low-profit	4	17.38%	16.37%	14.15%	3.76%	2.59%	1.20%
	12	23.37%	22.35%	19.12%	5.25%	3.99%	1.31%
High-profit	4	10.00%	7.23%	6.16%	4.09%	1.14%	2.98%
	12	11.03%	9.21%	6.26%	5.10%	3.15%	2.01%

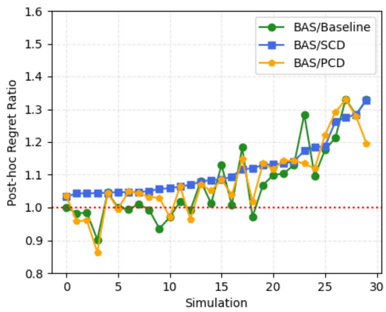
Figure 1 shows post-hoc regret comparisons between BAS and the proposed methods (Baseline, SCD, and PCD) for each run. The x-axis refers to the number of each simulation, and the y-axis refers to the ratio of the post-hoc regret achieved by BAS and the proposed methods (Baseline, SCD, and PCD) corresponding to that simulation. To easily read the comparisons, we sorted all simulations by the increasing order of the post-hoc regret ratios of BAS/SCD. The red dashed line where the post-hoc regret ratio is 1.0 represents the boundary line where (Baseline, SCD, or PCD) performs better or worse than BAS. When the point representing the post-hoc regret ratio of BAS/(Baseline, SCD, or PCD) falls above the red dashed line, it means that (Baseline, SCD, or PCD) performs better than BAS. Conversely, when the point falls below the red dashed line, it means BAS performs better than (Baseline, SCD, or PCD). Observing Figure 1, SCD outperforms BAS across all simulations in all 4 scenarios. While not as stable as SCD, PCD and Baseline also outperform BAS in most of the simulations. Compared with Figure 1a, there are more BAS/Baseline points that fall below the red dashed line in Figure 1b, while the number of BAS/SCD points and the number of BAS/PCD points that fall below the red dashed line are similar in Figure 1a and Figure 1b. The same phenomenon can be observed when comparing Figure 1c and Figure 1d, demonstrating the advantage of SCD and PCD over Baseline.



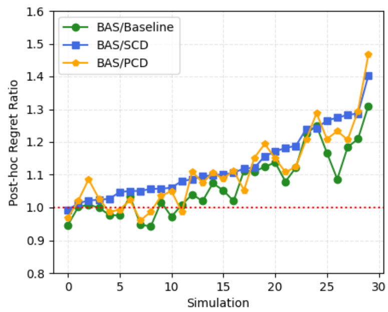
(a) Low-profit products and 4 quarters.



(b) Low-profit products and 12 months.



(c) High-profit products and 4 quarters.



(d) High-profit products and 12 months.

Figure 1: BAS/Baseline, BAS/SCD, and BAS/PCD for the production and sales problem.

F.2 Investment Problem

Table 7 and Table 8 report the mean post-hoc regrets and standard deviations across 30 simulations for all training methods on the investment problem. Compared among standard regression models, NN performs well and achieves the best performance in most cases, while Ridge and RF also have decent performances and obtain the smallest mean post-hoc regret in some cases.

Table 7: Mean post-hoc regrets and standard deviations of all methods for the investment problem when capital=25.

Stage num	4			12		
Transaction factor	0.01	0.05	0.1	0.01	0.05	0.1
SCD	**19.85±3.14**	**14.73±3.59**	**10.56±1.63**	**1513.31±185.03**	**666.96±91.54**	**260.27±34.32**
PCD	20.00±3.24	14.90±3.62	10.63±1.62	1524.69±191.19	675.27±95.10	263.97±35.09
Baseline	20.20±3.68	15.14±3.62	10.77±1.64	1540.47±186.90	686.84±92.49	269.07±34.47
NN	20.51±3.43	15.47±3.67	11.23±1.87	1563.78±199.67	699.30±101.58	277.31±32.99
Ridge	20.88±3.30	15.38±3.37	11.70±2.00	1588.11±200.48	703.74±97.62	276.51±32.14
k-NN	22.21±3.44	16.96±4.18	11.56±2.15	1643.46±198.96	722.73±79.93	285.73±41.26
CART	24.81±4.30	19.68±4.58	13.42±2.27	1845.40±285.85	832.02±129.30	333.71±51.84
RF	21.88±3.56	16.98±3.74	12.07±1.93	1563.94±190.01	700.31±77.70	279.84±34.73
TOV	64.11±4.91	51.53±9.97	39.87±2.67	2404.22±264.58	1147.61±114.54	502.05±46.67

Table 8: Mean post-hoc regrets and standard deviations of all methods for the investment problem when capital=50.

Stage num	4			12		
Transaction factor	0.01	0.05	0.1	0.01	0.05	0.1
SCD	**47.48±6.98**	**34.92±5.57**	**25.50±3.88**	**3846.20±420.94**	**1663.82±208.60**	**646.14±75.52**
PCD	47.67±6.64	35.22±5.98	25.63±3.93	3869.76±420.01	1679.17±205.01	652.57±74.45
Baseline	48.24±7.13	35.64±6.28	25.96±4.64	3941.09±437.57	1701.51±222.45	665.71±76.40
NN	48.98±7.19	36.42±5.70	26.62±4.07	4046.79±390.52	1736.59±232.15	680.94±70.76
Ridge	50.73±6.35	37.38±4.66	27.62±3.12	4019.04±454.78	1743.96±217.95	682.24±74.91
k-NN	53.49±8.52	39.69±7.22	28.12±3.86	4213.33±434.62	1797.32±206.92	702.48±94.53
CART	62.35±11.68	47.05±9.14	31.81±6.76	4723.27±529.86	2086.09±325.70	835.96±137.08
RF	52.49±6.73	39.07±6.50	27.75±3.84	3999.70±475.44	1748.02±201.68	696.46±75.14
TOV	158.56±11.19	126.22±8.86	99.83±7.02	6166.73±573.51	2860.05±288.85	1259.99±107.60

Table 9: Improvement ratios among Baseline, SCD, PCD, and standard regression models for the investment problem.

Capital	Stage num	Transaction factor	SCD VS BAS	PCD VS BAS	Baseline VS BAS	SCD VS Baseline	PCD VS Baseline	SCD VS PCD
25	4	0.01	3.25%	2.53%	1.53%	1.74%	1.01%	0.74%
		0.05	4.23%	3.15%	1.57%	2.70%	1.61%	1.12%
		0.1	6.03%	5.39%	4.12%	1.99%	1.33%	0.67%
	12	0.01	3.23%	2.50%	1.49%	1.76%	1.02%	0.75%
		0.05	4.07%	3.17%	1.78%	2.90%	1.68%	1.23%
		0.1	5.87%	4.53%	2.69%	3.27%	1.89%	1.40%
50	4	0.01	3.06%	2.68%	1.51%	1.58%	1.19%	0.39%
		0.05	4.12%	3.29%	2.14%	2.02%	1.18%	0.85%
		0.1	4.21%	3.71%	2.46%	1.79%	1.27%	0.52%
	12	0.01	3.84%	3.25%	1.47%	2.41%	1.81%	0.61%
		0.05	4.19%	3.31%	2.02%	2.22%	1.31%	0.91%
		0.1	5.11%	4.17%	2.24%	2.94%	1.97%	0.99%

Table 9 shows improvement ratios among the proposed 3 algorithms and BAS. Comparing "SCD vs BAS", "PCD vs BAS", and "Baseline vs BAS" performance under the same capital and the same stage number, we observe that the advantages of the proposed methods (SCD, PCD, and Baseline) over the standard regression approaches become more pronounced as the transaction factor increases. Besides, comparing “SCD vs Baseline" and “PCD vs Baseline" under the same capital and the same transaction factor but different stage numbers, the advantages of SCD and PCD over Baseline are amplified as the number of stages increases. This trend is consistent with the findings from the experiments on the production and sales problem. One interesting phenomenon is that under the same capital and the same transaction factor, the advantage of the proposed methods over BAS appears to be similar or even less obvious when the number of stages is 12 compared to when it is 4. This observation differs from the pattern seen in the production and sales problem experiments. We hypothesize that this divergence may be attributable to the fundamental differences between the problem settings. While the production and sales problem is a pure LP, the investment problem is an IP with several integrality constraints. The proposed methods relax these integrality constraints and treat the problem as an LP for the purpose of forward and backward propagation. The gaps between the original IP and the relaxed LP may accumulate as the number of stages grows larger, potentially diminishing the advantages of the Predict+Optimize approaches.

F.3 Nurse Rostering Problem

Table 10 reports the mean post-hoc regrets and standard deviations across 30 simulations for all training methods on the nurse rostering problem. Compared among standard regression models, NN always performs well and achieves the best performance, while Ridge and RF also have decent performances.

Table 11 shows improvement ratios among the proposed 3 algorithms and BAS. Comparing "SCD vs BAS", "PCD vs BAS", and "Baseline vs BAS" performance with different extra nurse payments, we observe that the advantages of the proposed methods (SCD, PCD, and Baseline) over the standard regression approaches become more pronounced as the extra nurse payment increases.

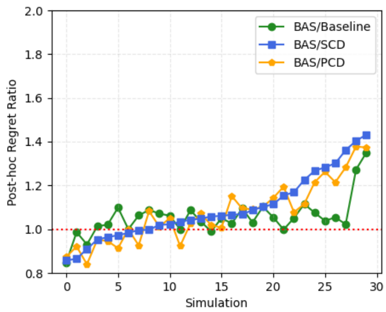
Figure 2 shows post-hoc regret comparisons between BAS and the proposed methods (Baseline, SCD, and PCD) for each run. To easily read the comparisons, we again sorted all simulations by

Table 10: Mean post-hoc regrets and standard deviations of all methods for the nurse rostering problem.

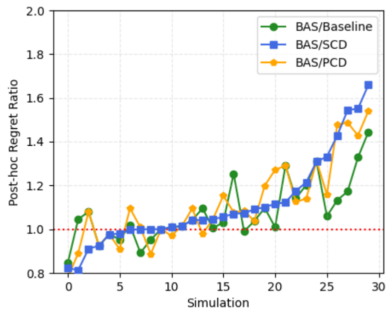
Extra nurse payment	15	20	25	30
SCD	**607.66±142.19**	**789.65±200.22**	**1038.29±255.42**	**1207.50±319.25**
PCD	622.05±153.64	805.11±224.99	1048.08±281.32	1240.48±332.39
Baseline	629.35±153.67	817.60±219.47	1083.45±259.68	1290.10±371.08
NN	662.34±169.17	863.02±214.50	1144.63±305.00	1369.81±373.20
Ridge	663.57±141.49	887.63±206.36	1146.56±297.33	1371.20±320.37
k-NN	758.49±135.75	1033.88±197.22	1309.92±255.86	1562.98±298.14
CART	965.16±207.67	1303.68±280.11	1645.59±350.32	1957.13±433.46
RF	680.47±148.20	870.50±221.81	1145.32±293.18	1378.68±333.73
TOV	10611.64±1574.11	10732.32±1504.12	10893.54±1485.37	11110.73±1344.15

Table 11: Improvement ratios among Baseline, SCD, PCD, and standard regression models for the nurse rostering problem.

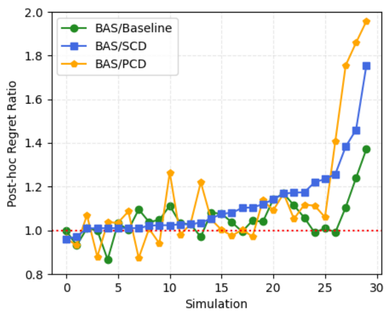
Extra nurse payment	SCD VS BAS	PCD VS BAS	Baseline VS BAS	SCD VS Baseline	PCD VS Baseline	SCD VS PCD
15	8.26%	6.08%	4.98%	3.45%	1.16%	2.31%
20	8.50%	6.71%	5.26%	3.42%	1.53%	1.92%
25	9.29%	8.44%	5.35%	4.17%	3.26%	0.93%
30	11.85%	9.44%	5.82%	6.40%	3.85%	2.66%



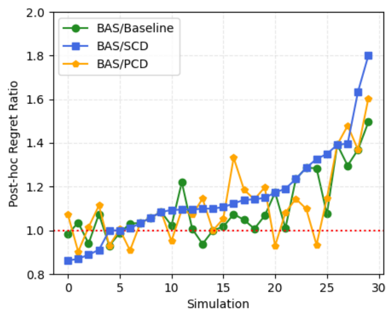
(a) Extra nurse payment = 15.



(b) Extra nurse payment = 20.



(c) Extra nurse payment = 25.



(d) Extra nurse payment = 30.

Figure 2: BAS/Baseline, BAS/SCD, and BAS/PCD for the nurse rostering problem.

the increasing order of the post-hoc regret ratios of BAS/SCD. Observing Figure 2, the proposed methods outperform BAS in most of the simulations. Since the nurse rostering problem is an IP with several integrality constraints, and the proposed methods just relax these constraints and treat the problem as an LP for the purpose of forward and backward propagation. We hypothesize that the gap between the original IP and the relaxed LP may diminish the advantages of the proposed methods, and thus, BAS sometimes performs slightly better than the proposed methods.

G Runtimes for the Experiments

Table 12: Average training time (in seconds) for the three benchmarks (in seconds).

Training time (s)	Production and sales				Investment				Nurse rostering
	Stage num = 4		Stage num = 12		Stage num = 4		Stage num = 12		Stage num = 7
	Low-profit	High-profit	Low-profit	High-profit	Capital = 25	Capital = 50	Capital = 25	Capital = 50	\
SCD	828.79±216.69	700.63±244.37	3287.99±809.72	2552.73±991.69	402.37±58.00	535.18±88.45	7734.01±1198.41	11216.01±1994.75	14949.59±3281.24
PCD	293.41±96.27	236.80±81.07	483.28±95.81	470.76±124.97	157.25±41.65	194.40±57.51	2639.72±648.22	4509.83±767.45	6801.54±1175.01
Baseline	157.72±50.85	100.09±32.50	195.42±35.03	169.58±45.62	56.04±14.73	61.49±18.30	669.01±261.78	797.61±282.70	2618.63±524.37
NN	70.58±24.78		97.60±46.17		49.24±18.24		70.81±29.81		61.41±5.28
Ridge	1.71±0.29		2.88±0.39		5.60±1.28		17.45±7.96		≤ 1
k-NN	0.98±0.96		1.03±0.24		1.92±0.62		11.35±0.99		
CART	0.77±0.19		2.46±0.39		5.79±1.45		27.30±2.19		
RF	24.82±1.13		91.93±1.80		358.19±4.26		1150.64±484.87		

In this paper, all models are trained with Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz processors on Google Colab. Since the testing time of different approaches is quite similar and close to being negligible, here, we only show the training time of each prediction model and do not include the testing time. At training time, the proposed Baseline, SCD, and PCD methods need to solve the LP. Training for the usual NN does not involve the LP at all, and so training is much faster (but gives much worse results).

There are two stopping criteria for SCD and PCD. We set the maximum iteration number of SCD and PCD as 5. Besides, if the difference between the post-hoc regret of two consecutive iterations is less than 1, we consider that the algorithm has converged. This is the second stopping criterion.

Table 12 shows the average training time across 30 simulations for the three problems. For the investment problem, since the training times under different transaction fees are similar when the capital and the number of stages are the same, we do not report them one by one but report only the average. For the nurse rostering problem, since the training times under different extra nurse payments are similar when the numbers of stages are the same, we also do not report them one by one but report only the average.

Since the proposed 3 methods require solving multiple LPs when training, their training times are usually longer than standard methods. But since the production and sales problem is an LP, the solving time of which is not too long, the training time of Baseline is around double of NN. Table 12 also shows that SCD and PCD usually converge in 4 iterations in the production and sales problem.

In the investment problem, the training times of Baseline are better than that of RF. The solving time of the OP, i.e., the difficulty of solving the OP, can largely affect the training times of the proposed methods. When the number of stages grows larger, the investment problem is more difficult to solve. Therefore, the training times of Baseline when there are 4 stages are quite comparable with that of NN, but the training times of Baseline when there are 12 stages are much larger than that of NN. In addition, when the OP becomes more complex, the number of iterations required for SCD and PCD convergence also increases. SCD and PCD convergence usually in 2-3 iterations when there are 4 stages, and usually in 3-4 iterations when there are 12 stages.

In the NRP, since the solving time of 1 NRP is large, the training times of the proposed methods are larger than standard regression methods, which indicates that one future research direction is the speed-vs-prediction accuracy tradeoffs on Multi-Stage Predict+Optimize.