Data-Driven Optimization: A Reproducing Kernel Hilbert Space Approach

Submitted to *Operations Research*
manuscript (Please, provide the manuscript number!)

Data-Driven Optimization: A Reproducing Kernel Hilbert Space Approach

Dimitris Bertsimas
Boeing Professor of Operations Research, Massachusetts Institute of Technology, Cambridge, MA 02139, dbertsim@mit.edu
Nihal Koduri
Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA 02139, nihalk@mit.edu

We present two methods, based on regression in reproducing kernel Hilbert spaces, for solving an optimization problem with uncertain parameters for which we have historical data, including auxiliary data. The first method approximates the objective function and the second approximates the optimizer. We provide finite sample guarantees and prove asymptotic optimality for both methods. Computational experiments suggest that at least the second method overcomes a curse of dimensionality that afflicts existing methods, extrapolates better to unseen data, and achieves a many-fold decrease in sample complexity even for small dimensions.

Key words: data-driven optimization, prescriptive analytics, reproducing kernel Hilbert space, regression, optimality

1. Introduction

A fundamental problem in optimization under uncertainty is

$$\min_z \mathbb{E}_y[c(z;y) \,|\, x = x^0], \tag{1}$$

where z is a decision, $c(z;y)$ is a given cost function, y is an uncertain parameter affecting the cost, and x^0 is an observation of auxiliary data that will be used to predict y. The only information we have is historical data $\{(x^i, y^i)\}_{i=1}^n$, where $x^i \in \mathbb{R}^{d_x}$ are the historical auxiliary data, and $y^i \in \mathbb{R}^{d_y}$ are the corresponding realizations of y.

An example of Problem (1) is the newsvendor problem, where z is the number of newspapers to order, y is the daily demand for newspapers, and x^0 is auxiliary data used to predict the demand, such as the temperature. Assuming that the cost per newspaper is d and the revenue per newspaper sold is r, the cost function is

$$c(z,y) = dz - r\min(z,y)\,.$$

We want to maximize the expected profit $r\min(z,y) - dz$, or equivalently minimize the cost $c(z;y)$, given today's temperature x^0. The only information we have is historical temperature and demand data $\{(x^i,y^i)\}_{i=1}^n$.

A natural approach is to use the data $\{(x^i,y^i)\}_{i=1}^n$ to build a machine learning model for predicting y from x (for example, linearly regress y on x). This produces an estimate of y, $\widehat{y}(x)$, for every x. We then solve

$$\min_z\ c(z;\widehat{y}(x^0))$$

instead of (1). Since $\widehat{y}(x^0)$ is an approximation of $\mathbb{E}[y\,|\,x=x^0]$, this approach solves the problem

$$\min_z\ c(z;\mathbb{E}[y\,|\,x=x^0])\,,$$

which is different from (1) because it does not take into account uncertainty in y.

Bertsimas and Kallus (2020) propose an approach to solving (1) that does take into account uncertainty in y. The authors use the fact that for several machine learning methods, the prediction of y from x^0 takes the form

$$\widehat{y}(x^0) = \sum_{i=1}^n w(x^i,x^0)y^i\,,$$

where $w(x^i,x)$ is a measure of closeness between x^i and x and depends on the machine learning method used. For example, in nearest neighbors $w(x^i,x^0) = \frac{1}{k}$ if x^i is a k-nearest neighbor of x^0, and $w(x^i,x^0) = 0$, otherwise. In CART, $w(x^i,x^0) = \frac{1}{d}$ for the d elements x^i in the same leaf as x^0, and $w(x^i,x^0) = 0$ for all other x^i. Bertsimas and Kallus (2020) also consider random forests, Nadaraya-Watson kernel regression, and local linear regression.

Bertsimas and Kallus (2020) take the perspective that these weights can be considered an approximation of the conditional distribution of y given $x = x^0$. Thus, they take

$$\min_z \sum_{i=1}^{n} w(x^i, x^0) c(z; y^i) \tag{2}$$

as an approximation of (1). They show that for w derived for several of the machine learning methods, the solution of (2) converges asymptotically as $n \to \infty$ to the solution of (1). Furthermore, they present very promising computational results in both synthetic and real world examples.

1.1. Our Approach

In this paper, we propose new methods for solving (1) based on "global" machine learning methods, as opposed to the "local" machine learning methods considered by Bertsimas and Kallus (2020). The primary difference between what we call local and global machine learning methods is that local machine learning methods predict by measuring closeness to existing data, while global machine learning methods predict by choosing a functional form of the prediction that minimizes some loss function on existing data. Global machine learning methods tend to be used more in practice for predicting real valued outputs (i.e., for regression as opposed to classification). For example, the most ubiquitous machine learning algorithm for predicting real valued outputs is linear regression by ordinary least squares (finding the best linear function that minimizes a square loss).

There are several reasons why one may prefer global methods over local methods. One is that local methods in some sense throw away all data that is not near the current observation, and so they need a lot of data (in particular, a lot of data close to the current observation) to work well. In contrast, global methods use all the data, so they should be better with less data. For the same reason, global methods should be better at extrapolating to outliers. Another reason to prefer global methods is the well-known fact that local methods suffer from a curse of dimensionality, their performance degrading exponentially as the dimension of the covariate vector x increases. Hastie et al. (2009, sec 2.5) give a good explanation of this phenomenon, the gist of which is that points near x^0 become much less representative of x^0 in higher dimensions. This becomes even

more problematic in the prescriptive setting because we are using points near x^0 not only to find the mean of y given $x = x^0$, but to describe the entire distribution of y given $x = x^0$. As Hastie et al. (2009) explain, global methods avoid the curse of dimensionality by not using distance as a measure of similarity. In our computational results, we provide evidence for all three of these claims in prescriptive problems: that the new global methods perform better with less data, that the new methods extrapolate better to outliers, and that the new methods perform better in higher dimensions.

Now that we have motivated our methods, we give a simple overview of them. We take a different perspective to that of Bertsimas and Kallus (2020), noticing that instead of estimating the conditional distribution of y given $x = x^0$, all we need to do is estimate the conditional expectation. We can do this with linear regression. In particular, if we wanted to estimate $\mathbb{E}[y \,|\, x]$ using linear regression, that is, we wanted to find some β such that $\mathbb{E}[y \,|\, x] \approx \beta' x$, we could compute

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} \|Y - X\beta\|_2^2 ,$$

where Y is a vector with elements y^i, and X is a matrix with rows x^i. The minimizer of this (assuming $X^T X$ is invertible) would take the form

$$\beta = (X^T X)^{-1} X^T Y ,$$

and the prediction of the y corresponding to x^0 would be

$$\mathbb{E}[y \,|\, x = x^0] \approx \beta^T x^0 = Y^T X (X^T X)^{-1} x^0 .$$

If we wanted to predict not y, but $c(z; y)$, we would compute

$$\min_{\beta(z)} \frac{1}{n} \sum_{i=1}^{n} \|c(z; Y) - X\beta(z)\|_2^2 .$$

The minimizer would take the form

$$\beta(z) = (X^T X)^{-1} X^T c(z; Y) ,$$

and the approximation of $\mathbb{E}[c(z;y)\,|\,x=x^0]$ would be

$$\mathbb{E}[c(z;y)\,|\,x=x^0] \approx \beta^T x^0 = c(z;Y)^T X(X^T X)^{-1}x^0\,,$$

where $c(z;Y)$ is the vector $(c(z;y^1),\ldots,c(z;y^n))^T$. We see that this approximation is of the form (2), so we can minimize over z to solve the original problem (1). This is the essence of the first new method. In Section 4, we derive the method from first principles using the framework of reproducing kernel Hilbert spaces, which allows us to generalize the approach to nonlinear predictions and prove asymptotic optimality.

In the second method, we predict the optimum decision z directly instead of finding an approximation of $\mathbb{E}[c(z;y)\,|\,x=x^0]$ and minimizing over the approximation. Let us use the same example of linear regression. Suppose we want to approximate the optimum decision z by a linear function of x. More specifically, we want to find β such that $\arg\min_z \mathbb{E}[c(z;y)\,|\,x] \approx \beta^T x$. We can do this by finding the β that minimizes the cost on the historical data:

$$\min_{\beta} \sum_{i=1}^{n} c(\beta^T x^i; y^i)\,.$$

In Section 5, we generalize the approach to predictions nonlinear in x and prove asymptotic optimality by using the framework of reproducing kernel Hilbert spaces.

The rest of the paper is structured as follows. In Section 2, we review the relevant literature and discuss our contributions to it. In Section 3, we provide a brief overview of reproducing kernel Hilbert spaces. In Section 4, we present the objective prediction method and associated probabilistic guarantees. in Section 5, we present the optimizer prediction method and associated probabilistic guarantees. In Section 6, we explain how to incorporate constraints and give examples of how the ideas in this paper can be used to solve other problems in operations research. In Section 7, we present our computational results. In Section 8, we conclude.

2. Literature Review

Two bodies of relevant literature, stochastic programming and robust optimization, develop general frameworks for modeling and solving optimization problems under uncertainty but do not take

data as a primitive. Birge and Louveaux (2011) give an overview of stochastic programming, while Bertsekas (1995) gives an overview of multiperiod stochastic programming. Ben-Tal et al. (2009) and Bertsimas et al. (2011) give an overview of robust optimization.

A related body of literature does take data as a primitive, but doesn't consider auxiliary data. Kleywegt et al. (2002), Shapiro (2003), and Shapiro and Nemirovski (2005) develop the sample-average approximation, which is the data-driven offspring of stochastic programming. Delage and Ye (2010) develop data-driven distributionally robust optimization. Bertsimas et al. (2018) develop a different data-driven robust optimization approach.

Machine learning takes data as a primitive and considers auxiliary data, but it is focused purely on prediction and not on prescription. We provide only a few basic references. Hastie et al. (2009) provide a broad overview of different machine learning methods. Vapnik (1998), Cucker and Smale (2002), Bousquet and Elisseeff (2002), and Poggio and Smale (2005) introduce classical techniques for proving theoretical properties. Some of these techniques we use in this paper.

The closest body of literature to this paper considers optimization under uncertainty with data and auxiliary data. However, in contrast to this paper, this body is focused on adapting what we have termed "local" machine learning methods. Notable references in this body of literature include Hannah et al. (2010), who use local kernel methods to approximate the objective, Hanasusanto and Kuhn (2013), who use local kernel methods to approximate a multistage objective, Bertsimas and Kallus (2020), who adapt several local methods to approximate the objective, and Bertsimas and McCord (2017) who adapt local methods to approximate the objective in multistage problems.

2.1. Contributions

This paper introduces the first general, asymptotically optimal approaches to prescriptive analytics based on loss-minimizing (what we have termed "global") machine learning methods. As discussed in Section 1.1, global methods have the potential to work better with less data, for extrapolation, and in high dimensions. Computational results in Section 7 corroborate each of these claims. We now discuss the novelty of the two approaches introduced in this paper in more detail separately.

One of the approaches introduced by this paper uses global machine learning to predict the objective. This has not been done before in the literature. In fact, as mentioned in the introduction, the previous literature such as Bertsimas and Kallus (2020), although able to be seen as approximating the objective using local methods, does not take the perspective that it is predicting the objective, but rather the perspective that it is estimating the conditional distribution of y. This new perspective, that what really needs to be done is predict the objective, not only leads to a new method, but also leads to interesting possibilities for extensions that are not possible with the old perspective, such as tractable prescriptive analytics when the decision affects the parameter (see Section 6).

The second approach introduced by this paper uses global machine learning to predict the optimizer. There is work in this vein, but none is both general and asymptotically optimal. For example, Ban and Rudin (2019) consider an approach to predicting the optimizer, but it only applies to the newsvendor problem. Bertsimas and Kallus (2020) suggest a more general way to predict the optimizer, but it is not asymptotically optimal, and they consider it impractical due to not being able to handle nonlinearities and constraints. We develop a method that can handle constraints and nonlinearities, and we prove asymptotic optimality. We also provide a reformulation of our method that scales, and we provide evidence of this by implementing the method on practical problems.

As a final contribution, we present our methods using a general framework of reproducing kernel Hilbert spaces that we hope can be applied to other problems in optimization under uncertainty. As a start, in Section 6, we show how the ideas in this paper can be used to create a potentially tractable and asymptotically optimal method to solve data-driven multi-stage optimization problems.

3. Overview of Reproducing Kernel Hilbert Spaces

In this section, we provide an overview of reproducing kernel Hilbert spaces to make the paper self contained. The standard reference is Aronszajn (1950).

A reproducing kernel Hilbert space is a type of Hilbert space (a set of functions) that behaves well pointwise. The pointwise behavior makes it ideal for working with data. As its name suggests,

a reproducing kernel Hilbert space is associated with a reproducing kernel. It is possible to define a reproducing kernel Hilbert space abstractly, by proving that if a Hilbert space satisfies some condition, then there exists an associated reproducing kernel. Here, we take a constructive approach. Starting with a positive definite kernel $K(\cdot,\cdot)$ over a domain $\mathcal{X}$, i.e., a function $K:\ \mathcal{X}\times\mathcal{X}\to\mathbb{R}$ such that

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j K(x^i,x^j)\geq 0 \quad \forall n\in\mathbb{N}, x^1,\dots x^n\in\mathcal{X}, a_1,\dots,a_n\in\mathbb{R},$$

we define next the Hilbert space generated by the kernel.

Definition 1. A reproducing kernel Hilbert space $\mathcal{H}$ generated by a positive definite kernel $K:\mathcal{X}\times\mathcal{X}\to\mathbb{R}$ is the closure of the set of functions

$$\left\{ g:\mathcal{X}\to\mathbb{R} \;\middle|\; g(x)=\sum_{l=1}^{L} a^l K(v^l,x), \text{ for } v^1,\dots,v^L\in\mathcal{X} \text{ and } L\in\mathbb{N} \right\}$$

with inner product of $g_1(x)=\sum_{l=1}^{L_1} a_1^l K(v_1^l,x)$ and $g_2(x)=\sum_{l=1}^{L_2} a_2^l K(v_2^l,x)$ defined as

$$\langle g_1,g_2\rangle=\sum_{l_1=1}^{L_1}\sum_{l_2=1}^{L_2} a_1^{l_1}a_2^{l_2}K(v_1^{l_1},v_2^{l_2}).$$

In other words, any function in the reproducing kernel Hilbert space generated by a kernel can be written as $\lim_{n\to\infty} g_1^n$, where g_1^n is a function of the form $\sum_{l=1}^{L_1} a_1^{l,n}K(v_1^{l,n},\cdot)$. The inner product of any two functions $\lim_{n\to\infty} g_1^n$ and $\lim_{n\to\infty} g_2^n$ is $\lim_{n\to\infty}\langle g_1^n,g_2^n\rangle$.

Several familiar function spaces are in fact reproducing kernel Hilbert spaces. If we choose K to be the linear kernel, then $\mathcal{H}$ is the space of all linear functions. If K is the polynomial kernel of degree d, then $\mathcal{H}$ is the space of all polynomials of degree less than or equal to d. If K is the Gaussian kernel, then $\mathcal{H}$ contains a functions arbitrarily close in sup-norm to any continuous function on a compact subset of $\mathcal{X}$. Thus, a reproducing kernel Hilbert space can be anything from the set of all linear functions to almost the set of all continuous functions.

The next result, known as the Representer Theorem, illustrates one of the major reasons why reproducing kernel Hilbert spaces are useful for working with data. It tells us that we can solve functional minimization problems over h in a reproducing kernel Hilbert space $\mathcal{H}$, as long as the objective only involves a term consisting of h evaluated at a finite set of points and a term consisting of the squared norm of h (the inner product of h with itself).

PROPOSITION 1 **(Representer Theorem).** *Fix $x^i \in \mathcal{X}$ and $y^i \in \mathcal{Y}$ for $i = 1, \ldots, n$. Let $\mathcal{H}$ be the reproducing kernel Hilbert space generated by a kernel $K : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$. Let $V : \mathbb{R} \times \mathcal{Y} \to \mathbb{R}$ be an arbitrary function and let $\lambda \geq 0$. There exists a solution to*

$$\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} V(h(x^i), y^i) + \lambda \|h\|_{\mathcal{H}}^2 \tag{3}$$

that takes the form

$$h^*(x) = \sum_{i=1}^{n} K(x^i, x) a^i \,, \tag{4}$$

where $a^i \in \mathbb{R}$ for $i = 1, \ldots, n$. We can represent a as the solution to the finite dimensional optimization problem

$$\min_{a \in \mathbb{R}^n} \sum_{i=1}^{n} V((\widehat{K}a)_i, y^i) + \lambda a^T \widehat{K} a \,, \tag{5}$$

where $\widehat{K}$ is the matrix with component $\widehat{K}_{ij} = K(x^i, x^j)$.

The proof of this theorem follows from the fact that any function in $\mathcal{H}$ can be written as the sum of a function of the form (4) and a function orthogonal to every function of the form (4). One can show that the first term in the objective (5) is independent of the orthogonal part, and the second term in the objective is increasing in the orthogonal part. Hence, the orthogonal part can be set to 0. We state and formally prove a multidimensional version of the Representer Theorem in Proposition EC.2 in Section EC.1 of the electronic companion to this paper.

In the next two sections, we use the concept of a reproducing kernel Hilbert space and the Representer Theorem in order to develop the two methods for solving (1).

4. Objective Prediction Method

In the introduction, we presented the objective prediction method as using regression to approximate the objective. Here, we derive the same method naturally starting from (1) using basic probability theory and the ideas from Section 3.

Conditional expectation is an L^2 projection, so for fixed z, the objective of (1) is exactly the solution to

$$\min_{h(z,\cdot)} \mathbb{E}[(c(z; y) - h(z, x))^2] \tag{6}$$

minimized over all measurable functions and then evaluated at x^0. Since we do not know the distribution of (x, y), instead of (6), we consider the empirical version of (6)

$$\min_{h(z,\cdot)} \frac{1}{n} \sum_{i=1}^{n} (c(z; y^i) - h(z, x^i))^2. \tag{7}$$

This is still an intractable functional minimization problem, so the next step is to restrict h to be in a Hilbert space $\mathcal{H}$ defined by a positive-definite kernel function $K(\cdot, \cdot)$ (see Definition 1). If we restrict h to be in $\mathcal{H}$, (7) becomes tractable, thanks to the Representer Theorem (Proposition 1). In fact, we can also add a regularization term to prevent overfitting:

$$\min_{h(z,\cdot)\in\mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (c(z; y^i) - h(z, x^i))^2 + \lambda \|h\|_{\mathcal{H}}^2. \tag{8}$$

The Representer Theorem tells us that any minimizer to (8) must take the form

$$h(z, \cdot) = \sum_{j=1}^{n} a^j(z) K(x^j, \cdot)$$

for some scalars $a^j(z) \in \mathbb{R}$. We can plug this expression back into (8) and solve for the minimizing $a^j(z)$ in closed form by setting the gradient with respect to a equal to zero. This solution, the minimizer of (8), is (assuming $\lambda > 0$)

$$h(z, x) = K(X, x)^T (\widehat{K} + \lambda n I)^{-1} c(z; Y)\,,$$

where $K(X, x)$ is the vector $(K(x^1, x), \ldots, K(x^n, x))^T$, $c(z; Y)$ is the vector $(c(z; y^1), \ldots, c(z; y^n))^T$, and $\widehat{K}$ is the kernel matrix, i.e., the $n \times n$ matrix with components $\widehat{K}_{ij} = K(x^i, x^j)$. We have that

$$\mathbb{E}[c(z; y) \,|\, x] \approx h(z, x) = K(X, x)^T (\widehat{K} + \lambda n I)^{-1} c(z; Y)\,,$$

and so the objective of (1) can be approximated by

$$h(z, x^0) = K(X, x^0)^T (\widehat{K} + \lambda n I)^{-1} c(z; Y)\,. \tag{9}$$

4.1. Tractability

Notice that if $c(z;y)$ is convex for fixed y, and $(\widehat{K}+\lambda nI)^{-1}K(X,x^0)$ is non-negative for fixed x^0, then (9) is convex for fixed x^0, and it is tractable to solve (1) using (9) as an approximation for the objective function. We observe empirically in the computational experiments in Section 7 that $(\widehat{K}+\lambda nI)^{-1}K(X,x^0)$ is effectively non-negative, with the negative coefficients being around six orders of magnitude smaller than the non-negative ones. Nevertheless, to complete the theory, we provide a generalization of the objective prediction method in Section EC.3 of the electronic companion to this paper, which produces an objective approximation of the form $h(z,x)=K(X,x)^TMc(z;Y)$ for some matrix M, where $K(X,x)^TM\geq 0$, after solving a convex optimization problem with n^2 variables. This approach can be thought of as finding the best approximation of the objective of (1) that is in $\mathcal{H}$ for fixed z and is a non-negative combination of $c(z;y^i)$ for fixed x^0.

The simpler approach, and the one we use in the computational experiments, is just to take

$$h(z,x)=\max\left(K(X,x^0)^T(\widehat{K}+\lambda nI)^{-1},0\right)c(z;Y)\,.$$

In any case, whether we use the approximation from Section EC.3 or the simpler approach, problem (1) is reduced to the convex optimization problem

$$\min_{z\in\mathcal{Z}} h(z,x^0)\,. \tag{10}$$

In (10), we are allowing the additional possibility of z needing to be restricted to some feasible set $\mathcal{Z}$, which we did not explicitly allow for in (1).

For a rough Big-O analysis, assume that a problem of the form $\min_{z\in\mathcal{Z}}\sum_{i=1}^n w_ic(z;y^i)$ can be reformulated as a linear optimization problem (LOP) with $D(n,d)$ variables, where d is the dimension of z, and assume that it takes $O(k^3\log(1/\epsilon))$ time to find an ϵ-optimal solution for an LOP with k variables. Also suppose that along with the training set of size n, we have a test set of size n_{test}. The objective prediction method takes $O(n^2d)$ time to compute the kernel matrix, $O(n^3)$ time to factor the matrix, $O((n^2+nd)n_{\text{test}})$ to compute the weights w_i for each test point, and

$O(D(n,d)^3 n_{\text{test}} \log(1/\epsilon))$ time to find an ϵ-optimal solution for each test point. Assuming d is at most $O(n)$, this results in a total time of $O(n^3 + n^2 n_{\text{test}} + D(n,d)^3 n_{\text{test}} \log(1/\epsilon))$.

This compares to, for example, the k-nearest neighbors method from Bertsimas and Kallus (2020), which takes $O(ndn_{\text{test}})$ time to compute the weights for each test point and $O(D(n,d)^3 n_{\text{test}} \log(1/\epsilon))$ time to find ϵ-optimal solutions for each test point, resulting in a total time of $O(ndn_{\text{test}} + D(n,d)^3 n_{\text{test}} \log(1/\epsilon))$. So if n_{test} is $O(n)$ (as it is in the standard 70%-30% train-test split), the objective prediction method takes about a factor of n times longer than the k-nearest neighbors method from Bertsimas and Kallus (2020), unless $D(n,d)$ is at least $O(n)$ or d is $O(n)$, in which case both the methods take about the same amount of time. Note also that k-nearest neighbors is the least computationally demanding method from Bertsimas and Kallus (2020).

4.2. Theoretical Guarantees

We now derive finite sample error bounds for the objective prediction method and use the finite sample error bounds to prove asymptotic convergence. There is a large literature on generalization bounds in machine learning. This literature can be seen as focused on showing that the prediction obtained by regressing y on x is close to $\mathbb{E}[y \,|\, x]$. Here we are regressing $c(z;y)$ on x, and then minimizing over z. We need to show not only that the prediction is close to $\mathbb{E}[c(z;y) \,|\, x]$, but also that it is uniformly close across z. We do this by leveraging the fact that Rademacher complexity theory gives us uniform bounds across classes of functions.

Assume $\{(x^i, y^i)\}_{i=1}^n \in \mathcal{X} \times \mathcal{Y}$ are independent and identically distributed according to some unknown distribution. Let (x, y) be a representative vector from that distribution. Let S denote the entire sample set $\{(x^i, y^i)\}_{i=1}^n$. Let $\mathcal{Z}$ be the set of possible decisions. Let $K(\cdot,\cdot)$ be a positive-definite kernel and let $\mathcal{H}$ be the associated reproducing kernel Hilbert space.

The goal is to show that the approximation produced by the objective prediction method is close to $\mathbb{E}[c(z;y) \,|\, x]$. The approximation produced by the objective prediction method is the solution to (8). It is well-known that because of Lagrangean duality,

$$\min_{h(z,\cdot)\in\mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (h(z,x^i) - c(z;y^i))^2 + \lambda \|h(z,\cdot)\|_{\mathcal{H}}^2 ,$$

is equivalent to

$$\begin{aligned} \min_{h(z,\cdot)\in\mathcal{H}} \ & \frac{1}{n}\sum_{i=1}^{n}(h(z,x^i)-c(z;y^i))^2 \\ \text{s.t. } & \|h(z,\cdot)\|_{\mathcal{H}}^2 \leq \eta. \end{aligned} \tag{11}$$

for some mapping between λ and η (for a proof, see Oneto et al. 2016). Let $h_{S,\eta}$ be the solution of Problem (11). The next proposition shows that $h_{S,\eta}$ is close to $\mathbb{E}[c(z;y)\,|\,x]$ uniformly over z, in the sense that its distance from $\mathbb{E}[c(z;y)\,|\,x]$ is almost as small as the function in $\mathcal{H}$ with the smallest distance from $\mathbb{E}[c(z;y)\,|\,x]$ that satisfies $\|h\| \leq \eta$, which we call h_η^*. In other words, the proposition shows that the approximation produced by the objective prediction method, $h_{S,\eta}$, is almost as close to $\mathbb{E}[c(z;y)\,|\,x]$ as the best possible approximation h_η^*, uniformly over z.

Proposition 2 **(Finite sample error bound for objective prediction method)**. *Let*

$$h_{S,\eta}(z,\cdot) = \operatorname*{arg\,min}_{h(z,\cdot)\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}(h(z,x^i)-c(z;y^i))^2$$

$$s.t.\ \|h(z,\cdot)\|_{\mathcal{H}}^2 \leq \eta.$$

and let

$$h_\eta^*(z,\cdot) = \operatorname*{arg\,min}_{h(z,\cdot)\in\mathcal{H}} \mathbb{E}_{x,y}\left[\left(h(z,x)-\mathbb{E}[c(z,y)\,|\,x]\right)^2\right]$$

$$s.t.\ \|h(z,\cdot)\|_{\mathcal{H}}^2 \leq \eta.$$

Assume the following.

1. *For some* κ, $\sup_{x\in\mathcal{X}} K(x,x) = \kappa^2 < \infty$.
2. *For some* C, $\sup_{y\in\mathcal{Y},z\in\mathcal{Z}} |c(z;y)| = C < \infty$.

Then, with probability $1-\delta$, *for all* $z\in\mathcal{Z}$ *simultaneously,*

$$\mathbb{E}_x\left[(h_{S,\eta}(z,x)-\mathbb{E}[c(z,y)\,|\,x])^2\right] - \mathbb{E}_x\left[\left(h_\eta^*(z,x)-\mathbb{E}[c(z,y)\,|\,x]\right)^2\right] \leq M_1\frac{1}{\sqrt{n}} + M_2\sqrt{\frac{\log 2/\delta}{2n}},$$

where $M_1 = 4(\kappa\sqrt{\eta}+C)\kappa\sqrt{\eta}$ *and* $M_2 = 2(\kappa\sqrt{\eta}+C)^2$.

Proposition 2 is proved in Section EC.2 of the electronic companion to this paper. It is an application of existing results about Rademacher complexity combined with a rewriting of the square loss $\mathbb{E}_{x,y}\left[\left(h(z,x)-c(z;y)\right)^2\right]$ as a distance from the true conditional expectation.

Both assumptions of Proposition 2 are boundedness conditions. The first assumption would be satisfied if $\mathcal{X}$ was compact and K was continuous or if K was a bounded kernel such as the Gaussian kernel. The second assumption would be satisfied if $\mathcal{Y}$ and $\mathcal{Z}$ were compact and c was continuous or if c was bounded. Philosophically, these boundedness conditions are not restrictive because arbitrarily large quantities do not exist in the real world applications. Nevertheless, they do create large constants that preclude the usefulness of Proposition 2 in any real finite sample setting. Proposition 2 is, however, useful for telling us the rate at which the distance between the objective approximation and the true objective goes to zero, which is the standard statistical learning rate of $\sqrt{\frac{\log(1/\delta)}{n}}$.

Next, we show asymptotic optimality. Proposition 3 formalizes that the approximation produced by the objective prediction method converges to $\mathbb{E}[c(z;y)\,|\,x]$ when the space $\mathcal{H}$ is large enough and we have enough sample points. Recall that the approximation produced by the objective prediction method is

$$\min_{h(z,\cdot)\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n}(h(z,x^i)-c(z;y^i))^2$$
$$\text{s.t. } \|h\|_{\mathcal{H}}^2 \leq \eta\,,$$

given the appropriate conversion between λ and η. To prove asymptotic optimality, we specialize to the case that $\mathcal{H}$ is generated by a polynomial kernel, so that $\mathcal{H}$ is the set of all polynomials up to a certain degree.

PROPOSITION 3 **(Asymptotic optimality of objective prediction method)**. *Let $\mathcal{H}_d$ be the space of polynomials of degree d and K_d be the polynomial kernel, $K_d(x,x') = (x^Tx'+\alpha)^d$ with $\alpha > 1$. Let*

$$h_{S,d,\eta}(z,\cdot) = \operatorname*{arg\,min}_{h(z,\cdot)\in\mathcal{H}_d} \frac{1}{n}\sum_{i=1}^{n}(h(z,x^i)-c(z;y^i))^2$$
$$s.t.\ \|h\|_{\mathcal{H}_d}^2 \leq \eta\,,$$

let

$$z_{S,d,\eta}(x) = \operatorname*{arg\,min}_{z\in\mathcal{Z}} h_{S,d,\eta}(z,x)\,,$$

and let

$$z^*(x) = \arg\min_{z \in \mathcal{Z}} \mathbb{E}[c(z;y) \,|\, x] \,.$$

We assume the following.

1. *For some C, $\sup_{y \in \mathcal{Y}, z \in \mathcal{Z}} |c(z;y)| = C < \infty$.*
2. *The function $\mathbb{E}[c(z;y) \,|\, x]$ is differentiable over $x \in \mathcal{X}$, and its partial derivative with respect to x is continuous over $x \in \mathcal{X}$ and $z \in \mathcal{Z}$.*
3. *The sets $\mathcal{X}$ and $\mathcal{Z}$ are closed and bounded.*
4. *The variable η is strictly increasing in d.*
5. *For every $x^0 \in \mathcal{X}$ and $\gamma > 0$, $\mathbb{P}_x(|x - x^0| < \gamma) > 0$.*

Then for any $x^0 \in \mathcal{X}$, the cost of $z_{S,d,\eta}(x^0)$ converges to the cost of $z^*(x^0)$ in probability. More specifically, for any $x^0 \in \mathcal{X}$ and for every $\epsilon > 0$ and $\delta > 0$, there is some D and N such that for all $d \geq D$ and $n \geq N(d)$, $\mathbb{P}_S \left(\left| \mathbb{E}[c(z_{S,d,\eta}(x^0); y) \,|\, x = x^0] - \mathbb{E}[c(z^*(x^0); y) \,|\, x = x^0] \right| > \delta \right) < \epsilon$.

We relegate the proof to Section EC.2 of the electronic companion to this paper. It involves using the Weierstrass Approximation Theorem to approximate $\mathbb{E}[c(z;y) \,|\, x]$ by a polynomial and then using Proposition 2 to show convergence to the approximation.

The assumptions of Proposition 3 are stronger than that of Proposition 2. Proposition 3 requires $\mathcal{X}$ and $\mathcal{Z}$ to be compact and $\mathbb{E}[c(z;y) \,|\, x]$ to be continuous. This is in order to use the Weierstrass Approximation Theorem. Additional regularity conditions on $\mathbb{E}[c(z;y) \,|\, x]$ and the distribution of x enable the conversion of the bound in Proposition 2, which is an average over x, to a bound that holds for each x. Proposition 3 says that under these assumptions, and as the degree of the polynomial kernel, η, and n increase, the decision produced by the objective prediction method converges to the true optimal decision.

5. Optimizer Prediction Method

In this section, we develop the optimizer predictor method starting from (1) using basic probability theory and the concepts from Section 3. We recognize that solving

$$\min_{z \in \mathbb{R}^d} \mathbb{E}[c(z;y) \,|\, x = x^0]$$

is equivalent to finding the vector-valued function $\widehat{z}(\cdot)$ that solves

$$\min_{z(\cdot)} \mathbb{E}[c(z(x);y)] \tag{12}$$

and evaluating $\widehat{z}$ at x^0. Now the original problem has been reduced to the standard machine learning framework of finding a function that minimizes some loss functional. Similarly to the previous method, we can find a good approximation of the minimizer of (12) by solving the empirical regularized version of (12) and restricting each component of $z(\cdot)$ to be in a reproducing kernel Hilbert space (associated with, say, a kernel K):

$$\min_{z^1(\cdot),\ldots,z^d(\cdot)\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} c(z^1(x^i),\ldots,z^d(x^i);y^i) + \lambda \sum_{t=1}^{d} \|z^t\|_{\mathcal{H}}^2 . \tag{13}$$

We solve the regularized empirical problem (13) by using a multidimensional version of the Representer Theorem (Proposition EC.2 in the electronic companion), which implies that the solution to (13) takes the form

$$\widehat{z}^t(x) = \sum_{i=1}^{n} K(x^i,x)a_i^t .$$

for some scalars a_i^t. By plugging this form back into (13), we obtain that the optimal solution to (13) is the above $\widehat{z}(\cdot)$, where a is the solution to

$$\min_{a^1,\ldots,a^d\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} c((\widehat{K}a^1)_i,\ldots,(\widehat{K}a^d)_i;y^i) + \lambda \sum_{t=1}^{d} (a^t)^T \widehat{K} a^t . \tag{14}$$

and where $\widehat{K}_{ij}$ is $K(x^i,x^j)$.

Note that if K is the linear kernel and $\lambda = 0$, then the optimizer prediction method is equivalent to a linear decision rule approach that is used by Ban and Rudin (2019) and Bertsimas and Kallus (2020, Electronic Companion). Similarly if K was the polynomial kernel, it would be equivalent to a polynomial decision rule approach. However, both the generality of K and the inclusion of a regularization term are crucial for the reformulation in Section 5.2 and the theoretical results in Section 5.3, including asymptotic optimality.

5.1. Tractability

We obtain an optimal decision by solving (14). Assuming c is convex, $c((\widehat{K}a^1)_i,\ldots,(\widehat{K}a^d)_i;y^i)$ is convex in $a^1,\ldots,a^d$ because the composition of a convex function with an affine function is convex. Since $\widehat{K}$ is positive semi-definite, $\lambda\sum_{t=1}^d (a^t)^T\widehat{K}a^t$ is convex quadratic in $a^1,\ldots,a^d$, and therefore (14) is a convex optimization problem in $a^1,\ldots,a^d$. In other words, we solve (1) by solving a convex optimization problem, (14), in nd variables.

As we did in Section 4, we provide a rough Big-O analysis of running time. Assume that $\min_z \sum_{i=1}^n w^i c(z;y^i)$ can be reformulated into an LOP with $D(n,d)$ variables, and that $\min_{a^1,\ldots,a^d} \frac{1}{n}\sum_{i=1}^n c((\widehat{K}a^1)_i,\ldots,(\widehat{K}a^d)_i;y^i)$ can be reformulated as an LOP with $D(n,nd)$ variables. Assume that it takes $O(k^3\log(1/\epsilon))$ time to find ϵ-optimal solutions for linear and quadratic optimization problems with k variables. Then the kernel optimizer prediction method takes $O(n^2d)$ time to compute the kernel matrix, $O(D(n,nd)^3\log(1/\epsilon))$ time to find an ϵ-optimal solution to the resulting LOP and $O(ndn_{\text{test}})$ time to find the final decisions. So the entire procedure takes a total of $O(n^2d+D(n,nd)^3\log(1/\epsilon)+ndn_{\text{test}})$ time. Again, for comparison, k-nearest neighbors from Bertsimas and Kallus (2020) takes $O(ndn_{\text{test}}+D(n,d)^3n_{\text{test}}\log(1/\epsilon))$ time. If $D(n,d)$ is $O(1)$ in n and $O(d)$ in d, then the kernel optimizer method takes around a factor of n times longer than k-nearest neighbors. However, if $D(n,d)$ is $O(n)$ in n (which is the case for a two-stage linear cost function, as we see in Section 7), then the kernel optimizer method is actually a factor of n_{test} times faster than k-nearest neighbors.

5.2. Alternative Formulation to Improve Numerical Performance

We find that (14), while convex, has subpar scaling in n and gives solvers such as Gurobi numerical trouble. The following is an alternative formulation that has excellent scaling and seemingly no numerical issues. Let $z^t=\widehat{K}^{-1}a^t$ for $t=1,\ldots,d$. Then problem (14) becomes

$$\min_{z^1,\ldots,z^d}\frac{1}{n}\sum_{i=1}^n c(z_i^1,\ldots,z_i^d;y^i)+\lambda\sum_{t=1}^d (z^t)^T\widehat{K}^{-1}z^t\,. \tag{15}$$

Compute the eigenvalue decomposition of $\widehat{K}$, $V\Sigma V^T$, choose a small quantity λ_{spec}, and approximate (15) as

$$\begin{aligned} \min_{z^1,\ldots,z^d} \ & \frac{1}{n}\sum_{i=1}^{n} c(z_i^1,\ldots,z_i^d;y^i) + \lambda \sum_{t=1}^{d} (\theta^t)^T\theta^t \\ \text{s.t. } & (\Sigma+\lambda_{\text{spec}}I)^{-1/2}V^T z^t = \theta^t, \quad t=1,\ldots,d \end{aligned} \tag{16}$$

In the largest computational example we considered in Section 7, formulation (16) solves in 40 seconds with $n=800$ (around 90,000 decision variables), and formulation (14) does not finish solving in 2 hours with $n=300$ (around 35,000 decision variables). One might wonder whether this improvement is due to the change of variables or due to the spectral regularization, i.e., the process of going from (15) to (16). Empirically, spectral regularization without the change of variables does not help.

The alternative formulation also leads to an interesting perspective on the optimizer prediction method. Take the case when z is univariate for simplicity. The alternative formulation tells us that

$$\min_z \mathbb{E}[c(z;y) \,|\, x = x^0]$$

can be solved by instead solving

$$\min_{z_1,\ldots,z_n\in\mathbb{R}} \frac{1}{n}\sum_{i=1}^{n} c(z_i;y^i) + \lambda z^T \widehat{K}^{-1} z$$

and taking the decision $z^0 = z^T\widehat{K}^{-1}K(X,x^0)$. If we took $\lambda=0$, the decision at point x^0 would be found by finding the optimal decisions z_i for the observed values of y_i, and weighting them according to $\widehat{K}^{-1}K(X,x^0)$. This is the same as regressing $\arg\min_z c(z;y^i)$ onto x^i. But when $\lambda>0$, the penalty term $\lambda z^T\widehat{K}^{-1}z$ binds the z_i together and prevents them from being strictly optimal for the respective y_i.

5.3. Theoretical Guarantees

We now derive a finite sample error bound for the optimizer prediction method. We do this by viewing the problem solved by the optimizer prediction method, (13), as a regression problem with

an arbitrary convex loss function. Existing machine learning results can bound the error of the optimizer prediction method when the dimension of z is 1. We extend these results to bound the error in the multidimensional case. After we derive the finite sample error bound, we use it to prove asymptotic optimality.

Assume $(x^i, y^i)_{i=1}^n \in \mathcal{X} \times \mathcal{Y}$ are independent and identically distributed according to some unknown distribution. Let (x, y) be a representative random vector from the same distribution. Let S denote the entire sample set $(x^i, y^i)_{i=1}^n$. Let $K(\cdot, \cdot)$ be a positive-definite kernel and let $\mathcal{H}$ be the associated reproducing kernel Hilbert space. Let $\mathcal{H}^d$ be Hilbert space that is the Cartesian product of d copies of $\mathcal{H}$, with inner product defined as

$$\langle (z^1, \ldots, z^d), (\tilde{z}^1, \ldots, \tilde{z}^d) \rangle_{\mathcal{H}^d} = \langle z^1, \tilde{z}^1 \rangle_{\mathcal{H}} + \ldots + \langle z^d, \tilde{z}^d \rangle_{\mathcal{H}} .$$

PROPOSITION 4 **(Finite sample error bound for optimizer prediction method)**. *Let z_S^λ be the solution to*

$$\min_{z(\cdot) \in \mathcal{H}^d} \frac{1}{n} \sum_{i=1}^n c(z(x^i); y^i) + \lambda \|z\|_{\mathcal{H}^d}^2 .$$

Assume the following.

1. *For some L and any $y \in \mathcal{Y}$, $c(\cdot; y)$ is Lipschitz with constant L.*
2. *For any $y \in \mathcal{Y}$, $c(\cdot; y)$ is convex.*
3. *For some κ, $\sup_{x \in \mathcal{X}} K(x, x) \leq \kappa^2$.*
4. *For some $C_0 \geq 0$ and any $y \in \mathcal{Y}$, $c(0; y) \leq C_0$.*

Then with probability $1 - \delta$, if $\lambda > 0$,

$$\mathbb{E}[c(z_S^\lambda(x); y)] \leq \frac{1}{n} \sum_{i=1}^n c(z_S^\lambda(x^i); y^i) + \frac{L^2 \kappa^2}{\lambda n} + \left(\frac{2L^2\kappa^2}{\lambda} + C_0 + \kappa L \sqrt{\frac{C_0}{\lambda}} \right) \sqrt{\frac{2 \log(2/\delta)}{n}}$$

The proof is in Section EC.2 of the electronic companion to this paper. The idea is to show that when $\lambda > 0$, z_S^λ is stable, meaning that it does not change too much if the data changes a little bit. Then we can use the stability property to bound the expected cost through McDiarmid's inequality.

Proposition 4 assumes a regularity condition on c (Conditions 1, 2, and 4) and a boundedness condition on $\mathcal{X}$ and $\mathcal{Y}$ (Conditions 3 and 4) and shows that this implies a bound on the expected cost

of the decision rule produced by the optimizer prediction method. Condition 2, the assumption that $c(\cdot; y)$ is convex for each y, is crucial because it allows us to characterize the minimizer. However, it is necessary anyway for tractability. Condition 1, Lipschitz continuity, is also reasonable. It would be satisfied if, for example, $c(\cdot; y)$ was piecewise linear and convex with bounded slopes. Condition 3 would be satisfied if $\mathcal{X}$ was compact. It would also always be satisfied for the Gaussian kernel. Condition 4 would be satisfied if c was continuous and $\mathcal{Y}$ was bounded, if c was bounded, or simply if taking the decision $z = 0$ meant that nothing happened, like in the newsvendor problem. Just like the last section, the rate at which the error decreases is the standard machine learning rate of $\sqrt{\frac{\log(1/\delta)}{n}}$.

We now use Proposition 4 to prove asymptotic optimality. Proposition 4 suggests that making the empirical cost small should also make the expected cost small. So we expect that if the objective prediction method asymptotically minimizes the empirical cost, it should asymptotically minimize the expected cost. In the next proposition we formalize this intuition, showing that the decision rule produced by the optimizer prediction method, z_S^λ, converges to the true optimal decision rule under the additional assumptions that $\mathcal{X}$ is compact, $\arg\min_z \mathbb{E}[c(z; y) \,|\, x]$ is continuous, K is a universal kernel like the Gaussian kernel, and that λ goes to 0 at the appropriate rate. (See Micchelli et al. (2006) for more information on universal kernels, but for our purposes it suffices to know that when $\mathcal{X}$ is compact, K being a universal kernel means that $\mathcal{H}$ is dense in the set of continuous functions.)

PROPOSITION 5 **(Asymptotic optimality of optimizer prediction method)**. *Let z_S^λ be the solution to*

$$\min_{z(\cdot)\in\mathcal{H}^d} \frac{1}{n}\sum_{i=1}^{n} c(z(x^i); y^i) + \lambda \|z\|^2_{\mathcal{H}^d}\,.$$

Assume the conditions of Proposition 4 are satisfied. Also assume the following.

1. *$\mathcal{X}$ is closed and bounded.*
2. *There is a continuous function $z^*(x)$ such that for all $x \in \mathcal{X}$, $z^*(x) = \arg\min_z \mathbb{E}[c(z; y) \,|\, x]$.*
3. *The kernel K is a universal kernel such as the Gaussian kernel or the exponential kernel.*
4. *As $n \to \infty$, $\lambda \to 0$ and $\frac{\lambda^2 n}{\log(1/\lambda)} \to \infty$.*

then $\mathbb{E}[c(z_S^\lambda(x);y)\,|\,x] \to \min_{z\in\mathbb{R}^d} \mathbb{E}[c(z;y)\,|\,x]$ *in probability except on a subset of* $\mathcal{X}$ *of zero measure.*

The proof is in Section EC.2 of the electronic companion to this paper. The condition that as $n\to\infty$, $\lambda\to 0$ and $\frac{\lambda^2 n}{\log(1/\lambda)} \to \infty$ is satisfied, for example, if $\lambda = \frac{1}{\sqrt[3]{n}}$.

6. Extensions

In this section, we present several extensions of the basic methodology we presented in earlier sections.

6.1. Objective Prediction with Constraints

Suppose that in Problem (1), we need z to satisfy the convex constraints

$$f_j(z) \le 0, \quad j \in J\,.$$

Because the objective prediction was developed with the requirement that z was restricted to be in $\mathcal{Z}$, this is simply a special case of the objective prediction section where $\mathcal{Z} = \{z\,|\,f_j(z)\le 0, j\in J\}$.

6.2. Optimizer Prediction with Constraints

Suppose that in Problem (1), we need z to satisfy the convex constraints

$$f_j(z) \le 0, \quad j \in J\,.$$

One approach is to take the reformulation of the kernel optimizer method (16), and simply add the constraints $f_j(z_i)\le 0$, $\forall j\in J$:

$$\begin{aligned}\min_{z^1,\ldots,z^d}\ & \frac{1}{n}\sum_{i=1}^n c(z_i^1,\ldots,z_i^d;y^i) + \lambda\sum_{t=1}^d (\theta^t)'\theta^t\\ \text{s.t. } & F^T z^t = \theta^t, \quad t=1,\ldots,d\\ & f_j(z_i)\le 0, \quad j\in J\,.\end{aligned}$$

Find $a = \widehat{K}^{-1}z$ and $z^0 = a^T K(X,x^0)$ as usual. If z^0 is not feasible, then project it onto the feasible region by solving the convex optimization problem

$$\begin{aligned}\min_{z\in\mathbb{R}^d}\ & \|z-z^0\|^2\\ \text{s.t. } & f_j(z)\le 0\ \ j\in J\,.\end{aligned}$$

Although this approach is intuitively appealing, it does not come with theoretical guarantees. The following approach does. Relax the constraints into the objective by letting

$$\Phi(z;y) = c(z;y) + \psi \sum_{j\in J} \max(0, f_j(z))^2 .$$

Instead of solving (1), we solve

$$\min_{z\in\mathbb{R}^d} \mathbb{E}_y \left[\Phi(z;y) \,|\, x = x^0 \right] ,$$

which converges to (1) as ψ goes to infinity.

All of the results from Section 5 go through when c is replaced by Φ. In particular, if $c(z;y)$ is convex in z for all y, then $\Phi(z;y)$ is convex in z for all y, and we obtain the decision z^0 by solving a convex optimization problem. The error bounds and convergence results also go through with c replaced by Φ. We rewrite the result for asymptotic convergence here for clarity.

COROLLARY 1 **(Corollary of Proposition 5)**. *Let $z_S^{\lambda,\psi}$ be the solution to*

$$\min_{z(\cdot)\in\mathcal{H}^d} \frac{1}{n} \sum_{i=1}^{n} \left(c(z(x^i);y^i) + \psi \sum_{j\in J} \max(0, f_j(z(x^i)))^2 \right) + \lambda \|z\|^2_{\mathcal{H}^d} .$$

Assume the following.

1. *For some L and for any $y \in \mathcal{Y}$, $c(\cdot;y)$ is Lipschitz with constant L.*
2. *For any $y \in \mathcal{Y}$, $c(\cdot;y)$ is convex.*
3. *For some κ, $\sup_{x\in\mathcal{X}} K(x,x) \le \kappa^2$.*
4. *For some C_0 and any $y \in \mathcal{Y}$, $c(0;y) \le C_0$.*
5. *$\mathcal{X}$ is closed and bounded.*
6. *For any ψ, there is a continuous function $z^*(x)$ such that for all $x \in \mathcal{X}$, $z^*(x) = \arg\min_z \mathbb{E}[c(z;y) \,|\, x] + \psi \sum_{j\in J} \max(0, f_j(z))^2$.*
7. *The kernel K is a universal kernel such as the Gaussian kernel or the exponential kernel.*
8. *As $n \to \infty$, $\lambda \to 0$ and $\frac{\lambda^2 n}{\log(1/\lambda)} \to \infty$.*
9. *The functions f_j for $j \in J$ are convex, Lipschitz, and bounded.*

Then, for any ψ,

$$\mathbb{E}[c(z_S^{\lambda,\psi}(x);y)\,|\,x] + \psi \sum_{j\in J} \max(0, f_j(z_S^{\lambda,\psi}(x)))^2$$

converges to

$$\min_{z\in\mathbb{R}^d} \mathbb{E}[c(z;y)\,|\,x] + \psi \sum_{j\in J} \max(0, f_j(z))^2$$

in probability except on a subset of $\mathcal{X}$ of zero measure.

Proof of Corollary 1. We apply Proposition 5 with $c(z;y)$ in Proposition 5 replaced by $c(z;y) + \psi \sum_{j\in J} \max(0, f_j(z))^2$ in Corollary 1. One can check that Condition 9 guarantees that $c(z;y) + \psi \sum_{j\in J} \max(0, f_j(z))^2$ is Lipschitz continuous with bounded constant across y. □

Even though the approximation $\widehat{z}(x)$ converges to the true feasible minimizer of (1) for all x asymptotically, for a finite sample, $\widehat{z}(x^0)$ may not be feasible, in which case we can select the decision to be the projection of $\widehat{z}(x^0)$ onto the feasible region as described above.

6.3. Objective Prediction when Decision Affects Parameter

In this section, we assume that the cost parameter y is actually a function of z. Moreover, in addition to the data $(x^1, y^1), \ldots, (x^n, y^n)$, we have data on past decisions taken $z^1, \ldots, z^n$.

In this case, the problem can be solved by considering $z^1, \ldots, z^n$ as part of the data. We replace the matrix of historical auxiliary data, X, by $[X, Z]$, the historical auxiliary data concatenated with the historical decisions. The objective of (1) is then approximated as

$$h(z,x) = c(z;Y)^T(\widehat{K} + \lambda n I)^{-1} K([X,Z],[x,z]) \tag{17}$$

where now x and z are used to compute the kernel matrix, i.e.,

$$\widehat{K}_{ij} = K([x^i, z^i], [x^j, z^j])\,.$$

Minimizing over z in (17) is a smooth nonlinear optimization problem in z. A local minimum can be found efficiently with gradient descent or, if there are constraints, the conditional gradient method. This is a potential improvement over Bertsimas and Kallus (2020) who approach the case where the decision affects the parameter by discretization.

6.4. Multistage Data-Driven Optimization

We briefly describe how we can apply the framework of this paper to the m-stage optimization problem

$$\min_{z_1, z_2(\cdot), \ldots, z_m(\cdot,\ldots,\cdot)} \mathbb{E}_y \left[Q\left(z_1, z_2(y_1), \ldots, z_m(y_1, \ldots, y_{m-1}), y_1, \ldots, y_T\right)\right], \tag{18}$$

where Q is a cost function, z_1 is our decision to be made now, and $z_2, \ldots, z_m$ are decisions to be made at future times that depend on future uncertainty. We assume that we have data $\{y_1^i, \ldots, y_T^i\}_{i=1}^n$. For simplicity of presentation, we assume that the decisions are univariate and there is no auxiliary data, but the extension to the multivariate case with auxiliary data should be apparent.

Let K^t be a positive-definite kernel $\mathcal{Y}^{t-1} \times \mathcal{Y}^{t-1} \to \mathbb{R}$. Let $\mathcal{H}^t$ be the reproducing kernel Hilbert space generated by the kernel K^t. We consider the empirical regularized version of the problem (18):

$$\min_{z_1 \in \mathbb{R}, z_2 \in \mathcal{H}^2, \ldots, z_m \in \mathcal{H}^T} \frac{1}{n} \sum_{i=1}^n Q\left(z_1, z_2(y_1^i), \ldots, z_m(y_1^i, \ldots, y_{m-1}^i), y_1^i, \ldots, y_m^i\right) + \sum_{t=1}^m \lambda_t \|z_t\|_{\mathcal{H}^t}^2 \tag{19}$$

This is a functional minimization problem, but using the Representer Theorem, we know that there exists a solution to it of the form

$$z_t(\cdot) = \sum_{i=1}^n a_t^i K^t(y^i, \cdot), \tag{20}$$

for some scalars a_t^i with $i = 1, \ldots, n$ and $t = 1, \ldots, m$. Plugging this form of z_t back into (19), we obtain the finite-dimensional optimization problem

$$\min_{z_1, a_2, \ldots, a_m} \frac{1}{n} \sum_{i=1}^n Q\left(z_1, (v_1^i)^T a_1, \ldots, (v_T^i)^T a_m, y_1^i, \ldots, y_m^i\right) + \sum_{t=1}^m \lambda_t a_t^T \hat{K}^t a_t \tag{21}$$

where $\hat{K}^t$ is the $n \times n$ matrix with $\hat{K}_{ij}^t = K^t(y_{0,t-1}^i, y_{0,t-1}^j)$, and v_i^t is the vector given by the ith row of this matrix. If Q is convex for fixed y, then this a convex optimization problem. The asymptotic optimality of this approach follows by a similar logic to the proof of asymptotic optimality in Section 5.

Table 1 Methods Implemented for Computational Results

abbreviation	explanation of method
`cart`	CART from Bertsimas and Kallus (2020)
`kerobj`	kernel objective prediction from Section 4
`keropt`	kernel optimizer prediction from Section 5
`nn`	k-nearest neighbors from Bertsimas and Kallus (2020)
`pp`	point prediction (predict y then optimize) using linear regression
`ppnn`	point prediction (predict y then optimize) using k-nearest neighbors
`rf`	random forest from Bertsimas and Kallus (2020)
`saa`	sample average approximation
`simopt`	simulated optimal decision using true conditional distribution

7. Computational Results

In this section, we compare the performance of the kernel objective prediction method from Section 4 and the kernel optimizer prediction method from Section 5 against several benchmarks (Table 1), including the strongest prescriptive methods from Bertsimas and Kallus (2020). We use two different problems of the form

$$\min_{z \in \mathcal{Z}} \mathbb{E}[c(z;y) \,|\, x]$$

with simulated training and test data. For each problem we report the performance of the methods listed in Table 1 using the abbreviations listed in the same table. For each method, instead of the mean cost of its decisions averaged over a test set C, we report the relative cost $\frac{C_{\text{saa}} - C}{C_{\text{saa}} - C_{\text{simopt}}}$, where C_{saa} is the mean cost of the decisions produced by `saa`, and C_{simopt} is the mean cost of the decisions produced by `simopt`. The relative cost is easier to interpret. SAA has a relative cost of 0, the true optimum decision has a relative cost of 1, and if a method has a relative cost of p, it has closed proportion p of the gap between SAA and the true optimum. For each figure, we also report for each instance the signal-to-noise ratio, as measured by the R^2 of the true conditional expectation $\mathbb{E}[y \,|\, x]$, and the prescriptive analog of the signal-to-noise ratio, defined as $\frac{C_{\text{saa}} - C_{\text{simopt}}}{C_{\text{saa}} - C_{\text{postopt}}}$, where C_{postopt} is the mean cost of the optimal decisions made when knowing the true realization of each y in the test set.

In the interest of keeping the simulations as fair as possible to all methods, we implement a systematic way of choosing hyperparameters. For each training set, we generate a validation set of the same size as the training set (with the size of the validation set capped at 100 samples), and

Table 2 Method Hyperparameters

abbreviation	hyperparameters
`cart`	depth, minimum samples to split, minimum samples per leaf
`kerobj`	γ of Gaussian kernel, regularization weight λ
`keropt`	γ, λ, ψ (see Section 6.2), λ_{spec} (see Section 5.2)
`nn`	number of nearest neighbors k
`pp`	None
`ppnn`	number of nearest neighbors k
`rf`	depth, minimum samples to split, minimum samples per leaf
`saa`	None
`simopt`	None

for each method we choose the hyperparameters that minimizes decision cost on the validation set. In Table 2, we list all of the hyperparameters we tune for each method. For `cart`, `nn`, `ppnn`, and `rf`, we attempt to do a comprehensive search over all reasonable hyperparameter choices. For `nn` and `ppnn`, we search over 50 values of k from 1 to $n/2$. For `cart` and `rf`, we search over 72 to 192 combinations of the depth from 1 to $\log_2(n) - 1$, minimum samples to split from 1 to n, and minimum samples per leaf from 1 to $n/2$. For `kerobj` and `keropt`, there are no natural bounds on the hyperparameter search space, so we need to find them manually. We do this by trying a wide range of parameters on a log scale on a small instance each of the newsvendor and product placement problems (described below), and expanding or reducing the range as necessary. We end up with two ranges for all of the computational instances in the paper, one range for all of the newsvendor problem instances and one range for all of the product placement problem instances. We search over no more than 100 combinations of parameters within these ranges.

In order to implement `cart`, `pp`, `ppnn`, and `rf`, we rely on the popular Python package scikit-learn version 0.21.2. To model all optimization problems, we use JuMP version 0.19.2. To solve all optimization problems, we use Gurobi version 8.1.1. For `kerobj` and `keropt`, we have a choice of kernel to use. We use the Gaussian kernel. For `kerobj`, we use the method described in Section 4 and not the more complex generalization in the electronic companion. For `keropt`, we use the reformulation in Section 5.2 and the relaxation approach of dealing with constraints in Section 6.2. All test sets contain 100 points. Each point in each figure is the average over 30 different training and test sets.

7.1. Newsvendor Problem

In the newsvendor problem, we need to decide each day how many newspapers to order. The amount of profit we make depends on the number of newspapers we order, but it also depends on the number of people who buy newspapers from us, a quantity which we will call y. In particular, we have a unit cost d of buying a newspaper and a unit revenue r of selling a newspaper.

We do not observe y before we order the newspapers, but we do observe some quantities that y may be correlated with, such as the weather. We summarize the quantities we can observe before ordering as x. The problem we address is then

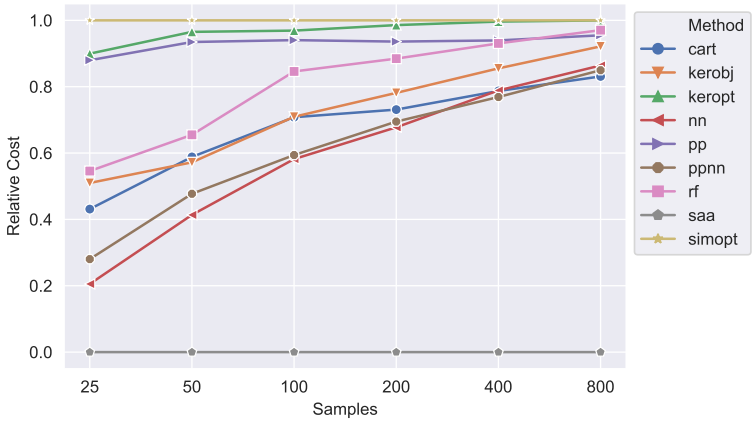
$$\min_{z \geq 0} \mathbb{E}[dz - r \min(z, y) \,|\, x].$$

We assume we have data $x^1, \ldots, x^n$ and $y^1, \ldots, y^n$.

Since we use simulated data, we have to make modeling decisions for x and y. We assume x_1 is log-normal and is supposed to represent the "newsworthiness" of the day. We assume x_2 is normal and is supposed to represent the temperature. We assume $y = \max(\beta_1 x_1 + \beta_2 x_2 + \epsilon, 0)$, where ϵ is exponentially distributed noise. We choose β_1 and β_2 so that y hovers around 100. Finally, we choose the unit cost as $d = 0.5$ and unit revenue as $r = 1$.

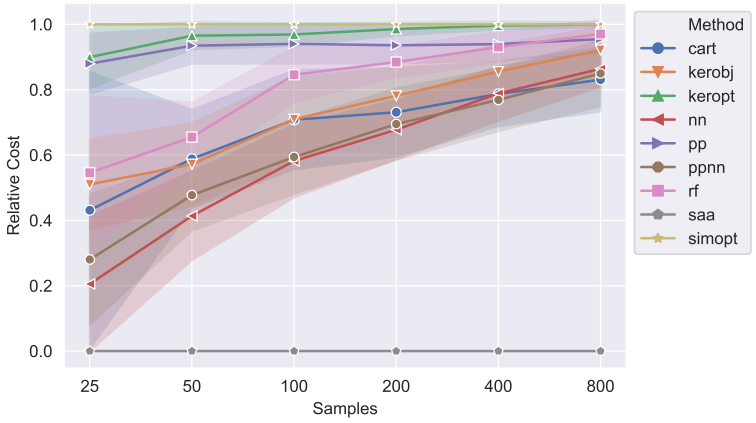
To compare the performance of the methods listed in Table 1, we will take Figure 1 as the baseline (with Figure 2 showing standard deviations). We will note how all other figures differ from the baseline. In the baseline, the kernel optimizer prediction method `keropt` with $n = 50$ performs better than the best method from Bertsimas and Kallus (2020) with $n = 800$. The kernel objective prediction method `kerobj` performs worse than `rf` but better than `nn`. The method `keropt` converges to the optimum, and `kerobj` appears on track to converge. Point prediction using linear regression `pp` works quite well, but this can be attributed to the true model being linear, as point prediction using nearest neighbors `ppnn` does much worse. These facts remain true if we increase the signal-to-noise ratio (Figure 3) or decrease it (Figure 4). The results are also stable when increasing the nonlinearity in the cost function by making $r = 4$ instead of $r = 1$ (Figure 5).

Figure 1 Newsvendor Problem (Baseline): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.84. Prescriptive signal-to-noise ratio is 0.55.

Figure 2 Newsvendor Problem (Baseline): Relative Cost vs. Samples

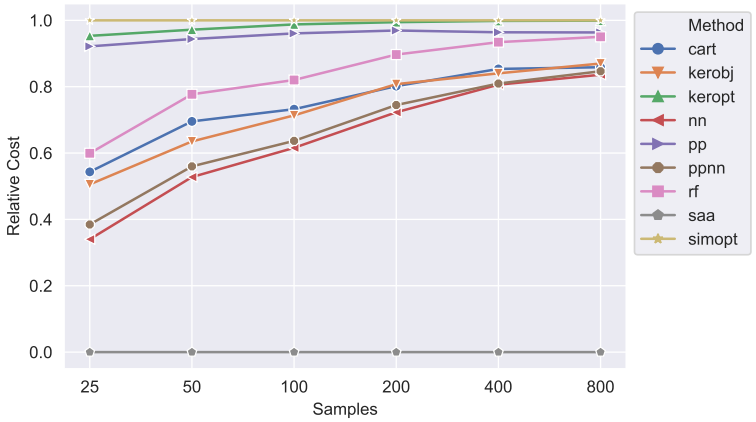


Note. Signal-to-noise ratio is 0.84. Prescriptive signal-to-noise ratio is 0.55.

Note that while absolute signal-to-noise ratios may seem high for the newsvendor problem, the newsvendor objective function amplifies the effect of noise, and the prescriptive signal-to-noise ratios are more reasonable.

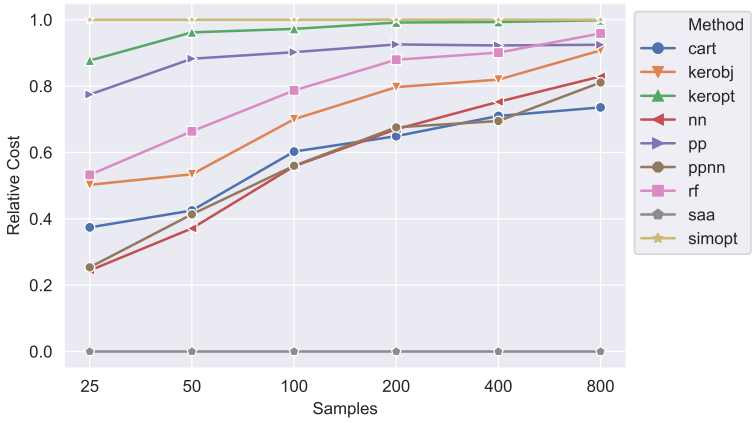
7.1.1. Running Time In Figure 6, we plot the mean running times for each method in the baseline setting. The running time is calculated as the sum of the training and test times after cross-validation has chosen hyperparameters. The test set size is fixed at 100 while the training set size varies. We also exclude running times for $n = 25$ because they are distorted by startup costs. We see that `keropt` consistently takes around 10 times longer to run than `nn`, and `kerobj` consistently

Figure 3 Newsvendor Problem (Increased Signal-to-Noise Ratio): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.93. Prescriptive signal-to-noise ratio is 0.67.

Figure 4 Newsvendor Problem (Decreased Signal-to-Noise Ratio): Relative Cost vs. Samples

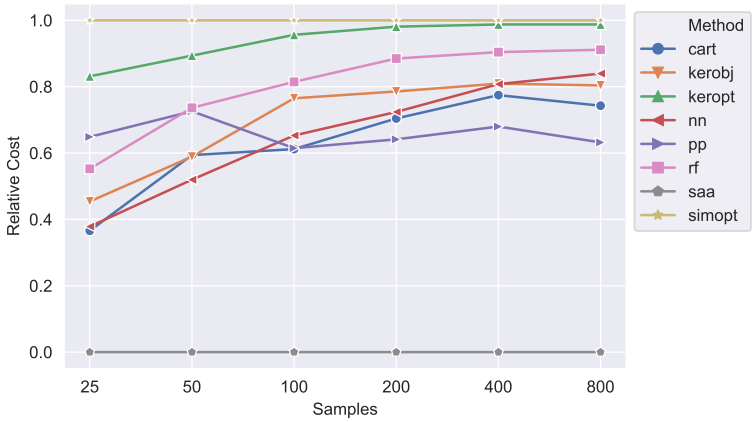


Note. Signal-to-noise ratio is 0.75. Prescriptive signal-to-noise ratio is 0.46.

has the same running time as `rf`, which seems to asymptotically be the same as the running time of `nn`. Thus, the Big-O analyses from Sections 4 and 5, which implied that both `kerobj` and `keropt` would take a factor of n times as long as `nn`, are too conservative here. Note also that the choice to make the test set fixed instead of scaling with n (as it would with a standard 70%-30% train-test split) disadvantages `keropt`, which requires very little time to test relative to the other methods.

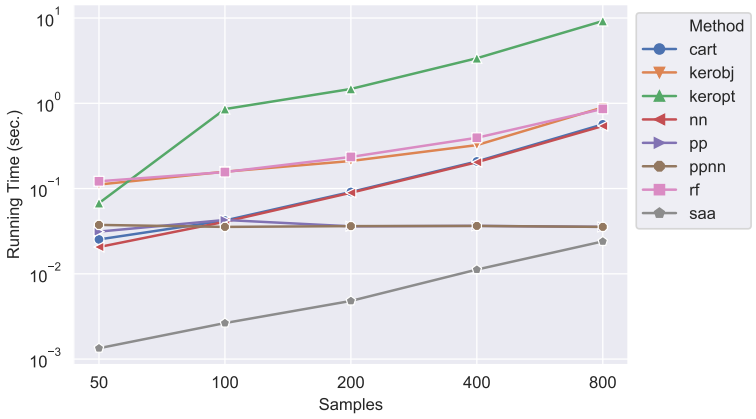
7.1.2. Extrapolation Next, we study how well the different methods extrapolate to unseen data. In order to do this, we draw from the 0% to 40% and 60% to 100% quantiles of the distribution of x_1 and x_2 to form the training set, and we draw from the 40% to 60% quantile of the distribution

Figure 5 Newsvendor Problem (Increased Nonlinearity in Cost Function): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.85. Prescriptive signal-to-noise ratio is 0.57. The method `ppnn` (average relative cost of 0.06) is excluded for readability.

Figure 6 Newsvendor Problem (Baseline): Running Time vs. Samples

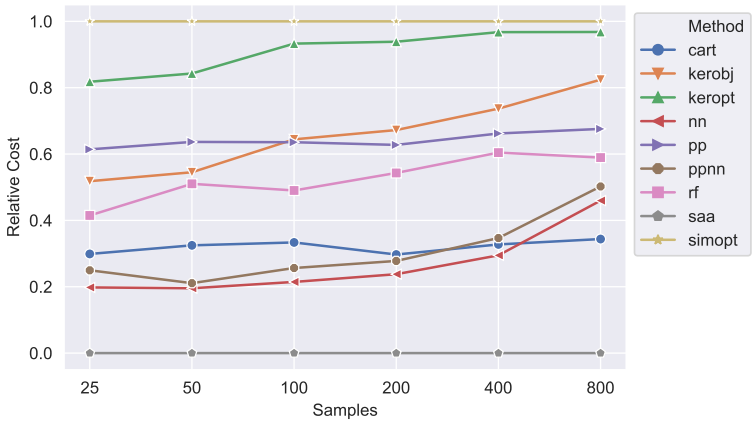


Note. Test set size is fixed at 100.

to form the test set. The relation of y and x remains the same as before. We call this procedure truncating the inner 20% of the training set distribution. We keep everything else, including all parameters and the training procedures for all of the methods, the same as in the baseline Figure 1. Figure 7 displays the results of this experiment. The methods `kerobj` and `keropt` are the only ones that perform better than `pp` and appear on track to recover the optimal decision.

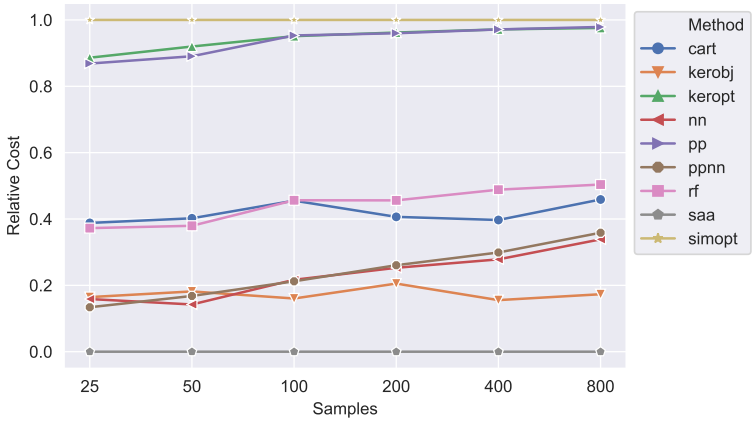
Figure 8 displays the results of an analogous experiment where the outer 20% of the training set distribution is truncated, and the test set is formed with the outer 20%. Here, `keropt` and `pp` recover the optimal decision while the other methods fail.

Figure 7 Newsvendor Problem (Inner 20% of Training Set Distribution Truncated): Relative Cost vs. Dimension with Truncation



Note. Signal-to-noise ratio for test set is 0.02. Prescriptive signal-to-noise ratio for test set is 0.10. The relative cost on each test set is cut off at 0.0 and 1.0 because of high variance in relative cost of methods `cart`, `nn`, `ppnn`, and `rf`.

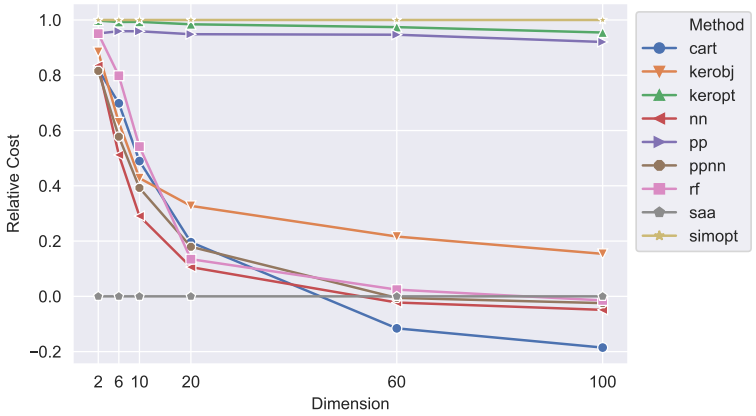
Figure 8 Newsvendor Problem (Outer 20% of Training Set Distribution Truncated): Relative Cost vs. Dimension with Truncation



Note. Signal-to-noise ratio for test set is 0.95. Prescriptive signal-to-noise ratio for test set is 0.81.

7.1.3. Effect of Covariate Dimension Since we partly motivated the development of `kerobj` and `keropt` by the curse of dimensionality for local methods in predictive machine learning, we lastly study the effect of increasing the dimension of x on performance in the prescriptive setting. In Figure 9, we increase the dimension of x_1, x_2, β_1, and β_2, while decreasing the magnitude of the components of β_1 and β_2 accordingly in order to maintain a stable signal-to-noise ratio. In Figure 9, we take $n = 800$ and report how the relative cost of the different methods changes with

Figure 9 Newsvendor Problem (High-Dimensional Dense β): Relative Cost vs. Dimension



Note. Signal-to-noise ratio is 0.85. Prescriptive signal-to-noise ratio is 0.62. Number of samples is 800.

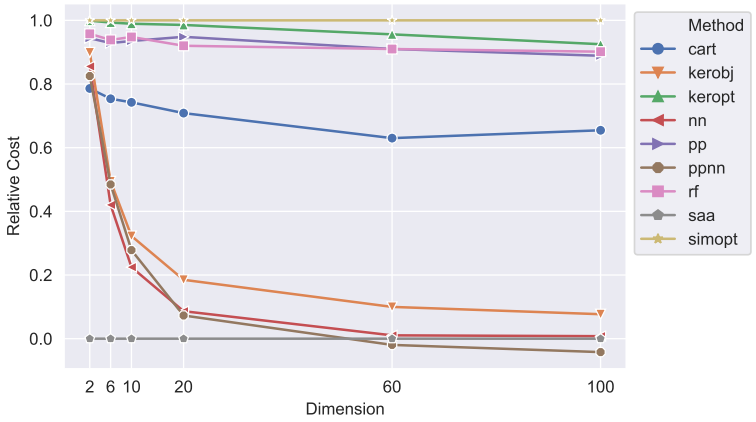
the combined dimension of β_1 and β_2. We can see that the curse of dimensionality in the predictive setting translates to the prescriptive setting. The local methods from Bertsimas and Kallus (2020) appear to have an exponential decrease in relative cost with an increase in covariate dimension. The method `keropt` overcomes the curse and has stable relative cost with increasing covariate dimension. The method `kerobj` does markedly better than the other methods from Bertsimas and Kallus (2020) but still is plagued by the curse of dimensionality.

Instead of x being high-dimensional and every component of x telling us a little bit about y, we can also have x being high-dimensional and only a few of the components tell us something about y. In Figure 10, we look at the extreme case when only one component of x_1 and one component of x_2 tell us something about y, i.e., only one component of β_1 and one component of β_2 is positive. Under this sparsity assumption, `keropt` does about the same as in the dense case. But in contrast to the dense case, the performance of `rf` and `cart` remains about stable.

7.2. Product Placement Problem

In the product placement problem, we need to decide how much inventory $z_1, \ldots, z_l$ of a product to place in each of l nodes of a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$. After we place the product, we observe a demand in each node $y_1, \ldots, y_l$, and we have to ship units of the product across arcs $\mathcal{A}$ so the inventory satisfies the demand. We pay some penalty for each unit of demand not satisfied. We let the cost of

Figure 10 Newsvendor Problem (High-Dimensional Sparse β): Relative Cost vs. Dimension



Note. Signal-to-noise ratio is 0.85. Prescriptive signal-to-noise ratio is 0.55. Number of samples is 800.

initial product placement be $h \in \mathbb{R}^{|\mathcal{N}|}$, the cost of shipping products along the edges be $g \in \mathbb{R}^{|\mathcal{A}|}$, and the penalty for not satisfying demand be $r \in \mathbb{R}^{|\mathcal{N}|}$.

We do not observe y before placing the products but we do observe some other related quantities x. Thus, the problem we address is

$$\min_{z \geq 0} \mathbb{E}[c(z;y) \,|\, x]$$

where

$$\begin{aligned} c(z;y) = h'z + \min_{f,p} \; & g'f + r'p \\ \text{s.t. } & Nf \geq z - y + p \\ & f \geq 0 \end{aligned} \tag{22}$$

and N is the node-arc matrix of $\mathcal{G}$. As usual, we don't know the distribution of x and y, but we have data $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$. The cost function (22) is convex, but it also has a two-stage linear structure. It turns out that for a two-stage linear cost function, the problem solved to obtain a decision for `kerobj`, the probrlem solved to obtain a decision for `keropt`, as well as the problems solved to obtain a decision for the methods from Bertsimas and Kallus (2020), can all be recast as LOPs using the standard technique of creating new decision variables for each data point (see, for example, Bertsimas and Tsitsiklis 1997, Section 6.5).

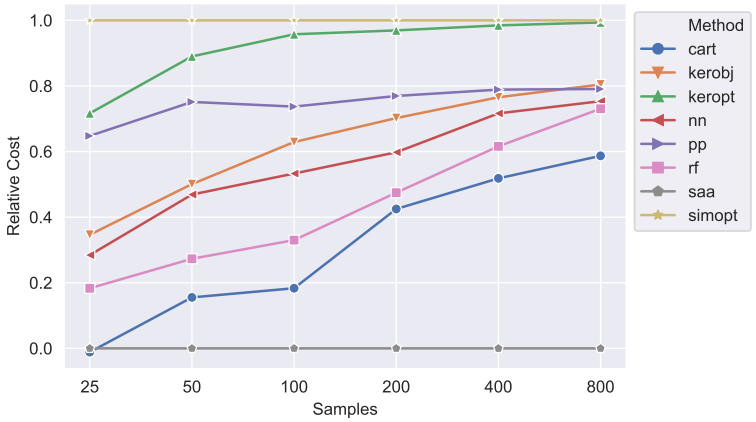
The following are further details for the structure of the cost function and the simulated data. The graph $\mathcal{G}$ has d nodes and $2d$ arcs. The first d arcs connect the nodes circularly. The last d arcs

are generated randomly once for each trial (i.e., 30 different sets of arcs are generated for each point plotted). We also provide results for a dense graph, which turn out to be similar. We use that x is log-normal with dimension d, and y is generated as $\max(\beta x + \epsilon, 0)$, where β is some $d \times d$ matrix (generated once randomly for each trial), and ϵ is exponentially distributed noise. The costs are $h = (2, \ldots, 2)$, $g = (1, \ldots, 1)$, and $r = (10, \ldots, 10)$.

Moving on to the results, Figure 11 is the baseline for the product placement problem with $d = 5$. Like in the newsvendor problem, `keropt` performs better with $n = 50$ data points than the strongest method from Bertsimas and Kallus (2020) does with $n = 800$ data points. Unlike in the newsvendor problem, `kerobj` clearly outperforms `rf` and `nn`. Again, `pp` performs well but has the unfair advantage of knowing the model is linear, as illustrated by the performance of `ppnn`. It is also apparent that, as we would expect from a point prediction algorithm, `pp` does not converge to optimality. When changing the signal-to-noise ratio (Figure 13 and Figure 14), the performance of `keropt` remains about the same. But the performance of `kerobj` relative to `nn` gets better with increased noise. In Figure 15, we go back to the baseline noise and increase the dimension (of x, y, and z, which all have the same dimension in this problem) to $d = 10$. The gulf between `keropt` and the other methods becomes even more dramatic, with `keropt` doing better with $n = 25$ than method other than `pp` does with $n = 800$. In Figure 16, we change the cost function by making the graph $\mathcal{G}$ dense, and the results are very similar.

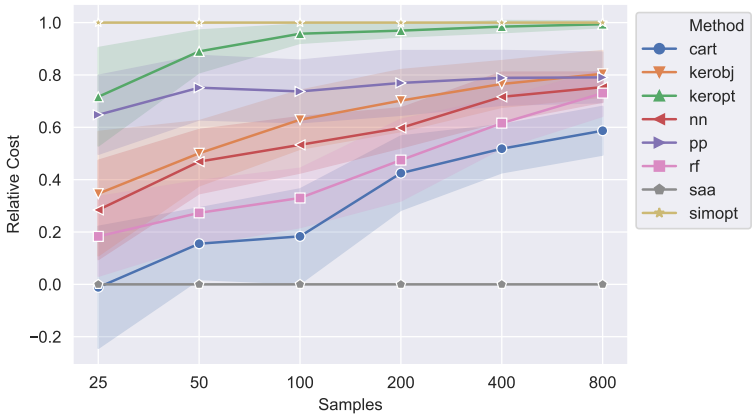
7.2.1. Running Time Figure 17 shows running times for the most computationally demanding setting, with dimension $d = 10$ and the dense graph $\mathcal{G}$. Running time is calculated as the sum of training time and test time after cross-validation, with the test set size fixed to 100 points. Since this is a two-stage problem, the number of variables in the linear or quadratic optimization formulations for each of the methods scales linearly with the number of data points for `kerobj`, `keropt`, and all methods from Bertsimas and Kallus (2020). As predicted in Sections 4 and 5, with the test size constant, this results on all of these methods scaling identically with n. If the test set size instead of being constant scaled linearly with n, we would expect `keropt` to scale better than the other methods, which we see in Figure 18 with `keropt` around 10 times faster than all of the other methods when $n = 800$.

Figure 11 Product Placement Problem (Baseline): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.59. Prescriptive signal-to-noise ratio is 0.54. The method `ppnn` (average relative cost of -0.03) is excluded for readability.

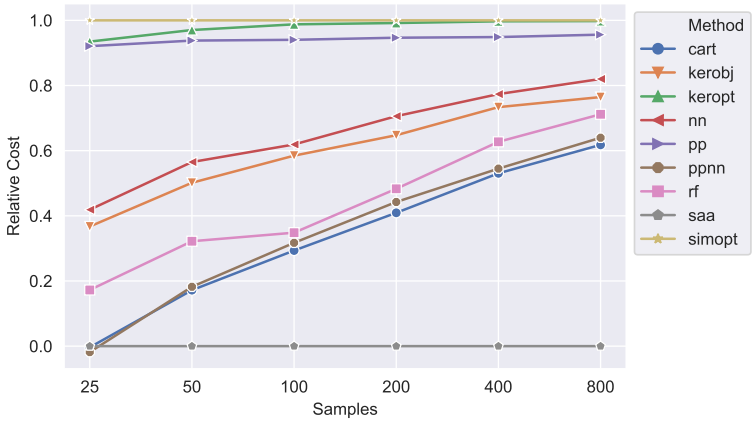
Figure 12 Product Placement Problem (Baseline): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.59. Prescriptive signal-to-noise ratio is 0.54. The method `ppnn` (average relative cost of -0.03) is excluded for readability.

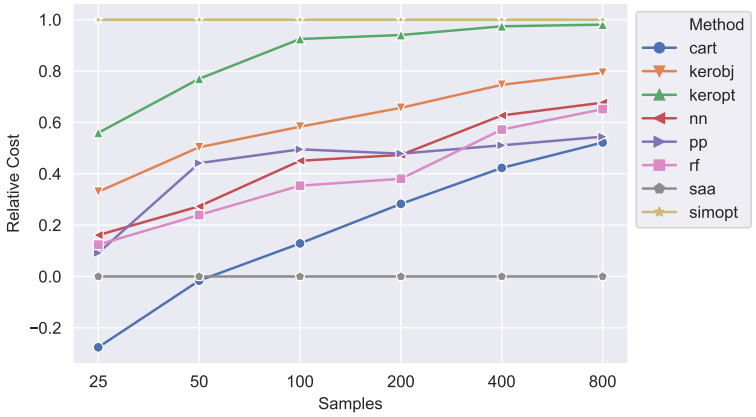
7.2.2. Extrapolation To assess how well the methods perform with extrapolating to unseen data, in Figure 19, we truncate the inner 20% of the distribution of x for the training data, and only use this inner 20% for the test data. Nothing else changes from the baseline Figure 11. Unlike in the newsvendor problem, this truncation seems to make the prescriptive problem easier. All methods outperform their baseline in Figure 11, but `keropt` still performs the best. In Figure 20, we truncate the outer 20% of the distribution of x for the training data, and use only the outer

Figure 13 Product Placement Problem (Increased Signal-to-Noise Ratio): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.93. Prescriptive signal-to-noise ratio is 0.84.

Figure 14 Product Placement Problem (Decreased Signal-to-Noise Ratio): Relative Cost vs. Samples

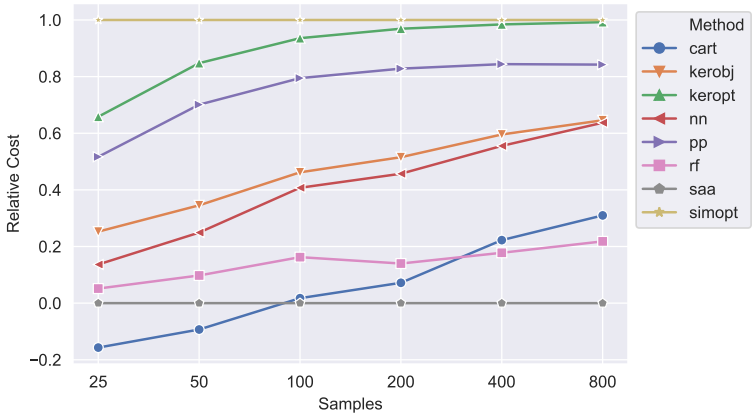


Note. Signal-to-noise ratio is 0.34. Prescriptive signal-to-noise ratio is 0.36. The method `ppnn` (average relative cost of -0.57) is excluded for readability.

20% for the test data. Like in the newsvendor problem, `keropt` and `pp` recover the optimal decision while the other methods fail.

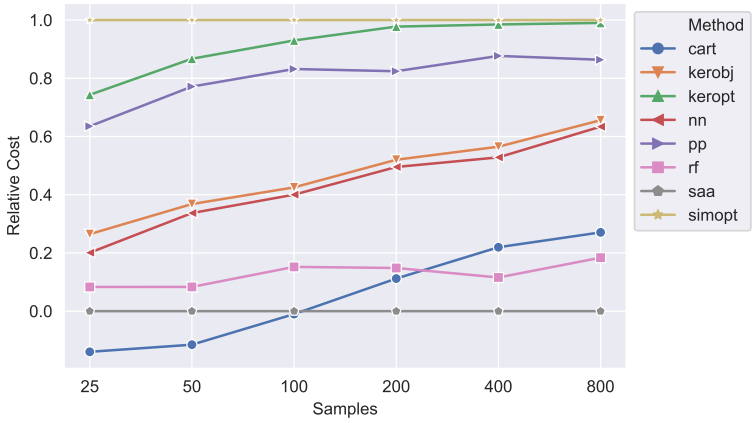
7.2.3. Effect of Dimension To close, we further explore the effect of varying dimension. Earlier in the section we increased the dimension of x and z at the same time. Here, we separate the effect of increasing x and z. In Figure 21, we fix the dimension of z and vary the dimension of x, keeping the signal-to-noise ratio constant. This leads to a curse of dimensionality like in the newsvendor problem, although less pronounced. The method `keropt` overcomes the curse. The method `kerobj` doesn't but still does better than the local methods. In Figure 22, we fix the

Figure 15 Product Placement Problem (Increased Dimension): Relative Cost vs. Samples



Note. Signal-to-noise ratio is 0.61. Prescriptive signal-to-noise ratio is 0.63. The method `ppnn` (average relative cost of -0.34) is excluded for readability.

Figure 16 Product Placement Problem (Increased Dimension and Dense Graph): Relative Cost vs. Samples



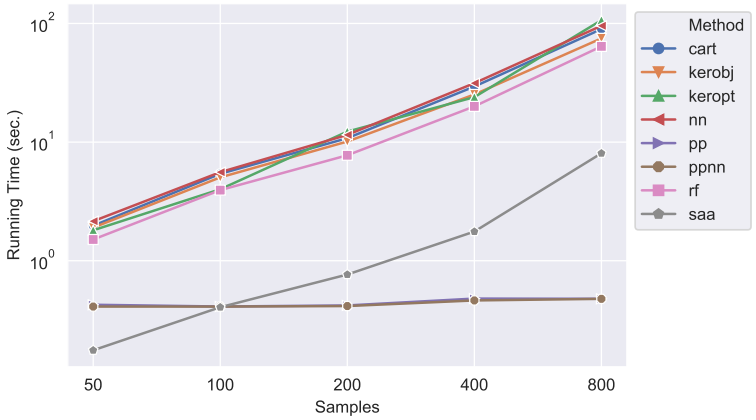
Note. Signal-to-noise ratio is 0.60. Prescriptive signal-to-noise ratio is 0.67. The methods `ppnn` (average relative cost of -0.31) is excluded for readability.

dimension of x and vary the dimension of z. We see that changing the dimension of z by itself does not hurt or help any of the methods.

8. Conclusion

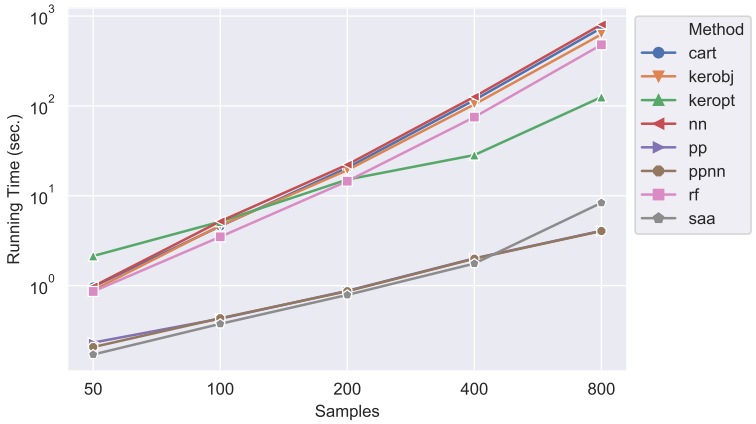
We started in Sections 3 by providing an overview of the framework of reproducing kernel Hilbert spaces. In Section 4 and Section 5, we used the framework to develop two new methods for solving

Figure 17 Product Placement Problem (Increased Dimension and Dense Graph): Running Time vs. Samples



Note. Test set size is fixed at 100.

Figure 18 Product Placement Problem (Increased Dimension and Dense Graph): Running Time vs. Samples



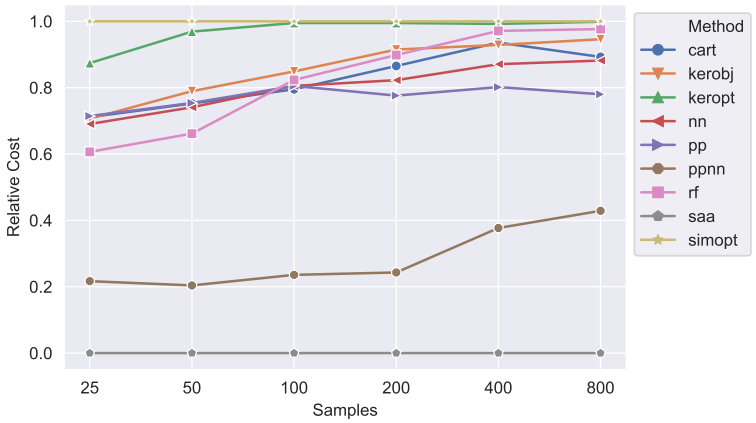
Note. Test set size is same as training set size.

the problem

$$\min_z \mathbb{E}_y[c(z;y) \,|\, x = x^0],$$

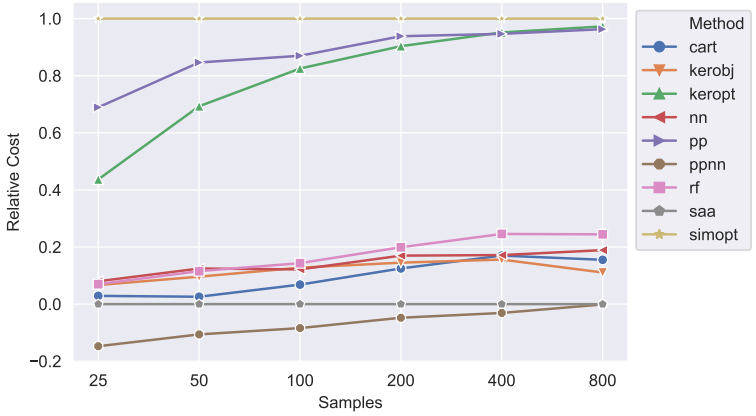
given historical data of x and y. We proved the asymptotic optimality of both methods and showed that their rates of convergence match standard rates from the machine learning literature. We provided tractable and scalable reformulations of both methods when the naive formulation was not tractable or scalable. In Section 6, we showed how to incorporate constraints and demonstrated a tractable way to handle the causal inference setting when the decision z affects y. To demonstrate the applicability of the ideas in this paper to other data-driven problems in operations research, we provided a preliminary proposal of a method to solve multistage optimization problems.

Figure 19 Product Placement Problem (Inner 20% of Training Set Distribution Truncated): Relative Cost vs. Samples



Note. Signal-to-noise ratio for test set is 0.00. Prescriptive signal-to-noise ratio for test set is 0.53.

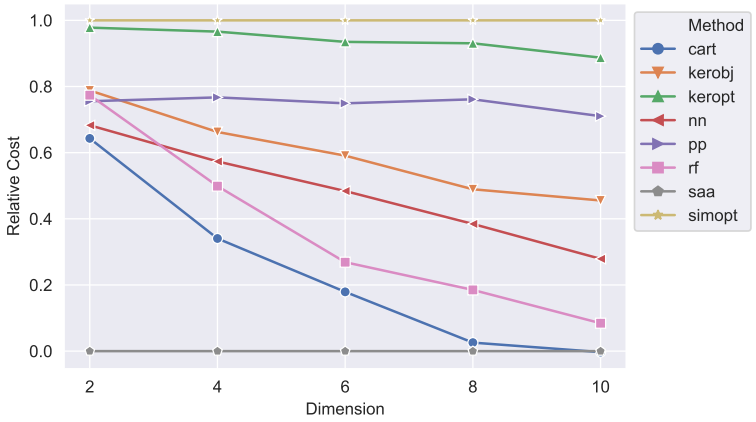
Figure 20 Product Placement Problem (Outer 20% of Training Set Distribution Truncated): Relative Cost vs. Samples



Note. Signal-to-noise ratio for test set is 0.85. Prescriptive signal-to-noise ratio for test set is 0.90.

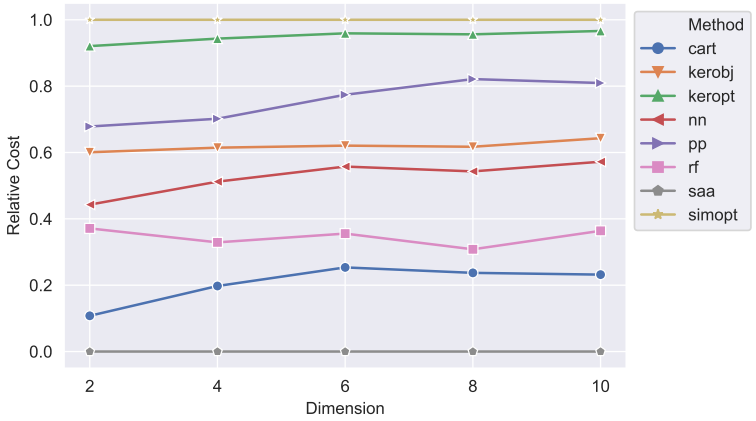
In Section 7, we examined the performance of the two new methods on two different problems compared to a variety of existing methods. One of the methods, the kernel objective prediction method, offered a moderate improvement over existing methods in certain situations. However, the other method, the kernel optimizer prediction method, performed substantially better than all existing methods in every condition tested. (We are disregarding point prediction with linear regression, which was sometimes comparable, but had the unfair advantage of knowing the form of the data-generation model.) The magnitude of improvement was such that in most conditions, the

Figure 21 Product Placement Problem (Varying Covariate Dimension): Relative Cost vs. Dimension



Note. Signal-to-noise ratio is 0.60. Prescriptive signal-to-noise ratio is 0.54. The method `ppnn` (average relative cost of -0.18) is excluded for readability. Number of samples is 100.

Figure 22 Product Placement Problem (Varying Decision Dimension): Relative Cost vs. Dimension



Note. Signal-to-noise ratio is 0.60. Prescriptive signal-to-noise ratio is 0.53. The method `ppnn` (average relative cost of -0.09) is excluded for readability. Number of samples is 100.

kernel optimizer prediction method achieved lower cost with $n = 50$ than any existing method did with $n = 800$. Moreover, when the training set distribution was truncated, it was the only one able to recover the optimal decision, and when the covariate dimension increased, it largely maintained its performance while the others' degraded exponentially. These improvements did not come at the expense of running time, as its running time scaled similarly to or better than the existing methods.

References

Aronszajn N (1950) Theory of reproducing kernels. *Transactions of the American Mathematical Society* 68:337–404.

Ban GY, Rudin C (2019) The big data newsvendor: Practical insights from machine learning. *Operations Research* 67(1):90–108.

Ben-Tal A, El Ghaoui L, Nemirovski A (2009) *Robust Optimization* (Princeton University Press).

Bertsekas DP (1995) *Dynamic Programming and Optimal Control* (Belmont, MA: Athena Scientific).

Bertsimas D, Brown DB, Caramanis C (2011) Theory and applications of robust optimization. *SIAM Review* 53(3):464–501.

Bertsimas D, Gupta V, Kallus N (2018) Data-driven robust optimization. *Mathematical Programming* 167(2):235–292.

Bertsimas D, Kallus N (2020) From predictive to prescriptive analytics. *Management Science* 66(3):1025–1044.

Bertsimas D, McCord C (2017) From predictions to prescriptions in multistage optimization problems. *submitted for publication* .

Bertsimas D, Tsitsiklis JN (1997) *Introduction to Linear Optimization* (Athena Scientific).

Birge JR, Louveaux F (2011) *Introduction to Stochastic Programming* (Springer).

Bousquet O, Elisseeff A (2002) Stability and generalization. *Journal of Machine Learning Research* 2:499–526, ISSN 15324435, URL `http://dx.doi.org/10.1162/153244302760200704`.

Cucker F, Smale S (2002) On the mathematical foundations of learning. *Bulletin of the American Mathematical Society* 39(1):1–49, ISSN 02730979, URL `http://dx.doi.org/10.1090/S0273-0979-01-00923-5`.

De Vito E, Rosasco L, Caponnetto A, Piana M, Verri A (2004) Some properties of regularized kernel methods. *Journal of Machine Learning Research* 5:1363–1390.

Delage E, Ye YY (2010) Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research* 58(3):595–612, ISSN 0030-364X, URL `http://dx.doi.org/10.1287/opre.1090.0741`.

Hanasusanto GA, Kuhn D (2013) Robust data-driven dynamic programming. *Advances in Neural Information Processing Systems* 827–835, ISSN 10495258, URL http://papers.nips.cc/paper/5123-robust-data-driven-dynamic-programming.

Hannah L, Powell WB, Blei DM (2010) Nonparametric density estimation for stochastic optimization with an observable state variable. *Advances in Neural Information Processing Systems 23* 820–828, ISSN 1523-4614, URL http://dx.doi.org/10.1287/msom.1110.0340.

Hastie T, Tibshirani R, Friedman J (2009) *The Elements of Statistical Learning* (New York: Springer-Verlag), 2nd edition.

Kleywegt AJ, Shapiro A, Homem-de Mello T (2002) The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization* 12(2):479–502, ISSN 1052-6234, URL http://dx.doi.org/10.1137/S1052623499363220.

Meir R, Zhang T (2003) Generalization error bounds for bayesian mixture algorithms. *Journal of Machine Learning Research* 4:839–860.

Micchelli CA, Xu Y, Zhang H (2006) Universal kernels. *Journal of Machine Learning Research* 7:2651–2667.

Mohri M, Rostamizadeh A, Talwalkar A (2018) *Foundations of Machine Learning* (MIT Press), 2nd edition.

Oneto L, Ridella S, Anguita D (2016) Tikhonov, Ivanov and Morozov regularization for support vector machine learning. *Machine Learning* 103:103–136.

Poggio T, Smale S (2005) The mathematics of learning: Dealing with data. *2005 International Conference on Neural Networks and Brain* 3:537–544, URL http://dx.doi.org/10.1109/ICNNB.2005.1614860.

Rosasco L, Poggio T (2012) Machine learning: A regularization approach, MIT 9.520 lecture notes. URL http://www.mit.edu/~9.520/spring12/index.html.

Scott C (2014) Statistical learning theory lecture notes. URL http://web.eecs.umich.edu/~cscott/past_courses/eecs598w14/index.html.

Shapiro A (2003) Stochastic programming. *Handbooks in Operations Research and Management Science* 10:353–425, ISSN 09270507, URL http://dx.doi.org/10.1016/S0927-0507(03)10006-0.

Shapiro A, Nemirovski A (2005) On complexity of stochastic programming problems. *Continuous Optimization* 99:111–146, URL http://dx.doi.org/10.1007/0-387-26771-9_4.

Steinwart I, Christmann A (2008) *Support Vector Machines* (Springer-Verlag).

Vapnik VN (1998) *Statistical Learning Theory* (John Wiley & Sons, Inc.), 1st edition.